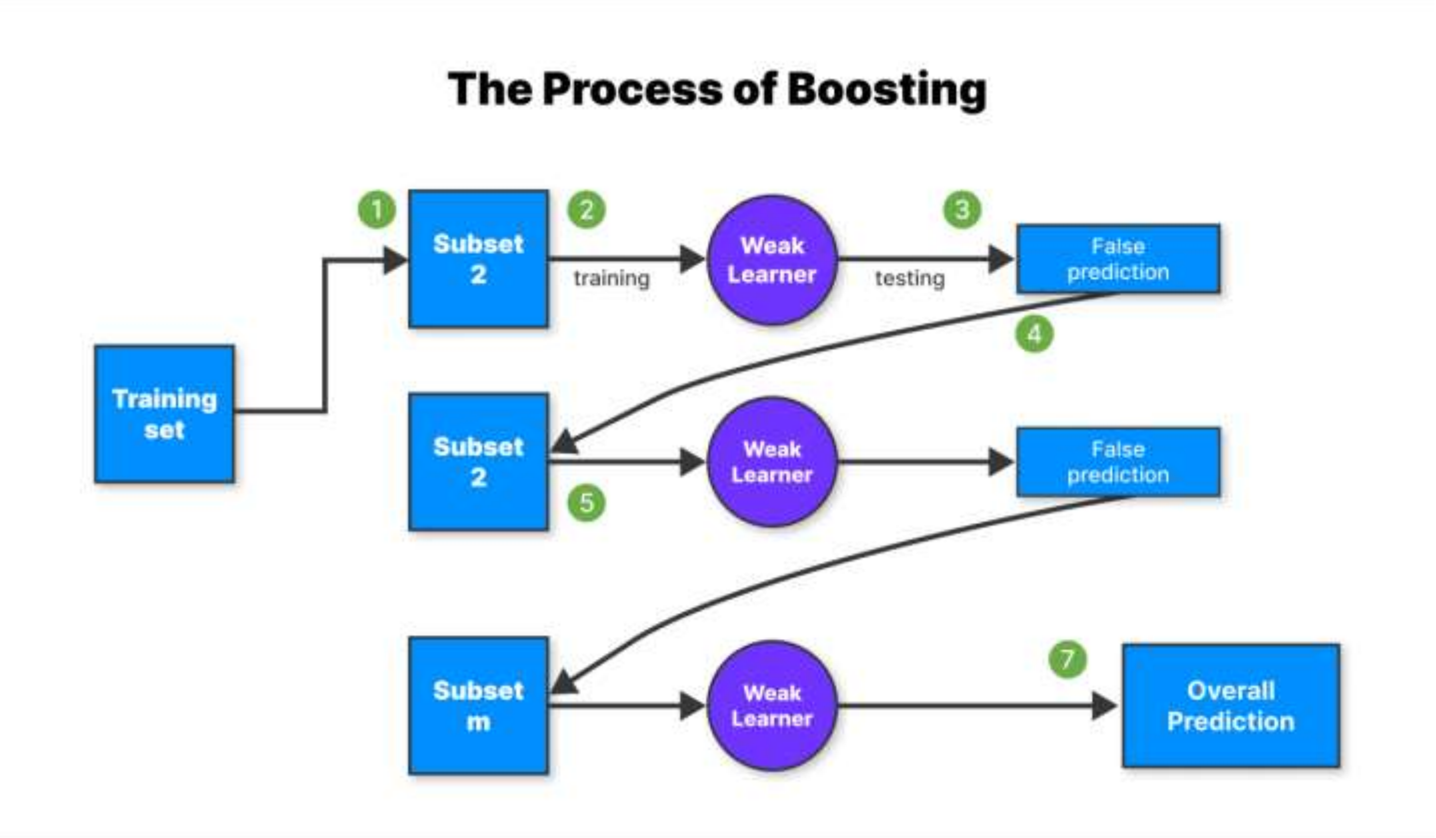
Boosting Algorithm

ENSEMBLE TECHNIQUES



Boosting *combines the weak learners to form a strong learner*.

Training models sequentially, each one trying to correct the errors made by the previous models.

