BAYESIAN NONPARAMETRIC GENERALIZATION OF TREE BASED MACHINE LEARNING APPROACHES

Guide: Dr. Sourabh Bhattacharya

Arindam Roy Chowdhury

March, 2021

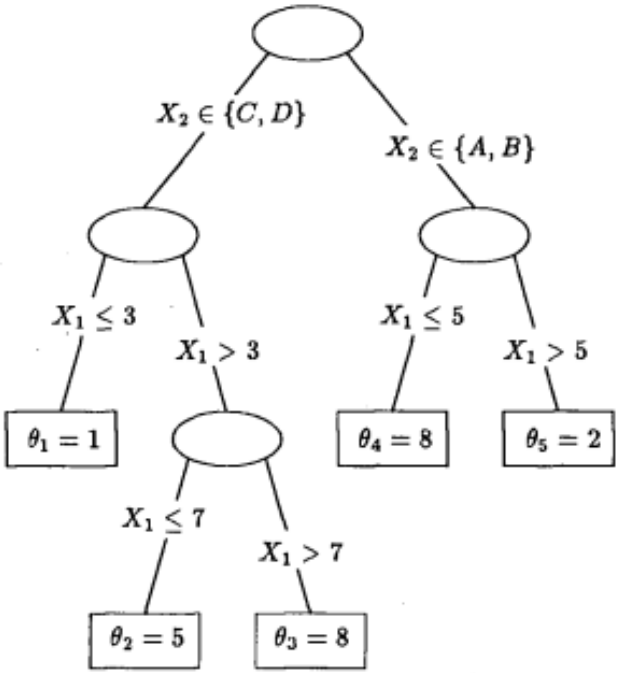
Table of Contents

Introduction

- The importance of tree based methods in machine learning research is undeniable. Classification and Regression Trees (CART) (Breiman, Friedman, Olshen & Stone 1984) have fundamental importance in this regard.
- Ensemble methods that combine a set of tree models, such as boosting , bagging and random forests have turned out to be much popular with time.
- Bayesian versions motivated by these methods, such as Bayesian CART and BART have established themselves as effective weapons for attacking challenging machine learning problems.

Table of Contents

The CART model



Assumptions:

- If x lies in the region denoted by the i^{th} terminal node, then $y|x$ has distribution $f(y|\theta_i)$, where f is a parametric family indexed by θ_i.
- The distribution of y values inside a terminal node are iid.
- Distribution of y values across different terminal nodes are independent.

Figure: A CART model with $y \sim N(\theta_g, \sigma^2)$

y_{ij} := j^{th} observation associated with the i^{th} terminal node.
n_i := number of observations in the i^{th} terminal node.

$$p(Y|X, \Theta, T) = \prod_{i=1}^{b} f(Y_i|\theta_i) = \prod_{i=1}^{b} \prod_{j=1}^{n_i} f(y_{ij}|\theta_i)$$

f is taken as required for the problem.

Table of Contents

The Mixture Model

We consider a situation when Y comes from a mixture distribution with k(random) components. Consider, for large M:

$$[Y|X, \Theta_1, ..., \Theta_M, T_1, ..., T_M] = \sum_{j=1}^{M} p_j f(Y|X, \Theta_j, T_j) \tag{1}$$

- For $j = 1, ..., M,\ p_j \in [0, 1]$ with $\sum p_j = 1$ and $(\Theta_j, T_j) \sim G$
- $G \sim DP(\alpha, G_0)$, the Dirichlet process with scale parameter α and some expected measure G_0

Due to discreteness of Dirichlet process, $(\Theta_i, T_i); i = 1, ..., M$, are coincident with positive probability. Hence [1] reduces to :

$$[Y|X, \Theta_1, ..., \Theta_M, T_1, ..., T_M] = \sum_{i=1}^{k} p_i^* f(Y|X, \Theta_i^*, T_i^*) \tag{2}$$

where $k \leq M,\ (\Theta_i^*, T_i^*); i = 1, ..., k$ are the distinct elements of $(\Theta_i, T_i);\ i = 1, ..., M$, and p_i^* is the sum of p_j over indices j such that (Θ_j, T_j) are coincident.

Reparameterization

Allocation variables: $Z = (z_1, \cdots, z_n)'$ where z_i denote the class to which the i^{th} point belongs.

Let $\Theta^*_M = \{(\Theta^*_l, T^*_l)\}_{l=1}^k$ denote the distinct components in Θ_M. Configuration vector $C = (c_1, \cdots, c_M)'$ where $c_j = l$ iff $(\Theta_j, T_j) = (\Theta^*_l, T^*_l)$ for $j \in \{1, \cdots, M\}$ and $l \in \{1, \cdots, k\}$.

Hence, the Model:

$$p(z_i = j) = \frac{1}{M}, \quad \forall j \in \{1, ..., M\}$$

$$(\Theta_j, T_j) \sim DP(\alpha, G_0)$$

$$p(y_i | z_i = j, \Theta_j, T_j) = f(y_i | \theta_{g(ij)})$$

$g(ij)$ denotes the terminal node to which the i^{th} sample point is assigned to by the j^{th} tree. Here, $\forall j, g(ij) \in \{1, ..., b_j\}$ where b_j denotes the number of terminal nodes in the j^{th} tree.

