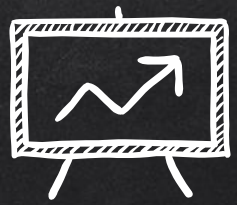


Tree Based Models

- Emma

Quiz

- ✘ What is Gini impurity?
- ✘ What is entropy?
- ✘ What is the difference and similarity between Random Forest and XGBoost/GBM?
- ✘ What are the key parameters to tune in training a tree based model?

Topics

- ✘ Decision Tree
- ✘ Bagging and Random Forest
- ✘ Boosting Tree
- ✘ Hyper parameter tuning

Decision tree

Types of Decision Tree

"Decision tree is a type of supervised learning algorithm, which is mostly used in classification problems. It works for both categorical and continuous input and output variables. In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) "

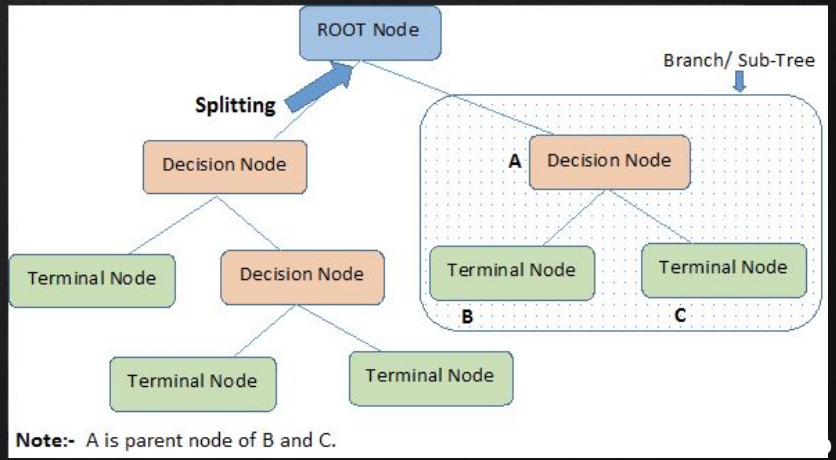
- ✘ Classification Trees: Decision Tree which has categorical target variable then it called as categorical variable decision tree.
- ✘ Regression Trees: Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.

Therefore, Decision Tree also called Classification tree And Regression Tree(CART)

https://en.wikipedia.org/wiki/Decision_tree_learning

Important Definitions related to Decision Tree

- **Root Node:** It represents entire population or sample and this further gets divided into two or more homogeneous sets.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
- **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
- **Branch / Sub-Tree:** A sub -section of entire tree is called branch or sub-tree.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

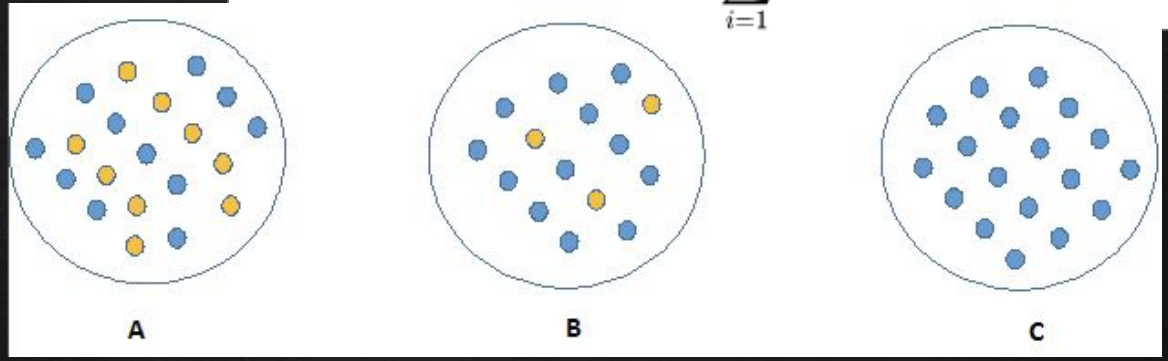


Regression Tree VS Classification Tree

1. Target
2. Prediction
3. Both the trees divide the predictor space (independent variables) into distinct and non-overlapping regions.
4. Algorithm – Both the trees follow a top-down greedy approach known as recursive binary splitting.
5. This splitting process is continued until a user defined stopping criteria is reached.
6. Pruning is used to prevent overfitting

Basics Algorithms to Split(grow) Tree

- ✘ Purity Algorithms
  - ρ_i = probability of item being labeled in group i
  - Gini Impurity = $\mathrm{I}_G(p) = \sum_{i=1}^{J}\left(p_i \sum_{k \neq i} p_k\right) = \sum_{i=1}^{J} p_i(1-p_i) = \sum_{i=1}^{J}(p_i - p_i^2) = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} p_i^2 = 1 - \sum_{i=1}^{J} p_i^2$
  - Entropy = $\mathrm{H}(T) = \mathrm{I}_E(p_1, p_2, \ldots, p_J) = -\sum_{i=1}^{J} p_i \log_2 p_i$

