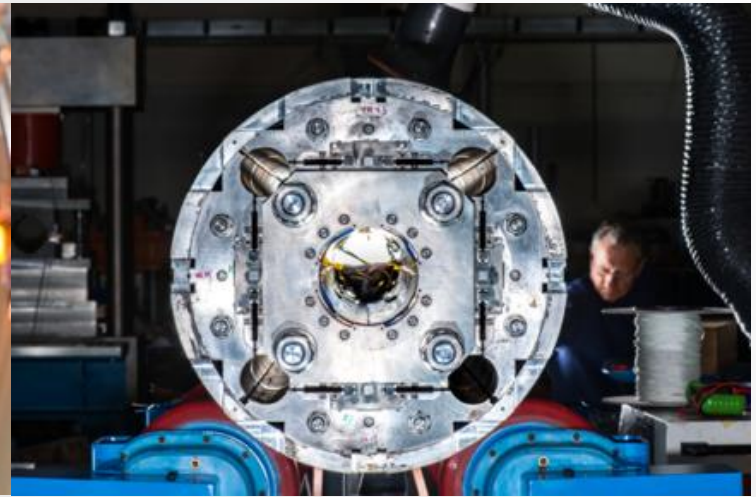
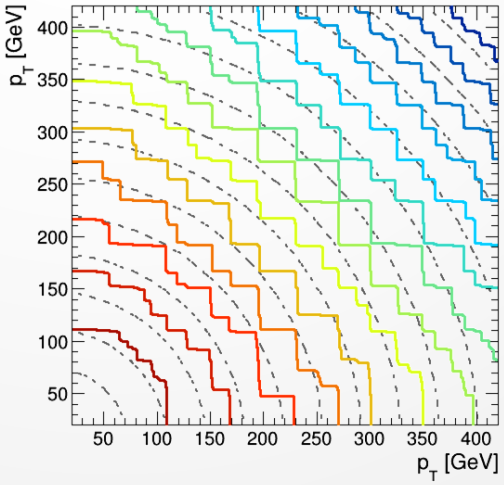
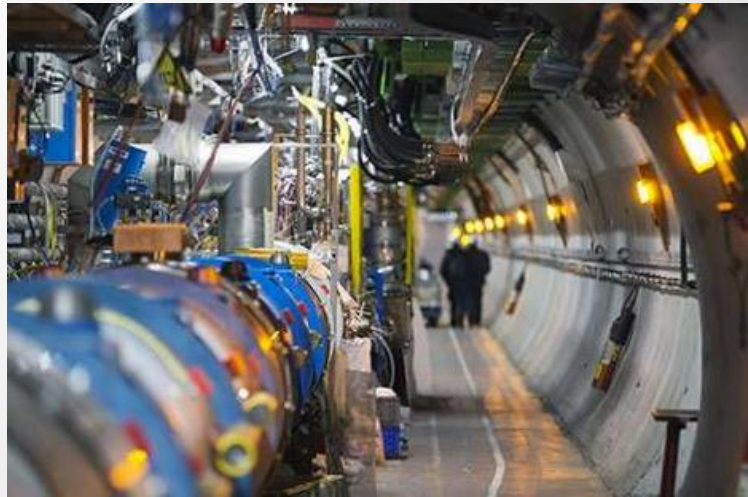
LEARNING EFT CLASSIFICATION FROM TREE BOOSTING

Boosted Information Trees

L. Lechner, S. Chatterjee, D. Schwarz, R. Schöfbeck (HEPHY), N. Frohner (TU Vienna)







EFT-ML SETTING

- Certainly no need to introduce (SM-) EFT to this audience!
- Simple dependence of cross sections on Wilson coefficients *differentially*