Choices for f

1. For Regression Trees, we consider two possible models:
   1. Mean Shift model, with $\theta_{g(ij)} = (\mu_{g(ij)}, \sigma_j^2)$:
   $$y_i|(z_i = j, \theta_{g(ij)}) \sim N(\mu_{g(ij)}, \sigma_j^2)$$
   2. Mean-variance Shift model, with $\theta_{g(ij)} = (\mu_{g(ij)}, \sigma_{g(ij)}^2)$:
   $$y_i|(z_i = j, \theta_{g(ij)}) \sim N(\mu_{g(ij)}, \sigma_{g(ij)}^2)$$

2. For classification trees, $y_i \in \{D_1, ..., D_Q\}$, where D_i denotes the different classes. In this case:
$$f(y_i|z_i = j, \theta_{g(ij)}) = \prod_{q=1}^{Q} (p_{g(ij),q})^{I(y_i \in D_q)}$$
where $\theta_g = (p_{g1}, ..., p_{gQ})$. Here, $p_{gq} := P(y_i \in D_q|z_i = j)$.

Table of Contents

Prior Specification G_0

$$p(\Theta, T) = p(\Theta|T)p(T)$$

We first describe $p(T)$. Then, we will describe $p(\Theta|T)$. Instead of specifying a closed-form expression, we specify $p(T)$ implicitly by a tree generating stochastic process.

1. Begin by setting T to be the trivial tree with single node (both root and terminal) denoted by η.
2. split terminal node η with probability $p_{\text{split}}(\eta, T)$.
3. If split occurs, assign it a splitting rule ρ according to the distribution $p_{\text{rule}}(\rho|\eta, T)$ to create right and left children nodes.
4. Continue the above steps 2 & 3 until termination.

Typical choices for $p_{\text{split}}(\eta, T)$:

1. $p_{\text{split}}(\eta, T) \equiv \alpha$ (constant).
2. $p_{\text{split}}(\eta, T) = \alpha(1 + d_\eta)^{-\beta}$, where d_η is the depth of that root and $\beta > 0$.

For $p_{\text{rule}}(\rho|\eta, T)$, we can take the uniform prior. i.e, uniformly choosing a variable to split on and taking a split point uniformly at random for that variable.

Prior Specification G_0: $p(\Theta_j|T_j)$ for Regression Trees

The simplest specification is the conjugate prior:

$$\mu_g|\sigma_j,\ \text{iid} \sim N\left(\bar{\mu}, \frac{\sigma_j^2}{a}\right)$$

and

$$\sigma_j^2 \sim IG\left(\frac{\nu}{2}, \frac{\nu\lambda}{2}\right)$$

Some analytic simplifications show that under the above prior, we get:

$$p(Y_j|X_j, T_j, Z) = \frac{ca^{b_j/2}}{\prod_{g=1}^{b_j}(n_{jg}+a)^{1/2}} \times \left(\sum_{g=1}^{b_j}(s_{jg}+t_{jg}) + \nu\lambda\right)^{-(n+\nu)/2}$$

- $Y_j = \{y_i : z_i = j\}$ and $A_{jg} = \{i : z_i = j, g(ij) = g\}$
- $s_{jg} = \sum_{i \in A_{jg}} (y_i - \bar{Y_{jg}})^2$
- $t_{jg} = [n_{jg}a/(n_{jg}+a)](\bar{Y_{jg}} - \bar{\mu})^2$

Prior Specification G_0: $p(\Theta|T)$ for Classification Trees

In Classification tree, y_i belongs to one of Q categories $D_1, ..., D_Q$. Prior for $\Theta_j = (p_1, ..., p_{b_j})$ is the standard dirichilet distribution of dimension $Q-1$ with $\alpha = (\alpha_1, ..., \alpha_Q)$:

$$p_1, ..., p_{b_j}|T \text{ iid} \sim Dirichlet(p_i|\alpha) \propto p_{i1}^{\alpha_1 - 1} ... p_{iQ}^{\alpha_Q - 1}$$

The above prior gives the following marginal distribution:

$$P(Y_j|X_j, T_j) = \left(\frac{\Gamma\left(\sum_q \alpha_q\right)}{\prod_q \Gamma(\alpha_q)} \right)^{b_j} \prod_{g=1}^{b_j} \frac{\prod_k \Gamma(n_{gq} + \alpha_q)}{\Gamma(n_g + \sum_q \alpha_q)}$$

- $q \in \{1, ..., Q\}$
- $n_{gq} = \sum_{i \in A_{jg}} I(y_i \in D_q)$
- $n_g = \sum_q n_{gq}$

Table of Contents

Allocation Variables Z

z_i denotes the class to which the i^{th} point belongs.

$$[y_i|z_i = j, C, k, \Theta^*_M] = f(y_i|\theta_{g(ij)})$$

Hence, for Regression Trees, we have:

$$[z_i = j \mid Y, Z_{-i}, C, k, \Theta^*_M] \propto \frac{1}{\sigma_j} \exp\left\{-\frac{1}{2\sigma_j^2}\left(y_i - \mu_{g(ij)}\right)^2\right\}$$

Similarly, for classification trees, $y_i \in \{D_1, ..., D_Q\}$, where D_i denotes the different classes. In this case:

$$[z_i = j \mid Y, Z_{-i}, C, k, \Theta^*_M] \propto \prod_{q=1}^{Q} (p_{g(ij),q})^{I(y_i \in C_q)} = p_{g(ij),q'} \ , \quad \text{where } y_i = D_{q'}$$

where $\theta_g = (p_{g1}, ..., p_{gQ})$. Here, $p_{gq} := P(y_i \in D_q | z_i = j)$.

