# **CHAPTER 5**

# **MACHINE LEARNING ALGORITHMS**

Machine learning algorithms are mathematical model mapping methods used to learn or uncover underlying patterns embedded in the data. Machine learning comprises a group of computational algorithms that can perform pattern recognition, classification, and prediction on data by learning from existing data (training set).

**5.1 Decision Tree**

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. A neat diagram of decision tree is shown in fig 5.1.

**5.1.1 Importance of Decision Tree**

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). In Decision Trees, for predicting a class label for a record we start from the root of the tree.

**5.1.2 How Decision Tree works?**

The decision of making strategic splits heavily affects a tree’s accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.



**Fig. 5.1 Decision Tree**

**5.1.3 Advantage and Disadvantage of Decision Tree**

**Advantage of Decision Tree**

1. It can be used for both classification and regression problems: Decision trees can be used to predict both continuous and discrete values i.e. they work well in both regression and classification tasks.
2. As decision trees are simple hence they require less effort for understanding an algorithm.
3. It can capture nonlinear relationships: They can be used to classify non-linearly separable data.
4. An advantage of the decision tree algorithm is that it does not require any transformation of the features if we are dealing with non-linear data because decision trees do not take multiple weighted combinations into account simultaneously.
5. They are very fast and efficient compared to KNN and other classification algorithms.
6. Easy to understand, interpret, visualize.
7. The data type of decision tree can handle any type of data whether it is numerical or categorical, or boolean.
8. Normalization is not required in the Decision Tree.
9. The decision tree is one of the machine learning algorithms where we don’t worry about its feature scaling. Another one is random forests. Those algorithms are scale-invariant.
10. It gives us and a good idea about the relative importance of attributes.
11. Useful in data exploration: A decision tree is one of the fastest way to identify the most significant variables and relations between two or more variables. Decision trees have better power by which we can create new variables/features for the result variable.
12. Less data preparation needed: In the decision tree, there is no effect by the outsider or missing data in the node of the tree, that’s why the decision tree requires fewer data.
13. Decision tree is non-parametric: Non-Parametric method is defined as the method in which there are no assumptions about the spatial distribution and the classifier structure.

**Disadvantage of Decision Tree**

1. Concerning the decision tree split for numerical variables millions of records: The time complexity right for operating this operation is very huge keep on increasing as the number of records gets increased decision tree with to numerical variables takes a lot of time for training.
2. Similarly, this happens in techniques like random forests, XGBoost.
3. Decision tree for many features: Take more time for training-time complexity to increase as the input increases.
4. Growing with the tree from the training set: Overfit pruning (pre, post), ensemble method random forest.
5. Method of overfitting: If we discuss overfitting, it is one of the most difficult methods for decision tree models. The overfitting problem can be solved by setting constraints on the parameters model and pruning method.
6. As you know, a decision tree generally needs overfitting of data. In the overfitting problem, there is a very high variance in output which leads to many errors in the final estimation and can show highly inaccuracy in the output. Achieve zero bias (overfitting), which leads to high variance.
7. Reusability in decision trees: In a decision tree there are small variations in the data that might output in a complex different tree is generated. This is known as variance in the decision tree, which can be decreased by some methods like bagging and boosting.
8. It can’t be used in big data: If the size of data is too big, then one single tree may grow a lot of nodes which might result in complexity and leads to overfitting.
9. There is no guarantee to return the 100% efficient decision tree.