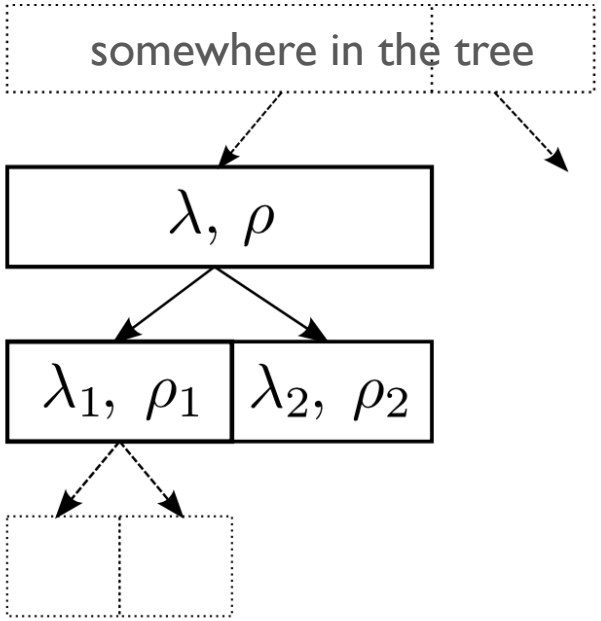
$$\begin{aligned}\Delta\sigma(z|\theta_i) &\propto |\mathcal{M}_{\rm SM} + \theta_i \mathcal{M}^i_{\rm BSM}|^2 \\ &= |\mathcal{M}_{\rm SM}|^2 + \theta_i {\rm Re}\mathcal{M}^*_{\rm SM}\mathcal{M}^i_{\rm BSM} + \theta_i\theta_j \mathcal{M}^{i*}_{\rm BSM}\mathcal{M}^j_{\rm BSM} \\ &= \Delta\sigma_{\rm SM}(z) + \theta_i \Delta\sigma^i_{\rm int.}(z) + \theta_i\theta_j \Delta\sigma^{ij}_{\rm BSM}(z).\end{aligned}$$

  - If not polynomial, then good reasons to truncate; (supported e.g. in [SMEFTsim])
- Simulated EFT events [Madgraph/MadWeight] come with extra information ('augmented' data)
  - Can compute (event-wise) "joint" LL ratio $r(x,z|\theta_2, \theta_1)$ and score vector $t(x,z|\theta)$
- pioneering series of work by K. Cranmer et al. [1506.02169, 1805.00013, 1805.00020, 1805.12244]
  - Can regress to the true (!) likelihood using the joint quantities, provided the loss is suitably simple
  - Exploited at various levels by NN classifiers

$$L[\hat{t}(x|\theta)] = \frac{1}{N} \sum_{(x_e,z_e)\sim p(x,z|\theta)} |t(x_e, z_{{\rm all},e}|\theta) - \hat{t}(x_e|\theta)|^2$$

- What can be done with BDTs? Focus: *Classification*, not regression.

FISHER INFORMATION IN TRADITIONAL CLASSIFICATION



- Fancy way to write the weak learner's majority output:

$$F_j = \Theta\left(\tfrac{1}{N}\textstyle\sum_{i\in j} y_i - \tfrac{1}{2}\right)$$

$$F(\mathbf{x}) = \sum_{j\in\mathcal{J}} \mathbb{1}_{\alpha_j}(\mathbf{x}) F_j$$

- Let's look at a weak learner in classification (CART) aiming to separate binary hypothesis (supervised training).
  Data: $\{y_i, \mathbf{x}_i\}_{i=1}^{N}$ with $y_i \in \{0,1\}$
- Find the next cut in feature *j* at position $(j, \alpha_j) = \arg\min_{j,\alpha_j}(\Delta H)$

  where the loss is $\Delta_H = \lambda_1 H(\rho_1) + \lambda_2 H(\rho_2) - \lambda H(\rho)$
  Purity ρ = S/(S+B), total yield **λ** = S+B

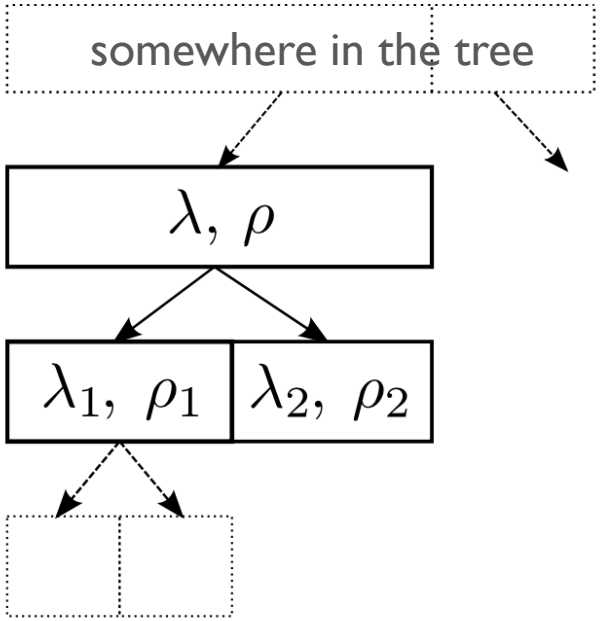
1. Let's chose the Gini impurity $H_G = \rho(1-\rho)$

   and we find $\Delta_{H_G} = -\left(\frac{1}{\lambda_1} + \frac{1}{\lambda_2}\right)^{-1} (\rho_1 - \rho_2)^2$

