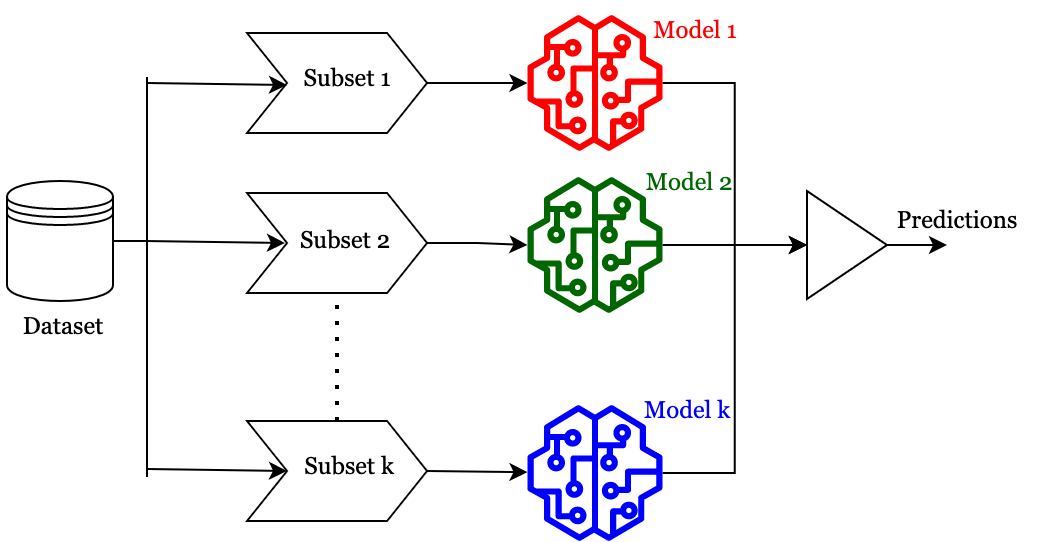
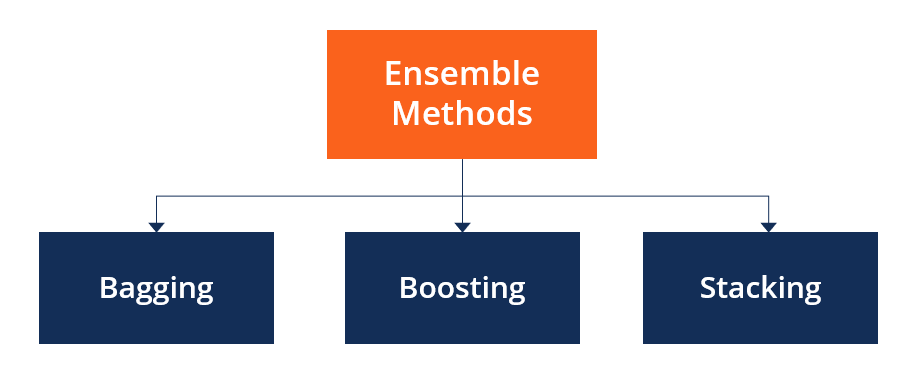
Ensemble learning is a supervised learning technique used in machine

learning to improve overall performance by combining the predictions from

multiple models.Ensemble methods help to improve the robustness/generalizability of the model.



Types of Ensemble methods : -



Basic ensemble methods 1.

**Averaging method:** It is mainly used for regression problems. The method consists of building multiple models independently and returning the average of the prediction of all the models. In general, the combined output is better than an individual output because variance is reduced. In the below example, three regression models (linear regression, xgboost, and random forest) are trained and their predictions are averaged. The final prediction output is pred\_final.

**Max voting:** It is mainly used for classification problems. The method consists of building multiple models independently and getting their individual output called ‘vote’. The class with maximum votes is returned as output.

**Bagging (BootStrap Aggregation**):- Bagging is a machine learning technique that's used to improve model accuracy and reduce variance in data sets by training multiple models on different subsets of data.

