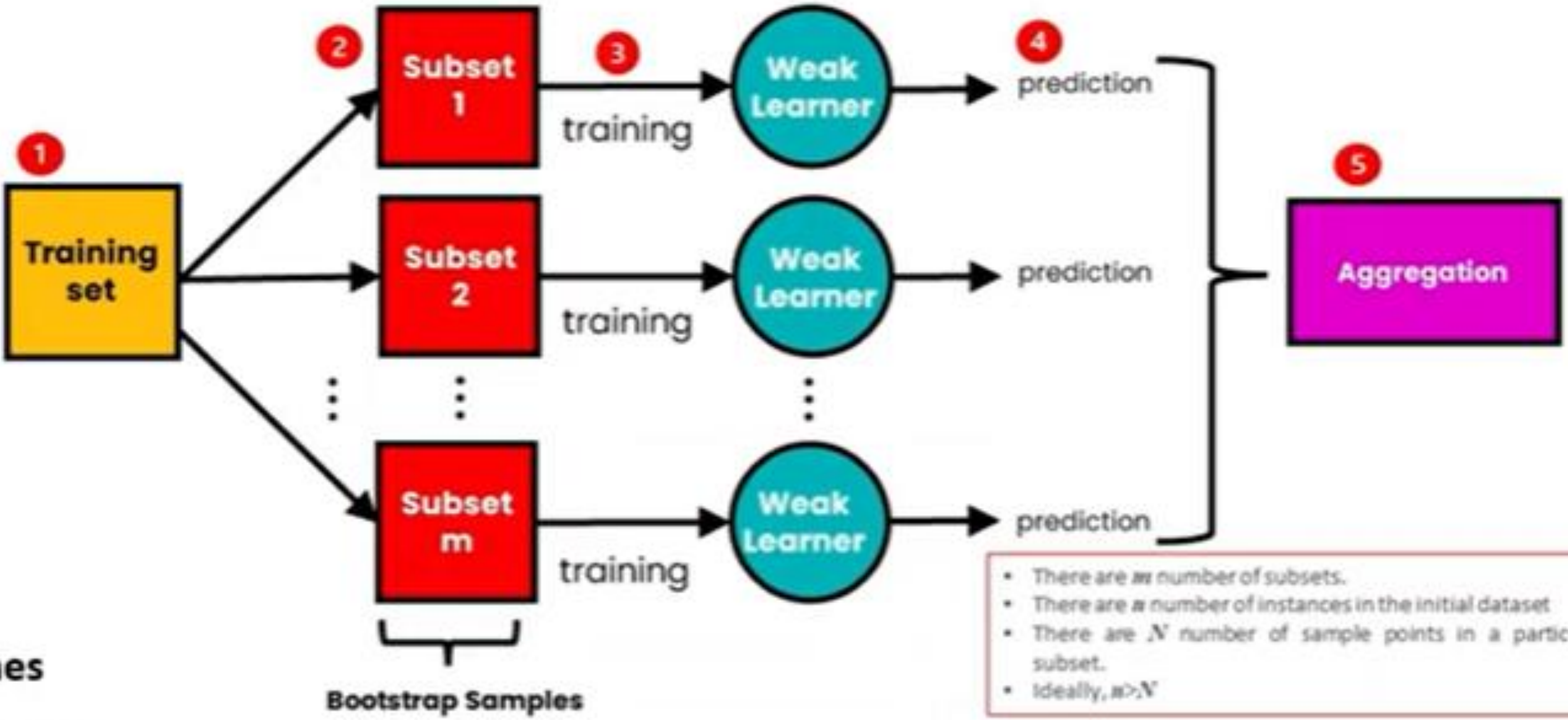
Ensemble Learning (Part II)

Pasting in Ensemble Learning:

- Training multiple models on different subsets of the training data and selecting subsets **without replacement** (each data point appears only in one subset).