   Penalty for small yields — Maximally different purities

2. The weak learner returns the majority label of the training data

FISHER INFORMATION IN TRADITIONAL CLASSIFICATION



- Fancy way to write the weak learner's majority output:

$$F_j = \Theta\left(\frac{1}{N}\sum_{i\in j} y_i - \frac{1}{2}\right)$$

$$F(\mathbf{x}) = \sum_{j\in\mathcal{J}} \mathbb{1}_{\alpha_j}(\mathbf{x}) F_j$$

- 1 step back: Take pdf p(x|**θ**) and score $t(x|\theta) = \nabla_\theta \log p(x|\theta)$
- The Fisher information is defined as

$$I_{ij}(\theta) = \mathbb{E}_\theta\left[\frac{\partial}{\partial\theta_i}\log p(x|\theta)\frac{\partial}{\partial\theta_j}\log p(x|\theta)\right]$$

- The Cramér-Rao bound states that, for an (unbiased) estimator of θ, we have $\mathrm{var}(\hat{\theta}) \geq \frac{1}{I(\theta)}$
- Now let's do a Poisson counting experiment: $I_{ij}(\theta) = \frac{1}{\lambda}\frac{\partial\lambda}{\partial\theta_i}\frac{\partial\lambda}{\partial\theta_j}$ and measure a cross section of a signal process

$$\lambda(\Delta\sigma_s) = b + s(\Delta\sigma_s) = b + \varepsilon\Delta\sigma_s\mathcal{L} = \frac{\varepsilon}{\rho}\Delta\sigma_s\mathcal{L}$$

- The Fisher information in the x-sec from a Poisson count is

$$I(\Delta\sigma_s) = \frac{1}{\lambda}\left(\frac{\partial\lambda}{\partial\Delta\sigma_s}\right)^2 = \frac{\varepsilon\rho\mathcal{L}}{\Delta\sigma_s} = \frac{1}{(\Delta\sigma_s)^2}\lambda\rho^2$$

↔ equivalent to H_G!
Constants and ∝ ρ drop out.

- Gini impurity is the Fisher optimum of xsec measurement!
[A. Valassi, 2019,, 2020]

INGREDIENT 1: THE WEAK LEARNER

- Use Fisher Information in Poisson EFT yield $\boldsymbol{\lambda}(\boldsymbol{\theta})$ to define node split in the CART[1] algorithm
- Learn to separate EFT hypothesis (instead of labels y=0,1):
- We're training for classification of $\boldsymbol{\theta}_{ref}$ from $\boldsymbol{\theta}_{ref}$ + $d\boldsymbol{\theta}$ at a reference point $\boldsymbol{\theta}_{ref}$. Single $\boldsymbol{\theta}$ direction.
- The impurity H_G is replaced by the $\boldsymbol{\theta}$-dependent general function: $H_F = \left(\frac{1}{\lambda}\frac{\partial\lambda}{\partial\theta}\right)^2$
- The loss for the node split becomes

$$\Delta_{H_F} = \underbrace{-I_1(\theta) - I_2(\theta) + I(\theta)}_{\text{Here are the Poisson } I_F \text{ for general } \boldsymbol{\lambda}(\boldsymbol{\theta})} = -\left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}\right)^{-1}\left(\frac{\partial}{\partial\theta}\log\lambda_1 - \frac{\partial}{\partial\theta}\log\lambda_2\right)^2$$

  - ρ is replaced by the Poisson score function ⟷ $\partial_\theta \log\lambda(\theta) = \lambda'(\theta)/\lambda(\theta)$
  - Same penalty term as for the Gini impurity
  - There are *no backgrounds*! Only events with w'=o that reduce the Poisson score ⟷ the classifier can learn that.
- ΔH_F defines only the node split. But what shall the weak learner *learn*?