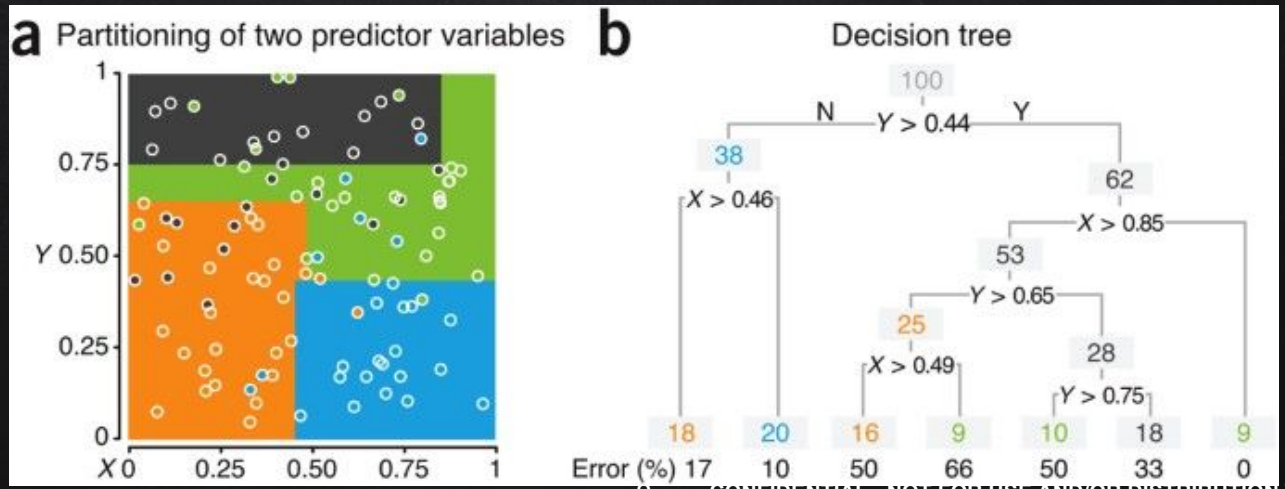


- ✘ Chi-square
- ✘ Reduction in Variance (MSE/RMSE)

https://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity

Recursive Binary Partition

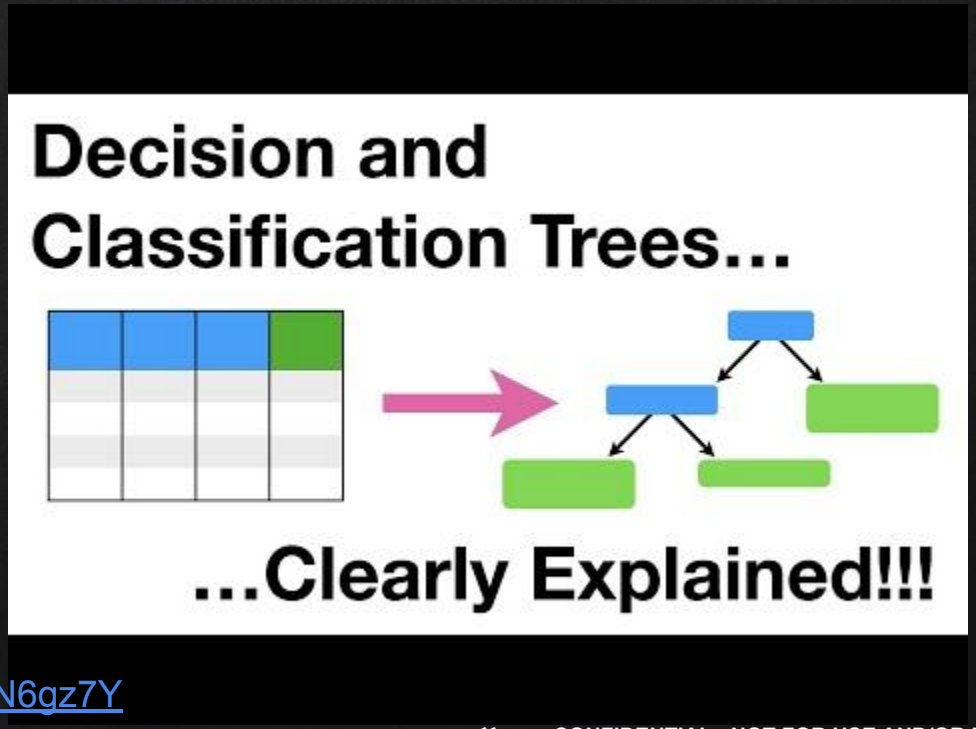
1. Initialize A with the entire data set.
2. Apply the partitioning algorithm to split A into two subpartitions, A 1 and A 2 . (Basic algorithms embedded here for best splitting)
3. Repeat step 2 on subpartitions A 1 and A 2 .
4. The algorithm terminates when no further partition can be made that sufficiently improves the homogeneity of the partitions.



