Module 10 – Part III
Advanced Boosting models

dmlc XGBoost

CatBoost

LightGBM

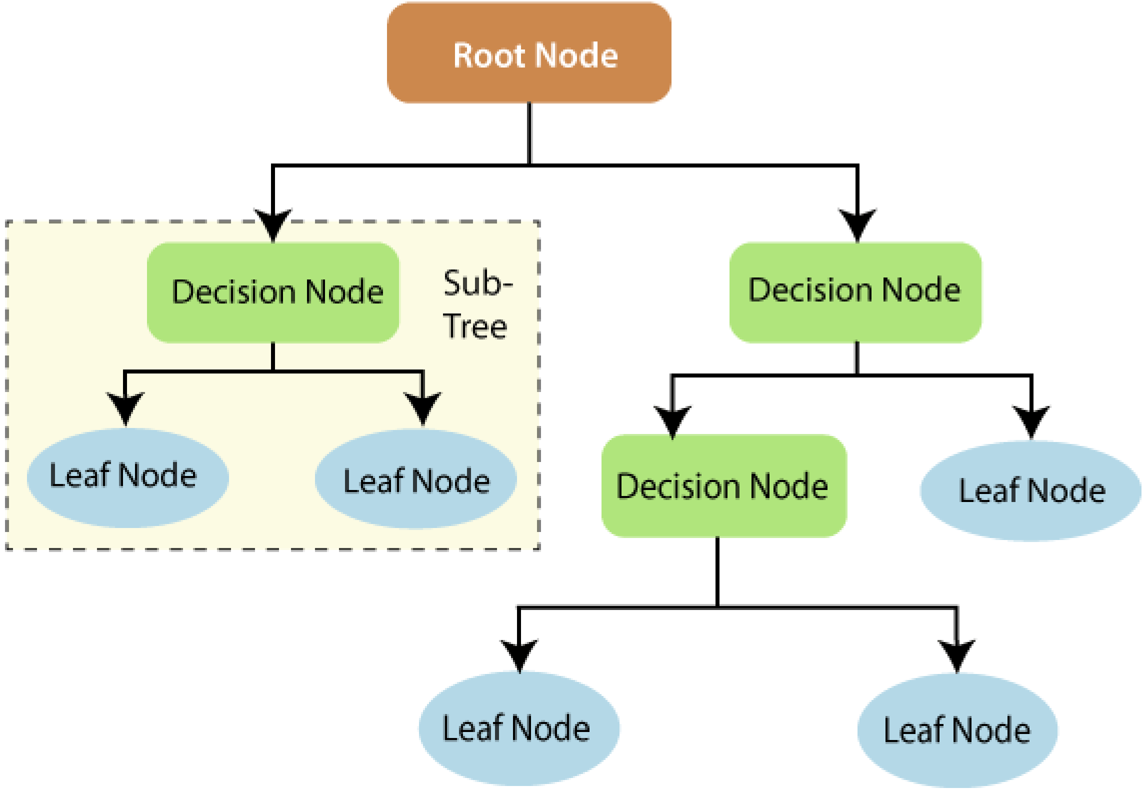
Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

Decision Trees Fundamental questions

- Four fundamental questions to be answered:

1) What feature and cut off to start with?
2) How to split the samples?
3) How to grow a tree?
4) How to combine trees?