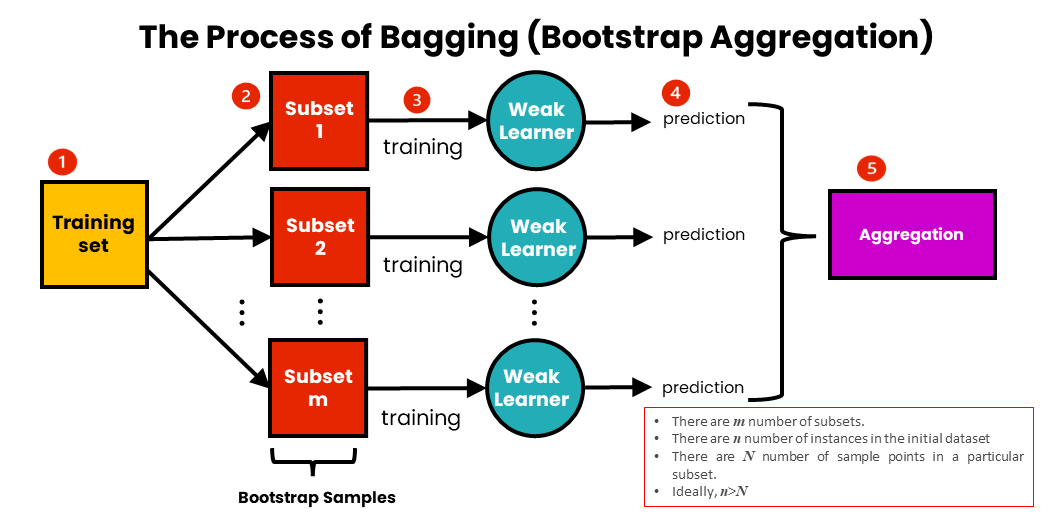
It's a type of ensemble learning, which is a method that uses a group of models to make better predictions by combining their strengths.

How bagging works:

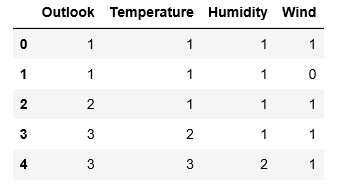
Randomly select data: A random sample of data is selected from the training set, with the possibility of choosing the same data points more than once. This is called a bootstrap sample.

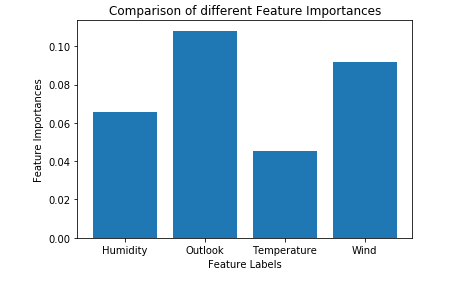
Train models: Multiple models are trained independently on the bootstrap samples.

Combine predictions: The predictions from all the models are combined to create the final prediction. This is usually done by averaging the predictions, but it can also be done by voting. Bagging is effective at reducing variance and overfitting, and it's especially useful when individual models are unstable or prone to overfitting. It can be applied to a variety of base learners, including decision trees, neural networks, and support vector machines.



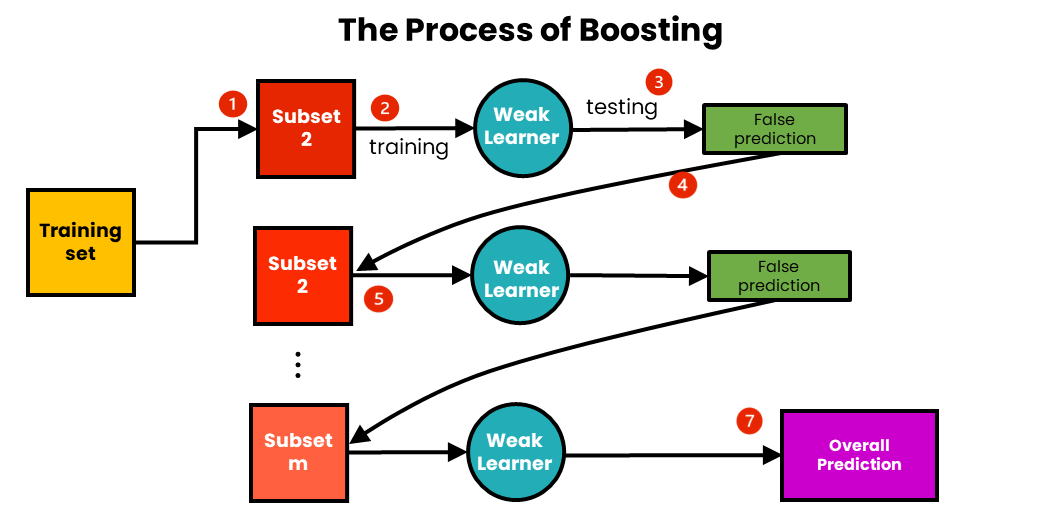
Extra Tree classifier : A technique to find out the most optimal way to choose the best features so as to predict the outcome . So we aim to find out the parameters which are highly dependent on the outcome. So we will be finding