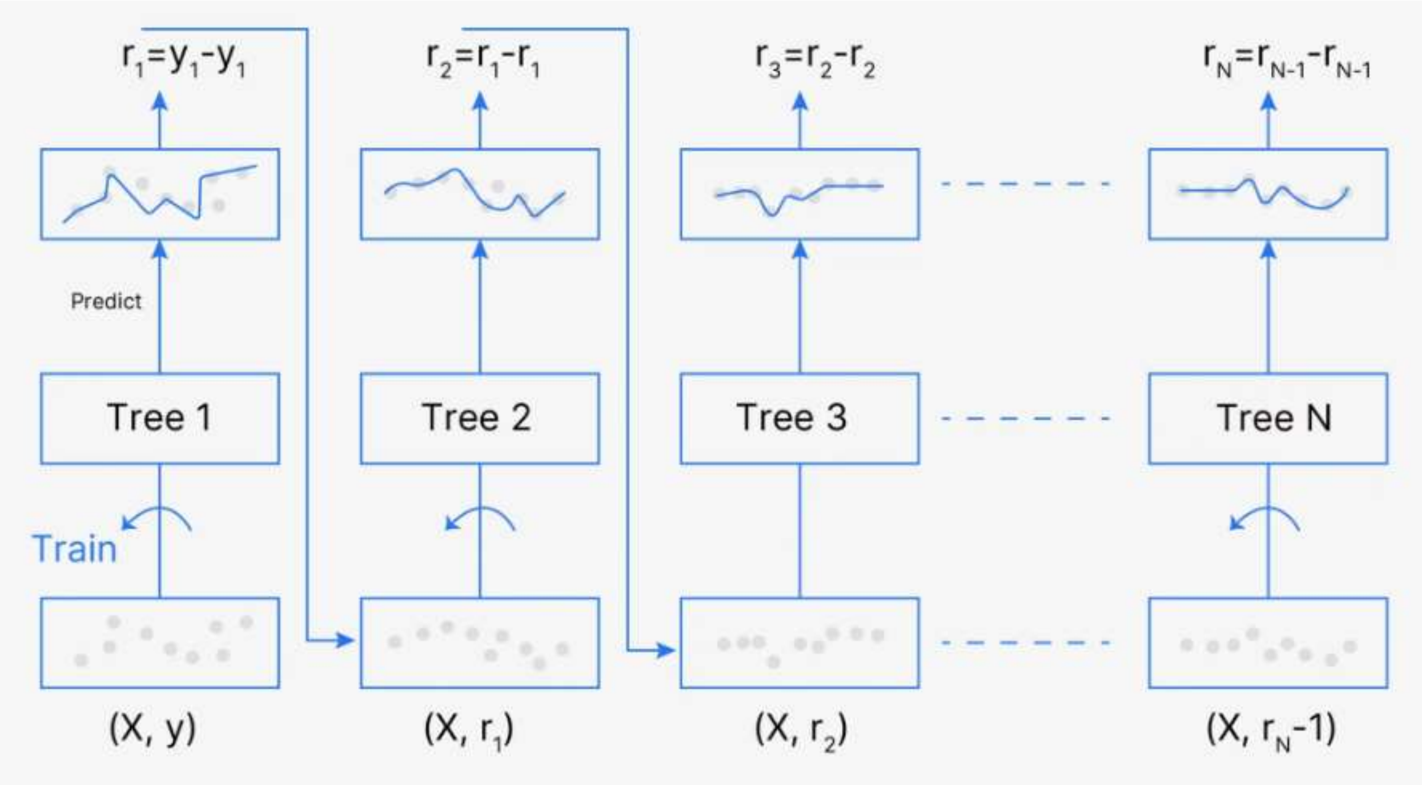
The final prediction combines the outputs of all models, usually through weighted voting or averaging.