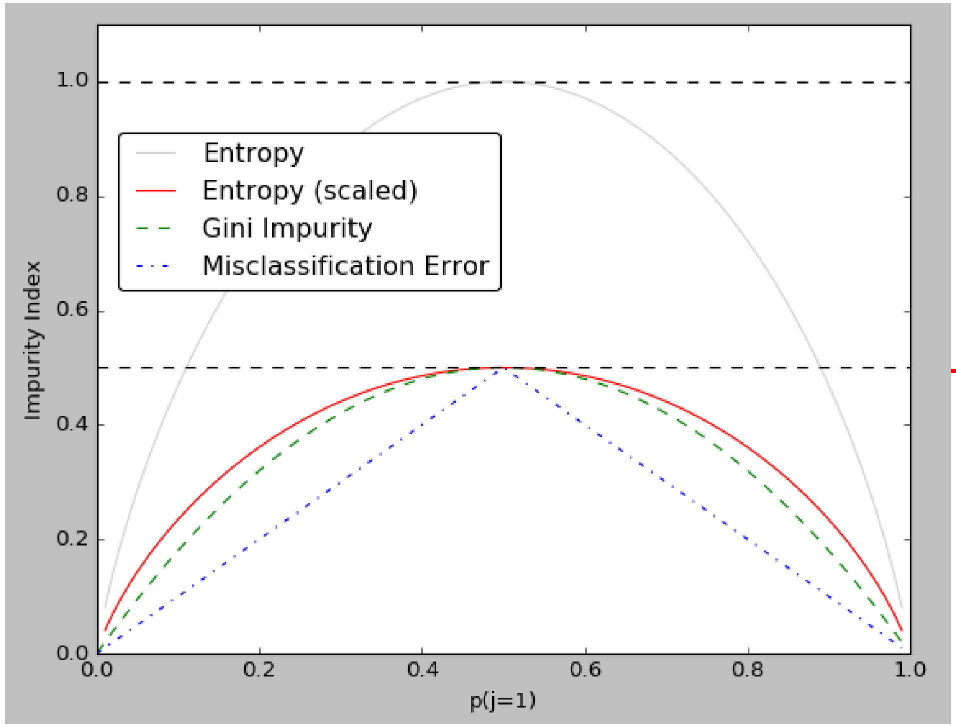


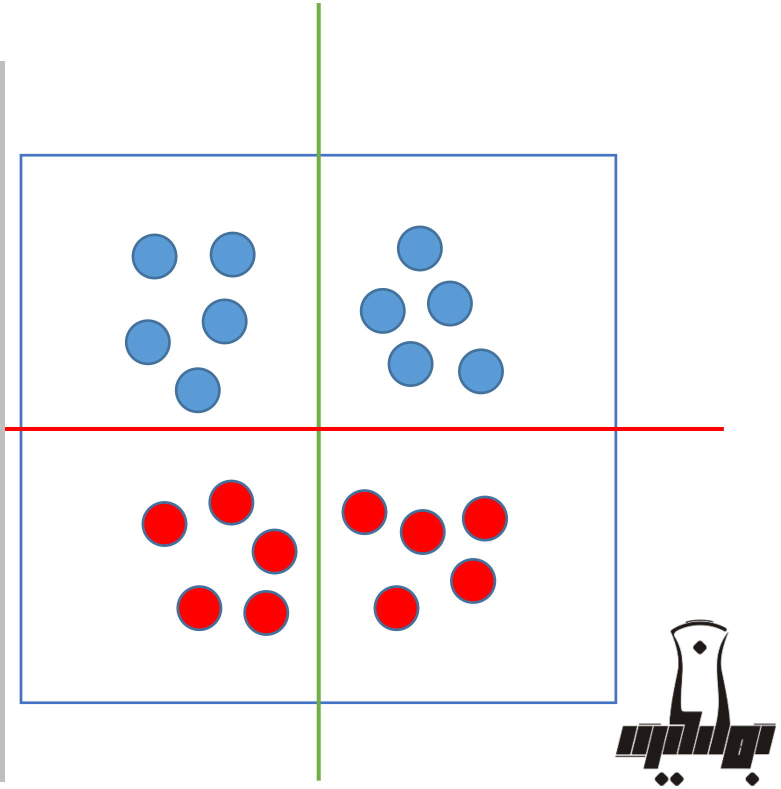
What feature and cut off to start with?

- Which feature and cut off adds the most information gain (minimum impurity)?

- Regression trees: MSE
- Classification trees:
  1. Error rate
  2. Entropy
  3. Gini Index

Control how a Decision Tree decides to split the data





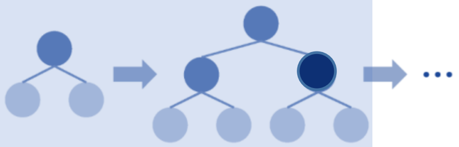
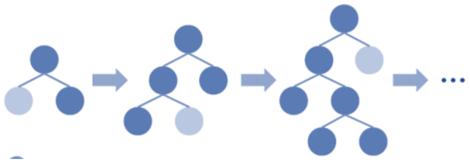
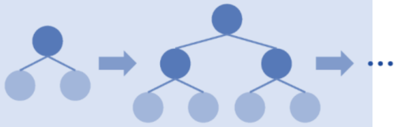
How to split the samples?

