

Multiple Loss Functions in TMVA

By Andrew Carnes





Last Time

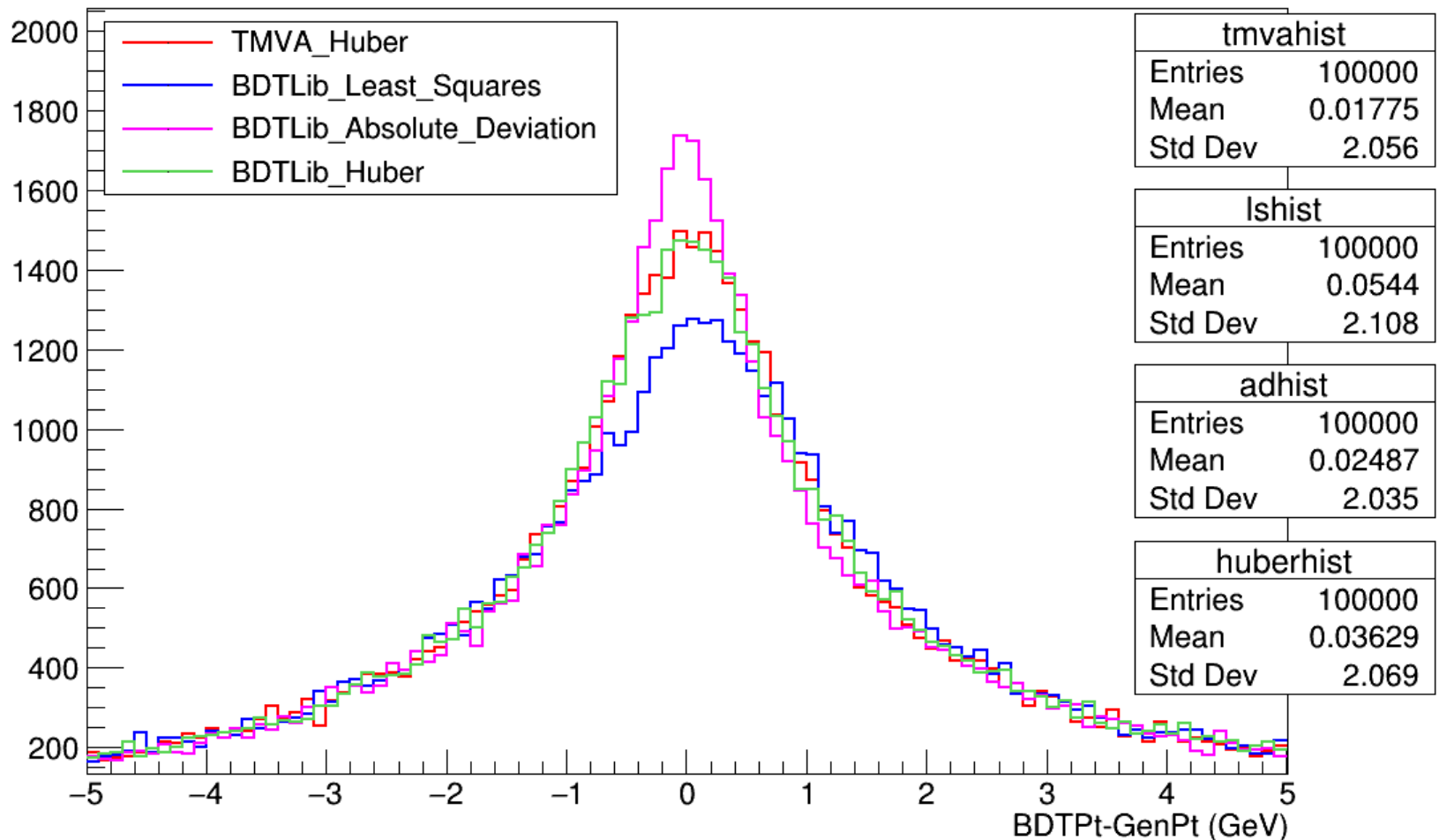
- <https://indico.cern.ch/event/495175/>
- Presented a talk on a Boosted Decision Tree (BDT) package I wrote, BDTLib <https://github.com/acarnes/bdt>
 - Implements the Gradient Boosted Decision Tree machine learning algorithm with multiple loss functions
 - The different choices in loss functions enable the user to focus on the data that matters to them
 - A nice feature currently unavailable in TMVA
- There was a consensus to integrate the capability for multiple loss functions into TMVA

This Time

- **Message of this talk:** Beginning to implement this feature into TMVA
 - **Current status** → **Benchmarking**
- Present benchmarks of the different loss functions on some available data
 - Look at the difference in predictions for some different loss functions on these data sets
- Present future plans