Type of Boosting Algorithm

- **GRADIENT BOOSTING**
- **ADABOOST (ADAPTIVE BOOSTING)**
- **XGBOOST**

Gradient Boosting Machine

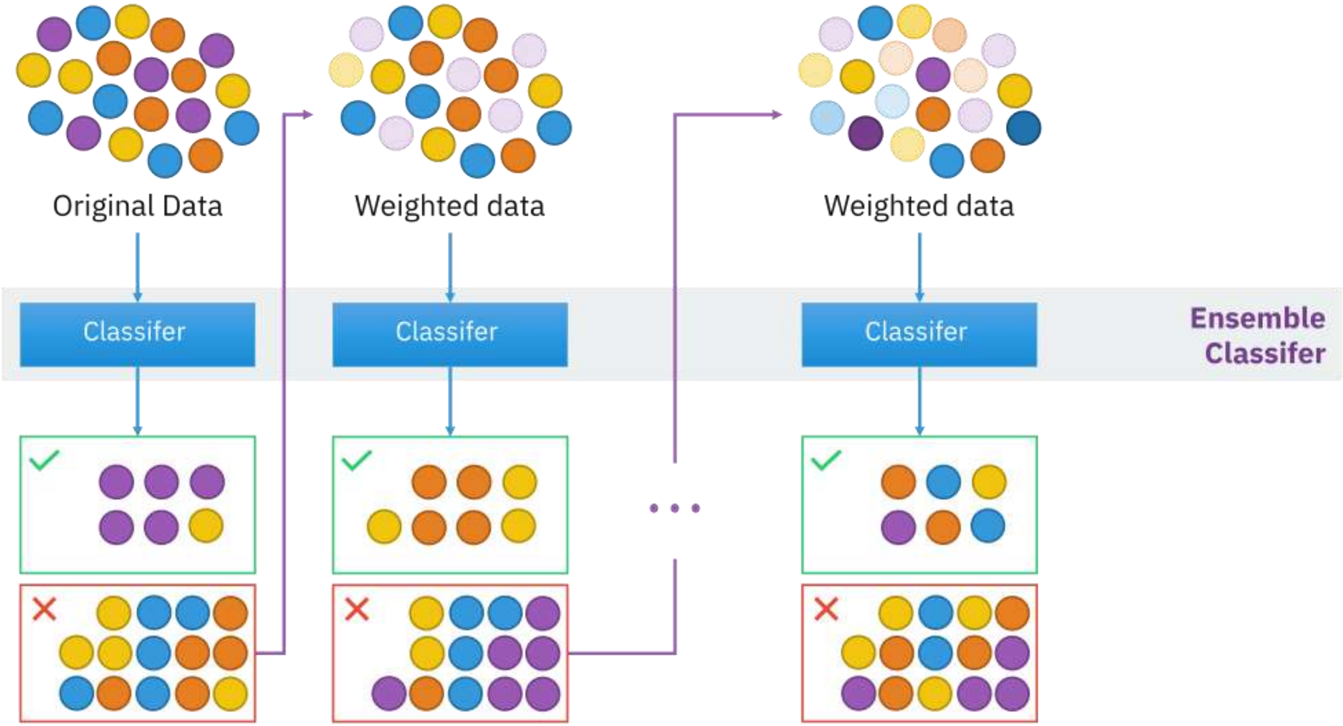
[Initial Prediction] --> [Compute Residuals] --> [Base Estimator 1 (on Residuals)] --(Update Prediction)--> [Compute New Residuals] --> [Base Estimator 2] --> ... --> [Final Model]



Gradient Boosting is a machine learning technique for regression and classification problems. It builds models in a sequential manner, where each new model aims to correct the errors made by its predecessor. The models are typically decision trees, and the final prediction is a weighted sum of all individual models.

Adaboosting Algorithm

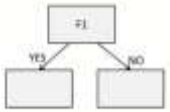
```
[Data with Weights] --> [Base Estimator 1] --(Adjust Weights)--> [Base Estimator 2] --(Adjust Weights)--> ... -->
[Base Estimator N]
                         \                                                  /
                          \--(Weighted Combination of Estimators)---> [Final Model]
```



Adaboosting Algorithm

- STEP1: SAMPLE WEIGHT CREATION
- STEP2: STUMP CREATION
- STEP3: STUMP SELECTION
- STEP4: CALCULATE TOTAL ERROR
- STEP5: CALCULATE AMOUNT OF SAY (OR) PERFORMANCE SAY
- STEP6: UPDATE WEIGHTS
- STEP7: NORMALIZE THE WEIGHTS
- STEP8: NEW SAMPLE FORMATION