[1]Classification And Regression Trees

INGREDIENT 2: BOOSTING (OR WHAT THE LEARNER LEARNS)

- Traditionally loss functions have the form L($-yF(x)$), i.e. sum products of prediction x truth.
- We classify $\boldsymbol{\theta}$ and $\boldsymbol{\theta}$ + d $\boldsymbol{\theta}$ → each augmented event enters with +w($\boldsymbol{\theta}$) and –w($\boldsymbol{\theta}$+d$\boldsymbol{\theta}$)
- Hence, the general form of the loss is the weight differentiated x (whatever we decide to predict)

$$F(\mathbf{x}) = -\sum_{j\in\mathcal{J}} \mathbb{1}_{\alpha_j}(\mathbf{x}) \frac{\partial w}{\partial \theta} F_j$$

F_j … What the weak learner shall learn (our choice)
$\partial w(\boldsymbol{\theta})/\partial \boldsymbol{\theta}$ … Because each event enters *twice* in the classification loss

- Boosting: a linear sum of greedy weak learners is fit iteratively to the residuals of the preceding iteration.
- It's important to have the boosting loss consistent with the I_F criterion (otherwise will just learn *this* loss)
- Choosing $F_j = \frac{\partial}{\partial\theta} \log \sum_{i\in j} w_i(\theta) = \frac{1}{\lambda_j(\theta)} \frac{\partial \lambda_j(\theta)}{\partial \theta}$ leads to $\mathbb{E}(F) = -\sum_{j\in\mathcal{J}} \frac{1}{\lambda_j(\theta)} \left(\frac{\partial \lambda_j(\theta)}{\partial \theta}\right)^2 = -\sum I_F$
- The weak learner shall return (predict) the Poisson score from all events in the node!
- Boosting Algorithm: 1) Fit a weak learner using I_F node-split; 2) return the score. 3) Take only a fraction η·F (η = learning rate), 4) remove η·F from the events' score, and 5) repeat!

BOOSTED INFORMATION TREES

- Result is a CART algorithm with dedicated node-split criterion and consistent boosting loss.
- Pseudo-code of weak learner in backup
- Dressed up with overtraining protection
- Nightly python2/3 code on GitHub

```
bit = BoostedInformationTree(
        training_features = training_features,
        training_weights      = training_weights,
        training_diff_weights = training_diff_weights,
        learning_rate = learning_rate,
        n_trees = n_trees,
        max_depth=max_depth,
        min_size=min_size,
        split_method='vectorized_split_and_weight_sums',
        weights_update_method='vectorized')

bit.boost()
```

Algorithm 2 Boosted Information Tree

Data: Dataset $\mathcal{D} = \{\mathbf{x}_i, \omega_i, \omega_i'\}_{i=1}^N$.
Input: number of boosting iterations B, learning rate η
Output: boosted learner $\boldsymbol{F}^{(b)}$
$\boldsymbol{F}^{(0)} \leftarrow 0$ {initialize boosted learner}
for $b \leftarrow 1, \ldots, B$ **do**
  $f^{(b)} \leftarrow \text{fit}\left(\left\{\mathbf{x}_i, \omega_i, \omega_i' - \eta\omega_i \boldsymbol{F}^{(b-1)}(\mathbf{x}_i)\right\}_{i=1}^N\right)$
  $\boldsymbol{F}^{(b)} \leftarrow \boldsymbol{F}^{(b-1)} + \eta f^{(b)}$
end

- Main aim: Minimal dependencies, very *very* short get-go time for the users [drop us an email].

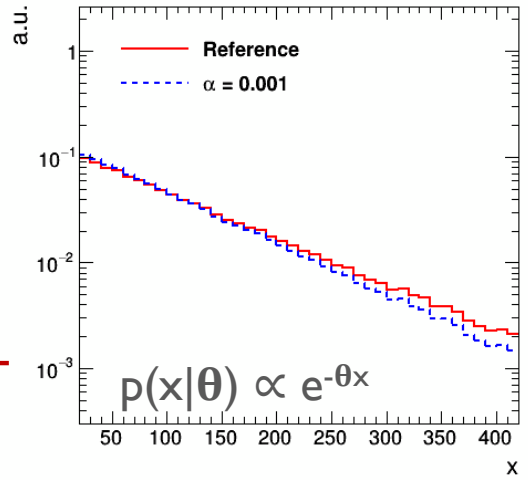
BOOSTED INFORMATION TREES: TOY MODELS

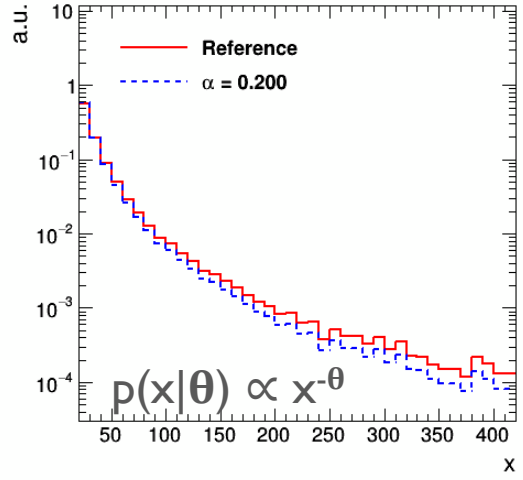
- Let's fit to analytically tractable models!

