

Applied modeling with BART: How, Why, and When

Sameer K. Deshpande

University of Wisconsin–Madison

18 May 2024

Overview

Motivations

Introducing BART

BART in practice

Parting Thoughts

Catch probability & QB rankings

- Which QB's throw the “most catchable” balls?
- If we knew $\mathbb{P}(\text{catch})$ exactly, we could rank QB's based on
 - ▶ Avg. $\mathbb{P}(\text{catch})$
 - ▶ Prop. of time $\mathbb{P}(\text{catch})$ exceeds a threshold (e.g., 80%)
 - ▶ Prop. of time QB threw to target w/ highest $\mathbb{P}(\text{catch})$ on play (EHCP)

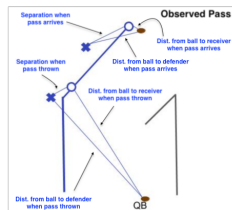
Catch probability & QB rankings

- Which QB's throw the “most catchable” balls?
- If we knew $\mathbb{P}(\text{catch})$ exactly, we could rank QB's based on
 - ▶ Avg. $\mathbb{P}(\text{catch})$
 - ▶ Prop. of time $\mathbb{P}(\text{catch})$ exceeds a threshold (e.g., 80%)
 - ▶ Prop. of time QB threw to target w/ highest $\mathbb{P}(\text{catch})$ on play (EHCP)

- $\mathbb{P}(\text{catch}) = F(\mathbf{x}_c, \mathbf{x}_r, \mathbf{x}_a)$

- ⊖ F is highly non-linear

- ⊖ F may involve complex interactions



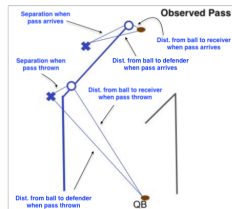
Catch probability & QB rankings

- Which QB's throw the “most catchable” balls?
- If we knew $\mathbb{P}(\text{catch})$ exactly, we could rank QB's based on
 - ▶ Avg. $\mathbb{P}(\text{catch})$
 - ▶ Prop. of time $\mathbb{P}(\text{catch})$ exceeds a threshold (e.g., 80%)
 - ▶ Prop. of time QB threw to target w/ highest $\mathbb{P}(\text{catch})$ on play (EHCP)

- $\mathbb{P}(\text{catch}) = F(\mathbf{x}_c, \mathbf{x}_r, \mathbf{x}_a)$

- ⊖ F is highly non-linear

- ⊖ F may involve complex interactions



- Uncertainty about F **propagates** to uncertainty in QB rankings!

A more general problem: nonparametric regression

- Observe $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$ and model

$$y_n = f(\mathbf{x}_n) + \sigma \epsilon_n; \epsilon_n \sim \mathcal{N}(0, 1)$$

- We have the following goals:
 - ▶ Prediction: value of $f(\mathbf{x}^*)$ & $y^* = f(\mathbf{x}^*) + \sigma \epsilon^*$
 - ▶ UQ: uncertainty intervals for $f(\mathbf{x}^*)$ & y^*
 - ▶ Variable importance/selection: on which X_j does f depend?

Approaches to learning f

- Assume $f(\mathbf{x}) = \sum_d \beta_d \phi_d(\mathbf{x})$
 - ▶ Basis $\{\phi_d\}$ could be linear, polynomial, splines, Fourier, etc.
 - ▶ Estimate β_d 's with OLS, LASSO, Bayesian linear regression, etc.
 - ☒ Correctly specifying $\{\phi_D\}$ is extremely hard!

Approaches to learning f

- Assume $f(\mathbf{x}) = \sum_d \beta_d \phi_d(\mathbf{x})$
 - ▶ Basis $\{\phi_d\}$ could be linear, polynomial, splines, Fourier, etc.
 - ▶ Estimate β_d 's with OLS, LASSO, Bayesian linear regression, etc.
 - ☒ Correctly specifying $\{\phi_D\}$ is extremely hard!
- Classification & regression trees
 - ▶ Train a single regression tree to approximate f
 - 😊 Interpretable, accurate, avoids pre-specifying form of f
 - 😞 Often unstable & non-smooth