Parameters Control Tree Grow

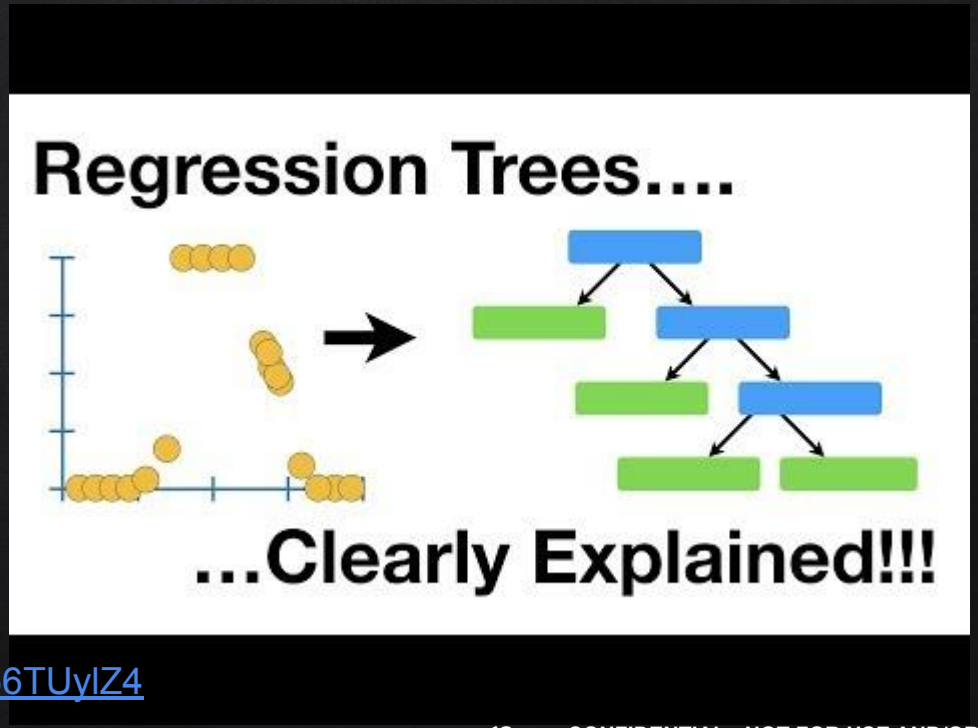
- Minimum samples for a node split
- Minimum samples for a terminal node (leaf)
- Maximum depth of tree (vertical depth)
- Maximum number of terminal nodes
- Maximum features to consider for split

Classification Tree



https://youtu.be/_L39rN6gz7Y

Regression Tree



https://youtu.be/g9c66TUylZ4

Bagging and Random Forest

Key Definitions

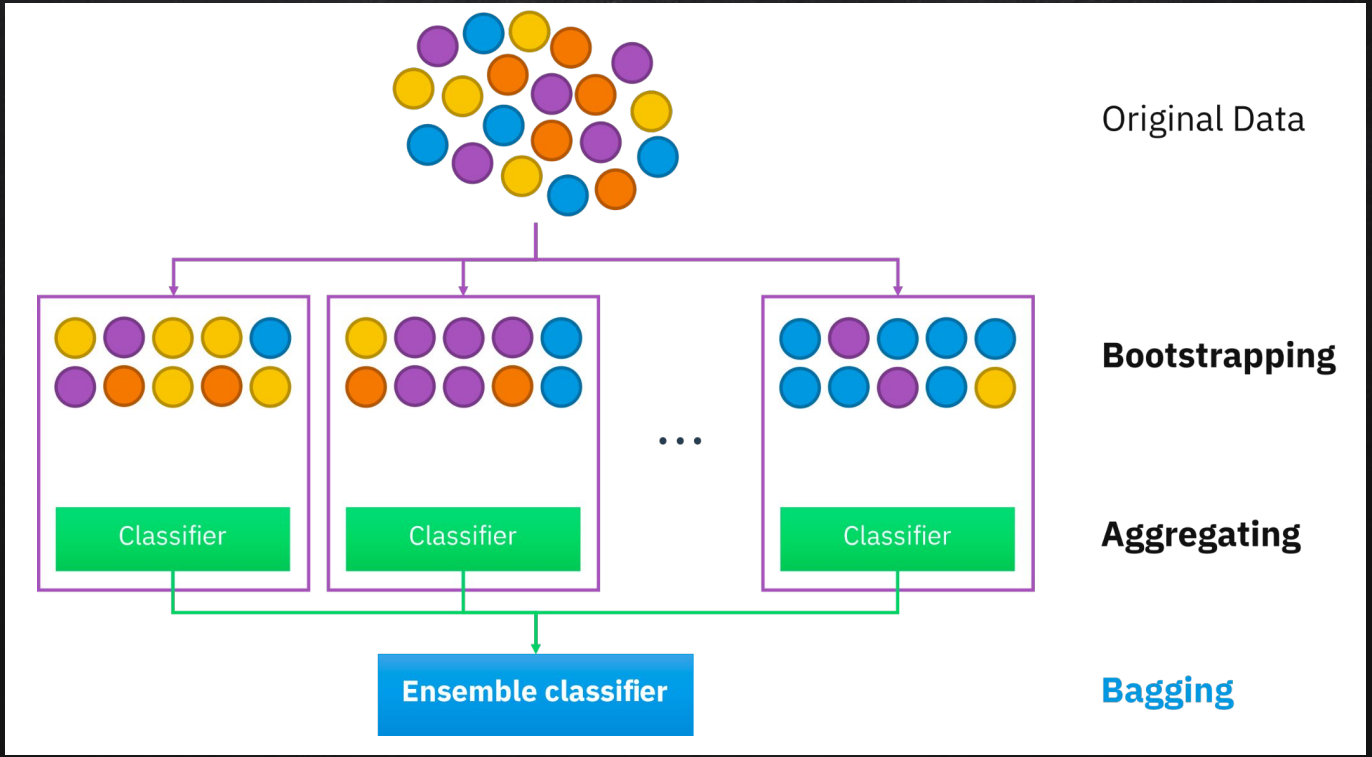
Ensemble "Forming a prediction by using a collection of models. "