$(\Theta_j, T_j) \sim DP(\alpha, G_0)$. Hence,

$$[(\Theta_j, T_j)|(\Theta, T)_{-j}] \sim \frac{\alpha}{\alpha + m - 1} G_0 + \frac{1}{\alpha + m - 1} \sum_{l=1, l \neq j}^{m} \delta_{(\Theta_l, T_l)}$$

Where $\delta_{(\Theta_l, T_l)}$ denotes the point mass at (Θ_l, T_l).

Let k_j denote the number of distinct values in Θ_{-jM}.

$$[c_j = l \mid Y, Z, C_{-j}, k_j, \Theta^*_M] = \begin{cases} \kappa q_{lj} & \text{if } l \in \{1, \cdots, k_j\} \\ \kappa q_{0j} & \text{if } l = k_j + 1 \end{cases}$$

$$q_{lj} = M_{lj} \left(\frac{1}{2\pi\sigma_j^2} \right)^{\frac{n_j}{2}} \exp \left[\frac{1}{2\sigma_j^2} \sum_{i:z_i=j} (y_i - \mu_{g(il)})^2 \right]$$

Where M_{lj} is the number of times θ^*_l occurs in Θ_{-jM}.

Since this is a non-conjugate case, q_{0j} is not available in closed form. Hence, we use Algorithm 8 in Neal (2000)

Algorithm 8 in Neal (2000)

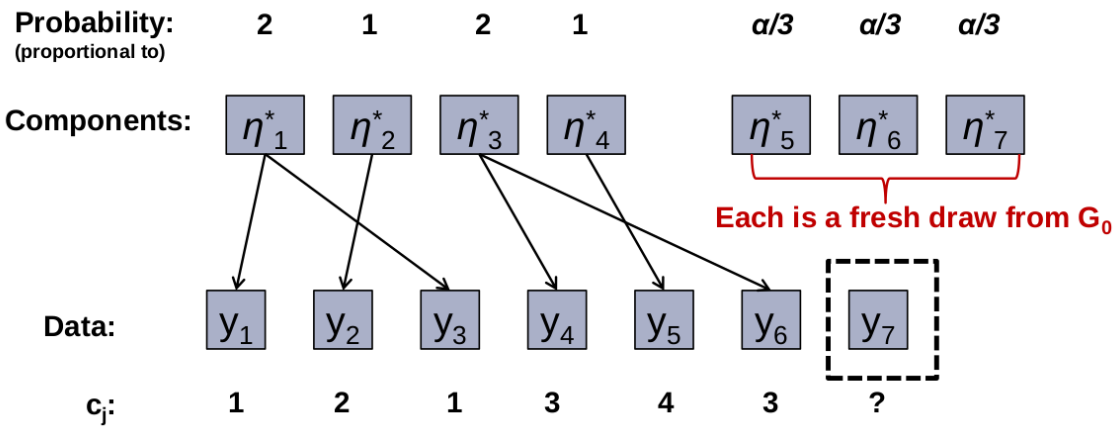
When updating c_j, we either:

- choose an existing component c from c_{-i} (i.e., all c_j such that $j \neq i$)
- choose a brand new component.

Let K_{-j} be the number of distinct components c_{-j}. Instead of integrating over G_0, we will add m auxiliary parameters, each corresponding to a new component independently drawn from G_0:

$$[\eta^*_{K_{-i}+1}, ..., \eta^*_{K_{-i}+m}] \equiv [(\Theta, T)^*_{K_{-i}+1}, ..., (\Theta, T)^*_{K_{-i}+m}]$$

Probability of selecting a new component is proportional to α. So, we divide α equally among the m auxiliary components.



Configuration Variables C and k

Let k_j denote the number of distinct values in Θ_{-jM}.

$$[c_j = l \mid Y, Z, C_{-j}, k_j, \Theta^*_M] = \begin{cases} \kappa q_{lj} & \text{if } l \in \{1, \cdots, k_j\} \\ \kappa q_{0j} & \text{if } l = k_j + 1 \end{cases}$$

$$q_{lj} = M_{lj} \left(\frac{1}{2\pi\sigma_j^2} \right)^{\frac{n_j}{2}} \exp \left[\frac{1}{2\sigma_j^2} \sum_{i:z_i=j} (y_i - \mu_{g(il)})^2 \right]$$

Where M_{lj} is the number of times θ^*_l occurs in Θ_{-jM}.

For q_{0j}, we modify our Gibbs sampling as described in Mukhopadhyay and Bhattacharya 2012 . Let $\theta^a = (\Theta^a, T^a)$ denote an auxiliary variable.