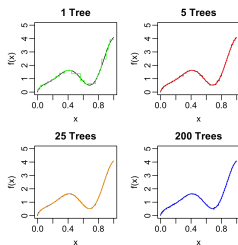
Approaches to learning f

- Assume $f(\mathbf{x}) = \sum_d \beta_d \phi_d(\mathbf{x})$
 - ▶ Basis $\{\phi_d\}$ could be linear, polynomial, splines, Fourier, etc.
 - ▶ Estimate β_d 's with OLS, LASSO, Bayesian linear regression, etc.
 - ☒ Correctly specifying $\{\phi_D\}$ is extremely hard!
- Classification & regression trees
 - ▶ Train a single regression tree to approximate f
 - 😊 Interpretable, accurate, avoids pre-specifying form of f
 - 😞 Often unstable & non-smooth
- Ensemble methods
 - ▶ Approximate f with a (weighted) average of “weak learners”:

$$\hat{f}(\mathbf{x}) = \sum_{m=1}^M w_m \hat{f}_m(\mathbf{x})$$

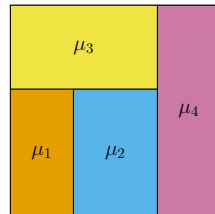
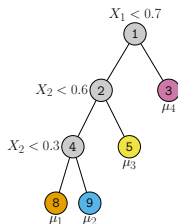
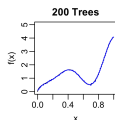
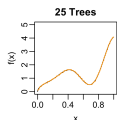
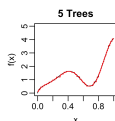
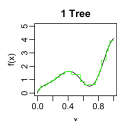
- ▶ Each \hat{f}_m may not fit data well but together they do
- 😊 Tremendous empirical success (e.g. Netflix Prize, Kaggle)

Step function approximations



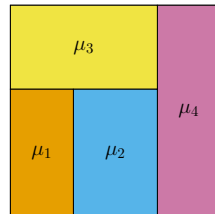
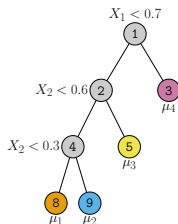
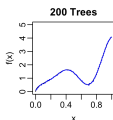
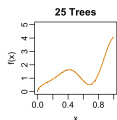
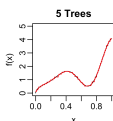
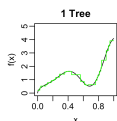
- Step functions are universal function approximators!

Step function approximations



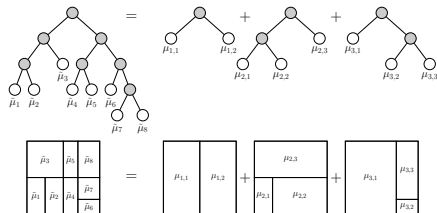
- Step functions are universal function approximators!
- Step functions can be represented as binary regression trees

Step function approximations



- Step functions are universal function approximators!
 - Step functions can be represented as binary regression trees
- ☒ Often need very deep tree to appx complicated f well

Sums of trees



- Sum of step functions is just another step function!
- Sums of regression trees is a more complicated regression tree!
- ☺ Averaging/ensembling introduces certain degree of smoothness

Digression: Pointillism



A Sunday afternoon on the island of La Grande Jatte, Georges Seurat

Source

Motivations

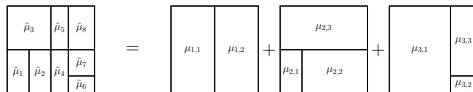
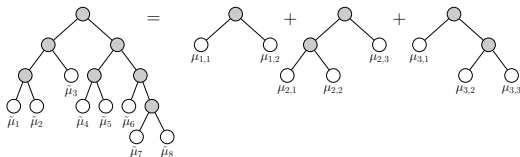
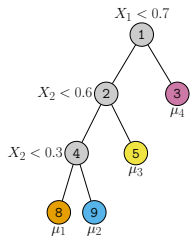
Introducing BART