Bagging "A general technique to form a collection of models by bootstrapping the data." Also called bootstrap aggregation

Random forest "A type of bagged estimate based on decision tree models. "

Variable importance "A measure of the importance of a predictor variable in the performance of the model. "

Bagging



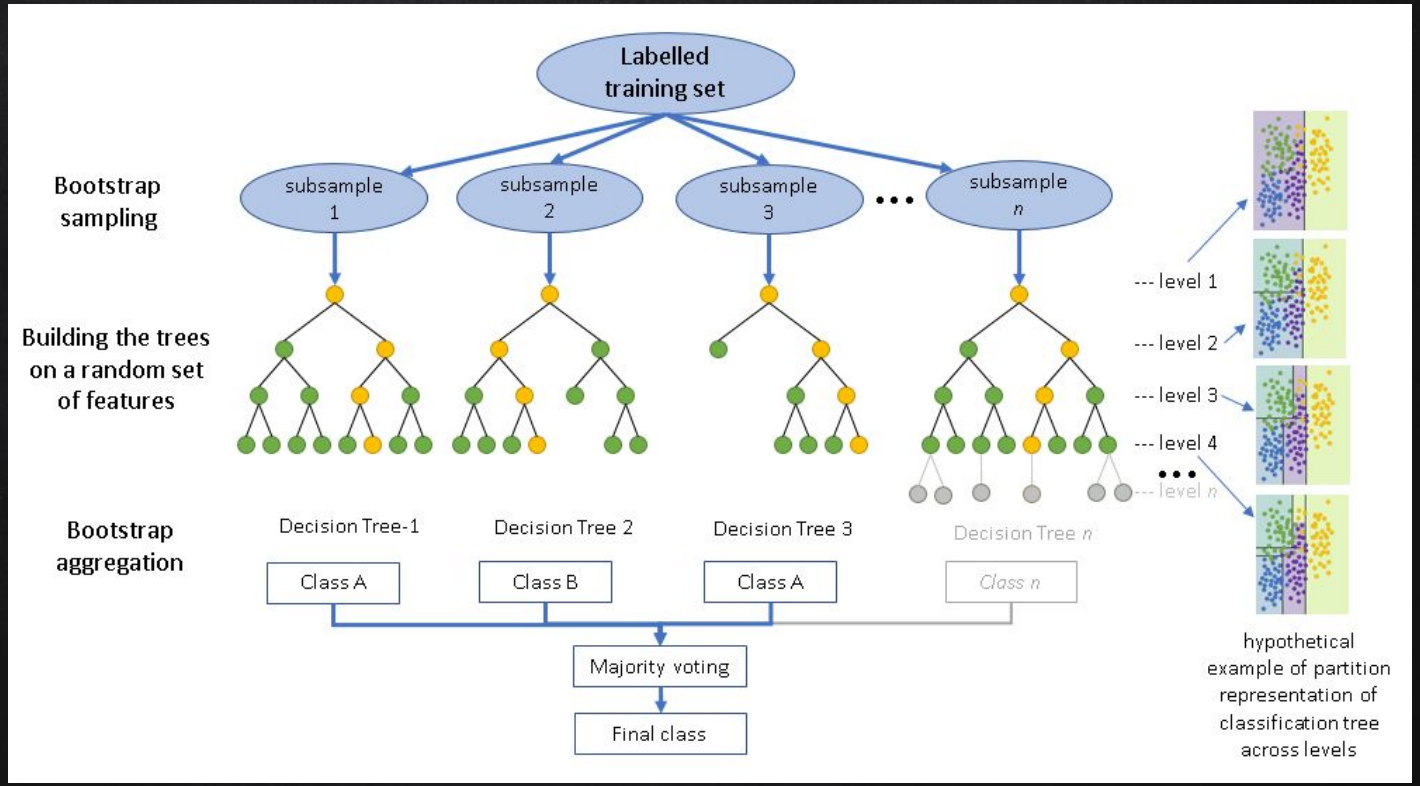
https://en.wikipedia.org/wiki/Bootstrap_aggregating

Bagging Algorithm

Bagging is like the basic algorithm for ensembles, except that, instead of fitting the various models to the same data, each new model is fitted to a bootstrap resample.