- if $c_j = c_l$ for some $l \neq j$, then $\theta^a \sim G_0$
- if $c_j \neq c_l \forall l \neq j$, set $\theta^a = \theta^*_{c_j}$

Next, replace q_{0j} with the tractable expression:

$$q^a_j = \alpha \left(\frac{1}{2\pi\sigma_\alpha^2} \right)^{\frac{n_j}{2}} \exp \left[\frac{1}{2\sigma_\alpha^2} \sum_{i:z_i=j} (y_i - \mu_{g(ia)})^2 \right]$$

once all the c_j are updated, k is obtained as the number of unique values in C.

Conditionals for $\theta_j = (\Theta_j, T_j)$

Consider the set:

$$Y_j = \{y_i : z_i = j\}$$

We simulate (T_j, Θ_j) by the posterior search metropolis - hastings algorithm. We perform this search for all the distinct components: (Y_j, T_j, Θ_j), $j \in \{1, ..., k\}$

We first simulate the tree T_j, and then simulate Θ_j given T_j

Tree Posterior Search

Dropping the subscript j for simplicity.

The transition kernel $q(T, T^*)$ generates a new tree T^* from the tree T by randomly choosing one of the four steps:

- GROW: Randomly pick a terminal node and split it into two new ones by randomly assigning it a splitting rule according to p_{RULE} used in the prior.
- PRUNE: Randomly pick a parent of two terminal nodes and turn it into a terminal node by collapsing the nodes below it.
- CHANGE: Randomly pick an internal node, and randomly reassign it a splitting rule according to p_{RULE} used in the prior.
- SWAP: Randomly pick a parent-child pair that are both internal nodes. Swap their splitting rules unless the other child has the identical rule, in which case swap the splitting rule of the parent with that of both children.

Using the above kernel, we have the following M-H algorithm to generate the markov chain $T_0, T_1, \ldots$:

1. Generate a candidate T^* from distribution $q(T_i, T^*)$
2. Define:

$$\alpha(T_i, T^*) := \min \left\{ \frac{q(T_i, T^*)}{q(T_*, T^i)} \frac{p(Y|X, T^*)P(T^*)}{p(Y|X, T^i)P(T^i)} \ , \ 1 \right\}$$

   Set $T_{i+1} := T^*$ with probability $\alpha(T_i, T^*)$ and T_i with probability $1 - \alpha(T_i, T^*)$.

Here, it is important to note that $p(Y|X, T)$ can be explicitly obtained by integrating over the parameters Θ.

Parameter Posterior

Once we have drawn T_j from its posterior, we next simulate Θ_j by gibbs sampling:

$$p(Y_j, \Theta_j | Z, T) \propto \prod_{g=1}^{b_j} \left(\prod_{i:g(ij)=g} \frac{1}{\sigma} \exp\left[-\frac{1}{2\sigma_j^2} (y_i - \mu_g)^2 \right] \right)$$
$$\times \prod_{g=1}^{bj} \frac{\sqrt{a}}{\sigma_j} \exp\left[\frac{a}{2\sigma_j^2} (\mu_g - \bar{\mu})^2 \right] \times \frac{\beta^\alpha}{(\sigma_j^2)^{\alpha+1}} \exp\left(-\frac{\beta}{\sigma_j^2} \right)$$

Hence, given σ_j^2, posterior for μ_g is $N(\mu_1, \sigma_1^2)$ where:

$$\sigma_1^2 = \left(\frac{a}{\sigma_j^2} + \frac{n_g}{\sigma_j^2} \right)^{-1} = \frac{\sigma_j^2}{a + n_g} \quad \text{and} \quad \mu_1 = \sigma_1^2 \left(\frac{a\bar{\mu}}{\sigma_j^2} + \frac{n_g \bar{Y_{gj}}}{\sigma_j^2} \right)$$

Again, given μ_g for all the terminal nodes of T_j, we have the posterior of σ_j^2 as:

$$\sigma_j^2 \mid (\mu_{j1}, ..., \mu_{jb_j}),\ T_j \sim IG\left(\frac{\nu}{2} + \frac{n_j}{2}, \frac{\nu\lambda}{2} + \frac{1}{2} \sum_{g=1}^{b_j} \sum_{i \in A_{jg}} (y_i - \mu_{jg})^2 \right)$$

where $A_{jg} = \{i : z_i = j, g(ij) = g\}$, i.e, the set of points that lie in the g^{th} terminal node of the j^{th} tree.

Summary Posterior Conditionals

We have either obtained the posterior conditional or a strategy to simulate from the posterior of the following quantities:

- $[z_i = j \mid Y, Z_{-i}, C, k, \Theta^*_M]$
- $[c_j = l \mid Y, Z, C_{-j}, k_j, \Theta^*_M]$
- $[k \mid Y, Z, C, \Theta^*_M]$
- $[T_j \mid Y, Z, C, k]$
- $[\Theta_j \mid Y, Z, C, k, T_j]$

Hence, we can perform Gibbs sampling to traverse the posterior space.