- There are m number of subsets.
- There are n number of instances in the initial dataset
- There are N number of sample points in a particular subset.
- Ideally, $n > N$

Examples:

- Pasting
- Random Patches
- Bag of Little Bootstraps

Decision Fusion Techniques

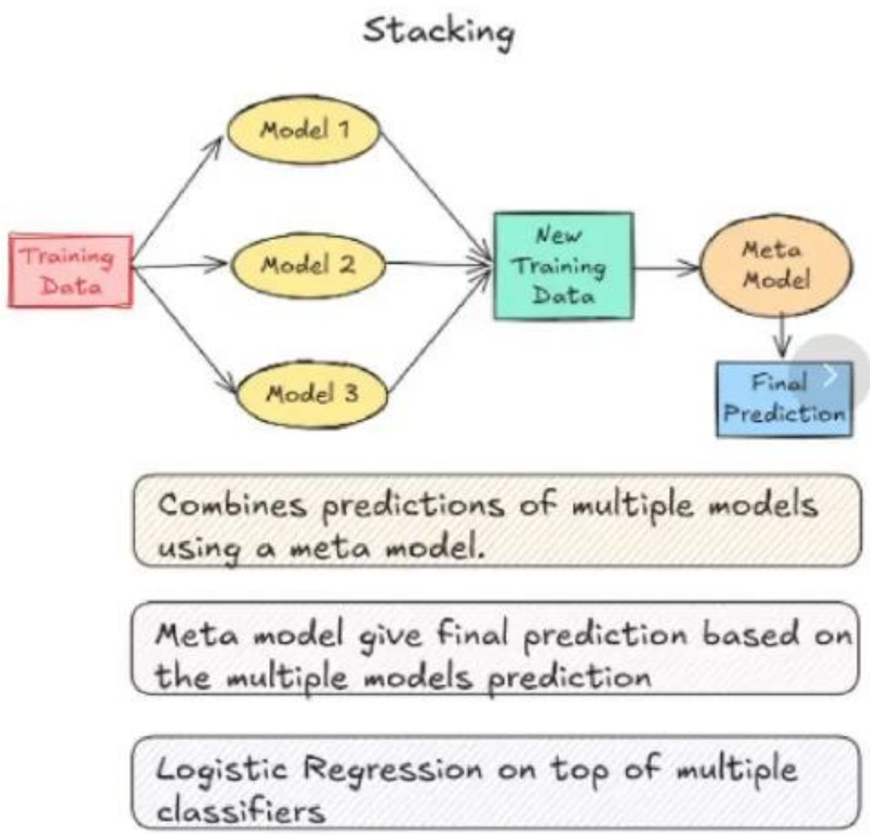
- Fixed Rule Fusion
  - The decision of each classifiers are fused using some fixed rules (eg. Majority Voting, Max(max/ min/ mean) etc)
- Trained Rule Fusion
  - A Classifier is trained using the decisions of all base classifiers to take a decision.