Method	Description
Pre-sorted and histogram based	This method sorts the data and creates histograms of the values before splitting the tree. This allows for faster splits but can result in less accurate trees.
GOSS (Gradient-based One-Side Sampling)	This method uses gradient information as a measure of the weight of a sample for splitting. Keeps instances with large gradients while performing random sampling on instances with small gradients.
Greedy method	This method selects the best split at each step without considering the impact on future splits. This method May result in suboptimal trees

Jon M. Huntsman School of Business
Utah State University

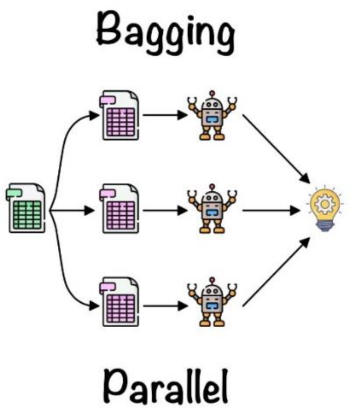
Pedram Jahangiry

How to grow a tree?

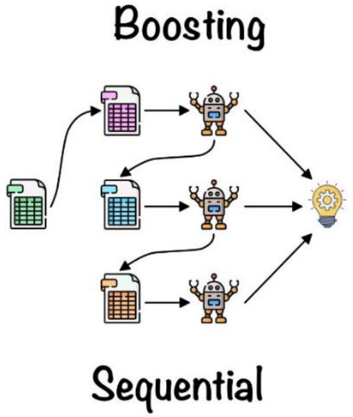
Algorithm	Description
Depth-Wise Level-Wise	Repeatedly splitting the data along the feature with the highest information gain, until a certain maximum depth is reached. Resulting in a tree with a balanced structure, where all leaf nodes are at the same depth. 
Leaf-wise	Repeatedly splitting the data along the feature with the highest information gain, until all leaf nodes contain only a single class. Resulting in a tree with a highly unbalanced structure, where some branches are much deeper than others. 
Symmetric	Builds the tree by repeatedly splitting the data along the feature with the highest information gain, until a certain stopping criterion is met (e.g. a minimum number of samples per leaf node). Resulting in a more balanced tree structure than leaf-wise growth. 

How to combine trees?

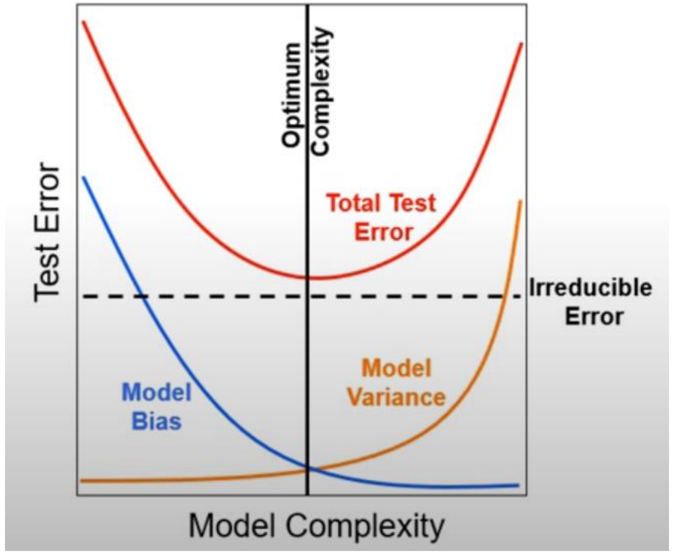
- Bagging consists of creating many “copies” of the training data (each copy is slightly different from another) and then apply the weak learner to each copy to obtain multiple weak models and then combine them.
- In bagging, the bootstrapped trees are independent from each other.

