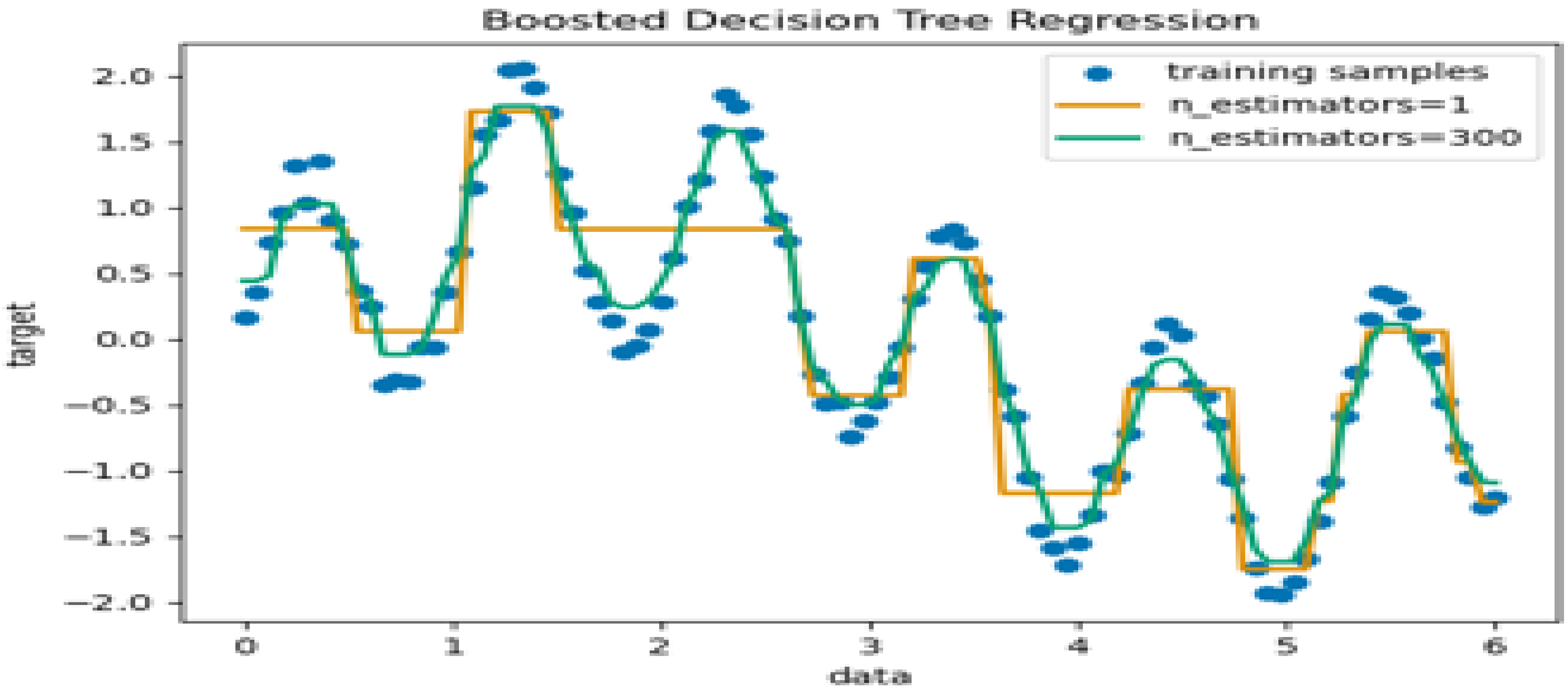
Ada Boosting

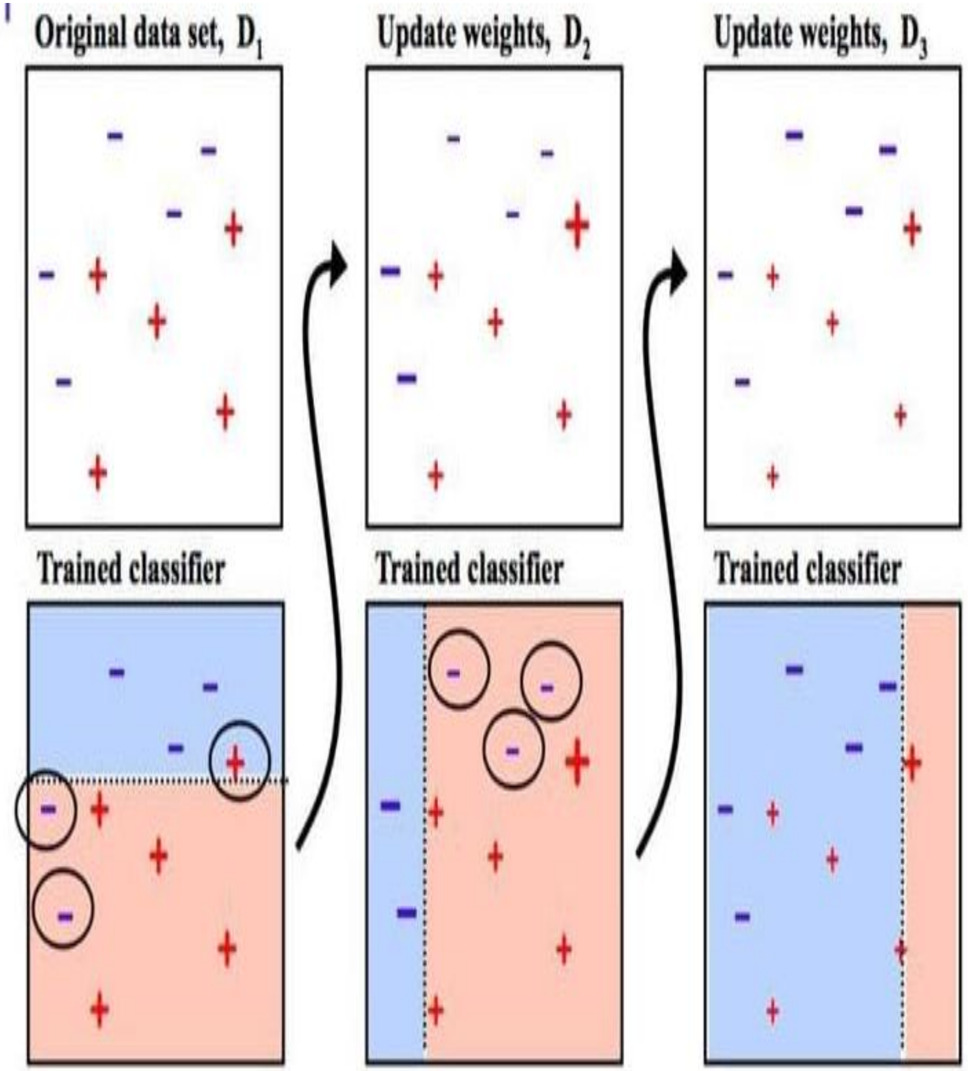
What is ada boosting?

An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.



What are the advantages of AdaBoost Regressor?

- It is easy to use as we do not have to do many hyperparameters tunning as compared to other algorithms.
- Adaboost increases the accuracy of the weak machine learning models.
- Adaboost has immunity from overfitting of data as it runs each model in a sequence and has a weight associated with them.



What is the difference between AdaBoost and boosting?

- Adaboost is computed with a specific loss function and becomes more rigid when comes to few iterations.
- Ada boost The shift is made by up-weighting the observations that are miscalculated prior and The trees are called decision stumps.
- But in gradient boosting, it assists in finding the proper solution to additional iteration modeling problem as it is built with some generic features
- The trees with weak learners are constructed using a greedy algorithm based on split points and purity scores. The trees are grown deeper with eight to thirty-two terminal nodes. The weak learners should stay a weak in terms of nodes, layers, leaf nodes, and splits

What is Hyperparameter in AdaBoost?

The hyperparameters that can be tuned for AdaBoost are: n_estimators: The number of weak learners to train iteratively. learning_rate: Controls the contribution of each classifier. There is a trade-off between learning_rate and n_estimators

The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data.

The data modifications at each so-called boosting iteration consist of applying weights w1, w2, …, wN to each of the training samples. Initially, those weights are all set to wi=1/N, so that the first step simply trains a weak learner on the original data.