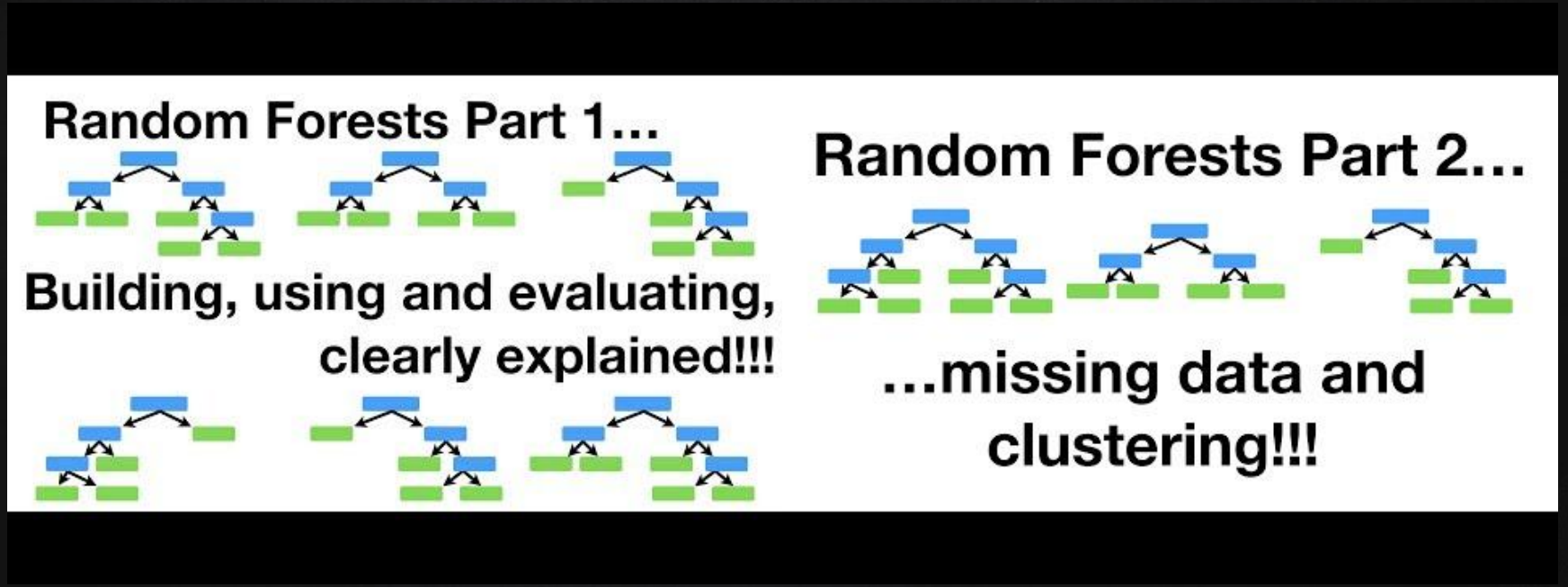
1. Initialize M , the number of models to be fit, and n , the number of records to choose (n < N). Set the iteration .
2. Take a bootstrap resample (i.e., with replacement) of n records from the training data to form a subsample and (the bag).
3. Train a model using and to create a set of decision rules .
4. Increment the model counter . If m <= M , go to step 2.
5. Sum average all f(x) (M trees)

Random Forest



https://catalyst.earth/catalyst-system-files/help/concepts/focus_c/oa_classif_intro_rt.html

Random Forest



https://youtu.be/J4Wdy0Wc_xQ

https://youtu.be/sQ870aTKqiM

Random Forest

This is an advanced algorithm add the bootstrap predictors for each tree split.

Major Tuning Parameters:

- ✓ Number of trees build
- ✓ Sample size of Observation
- ✓ Sample size of inputs
- ✓ Minimum samples for a node split
- ✓ Minimum samples for a terminal node (leaf)
- ✓ Maximum depth of tree (vertical depth)
- ✓ Maximum number of terminal nodes
- ✓ Maximum features to consider for split

Boosting Tree

Boosting Trees

Boosting trees, are a series of models, in which each successive model seeks to minimize the error of the previous model.

Several isoforms of boosting trees:

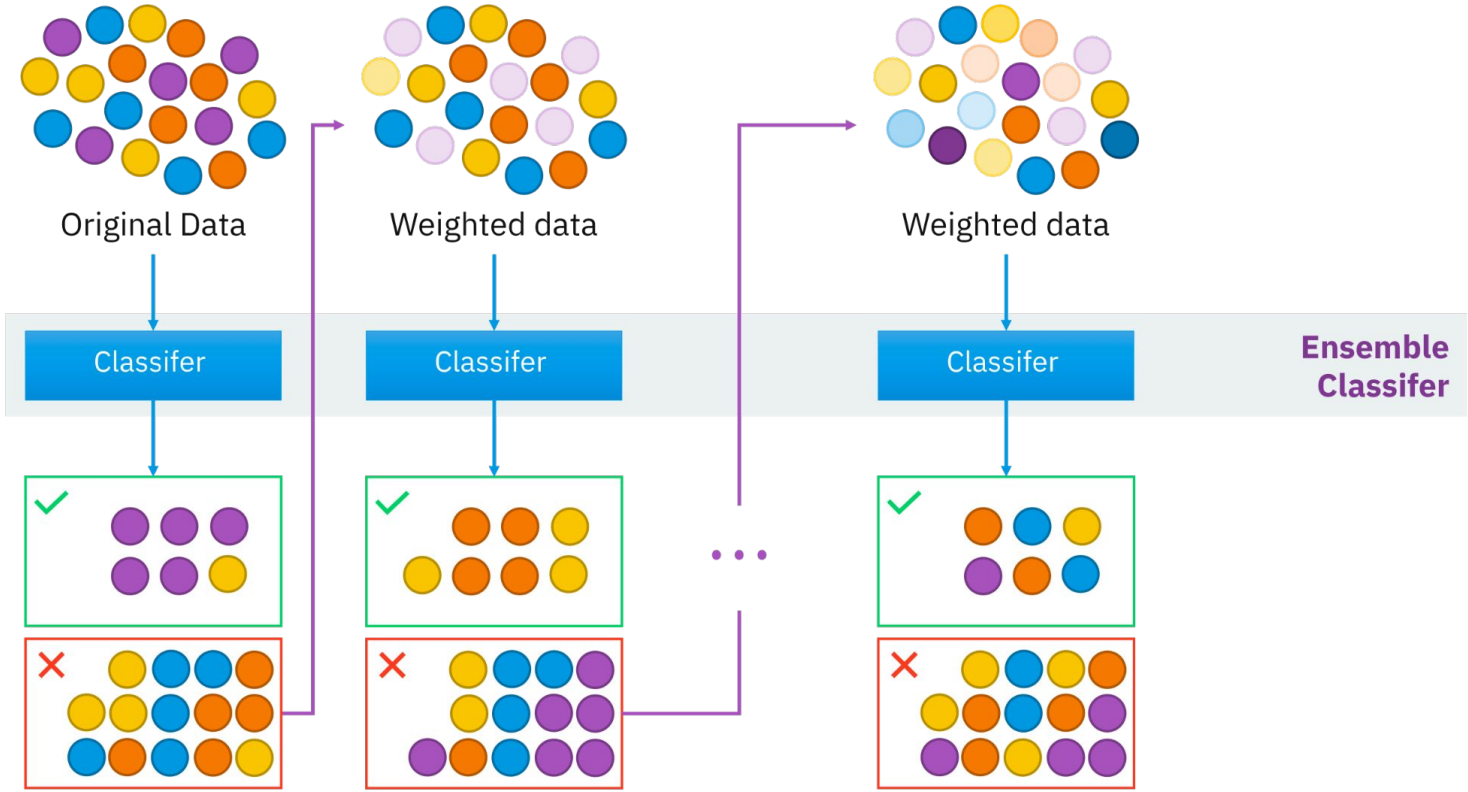
- Adaboost
- Gradient boosting
- Stochastic gradient boosting (XGB)

Normally boosting tree need more intensive hyperparameter tuning

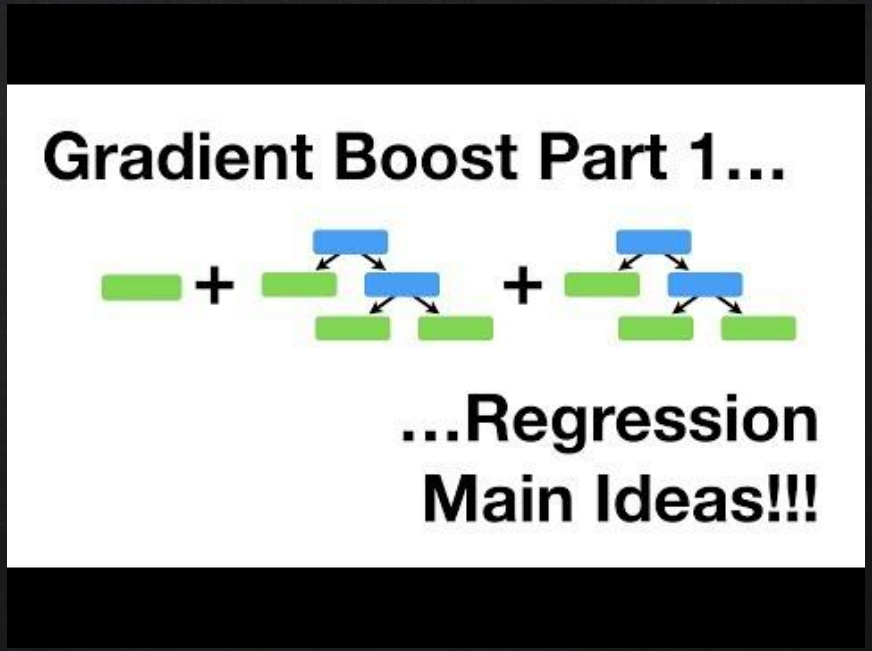
Comparison of different boosting methods:

https://neptune.ai/blog/gradient-boosted-decision-trees-guide

Boosting



Gradient Boosting



https://youtu.be/3CC4N4z3GJc (there are **4 episodes** for gradient boosting regression and classification

Advantage of XGB

- Regularization
- Parallel Processing
- High Flexibility
- Handling Missing Values
- Tree Pruning
- Built-in Cross-Validation
- Continue on Existing Model