**5.2 Random Forest**

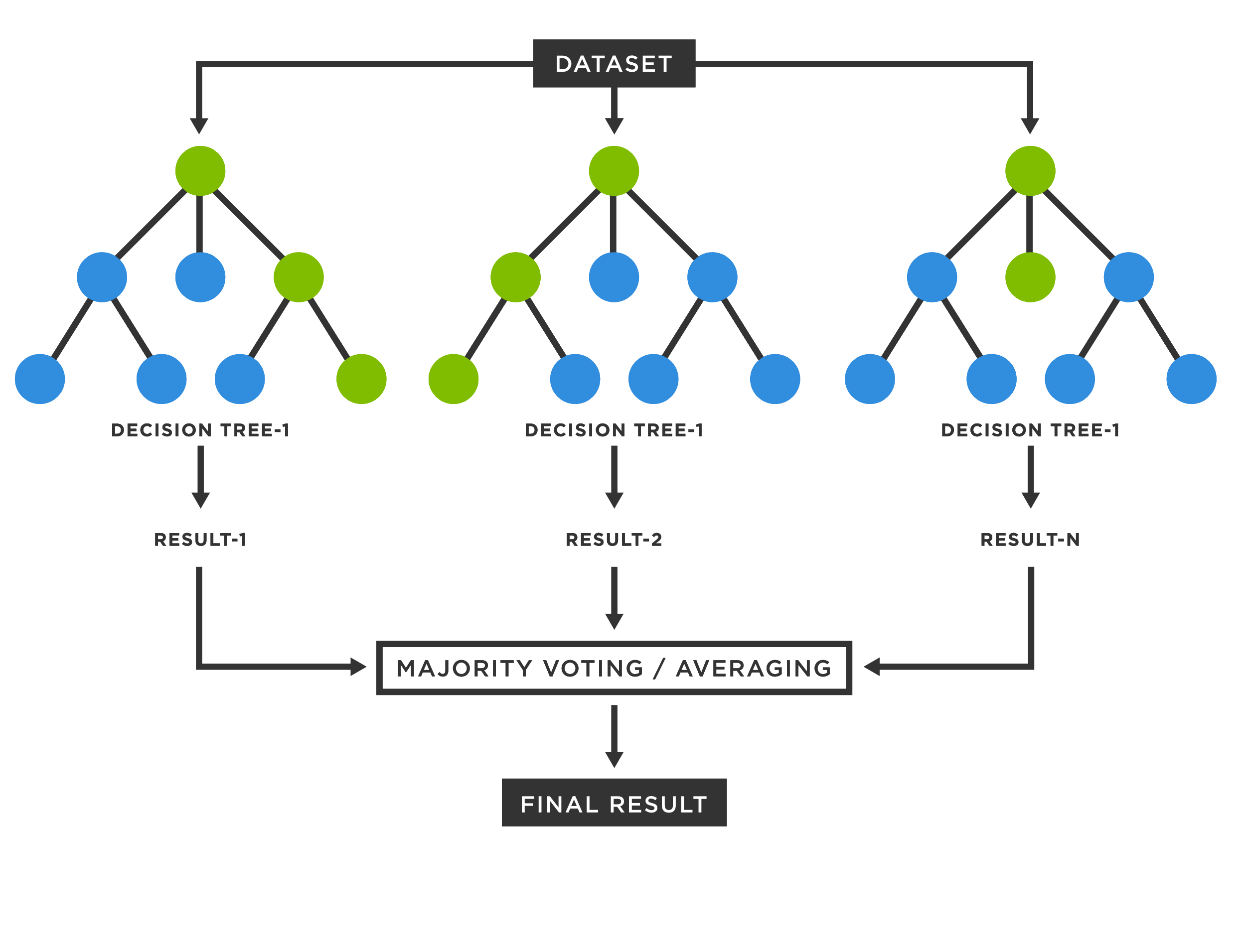
Random forests are a supervised Machine learning algorithm that is widely used in regression and classification problems and produces, even without hyperparameter tuning a great result most of the time. It is perhaps the most used algorithm because of its simplicity. It builds a number of decision trees on different samples and then takes the majority vote if it’s a classification problem. A neat diagram of random forest is shown in fig 5.2.

**5.2.1 Random Forest Importance**

Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction. Sklearn provides a great tool for this that measures a feature’s importance by looking at how much the tree nodes that use that feature reduce impurity across all trees in the forest. It computes this score automatically for each feature after training and scales the results, so the sum of all importance is equal to one.