Exponential model

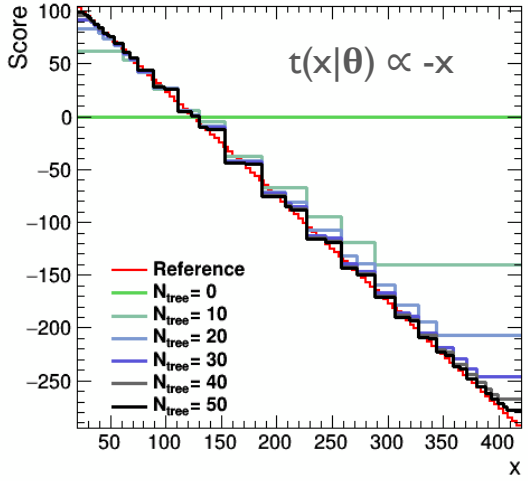
Power-law model

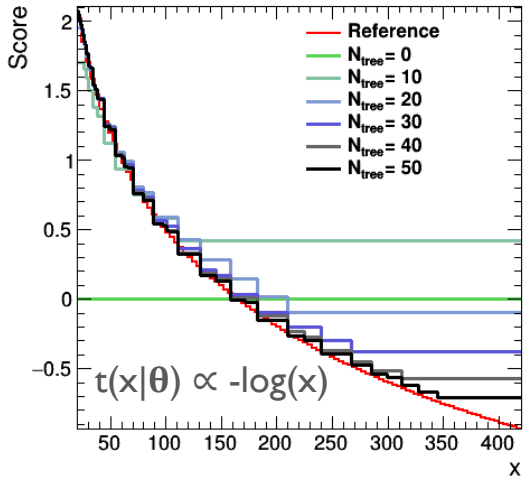
x ~ p(x|θ)



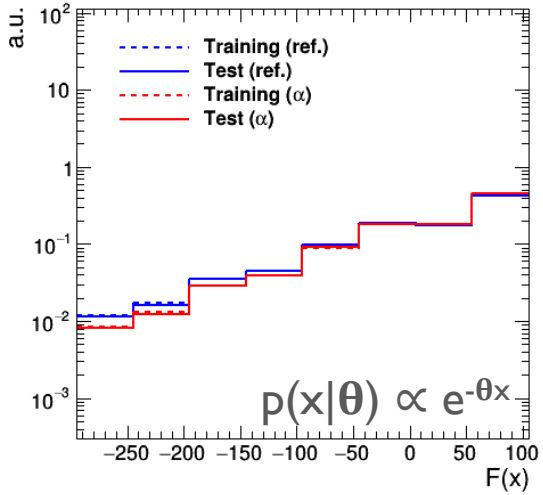


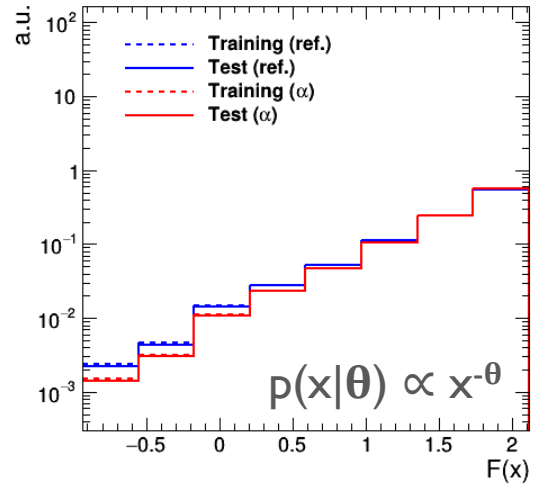
Learnt score vs. theory (red)