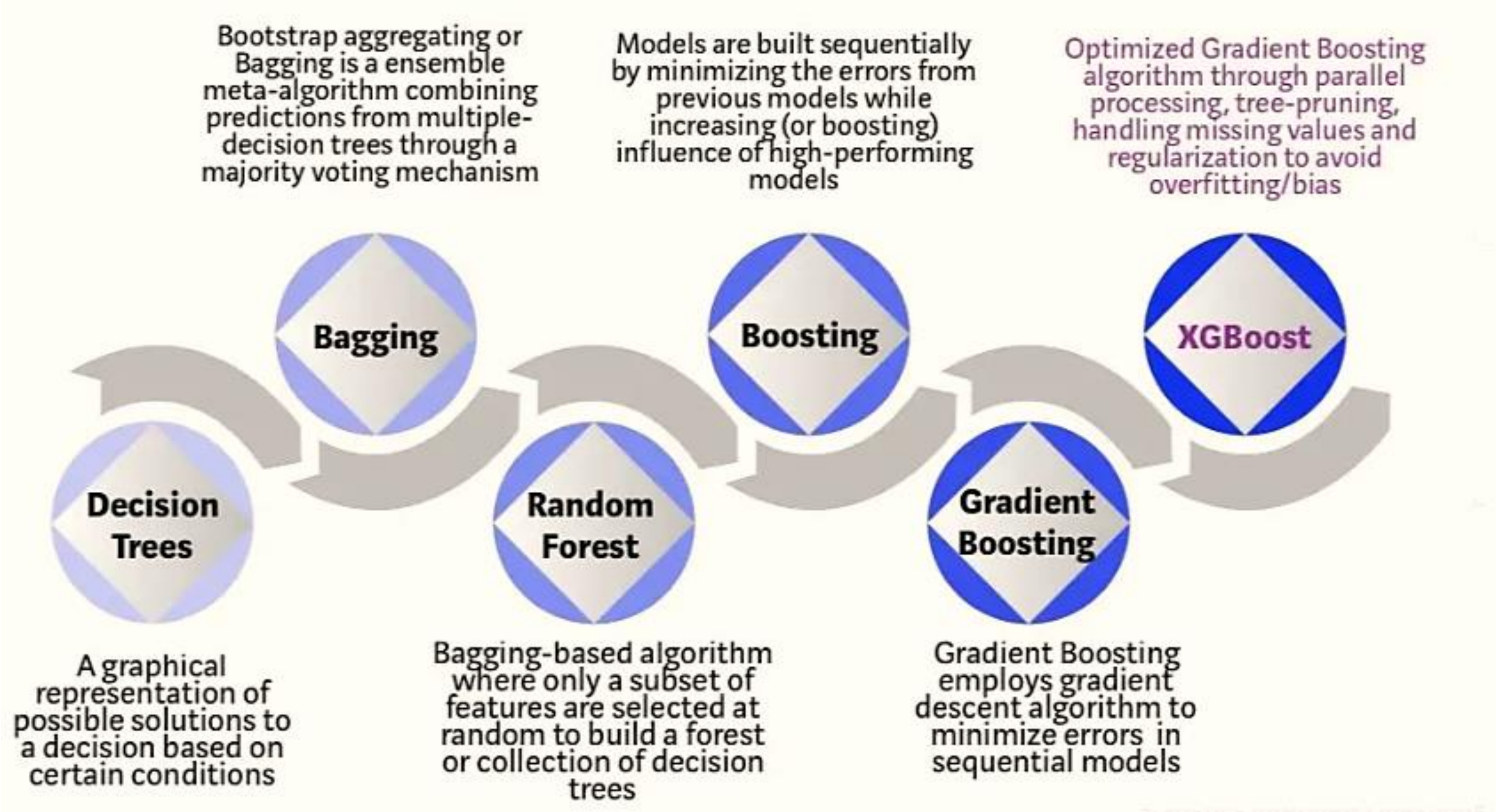


- Boosting consists of using the “original” training data and iteratively creating multiple models by using a weak learner. Each new model tries to “fix” the errors which previous models make.
- In boosting, each tree is grown using information from previous tree.



