XGBoost
ML Algorithm

Gavan VanOver

Philip Waymeyer

Widnie Dorilas

Zheding Zhao

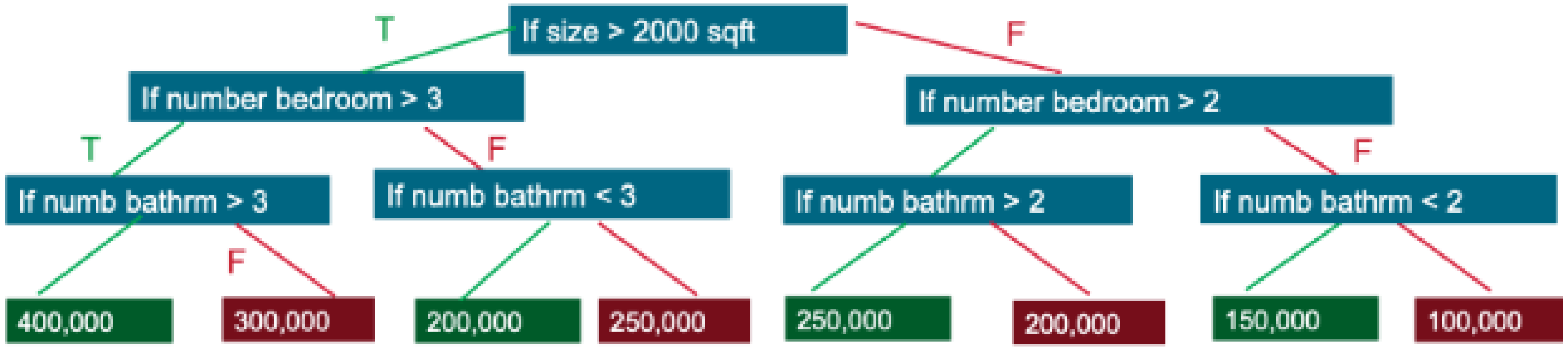
Intro to XGBoost

- Essentially an optimized Gradient Boosting machine learning library
    - Gradient Boosting is already an advanced variation on Decision Trees
- Open-Source
- Compatible with C++, Java, Python, Julia, Perl, and Scala
- Garnered notoriety around 2014 for consistently outperforming other algorithms in competitive environments such as Kaggle competitions

Intro to XGBoost

Basic Algorithm

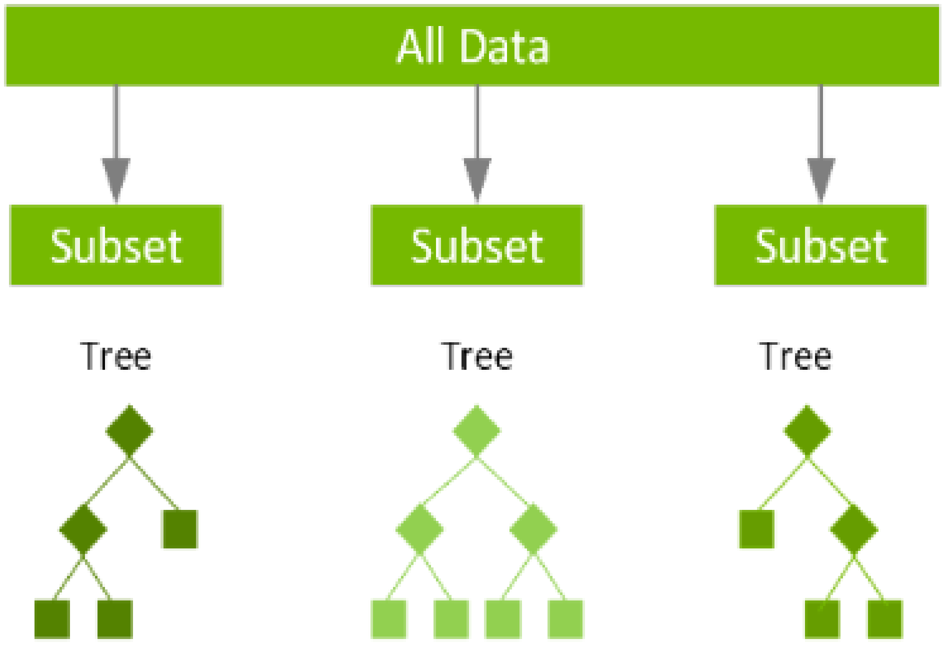
- Gradient Boost
  - Threshold Determination
  - Learning Rate
- Regularization
  - Lambda
  - Alpha
  - Gamma