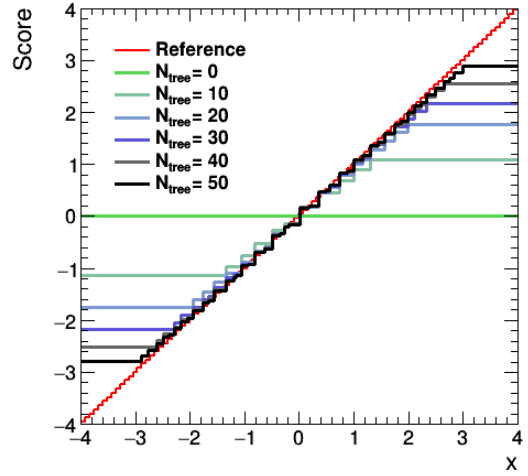


Train/test for classifier

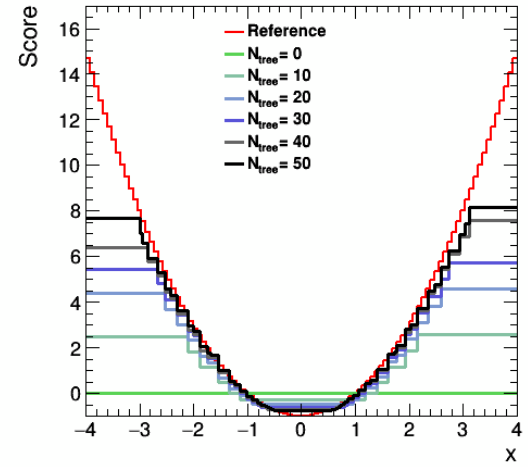




"Gaussian mean"



"Gaussian width"



OVERTRAINING & FURTHER TESTS

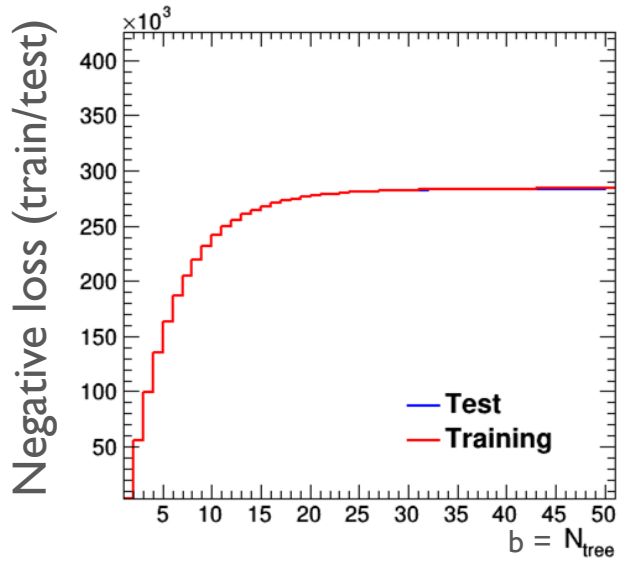
- If all weights (per-event scores) similar → overtraining mitigated, because there is no sampling stochastics in the weight-derivative computation; each event enters with both hypothesis
- If weight variance is large, overtraining situation is similar to BDT case
- Realistic SM-EFT examples show some weight variance, i.e., the discriminating information is in a subset
- Currently regularize with $N_{node,min}$, but look at options involving the distribution of weights/derivatives
- Preliminary toy studies show that we can extract information if it is dispersed in ≥2 features.
  - That's the benefit of boosting.

$$-L(\mathcal{D}, b) = \sum_{i \in \mathcal{D}} \left(\frac{\partial \omega_i(\theta)}{\partial \theta} \mathbf{F}^{(b)}(\mathbf{x_i}) \right)$$