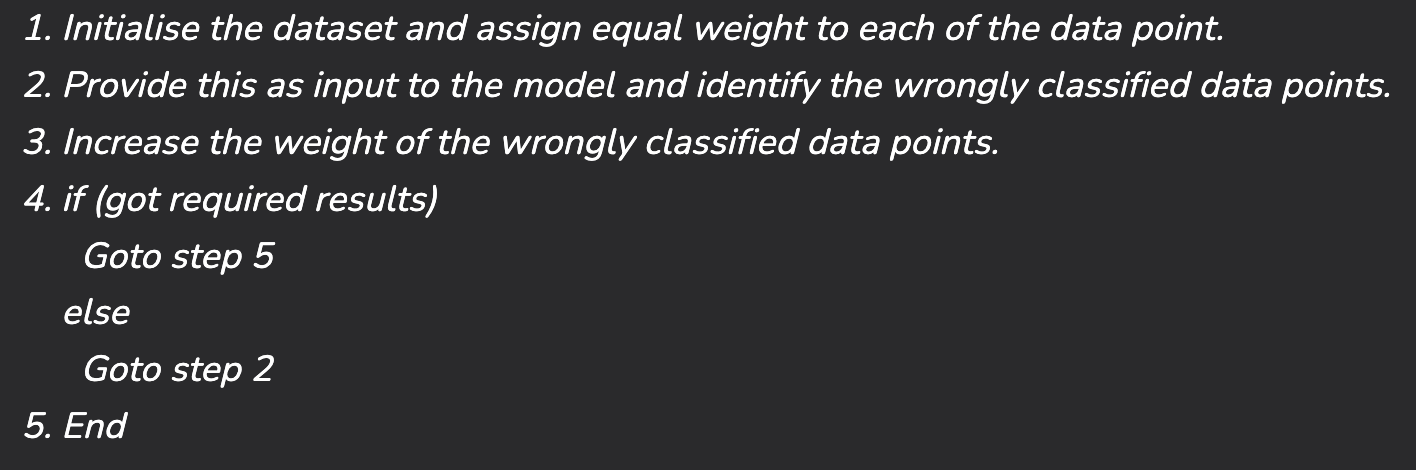
Boosting : A weak model is not enough for complex problems. So in such cases we combine various weak models to develop a single combined yet powerful model with more accuracy

Boosting : Boosting is an ensemble modeling technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.



Advantages of Boosting

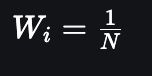
Improved Accuracy – Boosting can improve the accuracy of the model by combining several weak models’ accuracies and averaging them for regression or voting over them for classification to increase the accuracy of the final model. Robustness to Overfitting – Boosting can reduce the risk of overfitting by reweighting the inputs that are classified wrongly. Better handling of imbalanced data – Boosting can handle the imbalance data by focusing more on the data points that are misclassified

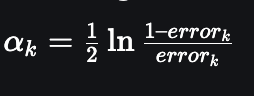
Training of Boosting Model 

1. **ADABOOST:-**

* The main idea behind AdaBoost is to iteratively train the weak classifier on the training dataset with each successive classifier giving more weightage to the data points that are misclassified.
* The final AdaBoost model is decided by combining all the weak classifier that has been used for training with the weightage given to the models according to their accuracies. The weak model which has the highest accuracy is given the highest weightage while the model which has the lowest accuracy is given a lower weightage.

Steps for ADABOOST:

1. Step1 – Initialize the weights For a dataset with N training data points instances, initialize N W i W i ​ weights for each data point with 
2. Step2 - Train a weak classifier Mk where k is the current iteration The weak classifier we are training should have an accuracy greater than 0.5 which means it should be performing better than a naive guess
3. Step3- Calculate the error rate and importance of each weak model Mk

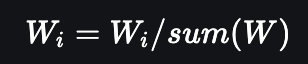
Calculate rate error\_rate for every weak classifier Mk on the training dataset Calculate the importance of each model α\_k using formula

1. Step4 – Update data point weight for each data point Wi

The formula for updating the weights will be w i = w i exp ⁡ ( − α k y i M k ( x i ) ) w i ​ =w i ​ exp(−α k ​ y i ​ M k ​ (x i ​ )) . Here yi is the true output and Xi is the corresponding input vector



1. Step5 - Normalize the Instance weight

normalize the instance weight so that they can be summed up to 1 using the formula 

1. Step6 - Repeat

We will train K classifiers and will calculate model importance and update the instance weights using the above formula