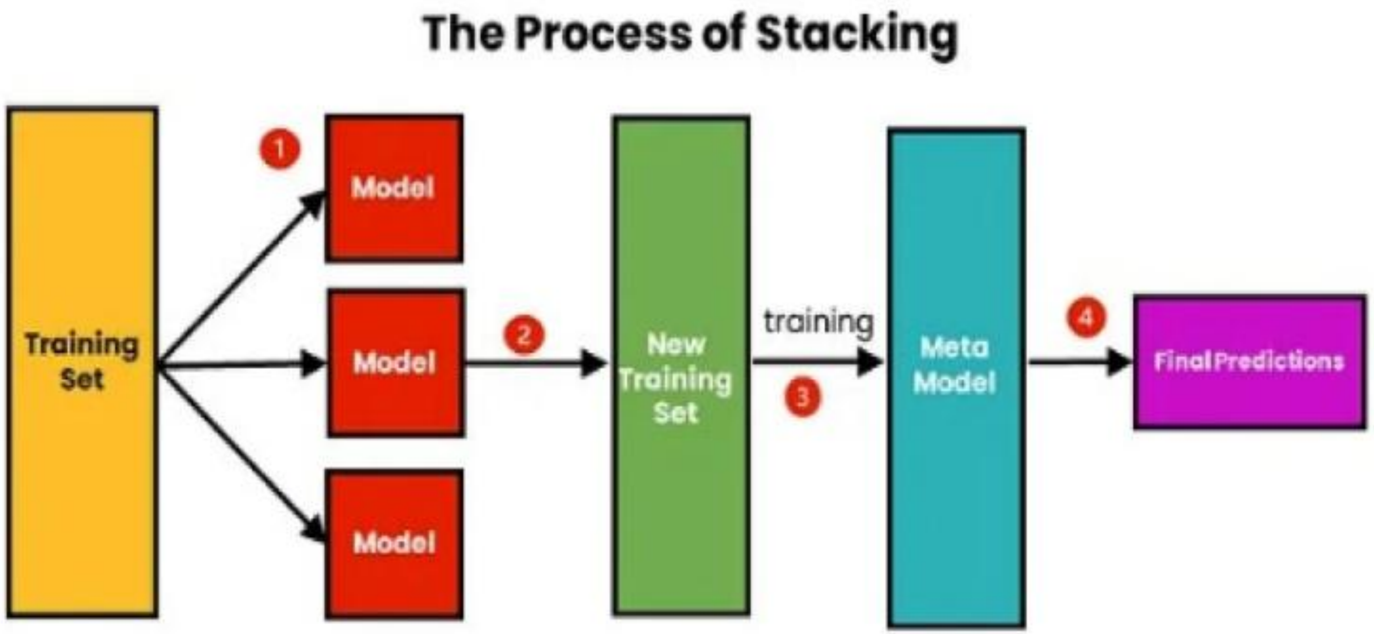
Stacking

- Purpose: Combine different types of models to leverage their diverse strengths.

- How it works: Stacking involves training multiple base models (which may be different types of models) and then using another model, called a "meta-model" or "stacker", to learn how to best combine the predictions of the base models. The meta-model takes the predictions of the base models as inputs to produce the final prediction.

- Example: Combining decision trees, SVM, and neural networks as base models, with a logistic regression or another model as the meta-model





The primary idea of stacking is to feed the predictions of numerous base models into a higher-level model known as the meta-model or blender, which then combines them to get the final forecast.

Preparing the Data: The first step is to prepare the data for modeling. This entails identifying the relevant features, cleaning the data, and dividing it into training and validation sets.

Model Selection: The following step is to choose the base models that will be used in the stacking ensemble.
A broad selection of models is typically chosen to guarantee that they produce different types of errors and complement one another.

Process of Stacking

Training the Base Models: After selecting the base models, they are trained on the training set. To ensure diversity, each model is trained using a different algorithm or set of hyperparameters.

Predictions on the Validation Set: Once the base models have been trained, they are used to make predictions on the validation set.

Developing a Meta Model: The next stage is to develop a meta-model, also known as a meta learner, which will take the predictions of the underlying models as input and make the final prediction. Any algorithm, such as linear regression, logistic regression, or even a neural network, can be used to create this model.

Training the Meta Model: The meta-model is then trained using the predictions given by the base models on the validation set. The base models' predictions serve as features for the meta-model.

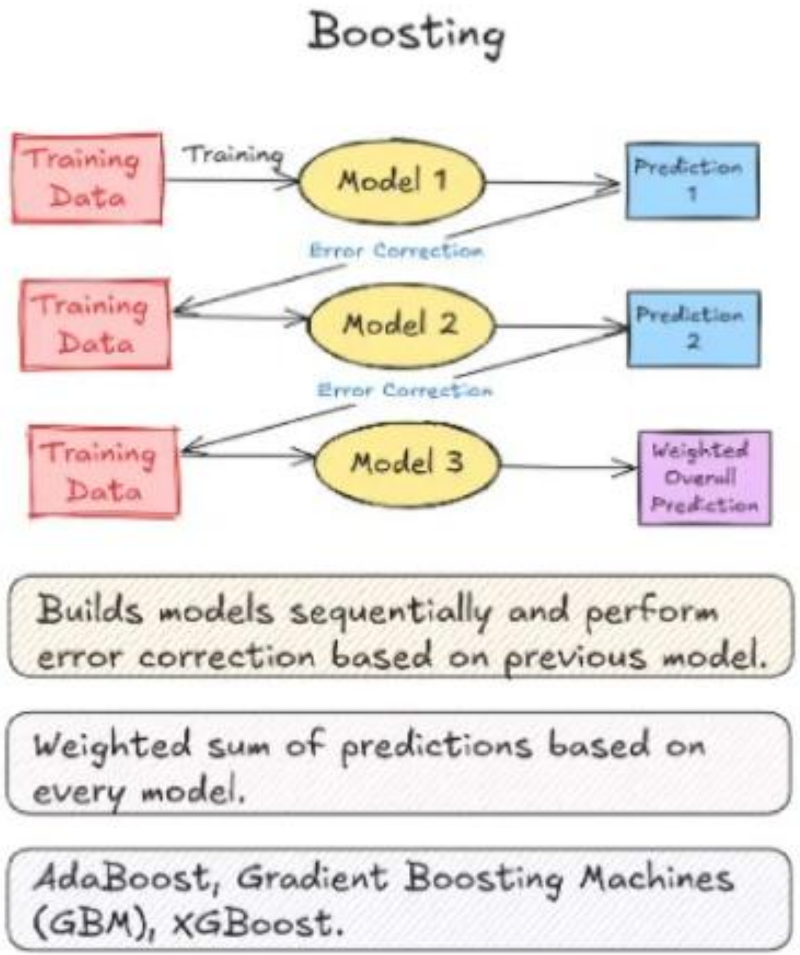
Making Test Set Predictions: Finally, the meta-model is used to produce test set predictions. The basic models' predictions on the test set are fed into the meta-model, which then makes the final prediction.