BART in practice

Parting Thoughts

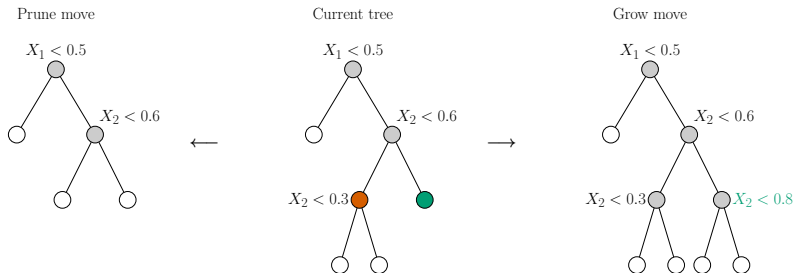
Bayesian Additive Regression Trees

- Regression: $y_n = f(\mathbf{x}_n) + \sigma \varepsilon_n; \varepsilon_n \sim \mathcal{N}(0, 1)$ & $\mathbf{x}_n \in [0, 1]^p$
- Main idea: approximate $f(\mathbf{x})$ with sum of M regression trees
- Prior encourages trees to be “weak learners”
- Gibbs sampler: update each tree conditionally on fit of all others



Posterior computation & implementation

- Metropolis-within-Gibbs: update each tree sequentially fixing others
 - ▶ Update decision tree with MH (randomly grow or prune tree)
 - ▶ Update leaf parameters / tree outputs conditional on tree
- No optimization involved!!



Outline

Motivations

Introducing BART

BART in practice

Parting Thoughts

Several implementations

- **BART**: [Sparapani, Spanbauer, & McCulloch \(2021\)](#)
 - ▶ Support for many extensions (e.g., classification, survival)
 - ▶ Based on efficient C++ regression tree class & sampler
- **dbarts**: [Dorie \(2020\)](#)
 - ▶ Makes it easy to include a sum-of-trees component in a larger model
 - ▶ E.g. $y_i = \mathbf{x}_i^\top \beta + f(\mathbf{x}_i) + \sigma \epsilon_i$
- **flexBART**: available at ([GitHub repo](#))
 - ▶ Flexibly handle categorical predictors and observations on networks
 - ▶ Much faster than **BART**
 - ▶ Still under active development
- **bartMachine**: [Kapelner & Bleich \(2014\)](#)
 - ▶ Core fitting written in Java
 - ☹ Non-trivial overhead in installing & setting up system
- **PyMC-BART**: [From the PyMC team](#)
 - ▶ Uses sequential Monte Carlo & is rather different than the above

Outline

Motivations

Introducing BART

BART in practice

Parting Thoughts

Variable importance & selection

- Variable importance in treed models is still area of on-going research
- For BART, counting # decision rules using X_j **not recommended**

Variable importance & selection

- Variable importance in treed models is still area of on-going research
- For BART, counting # decision rules using X_j **not recommended**
- Partial dependence plots
 - ▶ $\bar{f}_j(x) = n^{-1} \sum_i f(x_{i,1}, \dots, x_{i,j-1}, x, x_{i,j+1}, \dots, x_{i,p})$
 - ▶ Implemented in `dbarts::pdbart` and easy to do by hand

Variable importance & selection

- Variable importance in treed models is still area of on-going research
- For BART, counting # decision rules using X_j **not recommended**
- Partial dependence plots
 - ▶ $\bar{f}_j(\mathbf{x}) = n^{-1} \sum_i f(x_{i,1}, \dots, x_{i,j-1}, \mathbf{x}, x_{i,j+1}, \dots, x_{i,p})$
 - ▶ Implemented in `dbarts::pdbart` and easy to do by hand
- [Linero \(2016\)](#) modifies BART so that
 - ▶ Split on X_j with prob. θ_j (in prior & in MH transition)
 - ▶ θ given a sparsity-inducing Dirichlet prior
 - ▶ Adaptation: more accepted splits on $X_j \Rightarrow$ more proposed splits on X_j
 - ▶ Select X_j if more than 50% of ensembles involve a split on X_j

BART extensions

- Classification: probit w/ [Albert & Chib \(1993\)](#) data augmentation
- Survival models: [Sparapani et al. \(2016\)](#)
- Log-linear models: [Murray \(2019\)](#)
- Heteroscedasticity: [Pratola et al. \(2020\)](#)
 - ▶ $y_n = f(\mathbf{x}_n) + \sigma(\mathbf{x}_n)\varepsilon_n$, write $\log(\sigma^2(\mathbf{x}))$ as a sum-of-trees!
- Monotonic BART: [Chipman et al. \(2019\)](#)
- Estimating smooth functions
 - ▶ [Starling et al. \(2020\)](#): jumps μ_ℓ are Gaussian processes
- Varying coefficient models: [D. et al. \(2020+\)](#)
 - ▶ $Y = \beta_0(Z) + \beta_1(Z)X_1 + \cdots + \beta_p(Z)X_p + \sigma\epsilon$
 - ▶ E.g. time & demographic varying mediation effects
- Causal inference: [Hill \(2011\)](#) & [Hahn et al \(2020\)](#)