Table of Contents

Data Generation

- $0 \leq X_1, X_2 \leq 4$
- n = 1000

$$Y_i \sim \begin{cases} N(\mu_{g(i1)}, \sigma^2) & \text{w.p } 0.5 \\ N(\mu_{g(i2)}, \sigma^2) & \text{w.p } 0.5 \end{cases}$$

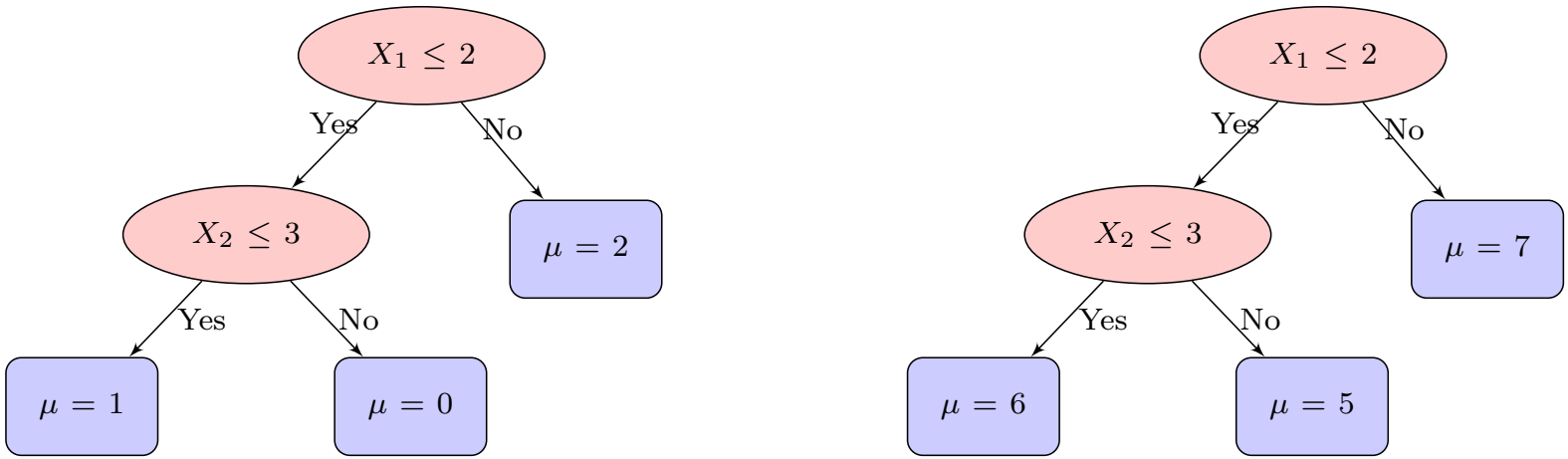


Figure: Data Generating process for Simulation

Gibbs Sampling

- One step in the Gibbs Sampler is the MH algorithm to update the trees in our mixture.
- As specified in Chipman(1998), any MH algorithm for CART models will have difficulty moving between local modes.
- Hence, we have repeatedly restarted the algorithm and let each of them converge to a local mode.
- Finally, we have taken the Forest that provides the highest posterior likelihood.

Non-informative Prior

- $P_{SPLIT} = e^{-\frac{(d_\eta+1)}{1000}}$
- Acceptance probability:
  $\alpha(T_i, T^*) = \min\left\{\frac{q(T_i,T^*)}{q(T_*,T^i)}\frac{p(Y|X,T^*)P(T^*)}{p(Y|X,T^i)P(T^i)}\ ,\ 1\right\} \approx \min\left\{\frac{p(Y|X,T^*)}{p(Y|X,T^i)}\ ,\ 1\right\}$

  $\because \frac{q(T_i,T^*)}{q(T_*,T^i)}\frac{P(T^*)}{P(T^i)} \approx 1$

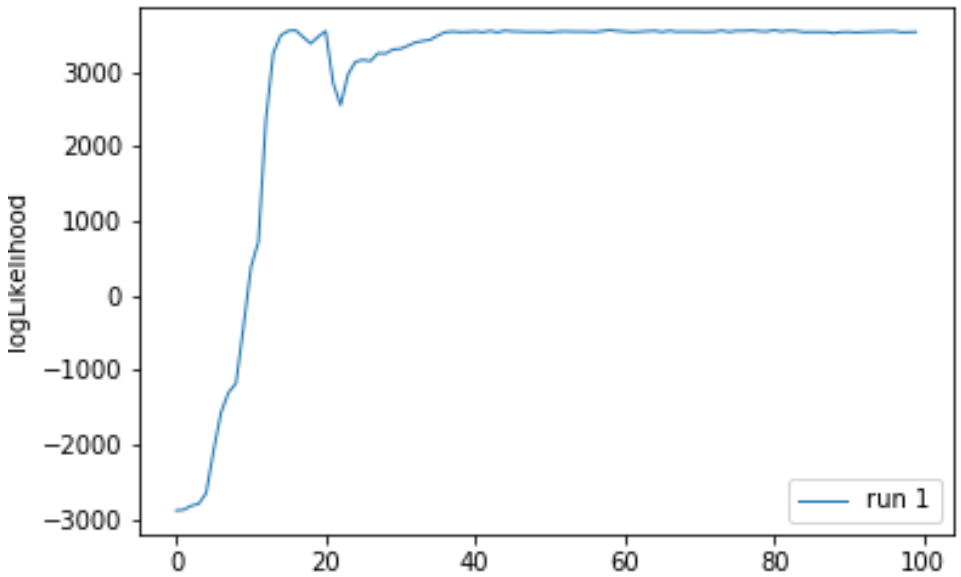


Figure: Log Likehood of the best run

Non-informative Prior (Results)

