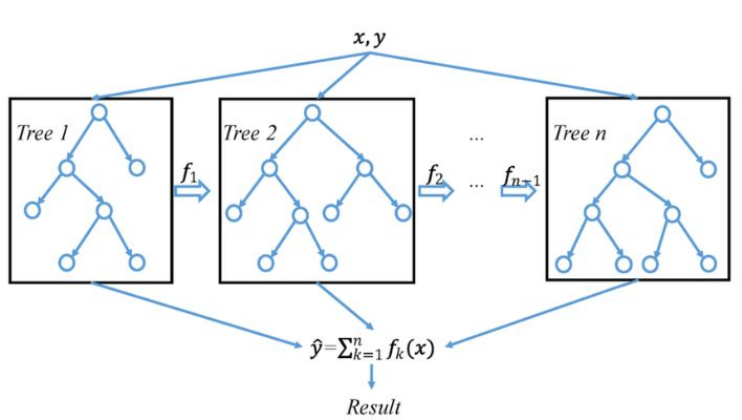
Extreme Gradient Boosting (XGBoost)

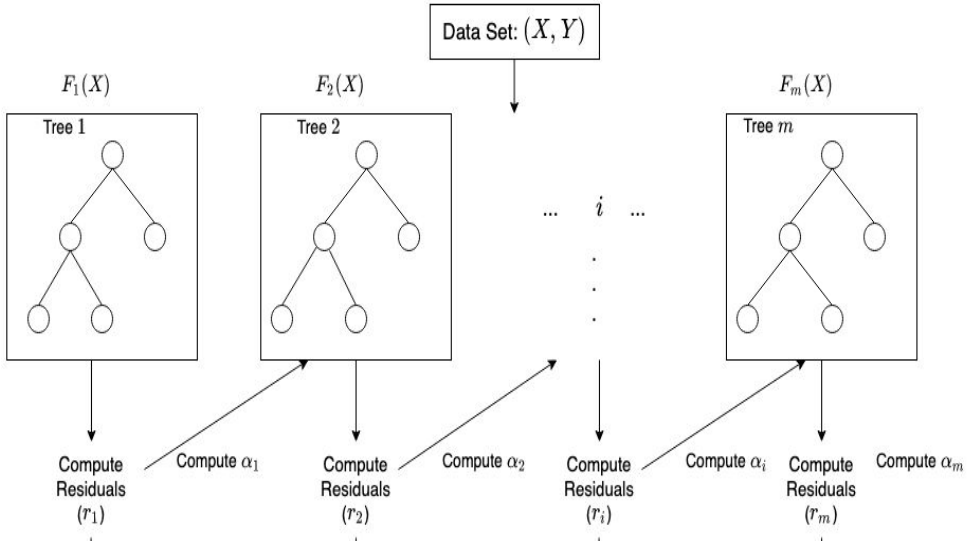


Group 6:

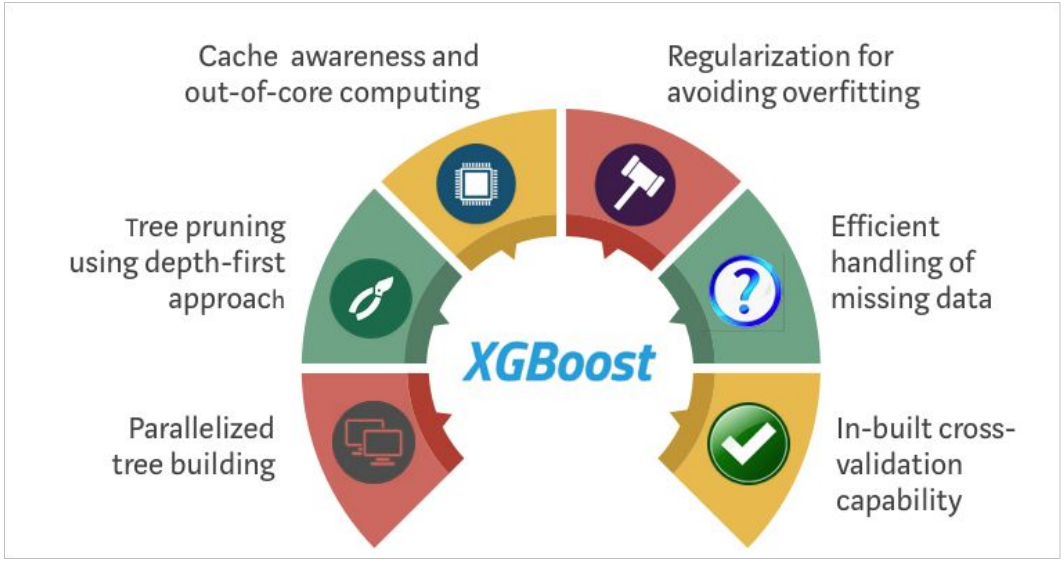
Armin Nouri
Chris Kusha
Jassleen Bhullar
Sara Douglas

