**Problem formulations and Methodology:**

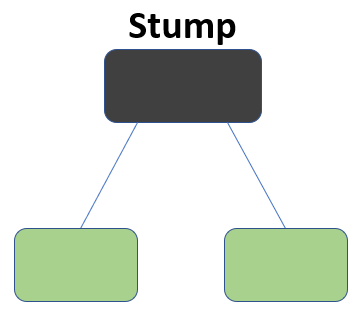
Boosting Model

Boosting is an ensemble modelling technique which improve the prediction power by converting a few weak learners to strong learners.

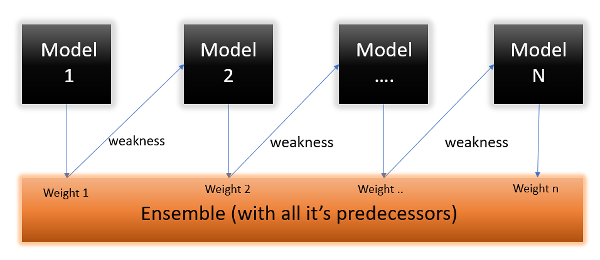
The principle behind boosting algorithms is that we first build a model on the training dataset and then build a second model to rectify the errors present in the first model. This procedure is continued until and unless the errors are minimized and the dataset is predicted correctly. Boosting algorithms work in a similar way, it combines multiple models (weak learners) to reach the final output (strong learners).

AdaBoost Algorithm

AdaBoost, also called Adaptive Boosting, is a technique in Machine Learning used as an Ensemble Method. The most common estimator used with AdaBoost is decision trees with one level which means Decision trees with only 1 split. These trees are also called **Decision Stumps**.

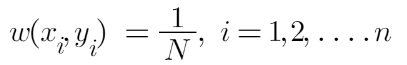


What this algorithm does is that it builds a model and gives equal weights to all the data points. It then assigns higher weights to points that are wrongly classified. Now all the points with higher weights are given more importance in the next model. It will keep training models until and unless a lower error is received.



## Methodology

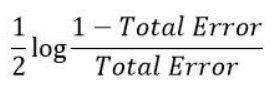
**Step1** - First, data points will be assigned some weights. Initially, all the weights will be equal.



**Step2** - We will create a decision stump for each of the features and then calculate the Gini Index of each tree. The tree with the lowest **Gini Index** will be our first stump.

[Gini Index aims to decrease the impurities from the root nodes (at the top of decision tree) to the leaf nodes (vertical branches down the decision tree) of a decision tree model.]

**Step3** - We will now calculate the “**Amount of Say**” or “**Importance**” or “**Influence**” for this classifier in classifying the data points using this formula:



The total error is nothing but the summation of all the sample weights of misclassified data points.

**Step4** - After finding the importance of the classifier and total error, we need to finally update the weights, and for this, we use the following formula:

net weight sample

The amount of, say (alpha) will be negative when the sample is correctly classified.

The amount of, say (alpha) will be positive when the sample is miss-classified.

**Step 5** – Now, we need to make a new dataset to see if the errors decreased or not. For this, we will remove the “sample weights” and “new sample weights” columns and then, based on the “new sample weights,” divide our data points into buckets.

**Step 6** – We are almost done. Now, what the algorithm does is selects random numbers from 0-1. Since incorrectly classified records have higher sample weights, the probability of selecting those records is very high.

**Step 7** - Now this act as our new dataset, and we need to repeat all the above steps until and unless a low training error is achieved.

**Step 8** - Suppose, with respect to our dataset, we have constructed 3 decision trees (DT1, DT2, DT3) in a sequential manner. If we send our test data now, it will pass through all the decision trees, and finally, we will see which class has the majority, and based on that, we will do predictions for our test dataset.