Model Evaluation: The final stage is to assess the stacking ensemble's performance. This is accomplished by comparing the stacking ensemble's predictions to the actual values on the test set using evaluation measures such as accuracy, precision, recall, F1 score, and so on.

Boosting

- Boosting is an ensemble learning method that combines a set of weak learners into a strong learner to minimize training errors.
- In boosting, a random sample of data is chosen, fitted with a model, and then sequentially trained. In other words, each model aims to make up for the shortcomings of the one before it.
- Intuitively, we do not have a super learner, but many bad learners. These bad learners are combined to obtain a strong learner with lower bias.
- So, Boosting is used to shift the models from **high** bias $\longrightarrow$ **low** bias.

Boosting


Boosting
Training Data
Training
Model 1
Prediction 1
Error Correction
Training Data
Model 2
Prediction 2
Error Correction
Training Data
Model 3
Weighted Overall Prediction
Builds models sequentially and perform error correction based on previous model.
Weighted sum of predictions based on every model.
AdaBoost, Gradient Boosting Machines (GBM), XGBoost.

Boosting

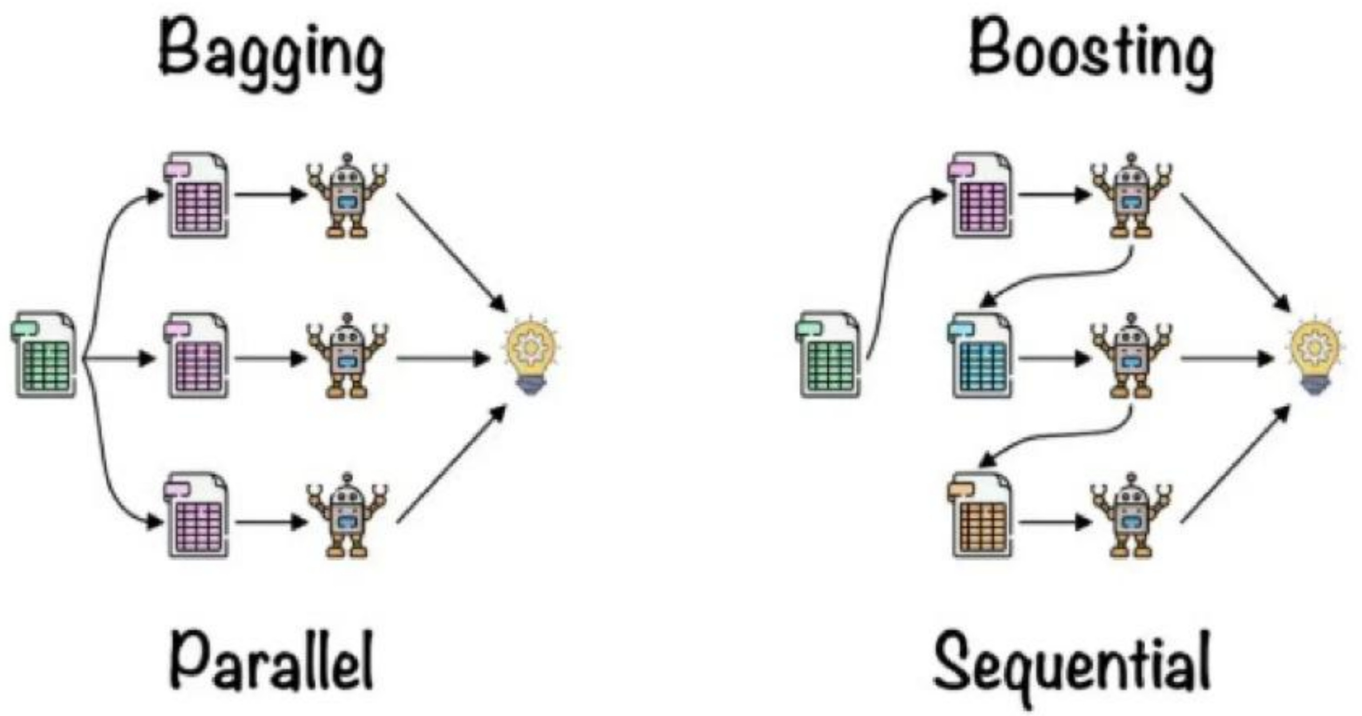
Purpose: Reduce bias and improve the performance of weak models.

