

EvoTrees.jl

Efficient Boosted Trees on
CPUs & GPUs



Jérémie Desgagné-Bouchard, FCAS
Head of Science - Evovest



Motivation

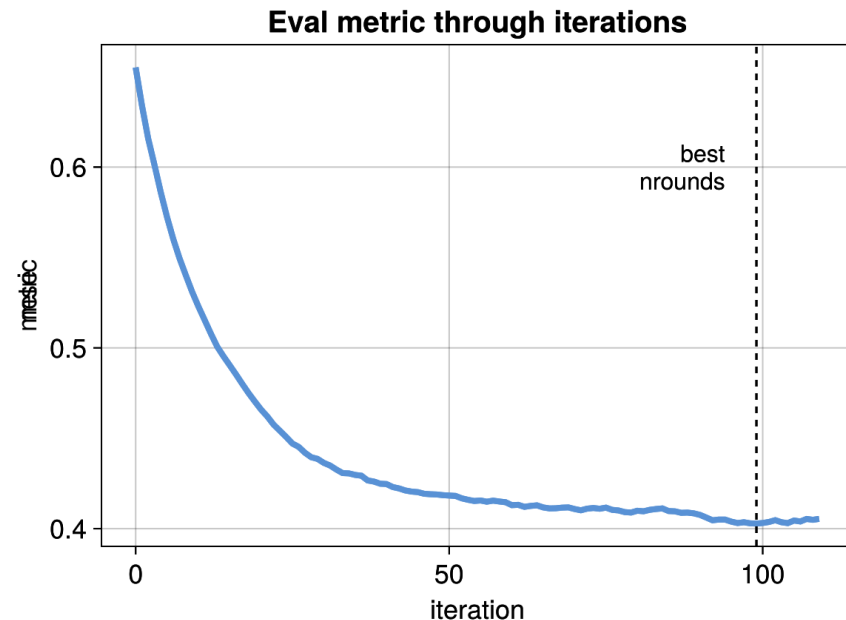
- **Overview of tree boosting algorithm**
 - And performance tricks used by EvoTrees.jl
- **Latest features**
 - Tree bagging
 - Oblivious trees
- **Alternatives to gradient-based losses**
 - mean-absolute-error
 - volatility-adjusted losses
- **Future development paths**
 - Improved acceleration
 - Multi-target losses

Basics

MLJ compliant API

| Row | target_name | | feature_names | | | | | | |
|-----|-------------------|-----------------|---------------|----------------|----------------|----------------|-----------------|-----------------------|--|
| | Survived Int64 | Pclass Int64 | Sex Cat... | Age Float64 | SibSp Int64 | Parch Int64 | Fare Float64 | Age_ismissing Bool | |
| 1 | 0 | 1 | male | 29.0 | 0 | 0 | 30.0 | false | |

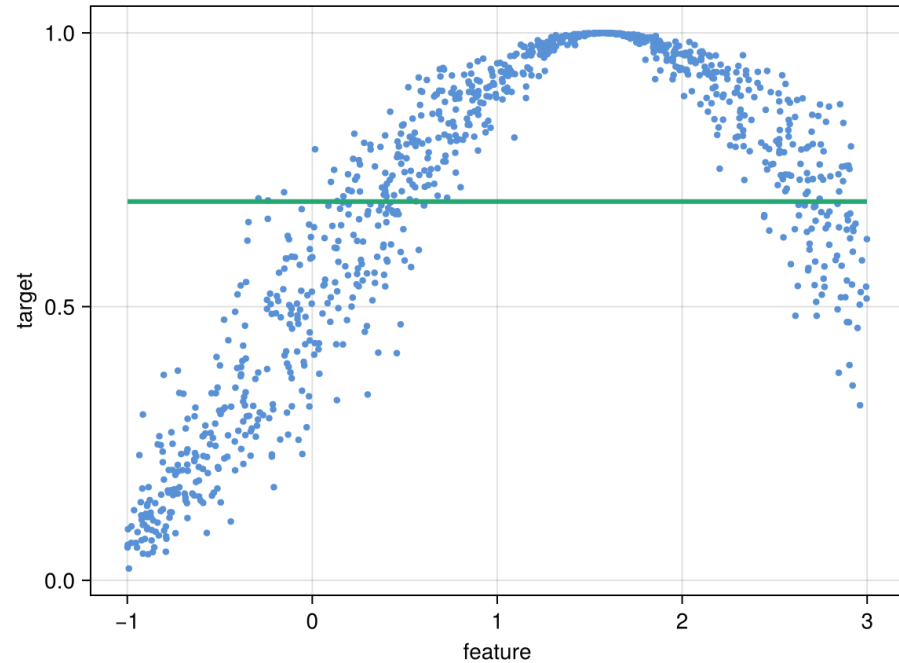
```
config = EvoTreeRegressor(  
    loss=:logloss,  
    nrounds=200,  
    early_stopping_rounds=10,  
    eta=0.05,  
    max_depth=5  
)  
  
model = EvoTrees.fit(  
    config, dtrain;  
    deval,  
    target_name,  
    feature_names  
)  
  
pred = model(deval)
```



