Top Down Particle Filtering For Bayesian Decision Trees

by Balaji Lakshminarayanan
Daniel M. Roy and Yee Whye Teh

Presented by Tom Rainforth

Decision Tree Overview

- Hierarchical partition of the input space into blocks represented by ‘leaves’.
- Typically the partitions are binary and axis aligned (including the case of this paper).
- At each leaf a simple model predicts the labels.
- Can be classification or regression - talk will focus on classification.
- Typically used in scenarios that require model to be easy to use for out of sample prediction, e.g. use by people without any statistical background.
- De-correlation between leaves allows them to act as ensemble methods rather than Bayesian model averaging.

Decision Tree Overview (2)

- Model at the leaves is usually very simple, typically just a deterministic class assignment or multinomial over classes.
- Many methods also use combinations of trees, either creating a ‘forest’ of trees, e.g. Random forests, or by combining small trees into larger trees, e.g. boosted decision trees, BART
- Methods that combine trees to a single tree are naturally ensemble methods due to lack of correlation between leaves.
- Random forests typically use bagging of both data points and of variables so that the individual trees are also uncorrelated. Therefore individual trees are weak classifiers.

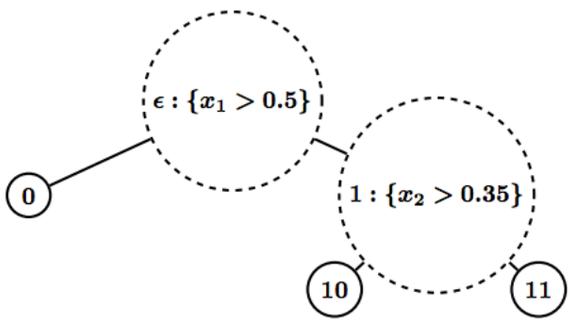
Decision Tree Overview (3)

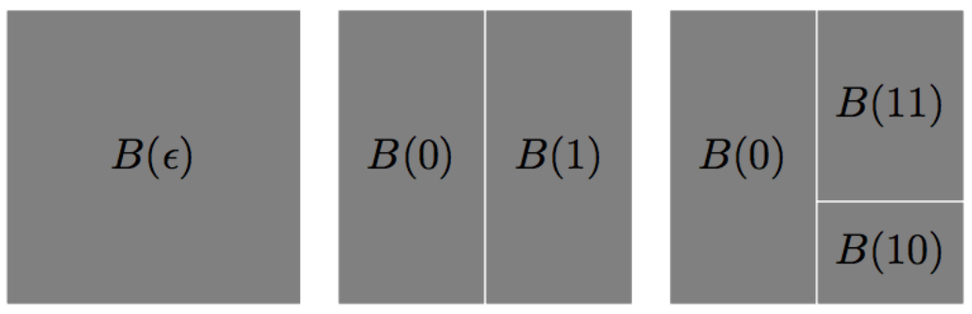
- Classically learnt in greedy top down fashion (ID3, C4.5 and CART).
- Bayesian methods prior to this paper exist which allow a prior to be placed on the trees.
    - Other advantages such as representing residual uncertainty in the posterior
- However, these work in an MH fashion and so cannot exploit hierarchical nature in structure - become slow to train.
- By using an SMC method where proposals split leaves, we can develop a Bayesian model working in a top-down fashion.
- This will produce a posterior over trees which we can either use as a forest or by extracting one or more of the best trees.

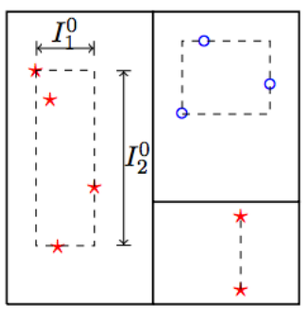
SMC Model

- Particularly in probabilistic programming setting, SMC methods usually work by sequentially “observing” subset of the data points in batches.
- However, they can also work by sequentially building the model.
- For this case, we are increasing the complexity of the tree at each stage and then observing the probability of the full dataset.
- This is a natural way to learn in a decision tree setting
  - The space occupied by the prior is combinatorial and has little structure
  - Latter partitions are contained within earlier partitions and the model is independent for data in different partitions.
- There are complications in the implementation of this with many probabilistic programs - talk about this later

Notation







The Algorithm

- Start with empty tree
- At each stage and for each particle draw from a proposal. Note this proposal can depend on the data including the class labels.
  - Deterministically chooses a node to consider from leaves not yet considered. If no candidates remain, the proposal tree is the current tree
    - Alternative scheme considers splitting all possible leaves at each stage
  - With some probability split that leaf, else mark that leaf as ‘finished’
  - If the leaf is to be split sample a dimension for split from non-trivial variables uniformly
  - Depending on variant, the split point is chosen randomly or calculate and sample from the conditional posterior if the next stage were the last stage.
- Weights are calculated as shown to left (likelihood collapsed Dirichlet multinomial) and normalised as required
- If there are no particles with any candidate leaves remaining, terminate.