If you don’t know how a decision tree works or what a leaf or node is, here is a good description from Wikipedia: “In a decision tree, each internal node represents a ‘test’ on an attribute (e.g., whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). A node that has no children is a leaf.”

By looking at the feature importance you can decide which features to possibly drop because they don’t contribute enough (or sometimes nothing at all) to the prediction process. This is important because a general rule in machine learning is that the more features you have the more likely your model will suffer from overfitting and vice versa.



**Fig 5.2 Random Forest**

**5.2.2 How Random Forest Works**

Random forest is a supervised learning algorithm. The “forest” it builds is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

Put simply: random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems.

Let’s look at random forest in classification, since classification is sometimes considered the building block of machine learning. Below you can see how a random forest would look like with two trees:

Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. Fortunately, there’s no need to combine a decision tree with a bagging classifier because you can easily use the classifier-class of random forest. With random forest, you can also deal with regression tasks by using the algorithm’s regressor.

Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model.

Therefore, in random forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

**5.2.3 Advantages and Disadvantages Random Forest Algorithm**

**Advantages**

1. One of the biggest advantages of random forest is its versatility. It can be used for both regression and classification tasks, and it’s also easy to view the relative importance it assigns to the input features.
2. Random forest is also a very handy algorithm because the default hyperparameters it uses often produce a good prediction result. Understanding the hyperparameters is pretty straightforward, and there’s also not that many of them.
3. One of the biggest problems in machine learning is overfitting, but most of the time this won’t happen thanks to the random forest classifier. If there are enough trees in the forest, the classifier won’t overfit the model.

**Disadvantages**

1. The main limitation of random forest is that a large number of trees can make the algorithm too slow and ineffective for real-time predictions. In general, these algorithms are fast to train, but quite slow to create predictions once they are trained. A more accurate prediction requires more trees, which results in a slower model.
2. In most real-world applications, the random forest algorithm is fast enough but there can certainly be situations where run-time performance is important and other approaches would be preferred. And, of course, random forest is a predictive modelling tool and not a descriptive tool, meaning if you’re looking for a description of the relationships in your data, other approaches would be better.

**5.3 Support Vector Machine**

Support Vector Machine or SVM is one of the most popular Supervised Reading algorithms, used for Scheduling and retrieval problems. Mainly, however, it is used for Distribution Problems in Machine Learning.

The goal of the SVM algorithm is to create a better line or decision line that can divide n-dimensional space into classes so that we can easily place a data point in the appropriate category in the future. This best decision-making limit is called the hyperplane.

SVM selects the extra points / vectors that help create the hyperplane. These extreme cases are called supporting vectors, which is why the algorithm is called Vector Support Machine. A neat diagram of Support Vector Machine is shown in fig 5.3

**5.5.1 Importance of Support Vector Machine**

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.



**Fig 5.3 Support Vector Machine**

**5.3.2 How SVM works**

SVM works by mapping the location of a high-resolution feature so that the data points are separated, even though the data can be categorized differently. A separator is found between sections, and the data is converted in such a way that the separator can be drawn as a hyperplane. After this, new data features can be used to predict which group the new record should belong to.

**5.3.3 Advantage and Disadvantage of SVM**

**Advantages of Support Vector Machine (SVM)**

1. Regularization capabilities: SVM has L2 Regularization feature. So, it has good generalization capabilities which prevent it from over-fitting.
2. Handles non-linear data efficiently: SVM can efficiently handle non-linear data using Kernel trick.
3. Solves both Classification and Regression problems: SVM can be used to solve both classification and regression problems. SVM is used for classification problems while SVR (Support Vector Regression) is used for regression problems.
4. Stability: A small change to the data does not greatly affect the hyperplane and hence the SVM. So, the SVM model is stable.

