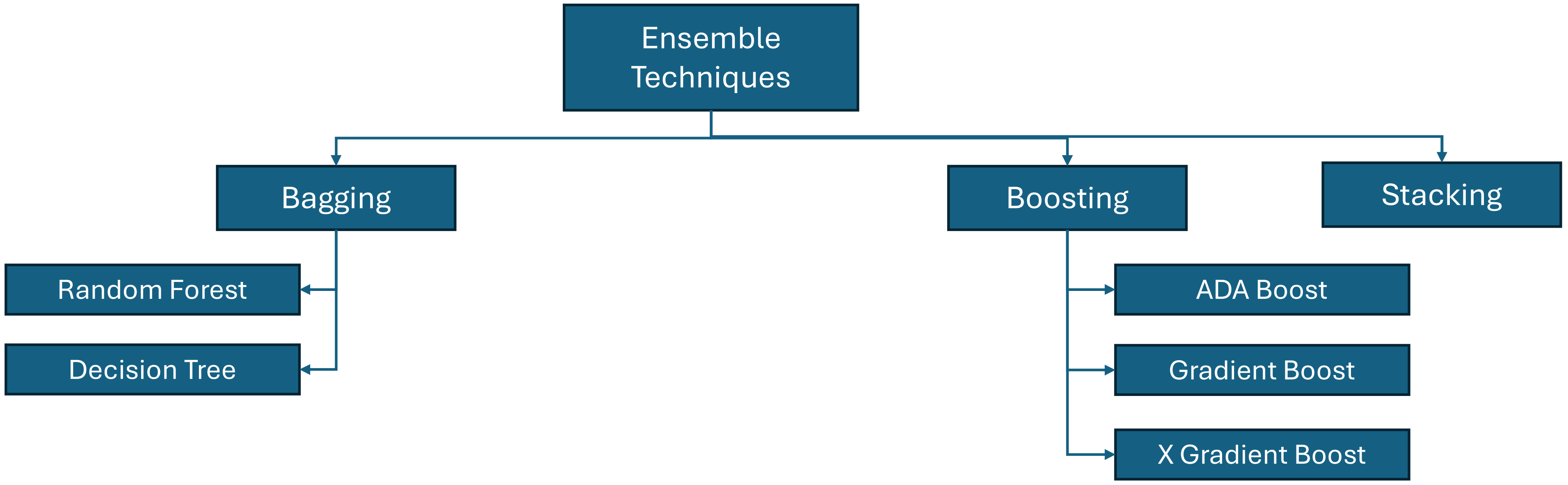
Ensemble Learning



Ensemble Techniques
Ensemble learning combines the predictions of multiple models like "weak learners" or "base models" to make a stronger, more reliable prediction. The goal is to reduce errors and improve performance.