F1	F2	F3	O/P	SAMPLE WEIGHT	BUCKETS
12	3	23	YES	0.07	0 - 0.07
23	5	45	YES	0.07	0.07 - 0.14
34	3	43	NO	0.07	0.14 - 0.21
21	4	65	YES	0.49	0.21 - 0.70
45	5	34	NO	0.07	0.70 - 0.77
12	2	23	NO	0.07	0.77 - 0.84
34	5	43	YES	0.07	0.84 - 0.93
16	6	45	YES	0.07	0.93 - 1



STUMPS

Stump selection based on Minimum R2 Score

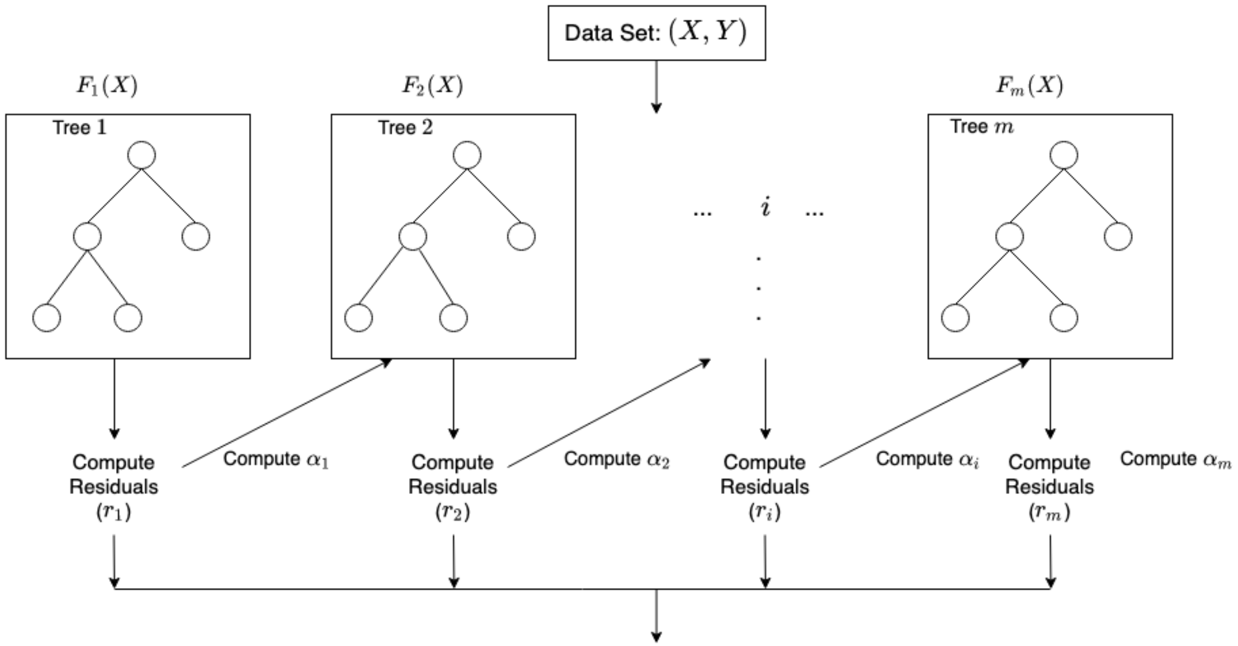
Total Error is the sum of all the errors in the classified record for sample weights.

$$\text{Amount of Say} = \frac{1}{2}\log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

$$\begin{aligned}\text{New Sample Weight} &= \text{sample weight} \times e^{\text{amount of say}} \\ &= \frac{1}{8}\, e^{\text{amount of say}}\end{aligned}$$

The main difference between these two algorithms is that Stochastic Gradient Boosting has a fixed base estimator (known as a **weak learner** or **base learner**) and **decision trees**. In contrast, in AdaBoost, we can change the base estimator to suit our needs.

XGBoost (Extreme Gradient Boosting)



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

XGBoost is an optimized implementation of gradient boosting designed for speed and performance. Developed by **Tianqi Chen** and collaborators, it introduces advanced features that make it stand out from traditional gradient boosting algorithms.

- One of the most important points is that XGBM implements parallel preprocessing (at the node level) which makes it faster than GBM
- XGBoost also includes a variety of regularization techniques that reduce overfitting and improve overall performance. You can select the regularization technique by setting the hyperparameters of the XGBoost algorithm