Training or test loss for validation

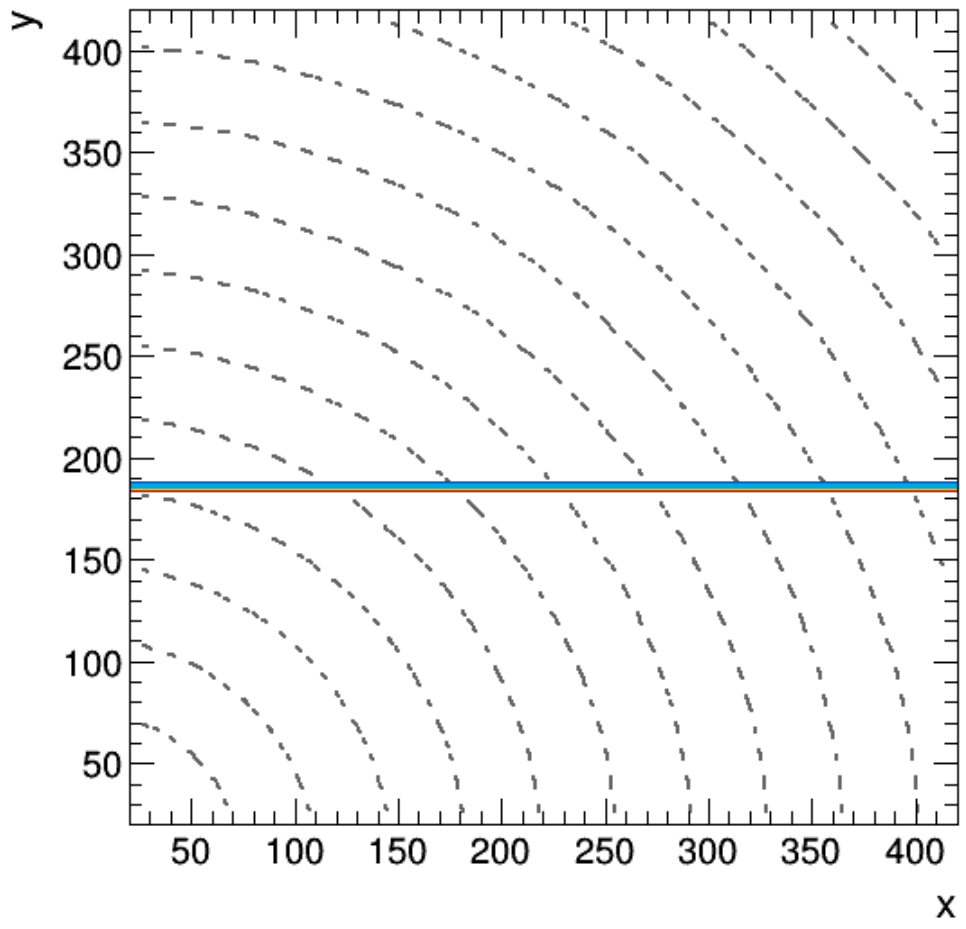
Training or test dataset

prediction up to tree b



No overtraining for small weight variance

2D EXPONENTIAL TOY



[Link] to animated gif

CLASSIFICATION WITH I_F = LEARNING THE SCORE

- We've started with classification, yet, the discriminator returns an estimate of the score.
  - How so? The algorithm has the computational complexity of *classification*, not regression!
  - Answer: linear approximation: Write likelihood as $p(x|\theta) = \frac{1}{Z(\theta)} p(t(x|\theta)|\theta) \exp(t(x|\theta) \cdot (\theta - \theta_0))$
  - In the exponential family! The score is a sufficient statistic for p(x|**θ**) [1805.00020]
- Clear in hindsight! Knowing the score *is* having the Fisher information.

- Let's *see* this: Fit a 2-node weak learner to *regress* to the score
  - 3-parameter estimator $\hat{t}(x|\alpha, F_L, F_R) = F_L \Theta(x < \alpha) + F_R \Theta(x \geq \alpha)$
- Use a χ^2 loss : $\chi^2(\alpha, F_L, F_R) = \mathbb{E}_{x,z}\left[\left(\frac{1}{\omega(\theta)}\frac{\partial \omega}{\partial \theta} - \hat{t}(x|\alpha, F_L, F_R)\right)^2\right]$

This loss is of the general form of Eq. 27 in [1805.00020]. On the same footing as SALLINO (TBC.)

- Eliminate the two equations for the regression predictions F_L/F_R, keep the one for the cut value:

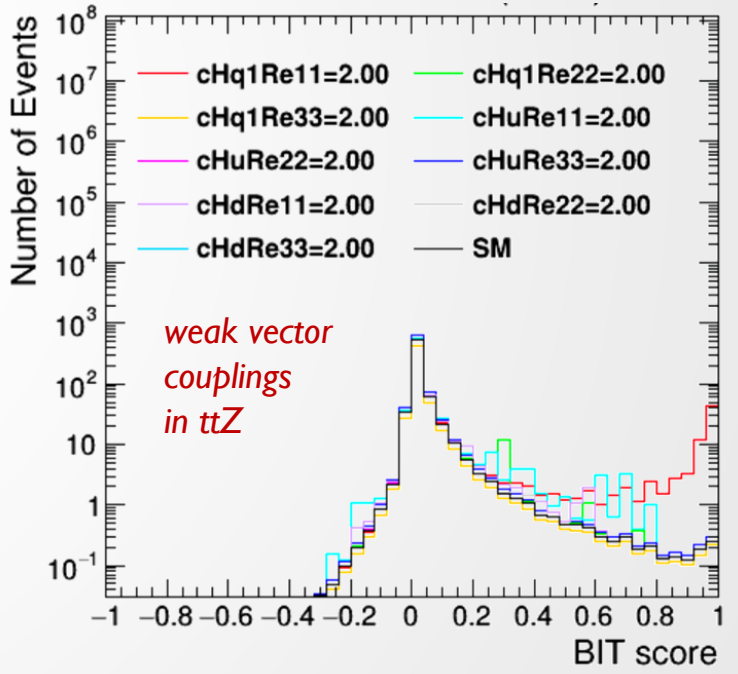
$$\chi^2(\alpha) = -\frac{1}{\lambda_L(\alpha)}\left(\frac{\partial \lambda_L}{\partial \theta}\right)^2 - \frac{1}{\lambda_R(\alpha)}\left(\frac{\partial \lambda_R}{\partial \theta}\right)^2$$