How it works: Boosting builds an ensemble **sequentially**, where each model is **trained to correct the errors made by the previous one**. Weights are assigned to instances that are misclassified, and subsequent models focus more on these challenging instances.

The final prediction is typically made by a weighted vote or average of all models.

Example: AdaBoost, Gradient Boosting, XGBoost, LightGBM, and CatBoost

Difference between bagging and boosting



Bagging vs Boosting

Aspect	**Bagging**	**Boosting**
Objective	Reduce variance by averaging predictions across multiple models.	Reduce bias by sequentially learning from model errors.
Model Training	Models are trained independently on different subsets of the data.	Models are trained sequentially, with each model learning from the errors of the previous one.
Data Sampling	Uses bootstrapped datasets (random sampling with replacement).	No bootstrapping; uses the entire dataset, but focuses on misclassified samples.
Model Combination	Voting for classification and averaging for regression.	Weighted combination of models based on their performance.
Overfitting Risk	Less prone to overfitting due to averaging across independent models.	Higher risk of overfitting, especially if the model becomes too complex.
Focus	Focuses on improving stability and reducing variance.	Focuses on improving accuracy by reducing bias.

Bagging vs Boosting

Aspect	Bagging	Boosting
Model Diversity	Models are trained in parallel and are often diverse due to bootstrapping.	Models are trained sequentially, and depend on the errors of previous models.
Computational Efficiency	Can be more computationally efficient due to parallelization.	Sequential training makes it more computationally intensive.
Base Learner	Typically uses strong learners like decision trees.	Typically uses weak learners like shallow decision trees.
Use Case	Suitable for reducing overfitting in high-variance models (e.g., Random Forest).	Suitable for tasks that require improving model accuracy on complex datasets (e.g., XGBoost, AdaBoost).
Common Algorithms	Random Forest, Bagging Classifier.	AdaBoost, Gradient Boosting Machines (GBM), XGBoost, LightGBM.