Algo Overview

Boosted algorithm

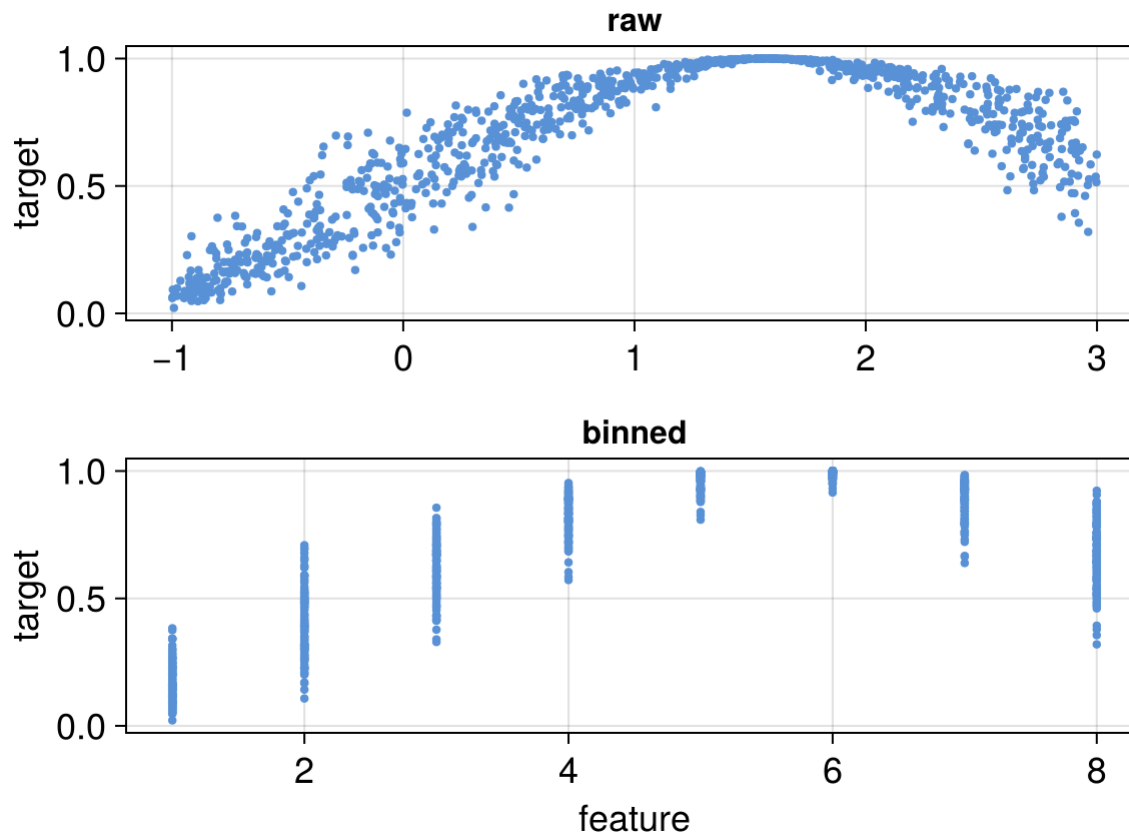
0.692



tree #1 = bias

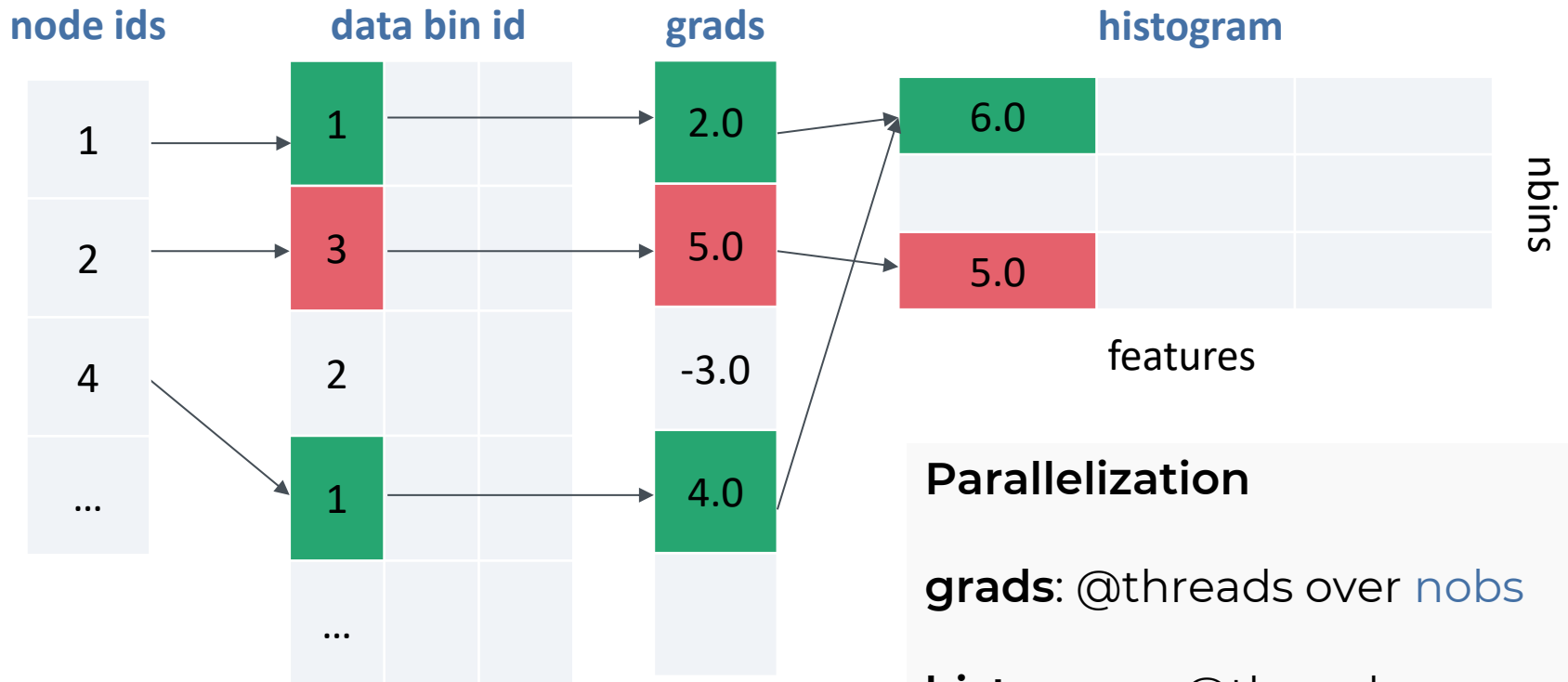
Algo – Histogram method

A core methodology for fast training used by all performant implementations (XGBoost, LightGBM, CatBoost)



Note how the binarization results in features that are scale and distribution invariant

Algo – Histogram method



Parallelization

grads: @threads over **nobs**

histogram: @threads over **features** (and **nobs** on GPU)

node id update: @threads over **nodes nobs**

Training Speed

CPU

| # obs | depth | EvoTrees | XGBoost | log-diff |
|-------|-------|----------|---------|----------|
| 100K | 6 | 0.83 | 1.31 | -20% |
| 100K | 11 | 3.43 | 3.33 | 1% |
| 1M | 6 | 5.50 | 13.57 | -39% |
| 1M | 11 | 15.72 | 18.79 | -8% |
| 10M | 6 | 79.99 | 147.04 | -26% |
| 10M | 11 | 185.47 | 189.04 | -1% |

GPU

| # obs | depth | EvoTrees | XGBoost | log-diff |
|-------|-------|----------|---------|----------|
| 100K | 6 | 1.64 | 0.66 | 40% |
| 100K | 11 | 17.98 | 3.77 | 68% |
| 1M | 6 | 3.53 | 3.19 | 4% |
| 1M | 11 | 30.11 | 8.58 | 55% |
| 10M | 6 | 21.56 | 27.49 | -11% |
| 10M | 11 | 65.96 | 49.85 | 12% |