→ Boosted Information Trees regress in score by means of I_F classification

STATUS / CODE / OUTLOOK

- 'Boosted Information Tree' algorithm
  - Classifies EFT hypothesis that are arbitrarily close in parameter space: d$\boldsymbol{\theta}$ vs. $\boldsymbol{\theta}$+d$\boldsymbol{\theta}$
  - Boosting loss and node split consistently maximize Poisson Fisher information.
  - Equivalent (bot not *computationally*) to regression in the score. It's still classification.
  - Can train on $\boldsymbol{\theta}_{ref}$ = SM or otherwise
  - At $\boldsymbol{\theta}_{ref}$ = SM , we only learn the linear EFT correction!
    - Benefitial for validity considerations ☺

- Preprint in preparation, nightly build of code on GitHub
- Looking into feasibility of, e.g., XGBoost implementation
- Similar in scope as WDR! [A. Valassi, 2019,, 2020]
  1. Treatment of background differs
  2. Boosting; trying to bring back the plug&play feeling from BDTs



… collecting first experience with real-life settings

Algorithm 1 Decision Tree Weak Learner fit Procedure

Data: Dataset $\mathcal{D} = \{\mathbf{x}_i, \omega_i, \omega'_i\}_{i=1}^{N}$.
Input: tree depth D, minimum terminal node size $N_{\min}$
Output: weak learner F

$\alpha \leftarrow (), Q \leftarrow (\alpha), \mathcal{J} \leftarrow \{\}$
while $Q \neq \emptyset$ **do**
  $\alpha \leftarrow \mathsf{pop}(Q)$
  $\boldsymbol{\pi} \leftarrow \arg\mathrm{sort}\ \mathbf{x} \in \mathcal{D}_\alpha$ {sorted indices for each dimension p}
  if $|\alpha| \leq D \wedge |\mathcal{D}_\alpha| > 2\zeta$ **then**
    $p^*, k^* \leftarrow \arg\max_{p,k \in \{\zeta+1,\ldots,|\mathcal{D}_\alpha|-\zeta\}} \left(\sum_{i=1}^{k} \omega_{\boldsymbol{\pi}_{p,k}}\right)^{-1} \left(\sum_{i=1}^{k} \omega'_{\boldsymbol{\pi}_{p,k}}\right)^2 + \left(\sum_{i=k+1}^{|\mathcal{D}_\alpha|} \omega_{\boldsymbol{\pi}_{p,k}}\right)^{-1} \left(\sum_{i=k+1}^{|\mathcal{D}_\alpha|} \omega_{\boldsymbol{\pi}_{p,k}}\right)^2$
    $c \leftarrow x_{\boldsymbol{\pi}_{p^*,k^*},p^*}$ from $\mathcal{D}_\alpha$
    if *c is a valid cut* **then**
      $\alpha_{\mathrm{L}} \leftarrow \alpha \cup (p^*, \leq, c)$
      $\alpha_{\mathrm{R}} \leftarrow \alpha \cup (p^*, >, c)$
      $Q \leftarrow Q \cup (\alpha_{\mathrm{L}}, \alpha_{\mathrm{R}})$
    else
      $\mathcal{J} \leftarrow \mathcal{J} \cup \left(\alpha, \left(\sum_{\omega \in \mathcal{D}_\alpha} \omega\right)^{-1} \Big/ \sum_{\omega' \in \mathcal{D}_\alpha} \omega'\right)$
    end
  else
    $\mathcal{J} \leftarrow \mathcal{J} \cup \left(\alpha, \left(\sum_{\omega \in \mathcal{D}_\alpha} \omega\right)^{-1} \Big/ \sum_{\omega' \in \mathcal{D}_\alpha} \omega'\right)$
  end
end
return $F \leftarrow \sum_{j \in \mathcal{J}} \mathbb{1}_{\alpha_j}(\mathbf{x}) \cdot F_j$