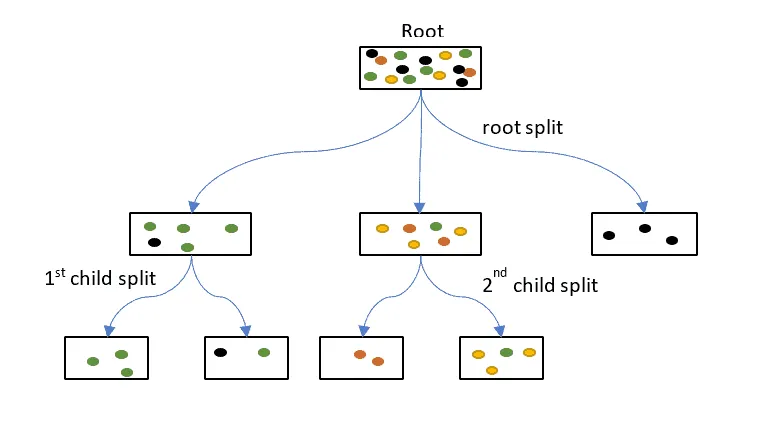
The final model M(X) will be an ensemble model which is obtained by combining these weak models weighted by their model weights

2.CATBOOST:

CatBoost operates on the principle of gradient boosting, which involves sequentially adding decision trees to minimize errors. It effectively handles categorical features without requiring preprocessing, reducing overfitting with techniques like symmetric weighted quantile sketch.

* We often encounter datasets that contain categorical features and to fit these datasets into the Boosting model we apply various encoding techniques to the dataset such as One-Hot Encoding or Label Encoding. But applying One-Hot encoding creates a sparse matrix which may sometimes lead to the overfitting of the model to handle this issue we use CatBoost. CatBoost automatically handles categorical features.
* It is designed for use on problems like regression and classification having a very large number of independent features.
* Catboost is a variant of gradient boosting that can handle both categorical and numerical features. It does not require any feature encodings techniques like One-Hot Encoder or Label Encoder to convert categorical features into numerical features. It also uses an algorithm called symmetric weighted quantile sketch(SWQS) which automatically handles the missing values in the dataset to reduce overfitting and improve the overall performance of the dataset.

CatBoost Data Preprocessing



CatBoost Data Preprocessing involves preparing data for training by handling categorical features efficiently and optimizing memory usage. It automatically handles categorical variables without requiring manual preprocessing steps like one-hot encoding. Additionally, CatBoost can work with missing values directly, simplifying data preparation. Utilizing a CatBoost pool encapsulates the dataset along with features, labels, and categorical feature indices, enhancing efficiency and simplifying data manipulation during training and prediction.

Working of CatBoot

CatBoost works by iteratively building decision trees to minimize errors and improve predictions. It efficiently handles categorical features, automatically handles missing values, and implements techniques to prevent overfitting

The choice between CatBoost, XGBoost, or LightGBM depends on various factors such as dataset characteristics, computational resources, and specific requirements of the problem. CatBoost is preferred when dealing with datasets containing categorical features, as it automatically handles them without preprocessing.

Key Concepts: -

**Gradient Boosting:**

* CatBoost is based on the gradient boosting framework, which combines multiple weak learners (typically decision trees) to create a strong learner. Each tree is trained to minimize the loss function based on the errors of the previous trees.

**Handling Categorical Features:**

* One of CatBoost's primary strengths is its ability to handle categorical features directly without the need for extensive preprocessing (like one-hot encoding).