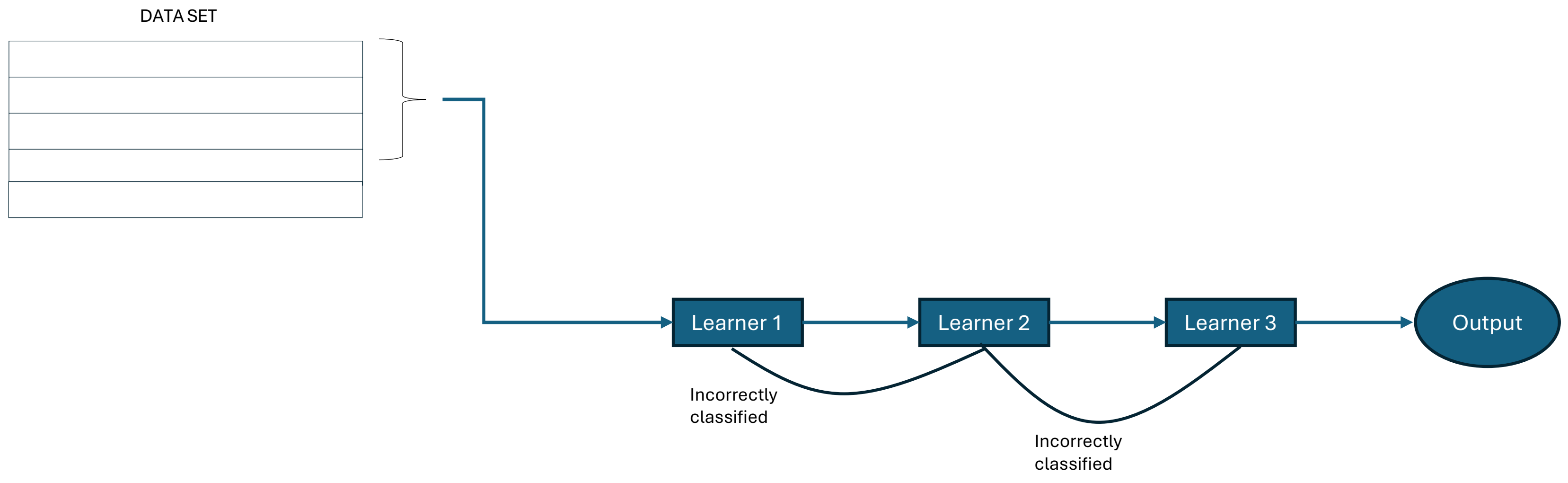
Algorithm’s Definition

Bagging Algorithm
Bagging models are trained independently on different subsets of the data. Their results are averaged or voted on.

Boosting Algorithm
Boosting algorithms are trained sequentially, with each one learning from the mistakes of the previous model. This will create a stronger, more accurate predictive model, focusing on improving predictions iteratively by addressing errors from previous models.

Stacking Algorithm
Stacking is a heterogenous parallel method that exemplifies what is known as meta-learning. Meta-learning consists of training a meta-learner from the output of multiple base learners. Stacking specifically trains several base learners from the same dataset using a different training algorithm for each learner. Each base learner makes predictions on an unseen dataset.

How ADA Boosting Works



Why Boosting
Boosting algorithm are used when the data is more complex or random

How ADA Boosting works
Portion of the dataset is fed into the base learner node. Here Learner 1.
From that node, incorrectly classified or identified data with higher weightage is fed into the next or sequential node.
Every node process the incorrect data and finally, output will be corrected or predicted data will come

ADA Boosting Steps

Step 1: When the algorithm is given data, it starts by Assigning equal weights to all training examples in the dataset. These weights represent the importance of each sample during the training process.

Step 2: In this step, Algorithm iterates with a few algorithms for a specified number of iterations or until a stopping criterion is met. The algorithm trains a weak classifier on the training data. Here the weak classifier can be considered a model that performs slightly better than random guessing, such as a decision stump (a one-level decision tree.)

Step 3: During each iteration, the algorithm trains the weak classifier on given training data with the current sample weights. The weak classifier aims to minimize the classification error, weighted by the sample weights.

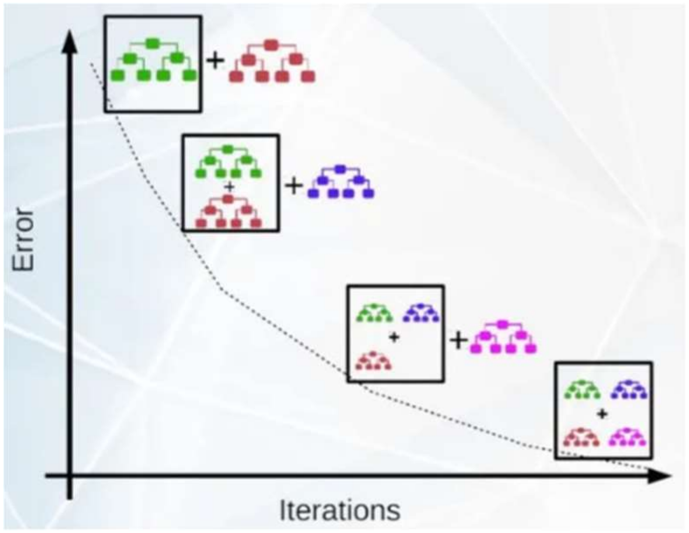
Step 4: After training the weak classifier, the algorithm calculates classifier weight based on the errors of the weak classifier. A weak classifier with a lower error receives a higher weight.

Step 5: Once the calculation of weight completes, the algorithm updates sample weights, and the algorithm gives assigns higher weights to misclassified examples so that more importance in subsequent iterations can be given.

