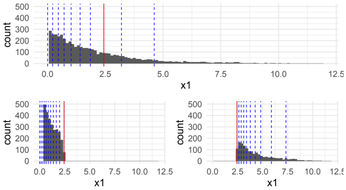
Introduction to Machine Learning

Modern Boosting Techniques



Learning goals

- XXX
- XXX

BEYOND XGBOOST

Next to XGBoost two other important modern boosting libraries exist:

- **LightGBM** by **Ke et al. (2017)**
- **CatBoost** by **Prokhorenkova et al. (2017)**

Both libraries extend the ideas of **XGBoost** in several areas:

1. Tree growing efficiency
2. Data sampling
3. Feature compression

Many of the the proposed ideas have later been implemented in **XGBoost** as well.

TREE GROWING EFFICIENCY

Recall: **XGBoost** grows a balanced tree of $\mathrm{max\_depth}$ and prunes leaves that do not improve the risk.

Leaf-wise (Best-first) Tree Growth allows the growing of unbalanced trees by comparing improvements between all possible leaves.

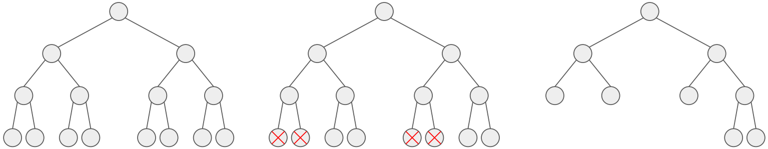


Figure: Tree with $\mathrm{max\_depth}$=3. Balanced tree (left), pruned balanced tree (middle), leave-wise grown tree (right)

DATA SAMPLING

Recall: **XGBoost** (and many other boosting libraries) use random data subsampling, i.e. stochastic gradient boosting.

Stochastic gradient boosting can be improved by *smarter* sampling strategies based on the values of the pseudo residuals.

Gradient-based One-Side Sampling (GOSS):

- To evaluate a split GOSS only uses the $a \cdot n$ observations with largest (absolute) gradients and samples $b \cdot n$ observations from the remaining.
- The randomly sampled observations with smaller gradients are weighted by $\frac{1-a}{b}$.
- Default values are $a = 0.2$ and $b = 0.1$.
- GOSS is only used after $\frac{1}{\nu}$ iterations of regular boosting steps.

DATA SAMPLING

Minimal Variance Sampling (MVS):

- MVS computes weights and selection probabilities for observations for a tree.
- The weighting is computed from the regularized absolute value $\hat{g}^{[m]}(\mathbf{x}^{(i)}) = \sqrt{g^{[m]}(\mathbf{x}^{(i)})^2 + \lambda h^{[m]}(\mathbf{x}^{(i)})^2}$.
- Observations with a value of $\hat{g}^{[m]}(\mathbf{x}^{(i)}) > \mu$ are always used and other observations are selected with probability $\frac{\hat{g}^{[m]}(\mathbf{x}^{(i)})}{\mu}$.
- μ has a closed-form nearly optimal solution for minimizing the risk of a tree base learner **(Ibragimov et al. 2019)**.
- For the tree fit each observation is weighted inversely proportional to its selection probability.

FEATURE COMPRESSION

For high dimensional (sparse) data it can be helpful to bundle similar features together to speed up split computations.

Exclusive feature bundling looks for mutually exclusive features, i.e. features that never take nonzero values simultaneously.

- A single histogram for approximate split finding in boosting can be built from multiple mutually exclusive features nearly without loss of information.
- Mutually exclusive features only occur in sparse data.
- This approach speeds up the histogram building from $\mathcal{O}(np)$ to $\mathcal{O}(nb)$ where b is the number of feature bundles.
- While finding the optimal bundling is *np*-hard, greedy approximations give good results empirically.

FEATURE COMPARISON OF BOOSTING FRAMEWORKS

	Parallel	GPU Support	Approx. splits	Categ. feats
XGBoost	*x*	*x*	*x*	
LightGBM	*x*	*x*	*x*	*x*
CatBoost	*x*	*x*	*x*	*x*
GBM				*x*
H2O	*x*	*x*	*x*	*x*
sklearn	*x*		*x*	*x*

FEATURE COMPARISON OF BOOSTING FRAMEWORKS

	Tree growing		Subsampling		
			Observations		Feats
	Depth-wise	Leaf-wise	Regular	Gradient-based	
XGBoost	*x*	*x*	*x*	*x*	*x*
LightGBM	*x*	*x*	*x*	*x*	*x*
CatBoost	*x*	*x*	*x*	*x*	*x*
GBM		*x*	*x*		
H2O	*x*		*x*		*x*
sklearn	*x*		*x*		*x*

BENCHMARK