Evolution of XGBoost





Decision Trees: A graphical representation of possible solutions to a decision based on certain conditions

Bagging: Bootstrap aggregating or Bagging is a ensemble meta-algorithm combining predictions from multiple-decision trees through a majority voting mechanism

Random Forest: Bagging-based algorithm where only a subset of features are selected at random to build a forest or collection of decision trees

Boosting: Models are built sequentially by minimizing the errors from previous models while increasing (or boosting) influence of high-performing models

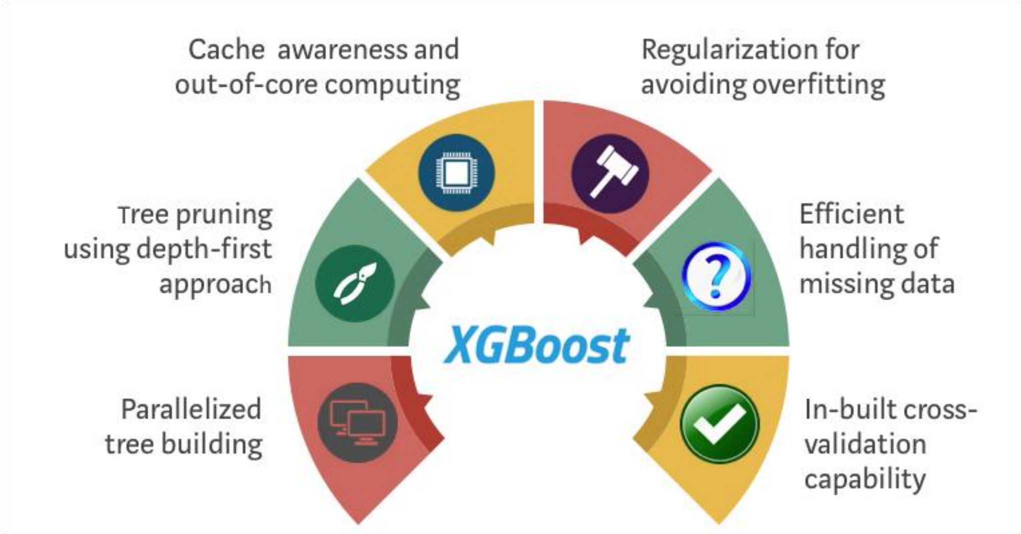
Gradient Boosting: Gradient Boosting employs gradient descent algorithm to minimize errors in sequential models

XGBoost: Optimized Gradient Boosting algorithm through parallel processing, tree-pruning, handling missing values and regularization to avoid overfitting/bias

XGBoost: eXtreme Gradient Boosting

- XGBoost is an open-source gradient boosting library developed by Tianqi Chen (2014) focused on developing efficient and scalable machine learning algorithms.
- Extreme refers to the fact that the algorithms and methods have been customized to push the limit of what is possible for gradient boosting algorithms.
- XGBoost includes several other features that can improve model performance, such as handling missing values, automatic feature selection, and model ensembling.

dmlc XGBoost


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

LightGBM (Light Gradient Boosted Machine)

- LightGBM is an open-source gradient boosting library developed by Microsoft (2016) that is fast and efficient, making it suitable for large-scale learning tasks.
- LightGBM can handle categorical features, but requires one-hot encoding, ordinal encoding or other preprocessing
- LightGBM includes several other features that can improve model performance, such as handling missing values, automatic feature selection, and model ensembling.

LightGBM

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry

CatBoost (Category Boosting)

- CatBoost is an open-source gradient boosting library developed by Yandex (2017) that is specifically designed to handle categorical data.
- CatBoost can handle categorical features directly, without the need for one-hot encoding or other preprocessing.
- CatBoost includes several other features that can improve model performance, such as handling missing values, automatic feature selection, and model ensembling.

XGBoost vs LightGBM vs CatBoost

	XGBoost	LightGBM	CatBoost
Developer	Tianqi Chen (2014)	Microsoft (2016)	Yandex (2017)
Base Model	Decision Trees	Decision Trees	Decision Trees
Tree growing algorithm	Depth-wise tree growth Leaf-wise is also available	Leaf-wise tree growth	Symmetric tree growth
Parallel training	Single GPU	Multiple GPUs	Multiple GPUs
Handling categorical features	Encoding required (one-hot, ordinal, target, label, …)	Automated encoding using categorical feature binning	No encoding required
Splitting method	Pre-sorted and histogram based	GOSS (Gradient based one-side sampling)	Greedy method

dmlc XGBoost

LightGBM

CatBoost

Jon M. Huntsman School of Business
Utah State University

Pedram Jahangiry