Tree Bagging

Tree bagging allows to train models similar to RandomForest

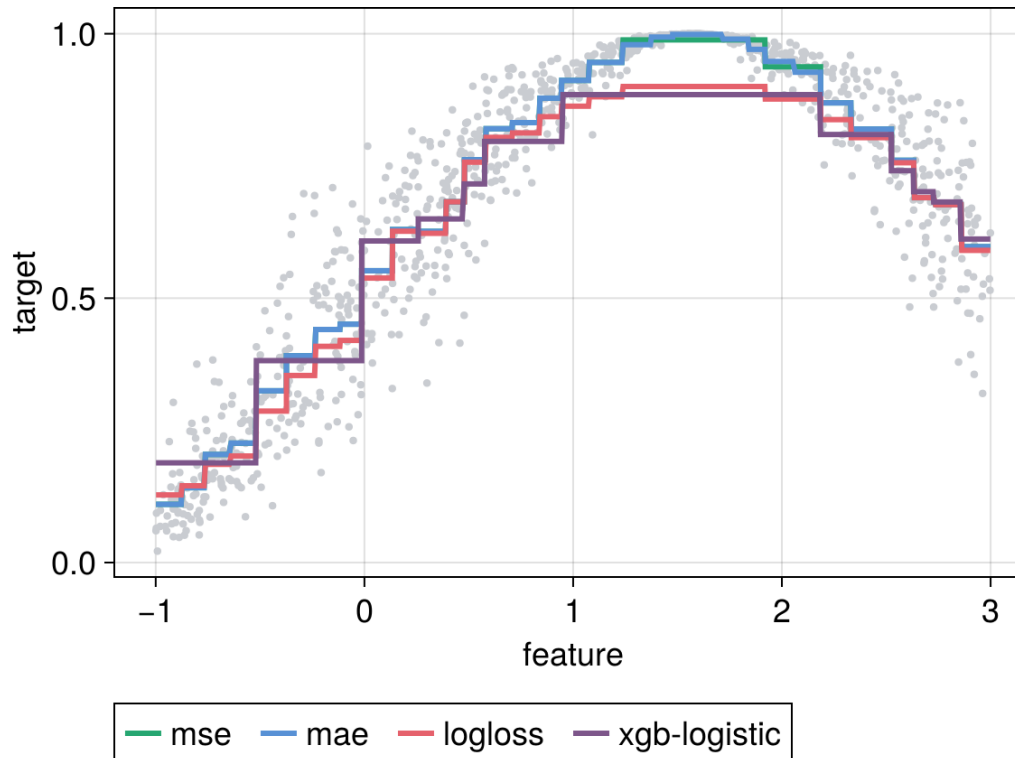
```
for tree in tree_size
  update_gradients!
  for bag in bag_size
    subsample_row_cols!
    grow_tree!
    predict!
  end
end
```

```
EvoTreeRegressor(  
  nrounds=1,  
  bagging_size=8,  
  eta=1.0,  
  row_sample=0.5,  
  col_sample=0.5,  
  loss=:mse,  
  kwargs...  
)
```


Tree Bagging

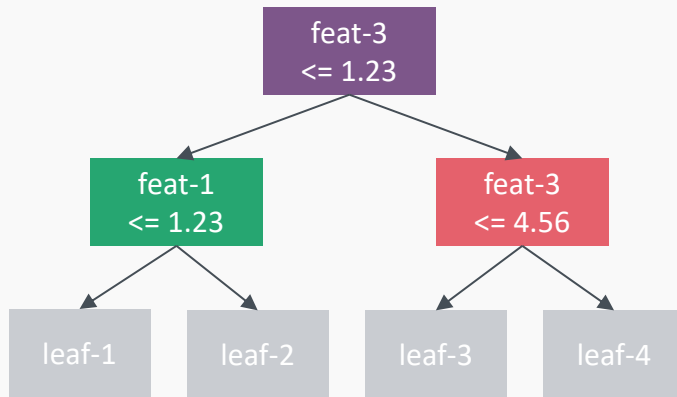
⚠ Only few loss functions can return a good model with a RandomForest emulation: $nrounds = 1$; $eta = 1.0$

Limitation intrinsic to gradient learning method. A single tree quality is limited to the quality of second-order approximation vs. the original loss curve.