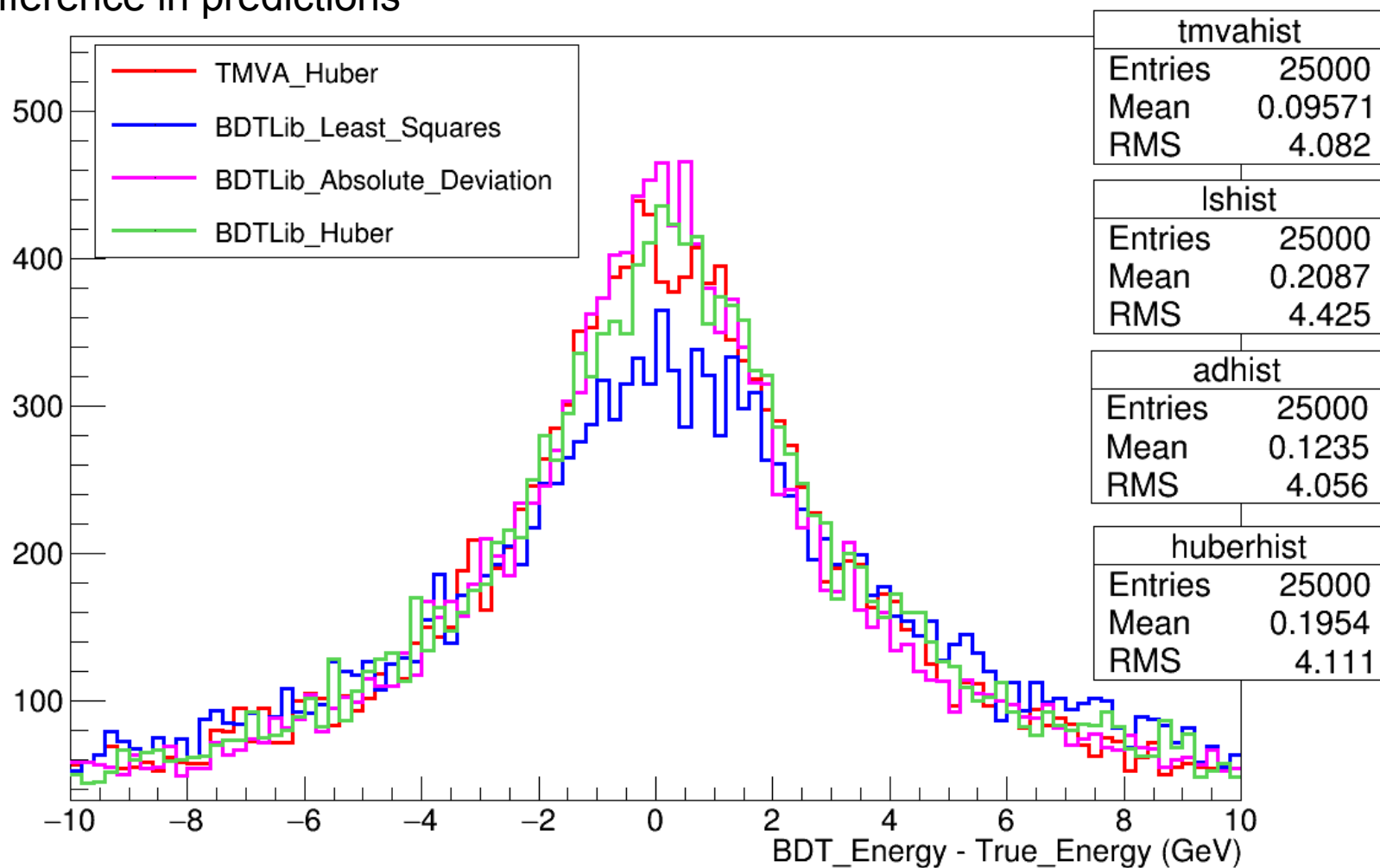
Benchmarking on CSC Pt Assignment

- Comparing some loss functions from BDTLib to TMVA's Huber Loss function in the context of regression
- Here we look at momentum assignment in the L1 Trigger's Cathode Strip Chambers using Monte Carlo
- We see an obvious difference in the predictions given different loss functions as expected



Benchmarking on Toy Calorimeter

- Again comparing some loss functions from BDTLib to TMVA's Huber Loss function in the context of regression
- Here we look at predicting the energy deposition in a toy calorimeter
 - Uses the sample from the page below
 - <https://www.hep1.physik.uni-bonn.de/people/homepages/tmva/tmvatutorial>
- Again we see a difference in predictions