Step 6: After updating the sample weights, they are normalized so that they sum up to 1 and Combine the predictions of all weak classifiers using a weighted majority vote. The weights of the weak classifiers are considered when making the final prediction.

Step 7: Finally, Steps 2–5 are repeated for the specified number of iterations (or until the stopping criterion is met), with the sample weights updated at each iteration. The final prediction is obtained by aggregating the predictions of all weak classifiers based on their weights.

Gradient Boosting Algorithm



Gradient Boosting
Gradient boosting is one of the ensemble methods which create multiple weak models and combine them to get better performance.

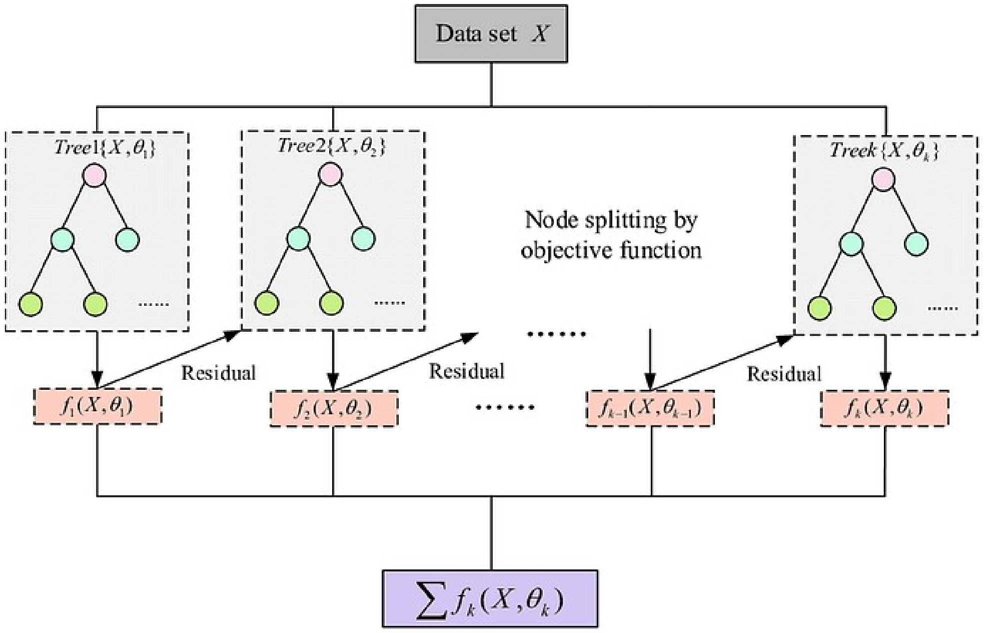
How Gradient Boosting works
Gradient boosting is an iterative optimization algorithm, used to minimize a given loss function or cost function.
In ML, we use this method to find the optimal values that minimize the difference between the predicted and the actual values.
In Gradient, each iteration new model is trained to minimize the loss function by adjusting its parameters using gradient descent.

Gradient Boosting Steps

To understand the gradient boosting, using the example dataset input feature (x) and continuous target variable (y)

Step 1: Algorithm will start with training a base model(usually a decision tree). This model gives predictions for the entire dataset.
Step 2: After making predictions, using the original values, it calculates the errors. These errors can be thought of as the parts of y that are still unexplained by the base model.
Step 3: Before applying the next sequenced model, the algorithm applies the gradient descent method to calculate the optimized parameters of the next model.
Step 4: Train a new model, and This model is fitted to minimize the difference between the predicted residuals and the actual residuals
Step 5: After training a new model, this algorithm calculates the new residuals by subtracting the updated predictions from the actual values of y. These residuals represent the remaining unexplained variability in the target variable.
Step 6: This whole process gets repeated iteratively, adding the new models to the ensemble while applying gradient descent on it and updating the predictions and residuals.
Step 7: At the final stage, the method combines the predictions from all the models where we consider the final predictions refined estimation of y.

XG Boosting or Extreme Gradient Boosting Algorithm



Gradient Boosting

XG boosting is similar to Gradient boosting, but with additional functionality. It has distributed computing or parallel process. It has more computational speed, model efficiency and cache optimization.

