Gradient boosting Hyper-parameters

Math behind Gradient Boosting: https://explained.ai/gradient-boosting/

	Parameter	Default Values	Description	Impact
Boosting Parameters	learning_rate	0.1	the effect of each tree on the outcome is shrunk by this factor.	> lower always preferred > inversely proportional to n_estimators > use high value for tuning and lower for final submissions
	n_estimators	100	the number of trees to fit sequentially	> tune using CV for a given learning rate > higher value for low learning rate but computationally expensive
	subsample	1	the fraction of observations to be used in individual tree	> typical value 0.8 > reduces variance in model > tune using CV
Tree-Specific Parameters	min_samples_split	2	the minimum number of observations required to split an internal node	> higher values prevent overfitting > risk of underfitting with too high values > tune using CV > ~0.5-2% of total observations
	min_samples_leaf	1	the minimum number of observations required in a terminal node for a split to be valid	> higher values prevent overfitting > risk of underfitting with too high values > tune using CV

	min_weight_fraction_leaf	0	same as min_samples_leaf but defined as a ratio of total number of observations	> higher values prevent overfitting > risk of underfitting with too high values > tune using CV
	max_depth	None	the maximum depth of each tree. None specified no limit on depth.	> lower values prevent overfitting > risk of underfitting with too low values > tune using CV > typical 5-20
	max_leaf_nodes	None	the maximum number of terminal nodes in each tree. If None then no limit	> lower values prevent overfitting > risk of underfitting with too low values > tune using CV
	max_features	None	the number of features to consider for each split	> typical sqrt to 30-40% of total features > possible values: - Int: actual number of features - Float: the fraction of total number of features - 'auto' or 'sqrt': square root of total features - 'log2': log to base 2 of total features - None: equal to total number of features
Miscellaneous Parameters	loss	deviance (class) ls (regression)	the cost function to be minimized by optimization	> use default value if not sure > options: - class: deviance / exponential - regr: ls / lad / huber / quantile
	init	None	estimator for getting the initial predictions	> BaseEstimator can be any class object with fit and predict functions
	random_state	None	set the seed of random number generator	> use fixed value for parameter tuning > try different values only if possible computationally

	verbose	0	select the type of output to be generated by model fit	>options: - 0: No output - 1: output for few trees (selected automatically) - >1: output all trees
	warm_start	False	if True, it will start from the previous call to fit and add more trees to it	> should be used judicially for advanced applications
	presort	'auto'	whether to presort data to speed up the finding of best splits	> mostly 'auto' works fine

XGBoost Hyper-parameters

	Parameter	Default Values	Decription & Impact
Boosting Parameters	eta/learning_rate	0.3	> Makes the model more robust by shrinking the weights on each step > Typical final values to be used: 0.01-0.2 (0.015,0.025,0.01,0.05,0.1..)
	min_child_weight	1	> Used to control over-fitting. Higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree. > Too high values can lead to under-fitting hence, it should be tuned using CV. > Typical Value Range - (1, 3, 5, 7....15)
	max_depth	6	> The maximum depth of a tree, same as GBM. > Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample. > Should be tuned using CV.Typical values: 3-10.
	gamma/min_split_loss	0	> Minimum loss reduction required to make a further partition on a leaf node of the tree. > The larger gamma is, the more conservative the algorithm will be. > Typical range: 0.05,0.1,0.3,0.5,0.7,0.9.1
	subsample	1	> Lower values make the algorithm more conservative and prevents overfitting but too small values might lead to under-fitting. > Typical values: 0.5-1
	colsample_bytree	1	> Similar to max_features in GBM. Denotes the fraction of columns to be randomly samples for each tree. > Typical values: 0.5-1
	lambda/reg_lambda	1	> L2 regularization term on weights. Increasing this value will make model more conservative. > Typical range: 0.01-0.1/ 1.0
	alpha/reg_alpha	0	> L1 regularization term on weights. Increasing this value will make model more conservative. > Typical range: 0.01-0.1/ 1.0

	scale_pos_weight	1	> Control the balance of positive and negative weights, useful for unbalanced classes. > A typical value to consider: sum(negative instances) / sum(positive instances).