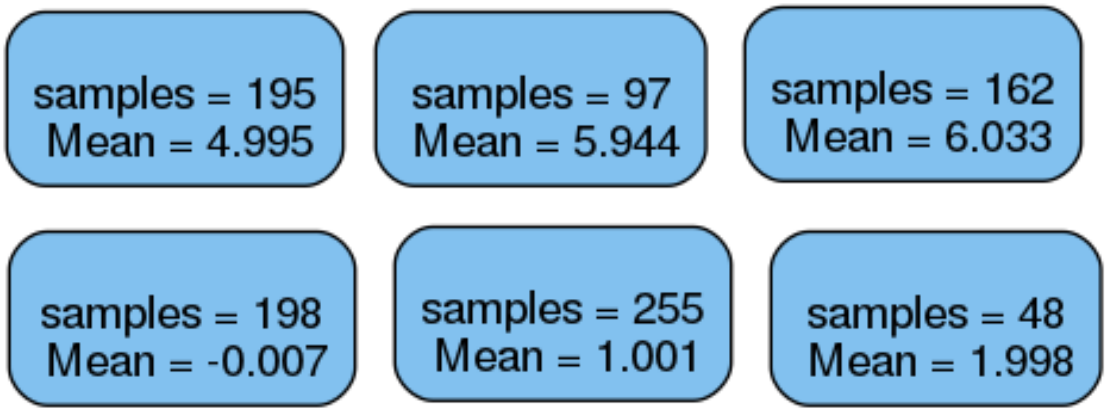


Figure: All leaf nodes taken as separate components in non informative prior

- We did not take any informative prior on the trees, and allowed for many different components in our mixture model.
- Taking all the leaves as separate components of the mixture will yield high posterior likelihood.

To avoid this, we will either need to specify a prior that favours splitting of the root node, or we can specify the number of components.

Informative Prior

We now consider the following prior for our trees:

$$p_{\text{split}}(\eta, T) = \alpha(1 + d_\eta)^{-\beta}$$

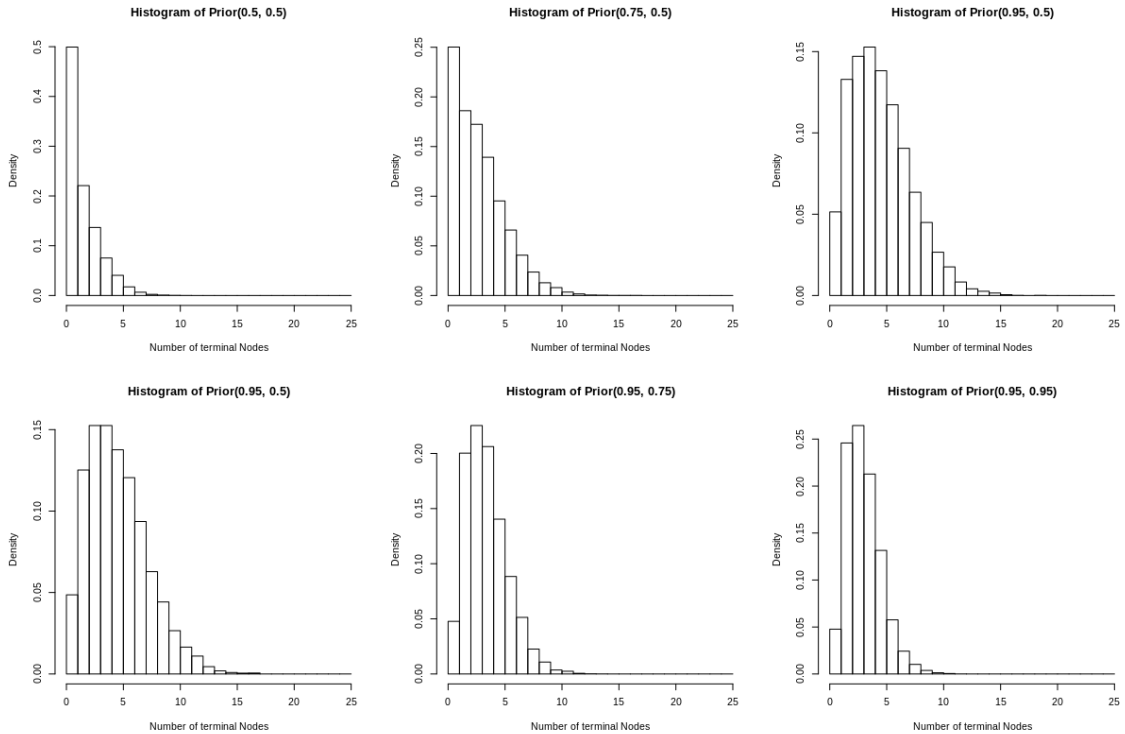


Figure: Prior Distribution on number of leaf nodes for different choices of (α, β)