Future Plans

- Benchmarks now available for comparison with future improvements
- Will start implementing the multiple loss functions into TMVA
- There are also plans to parallelize the BDTs in TMVA
 - Can search for the best cuts along each feature in parallel
 - Can reduce the BDT training time by a factor of the number of features
 - Can also parallelize the evaluation since the contribution from each tree doesn't depend on any of the others

Backup Slides

- BDT Algorithm Overview
- References



Brief BDT Algorithm Overview

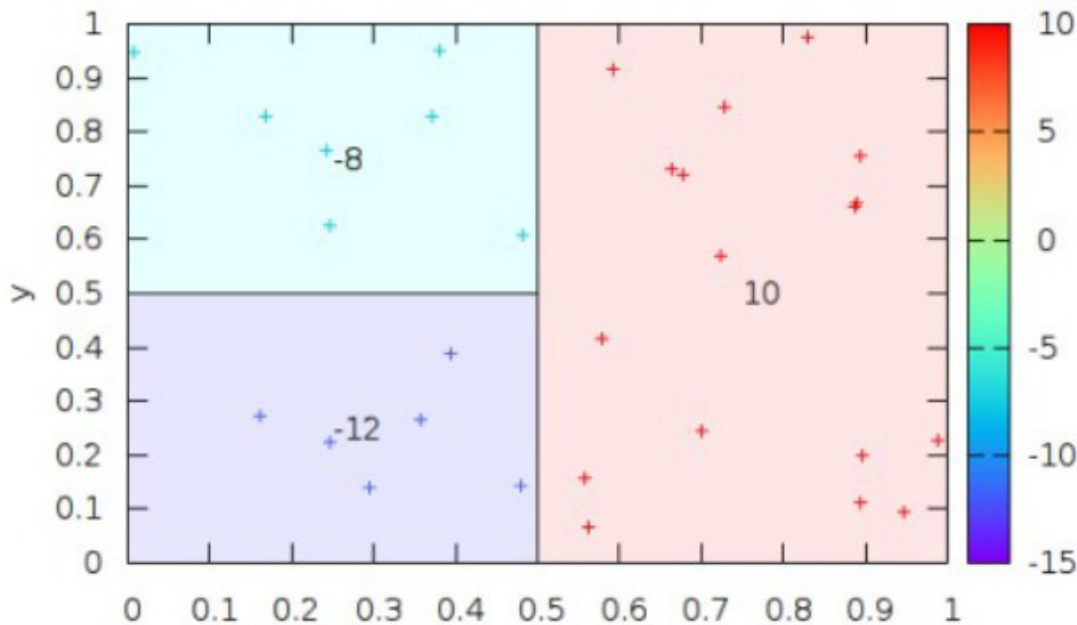


Fig 1. A decision tree with 3 terminal nodes

A Single Decision Tree

- Breaks up feature space into discrete regions using hyperplanes
- Fits a constant to each region
- The regions are greedily chosen to minimize a given Loss Function (a differentiable measure of the error)
- May be viewed as a series of decisions (shown below)

Boosting

- Make one tree, add another tree that corrects the predictions of the first
- Add another tree that corrects the net prediction of the first and second
- Continue the process
- End up with a collection of trees (Forest) and a net prediction
- $F(x) = T_0(x) + T_1(x) + T_2(x) + \dots + T_N(x)$

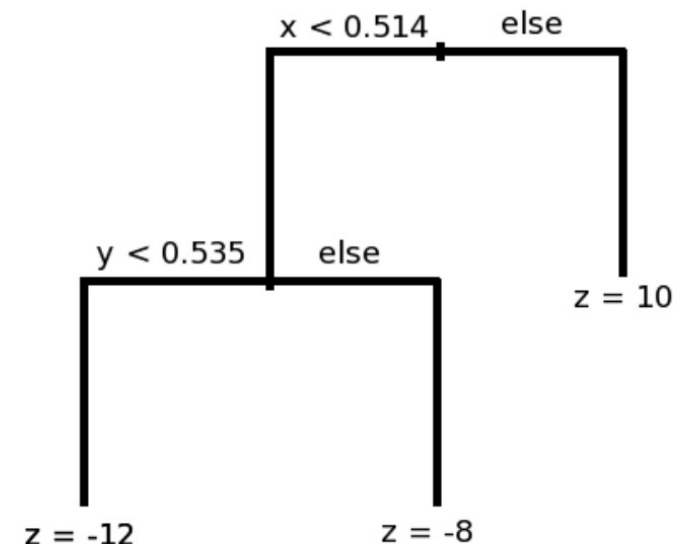


Fig 2. The same tree represented as a series of decisions

References

- Friedman, Jerome H. "Greedy function approximation: a gradient boosting machine." *Annals of statistics* (2001): 1189-1232.