Algorithm

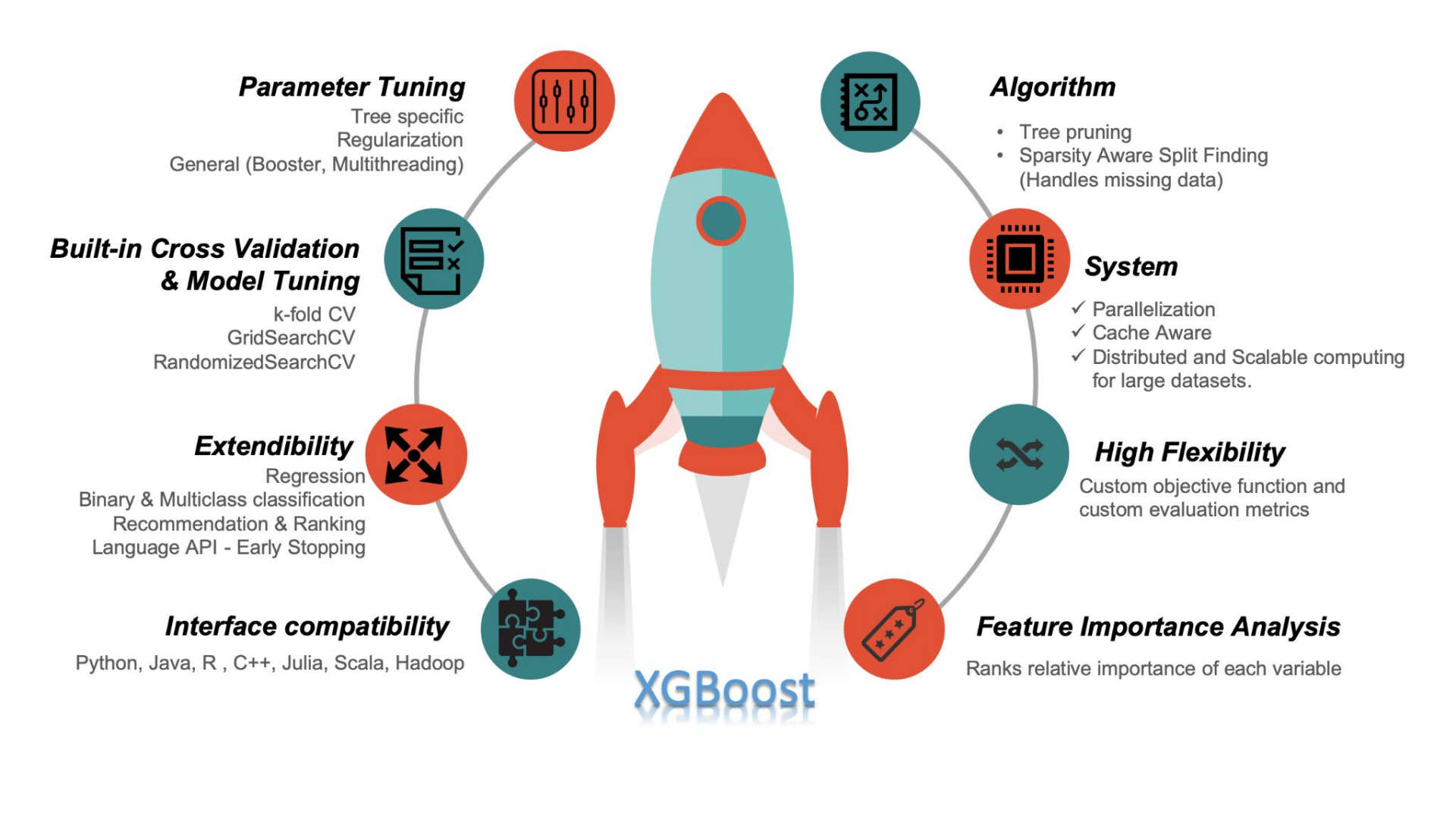
- Basic structure:
  - Random forest
    - Decision trees
- Boosting:
  - ensemble modelling
  - Strong - > weak
- Gradient Boosting