- We take $(\alpha, \beta) = (0.95, 1)$

Informative Prior(Results)

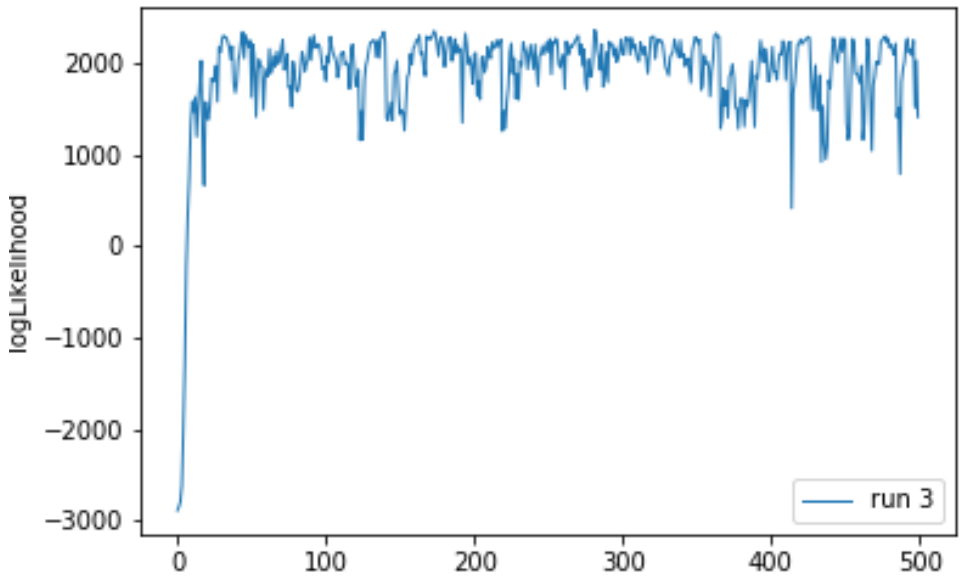


Figure: Log Likehood of the best run

We kept a maximum of 10 trees. Finally, 6 trees obtained with the following frequencies:

Tree	1	2	3	4	5	6	7	8	9	10
Sample Points Assigned	30	9	433	431	0	52	0	45	0	0

Informative Prior (Results)