Concluding remarks

- BART: approximate f with sum of regression trees
- Avoids detailed pre-specification of functional form of $f(\mathbf{x}) = \mathbb{E}[Y \mid X = \mathbf{x}]$
- Excellent performance off-the-shelf

Concluding remarks

- BART: approximate f with sum of regression trees
- Avoids detailed pre-specification of functional form of $f(\mathbf{x}) = \mathbb{E}[Y \mid X = \mathbf{x}]$
- Excellent performance off-the-shelf

Concluding remarks

- BART: approximate f with sum of regression trees
- Avoids detailed pre-specification of functional form of $f(\mathbf{x}) = \mathbb{E}[Y \mid X = \mathbf{x}]$
- Excellent performance off-the-shelf

Thanks, y'all!

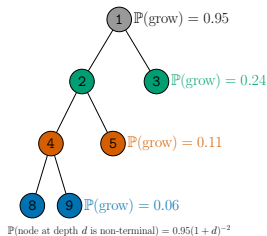
Email: sameer.deshpande@wisc.edu

Package: <https://github.com/skdeshpande91/flexBART>

Website: <https://skdeshpande91.github.io>

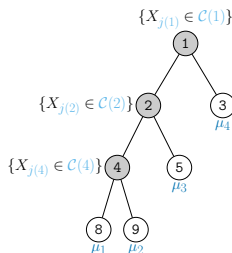
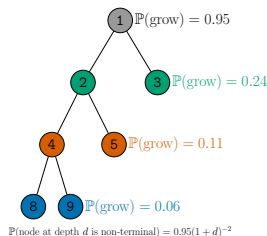
Preprint: [\[arXiv:2211.04456\]](https://arxiv.org/abs/2211.04456)

A prior over regression trees



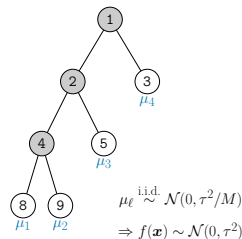
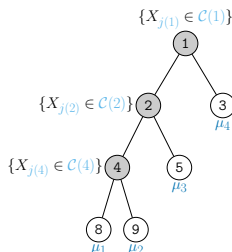
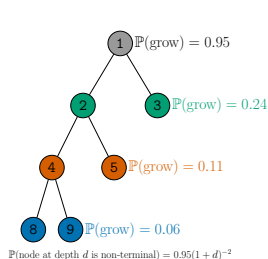
- Branching process prior on graphical structure
- Overwhelming prior prob. that tree depth ≤ 5

A prior over regression trees



- Branching process prior on graphical structure
- Overwhelming prior prob. that tree depth ≤ 5
- Decision rule $\{X_j \in \mathcal{C}\}$
 - ▶ X_j continuous: \mathcal{C} is an interval $[0, c)$
 - ▶ X_j categorical: \mathcal{C} is a discrete subset of X_j 's levels

A prior over regression trees



- Branching process prior on graphical structure
- Overwhelming prior prob. that tree depth $\gg 5$
- Decision rule $\{X_j \in \mathcal{C}\}$
 - ▶ X_j continuous: \mathcal{C} is an interval $[0, c]$
 - ▶ X_j categorical: \mathcal{C} is a discrete subset of X_j 's levels
- Leaf outputs $\mu_\ell \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \tau^2/M)$

Decision rule prior

1. Draw $j \sim \text{Multinomial}(\theta_1, \dots, \theta_p)$
where $\theta_j = \mathbb{P}(\text{split on } X_j)$
2. Compute set of all available values \mathcal{A}_j
 - ▶ \mathcal{A}_j determined by rules at ancestors
 - ▶ X_j continuous $\rightarrow \mathcal{A}$ is an interval
 - ▶ X_j categorical $\rightarrow \mathcal{A}$ is discrete set
3. Draw random subset \mathcal{C} from \mathcal{A}_j
 - ▶ X_j continuous: draw $c \sim \mathcal{U}(\mathcal{A}_j)$ and set $\mathcal{C} = [0, c]$
 - ▶ X_j categorical: assign elements of \mathcal{A}_j to \mathcal{C} with probability 0.5