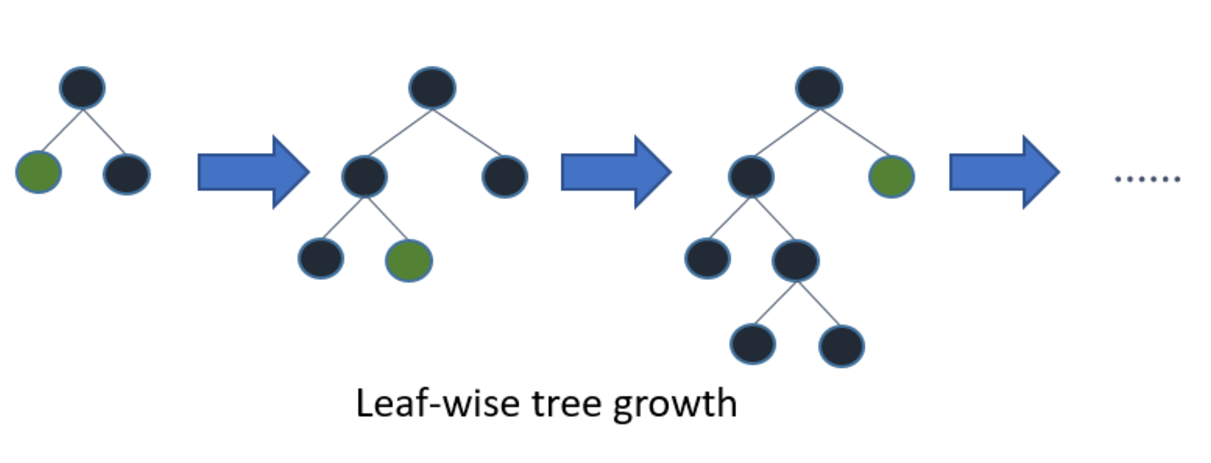
### **Advantages**

* **Handles Categorical Features:** CatBoost can work directly with categorical features, reducing preprocessing time and complexity.
* **Robustness to Overfitting:** The ordered boosting approach helps to reduce overfitting compared to traditional boosting methods.
* **High Performance:** CatBoost often achieves high accuracy and performs well with less tuning than other boosting methods.

**LightGBM:**

Constructs a strong learner by sequentially adding weak learners in a gradient descent manner.

It is designed for efficiency, scalability, and accuracy. It is based on decision trees designed to improve model efficiency and reduce memory usage.

Employs histogram-based algorithms for efficient tree construction. These techniques, along with optimizations like leaf-wise tree growth and efficient data storage formats, contribute to LightGBM’s efficiency and give it a competitive edge over other gradient boosting frameworks.

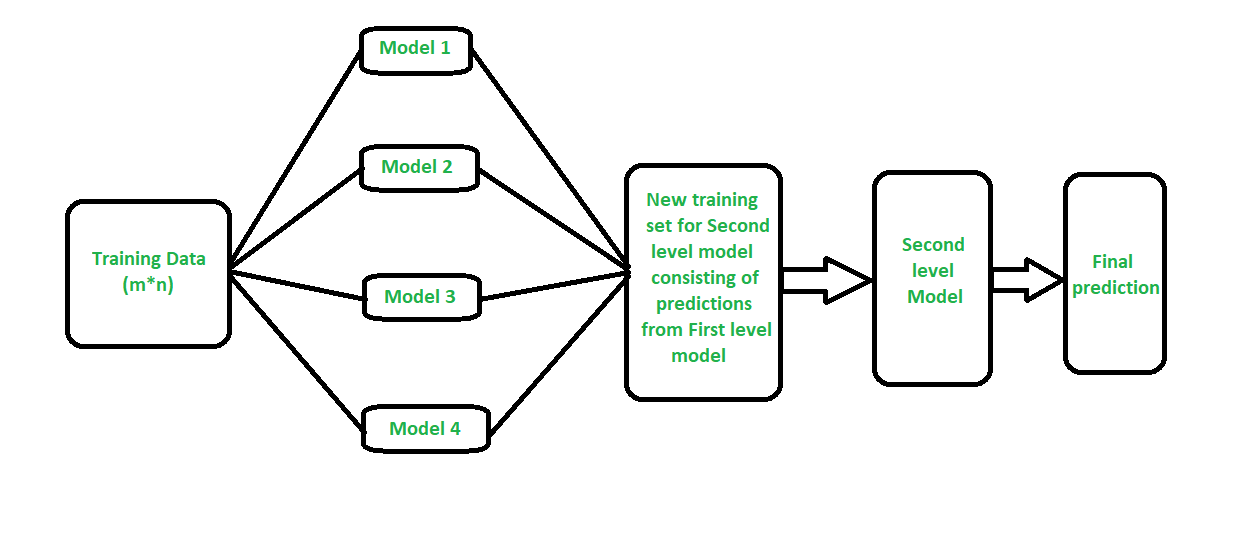
### **Key Concepts**

1. **Gradient Boosting:**
   * Like other gradient boosting methods, LightGBM builds an ensemble of weak learners (typically decision trees) in a sequential manner to minimize a specified loss function.
2. **Histogram-based Learning:**
   * LightGBM uses a histogram-based approach to bin continuous features, which reduces memory usage and speeds up computation.

### **Advantages**

* **Speed and Efficiency:** LightGBM is designed for high performance and low memory consumption, making it suitable for large datasets.
* **Scalability:** It can handle large amounts of data and provides better training speed than many other boosting algorithms.
* **Accuracy:** The leaf-wise growth strategy often leads to improved predictive performance.

**Stacking**

****

**It uses a meta-learning algorithm to learn how to best combine the predictions from two or more base machine learning algorithms.**

**The benefit of stacking is that it can harness the capabilities of a range of well-performing models on a classification or regression task and make predictions that have better performance than any single model in the ensemble.**