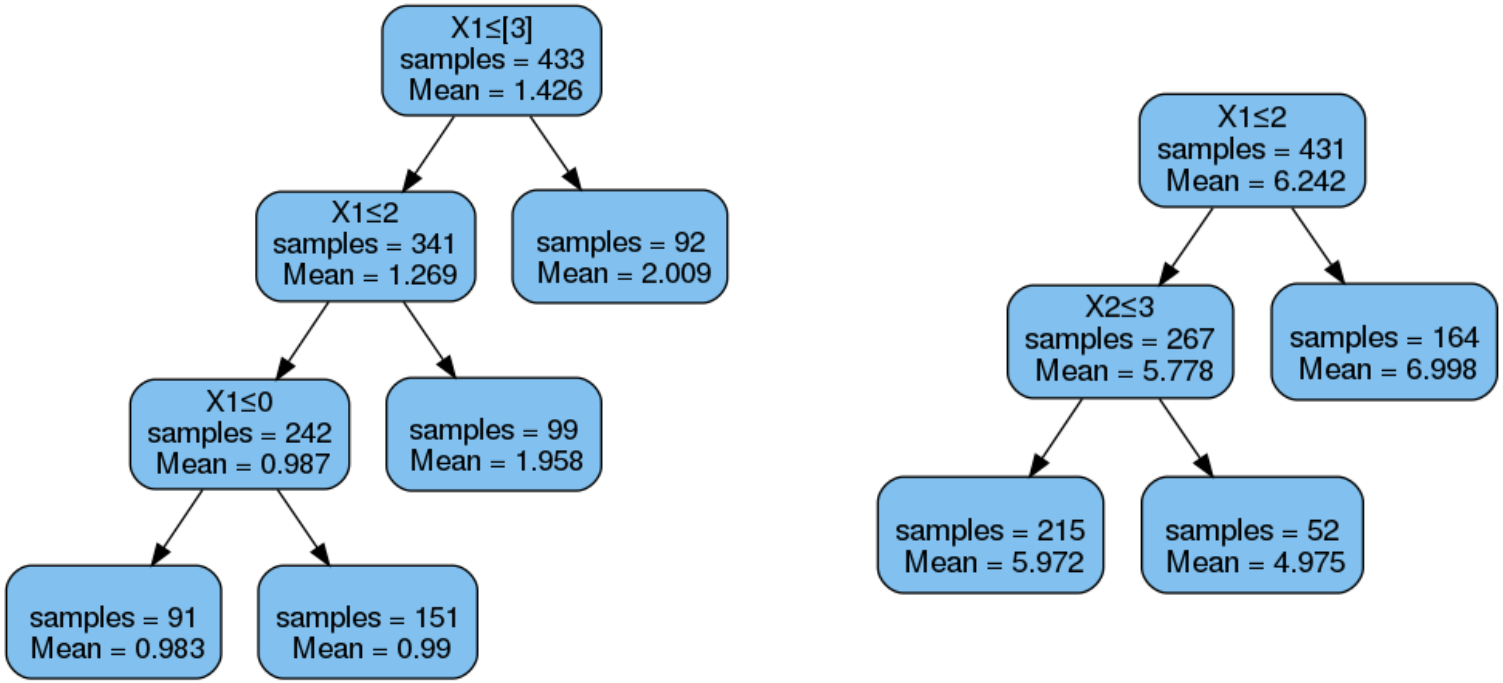


Figure: Estimates of Tree 3 and Tree 4 for informative prior

- One of the trees is accurately estimated.
- The other tree was not accurately estimated. The estimated tree is somewhat close to the true model.

Fixing the number of Components

Finally, we check how our algorithm performs when we already know the number of components from beforehand. We perform our posterior search by fixing $M = 2$.

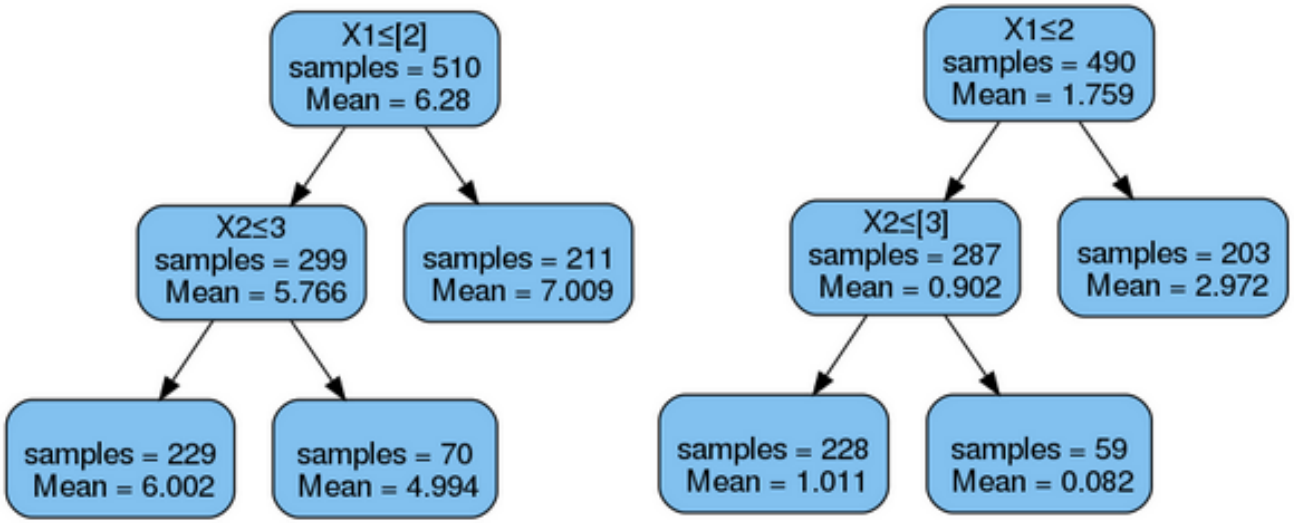


Figure: Estimated Trees for $M = 2$

We can see that both the trees are accurately estimated.

Conclusion

1. **Non-informative prior with $M \geq g$:**
   - The posterior search reaches the mode in very few steps.
   - It estimates all the leaves as separate components of the mixture.
   - Hence, taking an informative prior is important is we want to learn the random number of components.

2. **With Suitable Prior:**
   - The algorithm yields good results.
   - Though the predicted number of components was not accurate, if we look into the number of data points assigned to each of the estimated components, we can see that there are two major components and the remaining ones have very few data.
   - The major components closely resemble the oracle data generating process.

3. **Fixing The Number of Components:**
   - The algorithm performed extremely well when we fixed the number of components to the actual number of clusters.

Future Prospects

1. Bayesian versions motivated by various decision tree approaches, such as Bayesian CART, Bayesian Additive Regression Trees (BART), etc. have established themselves as effective weapons for attacking challenging machine learning problems. We are yet to show that this model includes all the aforementioned tree based approaches as simple special cases.
2. *Feature importance and variable selection*: In single decision trees, feature importance is calculated by the overall impurity reduction provided by that variable across all the nodes of the tree. Such an idea can be used, to evaluate feature importance of each variable in all the trees. Finally, we can denote the feature importance of a variable $F(X_j)$ as:

$$F(X_j) = \sum_{i=1}^{M} p_i F_i(X_j)$$

   where $F_i(X_j)$ denote the feature importance of X_j in the i^{th} tree. A variable selection procedure can be developed from this feature importance.
3. *Choosing Prior Parameters*: As seen in the simulation, choice of prior for the trees heavily influences the results. We need to explore the outcomes from various choices of α and β (for $P_{SPLIT} = \alpha(1 + d_\eta)^{-\beta}$) and suggest suitable choices for these parameters depending on the scenario.

Thank You