Algorithm 1 SMC for Bayesian decision tree learning

Inputs: Training data (X, Y)
Number of particles M
Initialize: $\mathsf{T}_0^{(m)} = E_0^{(m)} = \{\epsilon\}$
$\tau_0^{(m)} = \kappa_0^{(m)} = \emptyset$
$w_0^{(m)} = f(Y|\mathcal{T}_0^{(m)})$
$W_0 = \sum_m w_0^{(m)}$

for $i = 1 : \text{MAX-STAGES}$ **do**
  for $m = 1 : M$ **do**
    Sample $\mathcal{T}_i^{(m)}$ from $\mathbb{Q}_i(\cdot \mid \mathcal{T}_{i-1}^{(m)})$
      where $\mathcal{T}_i^{(m)} := (\mathsf{T}_i^{(m)}, \kappa_i^{(m)}, \tau_i^{(m)}, E_i^{(m)})$
    Update weights: (Here $\mathbb{P}, \mathbb{Q}_i$ denote their densities.)

$$w_i^{(m)} = \frac{\mathbb{P}(\mathcal{T}_i^{(m)})\, g(Y \mid \mathcal{T}_i^{(m)}, X)}{\mathbb{Q}_i(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})\, \mathbb{P}(\mathcal{T}_{i-1}^{(m)})} \tag{8}$$

$$= w_{i-1}^{(m)} \frac{\mathbb{P}(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})}{\mathbb{Q}_i(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})} \frac{g(Y \mid \mathcal{T}_i^{(m)}, X)}{g(Y \mid \mathcal{T}_{i-1}^{(m)}, X)} \tag{9}$$

  end for
  Compute normalization: $W_i = \sum_m w_i^{(m)}$
  Normalize weights: $(\forall m)\, \bar{w}_i^{(m)} = w_i^{(m)} / W_i$
  if $\left(\sum_m (\bar{w}_i^{(m)})^2\right)^{-1} < \text{ESS-THRESHOLD}$ **then**
    $(\forall m)$ Resample indices j_m from $\sum_{m'} \bar{w}_i^{(m')} \delta_{m'}$
    $(\forall m)\ \mathcal{T}_i^{(m)} \leftarrow \mathcal{T}_i^{(j_m)}$; $w_i^{(m)} \leftarrow W_i / M$
  end if
  if $(\forall m)\, E_i^{(m)} = \emptyset$ **then**
    exit for loop
  end if
end for
return Estimated marginal probability W_i / M and
  weighted samples $\{w_i^{(m)}, \mathsf{T}_i^{(m)}, \kappa_i^{(m)}, \tau_i^{(m)}\}_{m=1}^M$.

Subtleties of the Proposal

- As all leaves will be considered for splitting exactly once before the tree is complete, it is valid to use a deterministic method for choosing which leaf to split at each stage
- In the presence of many irrelevant features, then uniformly sampling from all variables may be inefficient
  - This is much more predominant if we want splits that are non-axis aligned
- As both the prior and proposal probabilities must be explicitly calculated, bagging techniques used elsewhere in the decision tree algorithms can be inappropriate due to the potential to have combinatorial ways to subsample if data subsets are not unique.
- The fact that we cannot use bagging means that even if we use the method to generate forests, the forests will be strong classifiers but correlated unlike random forests. There are also correlations introduced by the resampling.
  - Even though we can use trees in a forest fashion, this will only be Bayesian model averaging and not a true ensemble method.

Complications For PP

- The overall likelihood is no longer the product of the likelihoods from individual “observations”
  - By having an extra primitive we could capture the idea of observations that do not factor to form the likelihood
- We may want very aggressive proposal distributions without having to explicitly define them in the program
  - Adaptive proposals
  - Prior and proposal probabilities no longer cancel

My Work

- Decision trees are typically used with the aim of making ‘interpretable’ classifiers.
- For large, complex data then classical binary, axis aligned decision trees need to be very large to offer competitive performance.
- Therefore the aim of interpretability is somewhat lost.
- By making the trees non-binary and non-axis aligned, then comparable performance requires a much smaller tree
  - In particular the tree will be less deep
  - Broad trees are more interpretable as any particular data point only passes through a small number of data points and nearby points in the variable space are nearby in the tree space.
- In classical trees it is often also difficult to establish a ‘key’ decision that leads to a classification
  - SMC framework should naturally lead to the most important splits being towards the top of the tree

My Work - Complications

- Non axis alignment is equivalent to using a linear metric at each stage.
  - This means that the dimension of the split direction is no longer one and so sampling from this efficiently becomes more important.
  - Even when variables have been sampled for the split, the tree will still not perform well unless the weights are appropriate.
    - We therefore use an aggressive proposal to guide the sampling towards weights that will give good performance - canonical correlation analysis on sample of variables.
- We are no longer sampling a single split point
  - Again higher dimension - optimal proposal kernel would be inappropriate
  - Currently just sampling this from the prior which is a Dirichlet.
- Need more carefully defined prior to deliver trees that are interpretable, rather than just to avoid over-fitting.