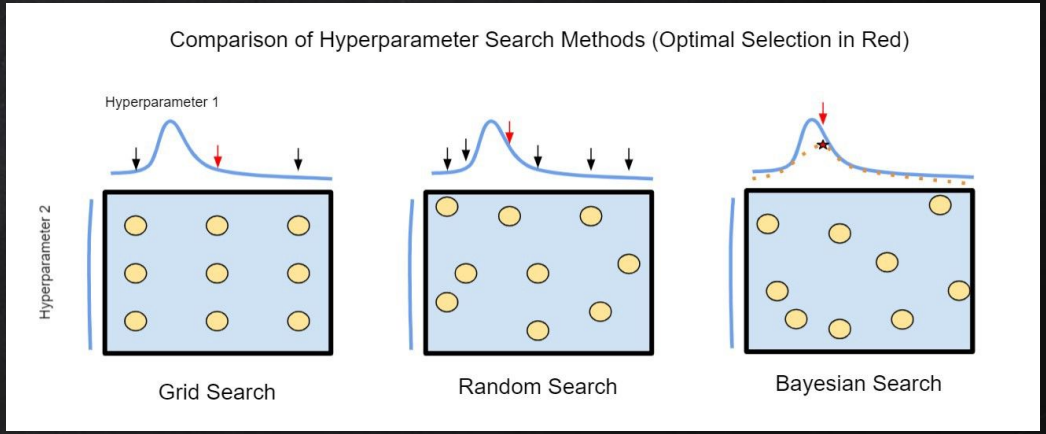
XGB Hyper-Parameters Example

- eta – Analogous to learning rate in GBM
- min_child_weight
- max_depth [default=6]
- Max_leaf_nodes
- gamma [default=0]
- max_delta_step [default=0]
- subsample [default=1]
- colsample_bytree [default=1]
- colsample_bylevel [default=1]
- lambda [default=1]
- alpha [default=0]
- scale_pos_weight [default=1] – A value greater than 0 should be used in case of high class imbalance as it helps in faster convergence.

Name	Default	Range	Effect	Notes/Tips
n_estimators	100	[1, inf)	Increasing may improve scores with large data.	The number of trees in the ensemble.
learning_rate **alias:eta**	0.3	[0, inf)	Decreasing prevents overfitting.	Shrinks the tree weights in each round of boosting.
max_depth	6	[0, inf)	Decreasing prevents overfitting.	The depth of the tree. 0 is an option in a loss-guided growing policy.
gamma **alias: min_split_loss**	0	[0, inf)	Increasing prevents overfitting.	Low values, usually lower than 10, are standard.
min_child_weight	1	[0, inf)	Increasing prevents overfitting.	The minimum sum of weights required for a node to split.
subsample	1	(0, 1]	Decreasing prevents overfitting.	Limits the percentage of training rows for each boosting round.
colsample_bytree	1	(0, 1]	Decreasing prevents overfitting.	Limits the percentage of training columns for each boosting round.
colsample_bylevel	1	(0, 1]	Decreasing prevents overfitting.	Limits the percentage of columns for each depth level of the tree.
colsample_bynode	1	(0, 1]	Decreasing prevents overfitting.	Limits the percentage of columns to evaluate splits.
scale_pos_weight	1	(0, inf)	Sum(negatives)/ Sum(positives) balances data.	Used for imbalanced datasets. See *Chapter 5*, *XGBoost Unveiled*, and *Chapter 7*, *Discovering Exoplanets with XGBoost*.
max_delta_step	0	[0, inf)	Increasing prevents overfitting.	Only recommended for extremely imbalanced datasets.
lambda	1	[0, inf)	Increasing prevents overfitting.	L2 regularization of weights.
alpha	0	[0, inf)	Increasing prevents overfitting.	L1 regularization of weights.
missing	None	(-inf, inf)	Finds optimal null values.	Replace null values with numerical value like -999.0, then set equal to -999.0. See *Chapter 5*, *XGBoost Unveiled*.