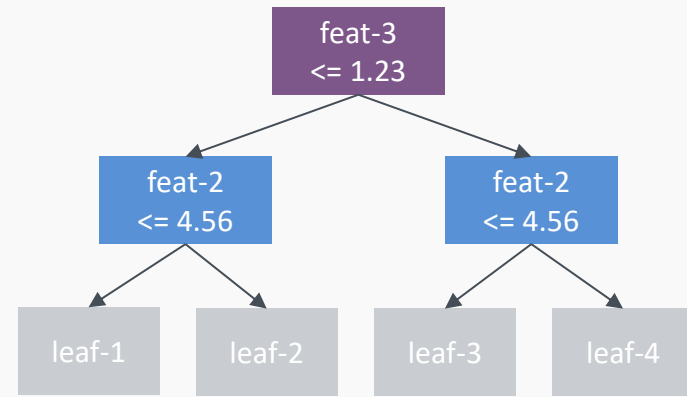
Oblivious Tree Structure

Binary



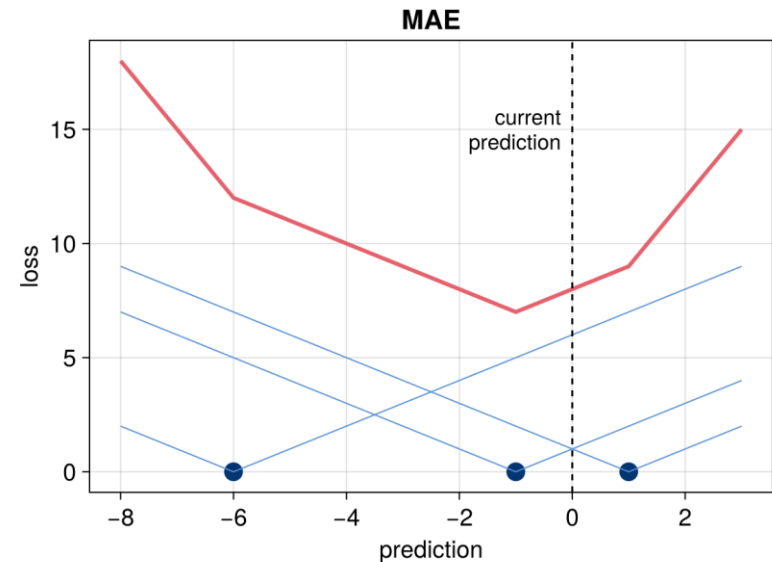
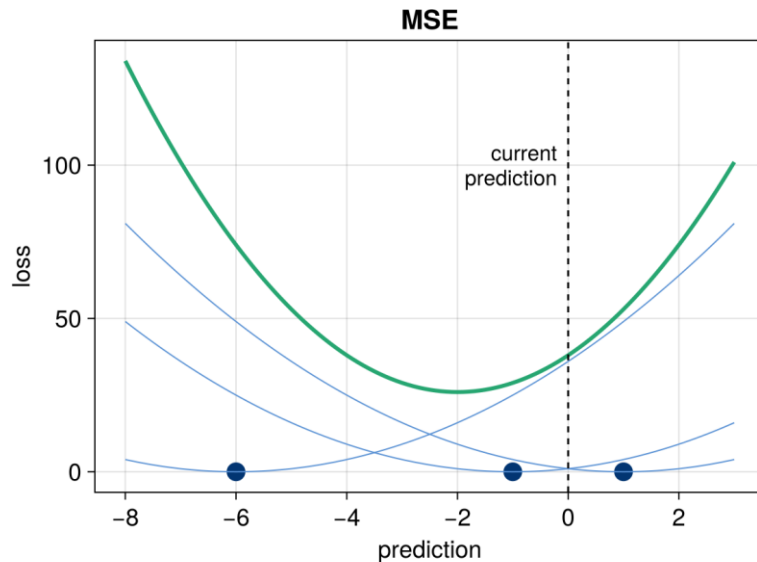
```
EvoTreeRegressor(  
  tree_type = :binary,  
  kwargs...  
)
```

Oblivious



```
EvoTreeRegressor(  
  tree_type = :oblivious,  
  kwargs...  
)
```

Loss Without Gradients - MAE



Are the 2 first moments needed if all we want is to move prediction towards the mean?