**Disadvantages of Support Vector Machine (SVM)**

1. Choosing an appropriate Kernel function is difficult: Choosing an appropriate Kernel function (to handle the non-linear data) is not an easy task. It could be tricky and complex.
2. In case of using a high dimension Kernel, you might generate too many support vectors which reduce the training speed drastically.
3. Extensive memory requirement: Algorithmic complexity and memory requirements of SVM are very high. You need a lot of memory since you have to store all the support vectors in the memory and this number grows abruptly with the training dataset size.
4. Requires Feature Scaling: One must do feature scaling of variables before applying SVM.
5. Long training time: SVM takes a long training time on large datasets.
6. Difficult to interpret: SVM model is difficult to understand and interpret by human beings unlike Decision Trees.

**5.4 XGBoost**

XGBoost is termed as Extreme Gradient Boosting Algorithm which is again an ensemble method that works by boosting trees. Boost makes use of a gradient descent algorithm which is the reason that it is called Gradient Boosting. The whole idea is to correct the previous mistake done by the model, learn from it and its next step improves the performance. The previous results are rectified, and performance is enhanced.

This gets continued until there is no scope of further improvements. Regularization is the feature that is dominant for this type of predictive algorithm. It is fast to execute and gives good accuracy. This algorithm is commonly used in Kaggle Competitions due to the ability to handle missing values and prevent overfitting. There are again a lot of hyperparameters that are used in this type of algorithm like a booster, learning rate, objective, etc.

The performance of XGBoost is no joke — it’s become the go-to library for winning many Kaggle competitions. Its gradient boosting implementation is second to none and there’s only more to come as the library continues to garner praise.

**5.4.1 Importance of XGBoost**

XGboost is a gradient boosting library. It provides parallel boosting trees algorithm that can solve Machine Learning tasks. It is available in many languages, like C++, Java, Python, R, Julia, Scala. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements Machine Learning algorithms under the Gradient Boosting framework. It provides a parallel tree boosting to solve many data science problems in a fast and accurate way.

**5.4.2 How XGBoost Works?**

XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models.

When using gradient boosting for regression, the weak learners are regression trees, and each regression tree maps an input data point to one of its leaf’s that contains a continuous score. XGBoost minimizes a regularized (L1 and L2) objective function that combines a convex loss function (based on the difference between the predicted and target outputs) and a penalty term for model complexity (in other words, the regression tree functions). The training proceeds iteratively, adding new trees that predict the residuals or errors of prior trees that are then combined with previous trees to make the final prediction. It's called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

**5.4.3 Advantage and Disadvantage of XGBoost**

**Advantage**

1. Regularization: XGBoost has in-built L1 (Lasso Regression) and L2 (Ridge Regression) regularization which prevents the model from overfitting. That is why, XGBoost is also called regularized form of GBM (Gradient Boosting Machine).
2. Parallel Processing: XGBoost utilizes the power of parallel processing and that is why it is much faster than GBM. It uses multiple CPU cores to execute the model.
3. Handling Missing Values: XGBoost has an in-built capability to handle missing values. When XGBoost encounters a missing value at a node, it tries both the left and right hand split and learns the way leading to higher loss for each node. It then does the same when working on the testing data.
4. Cross Validation: XGBoost allows user to run a cross-validation at each iteration of the boosting process and thus it is easy to get the exact optimum number of boosting iterations in a single run. This is unlike GBM where we have to run a grid-search and only a limited values can be tested.
5. Effective Tree Pruning: A GBM would stop splitting a node when it encounters a negative loss in the split. Thus, it is more of a greedy algorithm. XGBoost on the other hand make splits up to the max\_depth specified and then start pruning the tree backwards and remove splits beyond which there is no positive gain.

**Disadvantage**

1. XGBoost does not perform so well on sparse and unstructured data.
2. A common thing often forgotten is that Gradient Boosting is very sensitive to outliers since every classifier is forced to fix the errors in the predecessor learners.
3. The overall method is hardly scalable. This is because the estimators base their correctness on previous predictors, hence the procedure involves a lot of struggle to streamline.