Advantages



- Consists of hyperparameters than can be tuned
- Handles missing values
- Tree pruning
- Provides intuitive features, including parallelisation, distributed computing, and cache optimization
- Core algorithm is parallelizable - high speed and performance
- Can be used with multiple base learners - Tree Base learner is most used
- High flexibility - can define custom optimization objectives and evaluation criteria
- Built in cross-validation


Parameter Tuning
Tree specific
Regularization
General (Booster, Multithreading)
Built-in Cross Validation & Model Tuning
k-fold CV
GridSearchCV
RandomizedSearchCV
Extendibility
Regression
Binary & Multiclass classification
Recommendation & Ranking
Language API - Early Stopping
Interface compatibility
Python, Java, R , C++, Julia, Scala, Hadoop
Algorithm
• Tree pruning
• Sparsity Aware Split Finding (Handles missing data)
System
✓ Parallelization
✓ Cache Aware
✓ Distributed and Scalable computing for large datasets.
High Flexibility
Custom objective function and custom evaluation metrics
Feature Importance Analysis
Ranks relative importance of each variable
XGBoost

Disadvantages

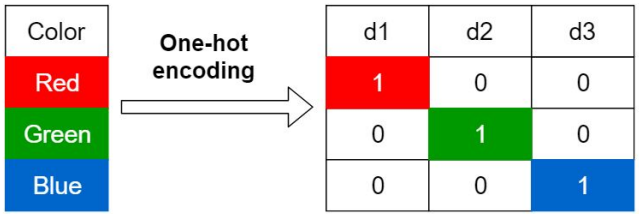
- Sensitive to outliers
- Must manually create dummy variables for categorical features before feeding into models
- Overfits if the model is not stopped early
- Can be less effective on smaller datasets
- Less effective on sparse and unstructured data
- Training time - high for large dataset

How does XGBoost compare?

- Uses the same model representation and inference as Random Forest
- An upgrade of gradient boosting, improving on bagging techniques
- It's trained models are decision trees
- A more regularized form of gradient boosting
- Depends on data/needs - other models can work better - accuracy may not be the only goal

Data Pre-Processing

- Requires the same level of normalization that most ML algorithms require
- Super sensitive to outliers
- Data must be structured as a table of feature vectors
  - Features can be numeric or categorical
  - Numeric features should be scaled (Z-scored)
  - Categorical features should be encoded using one hot encoding (dummy variables)



Hyperparameters (for tree learners)

Learning_rate / eta

- Affects how quickly the model fits the residual error using additional base learners
- A lower learning rate needs a higher number of boosting rounds

Num_boosting_rounds

- Number of estimators

Max_depth

Maximum number of decision points in a branch of a decision tree

Hyperparameters pt.2

Colsample_bytree (between 0 and 1)

- Percentage of features used per tree
- Smaller colsample causes higher regularization

Subsample

- Percentage of samples used per tree in a boosting round
- Be careful of overfitting or underfitting!

Hyperparameters pt.3

Regularization: Places a limit on model complexity, aiming to reduce variance, while limiting increase in bias.

- Gamma: minimum loss reduction allowed for a split to occur
  - Sets the threshold for gain needed to create new nodes in a tree
- Alpha: L1 regularization on leaf weights
  - L1 Penalty: absolute value of magnitude of coefficients (causes some to go to 0 and drop out)
  - Larger values mean more regularization
- Lambda: L2 regularization on leaf weights
  - L2 Penalty: square of the coefficients, will reduce all of them by a degree but not eliminate any
  - Causes leaf weights to smooth out

How can we tune them?

Two main methods:

- Grid Search: checks every combination of hyperparameters to find the combination with the best score.
  - Additional options and parameters cause grid searches to increase exponentially

- Random Search: user specifies how many random combinations of hyper parameter values to check.
  - Good for large hyperparameter spaces but can take a long time to find a good fit.