Alternative perspective for tree-split gain: any metric derived from histogram statistics.

For *MAE proxy*: gain = sum of absolute change in predicted value.

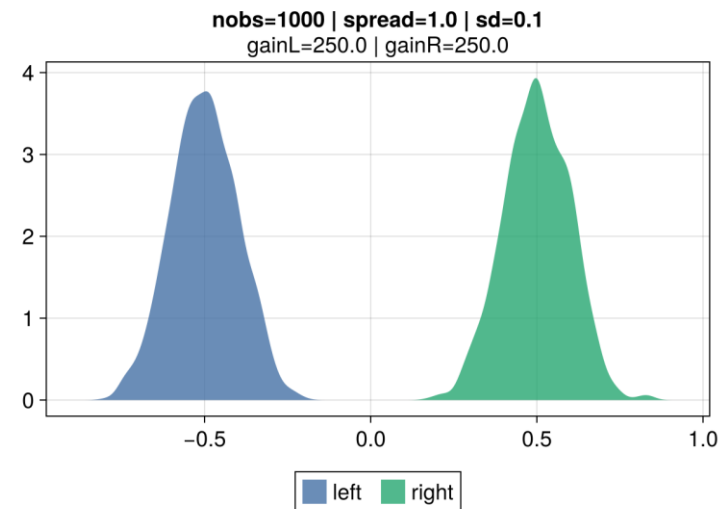
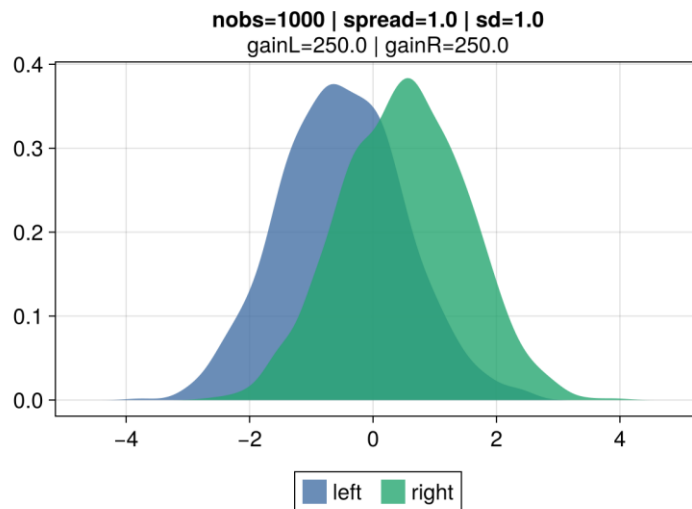
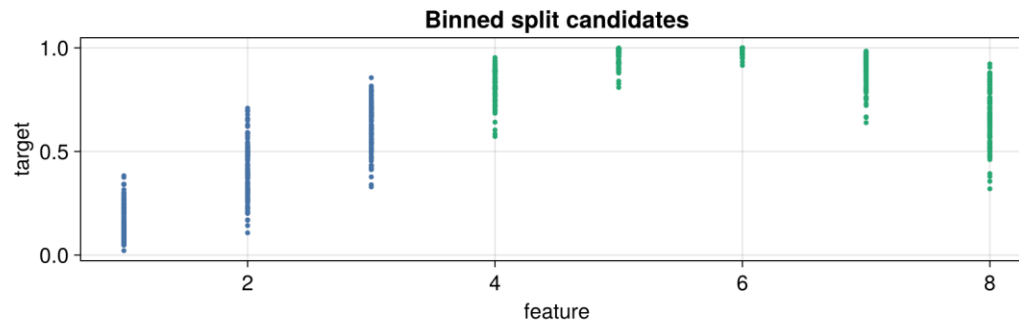
A single metric is needed, the residual: ***target - pred***

Cannot work beyond depth 2 without a trick gain valuation: gains need to be derived from the context of the parent node statistics.

Loss Without Gradients - Credibility

All other things being equal, the gain for two split candidates will be the same regardless of the volatility of observations within each leaf.

In regular regression, the gain is entirely driven by the mean of observations.