Gradient boosting is slow as it sequential when compared to XG boosting process.

XG Boosting Steps

Step 1: Initialization

Base Model: Usually, the model starts with a simple prediction for all instances, like the mean of the target variable.

In this step, we would have the initial prediction model that serves as the starting point, and the algorithm will iteratively improve upon it.

Step 2 : Iterative improvement

Adding Weak Learners Sequentially: The core idea is to add weak learners sequentially. A weak learner is a model that is slightly better than random guessing.

Residuals/Error Calculation: After each tree is added, the algorithm calculates the residuals or errors (difference between the predicted and actual values).

Step 3: Gradient Descent

Negative gradient computation: In each iteration, the algorithm computes the negative gradient of the loss function with respect to the prediction. This gradient is the steepest descent, showing how the prediction should be changed to reduce the loss.

Fit New Model to Gradient: A new weak learner (tree) is then fitted to these gradients.

Step 4: Update Model with Learning Rate

Apply Learning Rate: The predictions of the new tree are scaled by a parameter known as the learning rate

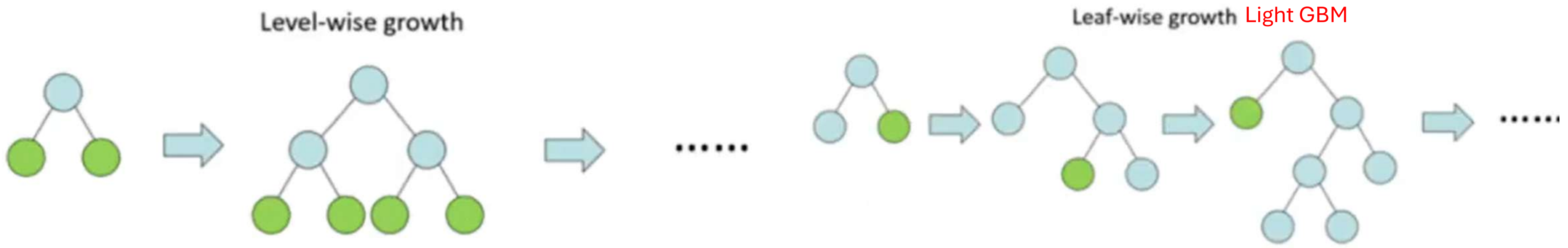
Update the Model: The scaled predictions are added to the previous predictions to update the model. This learning rate controls how fast the model learns and helps in preventing overfitting.

XG Boosting Steps Continuation…

Step 5: Regularization
Various methods are used to regularize the model, like setting a maximum depth for trees, minimum samples for a leaf, and the number of trees or iterations
Step 6: Stopping Criteria
Determine when to stop: The algorithm stops adding trees either after a certain number of trees are added or when the improvement drops below a threshold.

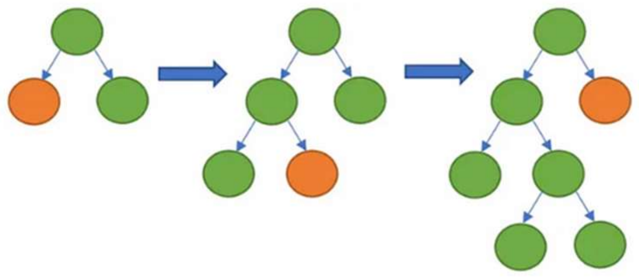
After the final step, we would get the optimized, predicted model.

Light Gradient Boosting Method (GBM) Algorithm



Light GBM Boosting
Light GBM adopts a leaf-wise growth strategy, as opposed to tradition level-wise approach.
It is fast, distributed, high performance gradient boosting framework based on decision tree.
It splits the tree leaf wise. It handles huge amount of data easily and poor in small dataset.

How Light GBM works



- The tree grows by adding a leaf that provides the maximum gain at each expansion step. In other words, only one leaf node is added at each expansion step.
- This usually results in deeper but narrower trees. Deeper trees can offer more flexibility to capture complex feature relationships.
- It is generally faster compared to the level-wise approach.

LightGBM's leaf-wise strategy focuses on expanding the tree by adding leaves that provide the maximum gain, offering advantages such as increased depth for capturing intricate patterns and faster processing.