Popular Boosting Algorithms

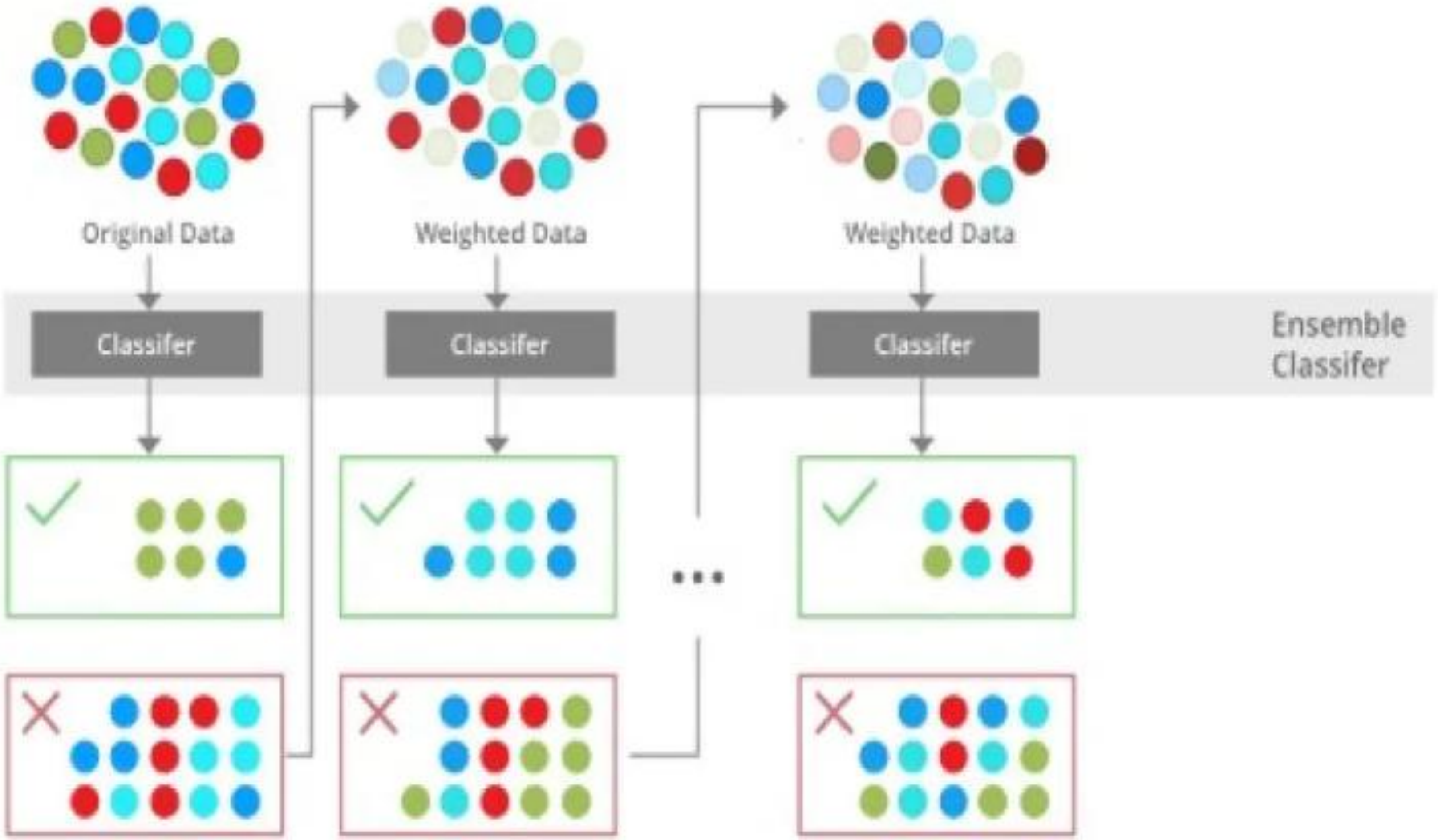
AdaBoost (Adaptive Boosting)
AdaBoost is one of the first and most well-known Boosting algorithms.

In AdaBoost, each model is trained sequentially, with greater emphasis placed on the data points that were misclassified by the previous models.

After training each model, AdaBoost assigns a weight to that model's prediction, which is based on its accuracy. The final output is a weighted combination of the predictions from all models.

Use Case: AdaBoost is often used in binary classification tasks such as spam detection or fraud detection, where the data may contain outliers or noisy samples that the algorithm can learn to handle.

.


Original Data
Weighted Data
Weighted Data
Classifer
Classifer
Classifer
Ensemble
Classifer
...

Popular Boosting Algorithms

2. Gradient Boosting Machines (GBM)
Gradient Boosting Machines (GBM) is a generalization of AdaBoost that applies gradient descent to optimize the performance of the model.

Each subsequent model is trained to correct the residual errors (i.e., the difference between the actual and predicted values) from the previous models, which helps improve accuracy.

Use Case: GBM is widely used in predictive analytics tasks such as predicting credit default risk, customer churn, and sales forecasting. It's known for its strong predictive performance in both classification and regression tasks

Popular Boosting Algorithms

3. XGBoost (Extreme Gradient Boosting)
XGBoost is an optimized implementation of Gradient Boosting that has gained immense popularity for its speed and performance.

XGBoost incorporates several additional features, including regularization to prevent overfitting and parallel processing for faster training.

Use Case: XGBoost is commonly used in data science competitions (like Kaggle) and in real-world applications such as time-series forecasting, marketing analytics, and recommendation systems.