Loss Without Gradients - Credibility

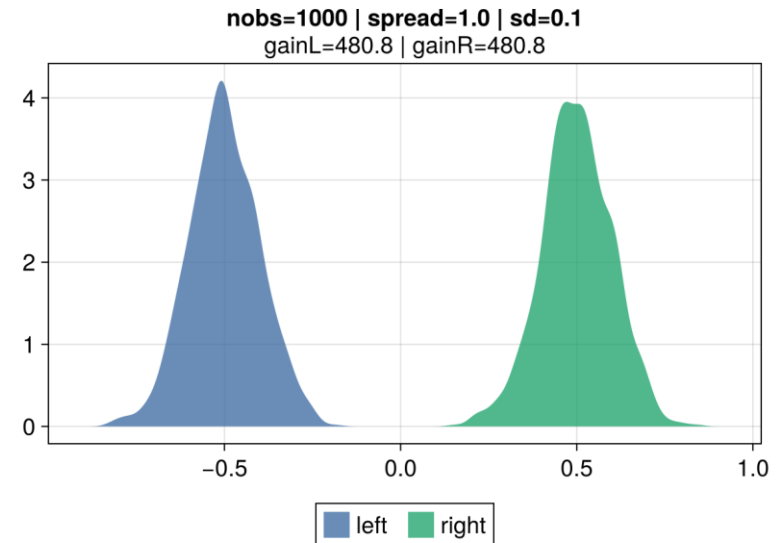
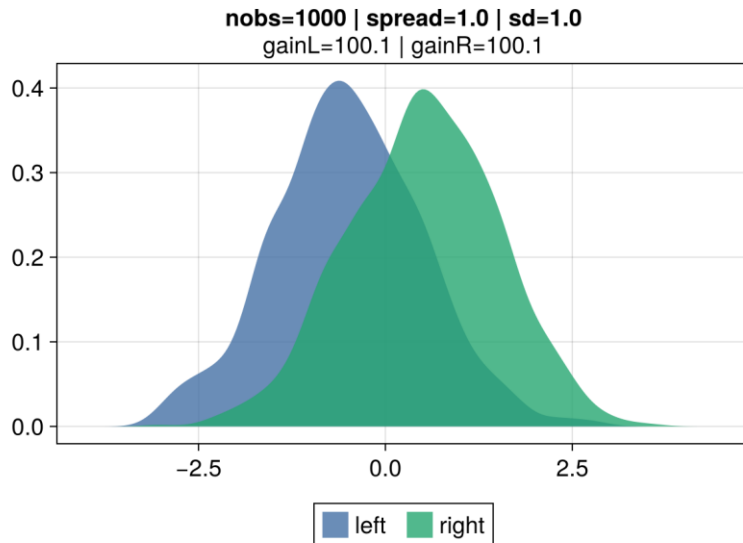
The gain is derived as the credibility-adjusted leaf mean.

Credibility is inversely proportional to the volatility of the leaf's observations.

$$Z = VHM / (VHM + EVPV)$$

Variance of the Hypothetical Means (VHM): if large differences between candidate means are expected, a greater credibility is assigned.

Expected Value of the Process Variance (EVPV): if the data generation process of a given candidate has high volatility, a lower credibility is assigned.



Performance Benchmarks

MSE metrics on regression problems

| model | mse | mae | cred_var | cred_std |
|--------|-------|-------|----------|----------|
| boston | 6.3 | 5.71 | 5.95 | 5.43 |
| year | 74.9 | 74.3 | 74.6 | 74.2 |
| msrank | 0.55 | 0.549 | 0.551 | 0.549 |
| yahoo | 0.565 | 0.567 | 0.589 | 0.568 |

Roadmap

- **GPU improvement**

- SciML's small grant to bring GPU at XGBoost level (USD 1,500)
- <https://github.com/Evovest/EvoTrees.jl/issues/288>

- **Multi-target support**

- Tree structure where leaves generate a vector of predictions, one for each of the multiple target variables

```
EvoTrees.fit(  
    config,  
    dtrain;  
    target_name = [y1, y2, y3],  
    kwargs...  
)
```

Roadmap

Autodiff: Support user defined loss

```
function my_mse(p, y)
    return (x - y)^2
end

function _mse_grad(p, dp, y)
    autodiff(Reverse, my_mse, Active, Duplicated(p, dp), Const(y))
    return nothing
end

autodiff(Forward, _mse_grad,
    Duplicated(p, vp),
    Duplicated(dp, hess),
    Const(y),
)
```

- Easy for single parameter loss, but also trivial to manually implement
- Complication arises for complex loss like penalized
- Consider 1st-order heuristics for training (similar to *MAE* loss)

Questions?