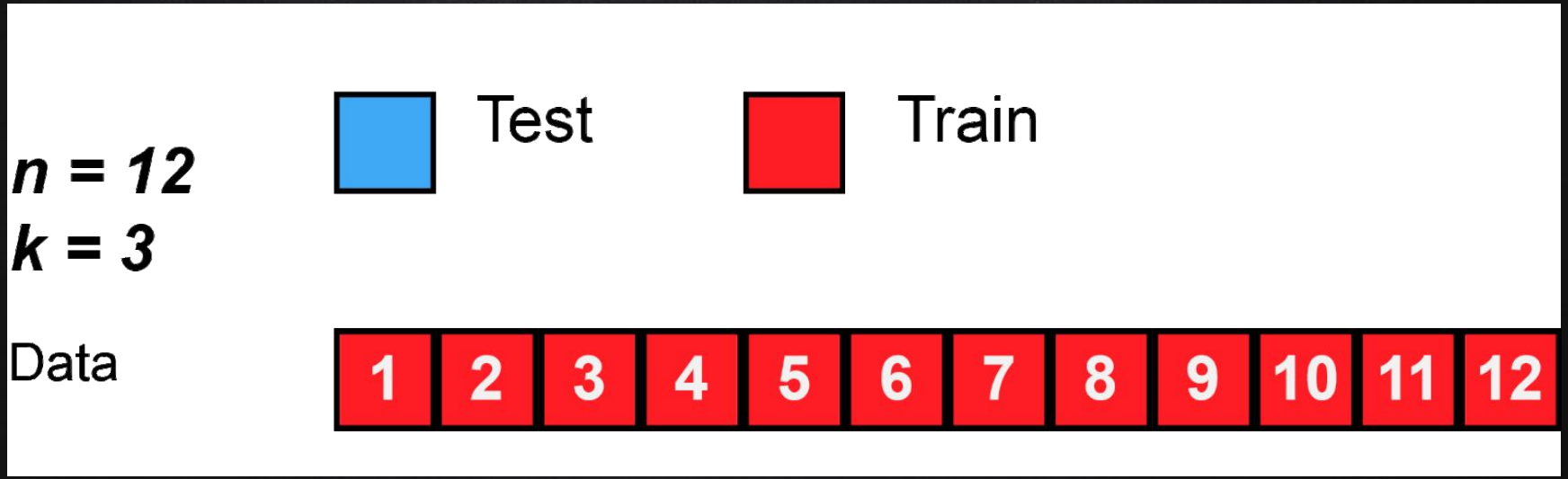
Hyper Parameter Tuning

- ✘ Cross validation
- ✘ Coarse tune -> fine tune
- ✘ Hyperparameter search
  - Grid search
  - Random search
  - Bayes search
- ✘ Early stopping

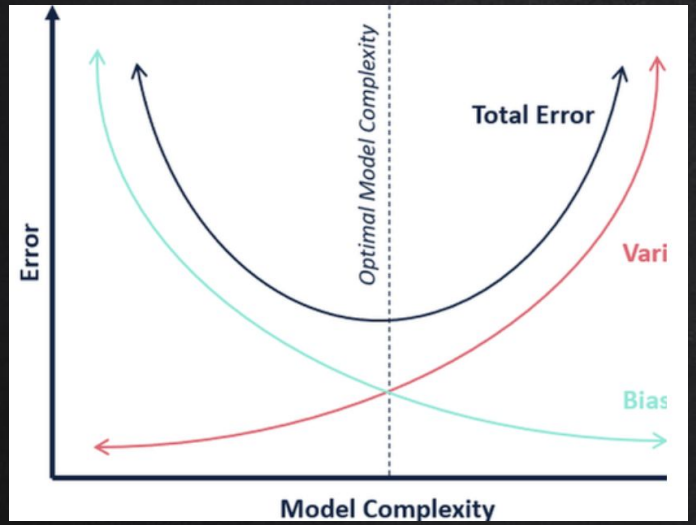


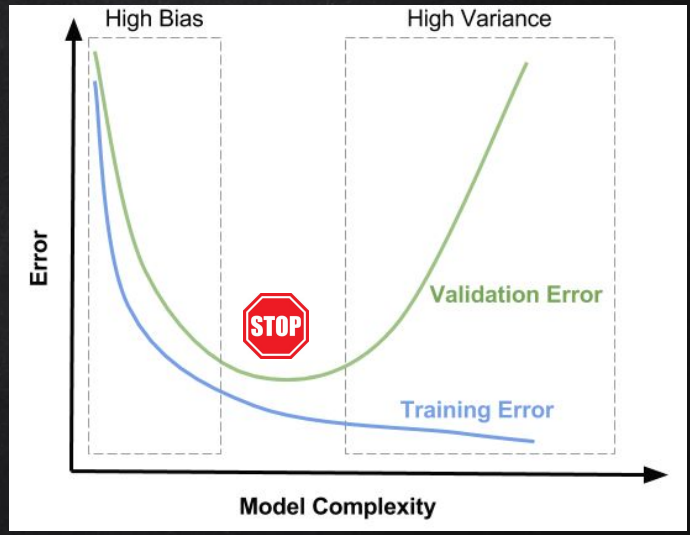
https://scikit-learn.org/stable/modules/grid_search.html
https://towardsdatascience.com/hyperparameter-tuning-always-tune-your-models-7db7aeaf47e9

Cross Validation



Bias-Variance trade-off





1) High Bias – underfitting
2) Goldilocks Zone – just right
3) High Variance – overfitting

Recommended Readings

https://online.stat.psu.edu/stat555/node/100/

https://www.slideshare.net/ShangxuanZhang/xgboost

https://towardsdatascience.com/machine-learning-part-18-boosting-algorithms-gradient-boosting-in-python-ef5ae6965be4

https://towardsdatascience.com/decision-tree-in-python-b433ae57fb93

https://medium.com/@rishabhjain_22692/decision-trees-it-begins-here-93ff54ef134#:~:text=Chi%2DSquare,Success%E2%80%9D%20or%20%E2%80%9CFailure%E2%80%9D.

Package of Trees detail parameters

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

https://xgboost.readthedocs.io/en/latest/parameter.html

References

Bruce, Peter,Bruce, Andrew,Gedeck, Peter. Practical Statistics for Data Scientists (Kindle Location 5267). O'Reilly Media. Kindle Edition.

https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/