Intro to XGBoost

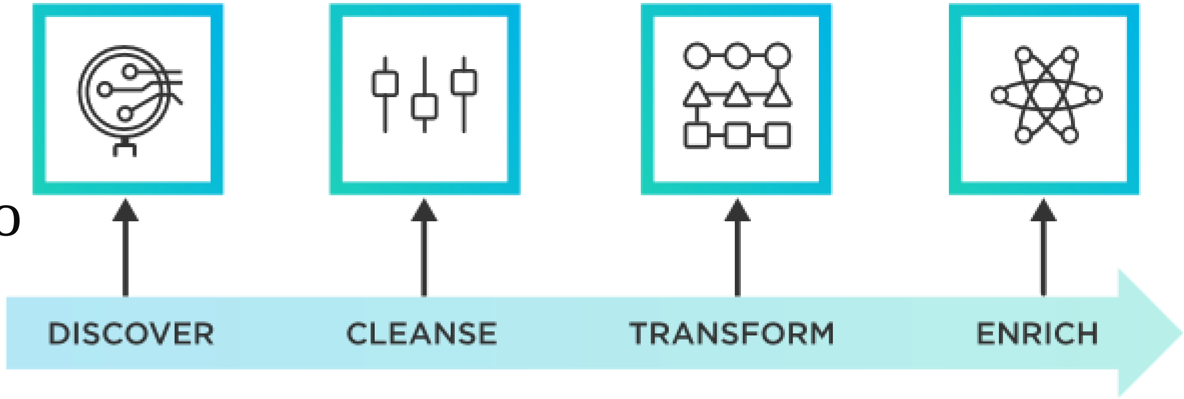
Handle of Dataset & Beyond Statistics

- Approximate Greedy Algorithm
- Sparsity-Aware Split Finding
- Parallel Learning
- Weighted Quantile Sketch
- Cache-Aware Access
- Blocks for Out-of-Core Computation



Data Processing for XGBoost

- First step is importing the data set you need
- Remove columns that will add noise
- XGBoost only manages numeric vectors
  - A way of changing categorical data to numeric data using *one-hot encoding AKA dummy contrast coding*
- *Null* values can be dealt with but don't have to be as XGBoost can internally handle missing values



XGBoost Hyperparameters - General Parameters

- Most of the general parameters are managed by the algorithm, and they do not need to be manually set.
- A few that a user may want to change for one reason or another include:
- booster[default=gbtree]
  - Can be 'gbtree', 'gblinear', or 'dart'.
- verbosity[default=1]
  - Can be 0 (silent), 1 (warning), 2 (info), 3 (debug).
- nthread[default to maximum number of threads available if not set]

XGBoost Hyperparameters - Booster Parameters

- XGBoost has two types of boosters: tree booster and linear booster.
- There are many booster parameters in XGBoost, so only a handful will be showcased.
- eta[default=0.3]
  - Range[0,1]
- gamma[default=0], lambda[default=1], alpha[default=0]
- max_depth[default=6]
- scale_pos_weight[default=1]
  - XGBoost recommends sum(negative instances) / sum(positive instances).
- subsample[deafult=1]

XGBoost Hyperparameters - Learning Task Parameters

- These specify the learning task and the objective.
- objective[default=reg:squarederror]
  - There are many values to use, such as reg:logistic, binary:logistic, binary:hinge, count:poisson, and multi:softmax.
- eval_metric[default according to objective]
  - Defaults include rmse for regression, error for classification, and mean average precision for ranking. Multiple evaluation metrics can be added.
- seed[default=0]

Advantages

- Works for both regression and classification
  - Rather than needing both linear and logistic regression
- Built-in ability to deal with missing values
- Nonparametric (no assumptions on data)
- Many hyperparameters to help the learning process
  - Regularization (prevent overfitting), learning rate, etc.
- Parallelization
  - Faster than other ensemble models (bagging or boosting)
  - Proficient memory usage

Disadvantages

- If not handled properly the model is likely to overfit
- More difficult to interpret
    - Much more complex than linear or even logistic regression
- Extremely sensitive to outliers
- Doesn't perform well on unstructured data

Questions?