A Guide to Random Forest in Machine Learning

Nisha Arya *Updated on* *November 9, 2023* *AI/ML and Deep Learning*

The Random Forest algorithm is a versatile and powerful tool capable of handling various data-driven challenges for machine learning. The concept of Random Forest took birth because of the need for simplicity and ensemble learning. **In Layman's terms, Ensemble Learning** is stacking together a lot of classifiers to improve performance.

In this article

What is a Random Forest?

Random Forests is a Supervised Learning algorithm and is one of the most flexible and easy-to-use algorithms. **Supervised Learning** is when the algorithm learns on a labeled dataset. We know the answers. However, the algorithm iteratively makes predictions on the training data under the supervision of the labeled dataset to help make corrections.

Random Forest leverages a multitude of decision trees to create a ‘forest’ that is more accurate and robust than its individual components. Each decision tree in the Random Forest makes decisions based on the given data. When these trees work together, they form a ‘forest’ that can handle complex problems much better than any single tree could.

Why Random Forest?

The algorithm of Random Forest helps avoid common challenges like overfitting of the model. With overfitting, a model works well on its training data but not on new, unseen data. The Random Forest approach is a favorite among data scientists for solving both classification problems, where we categorize data, and regression tasks, where we predict a number.

The ‘forest’ created by the Random Forest algorithm is trained through Bagging. **Bagging** aims to reduce the complexity of models that overfit the training data. Bagging consists of aggregation and bootstrapping; combining the words gives us ‘Bagging.’ Bootstrapping is a sampling method where we choose a sample out of a set using the replacement method, making the selection procedure completely random.

The opposite of Bagging is **Boosting,** which aims to increase the complexity of models that suffer from high bias, resolving under-fitting. To know about Bagging and boosting, please check the article about the concept of **decision trees**.

The Random Forest algorithm produces its outcomes based on the predictions generated by the Decision Trees. This prediction takes the average or mean output from the various Decision Trees. An increase in the number of trees increases the precision of the outcome.

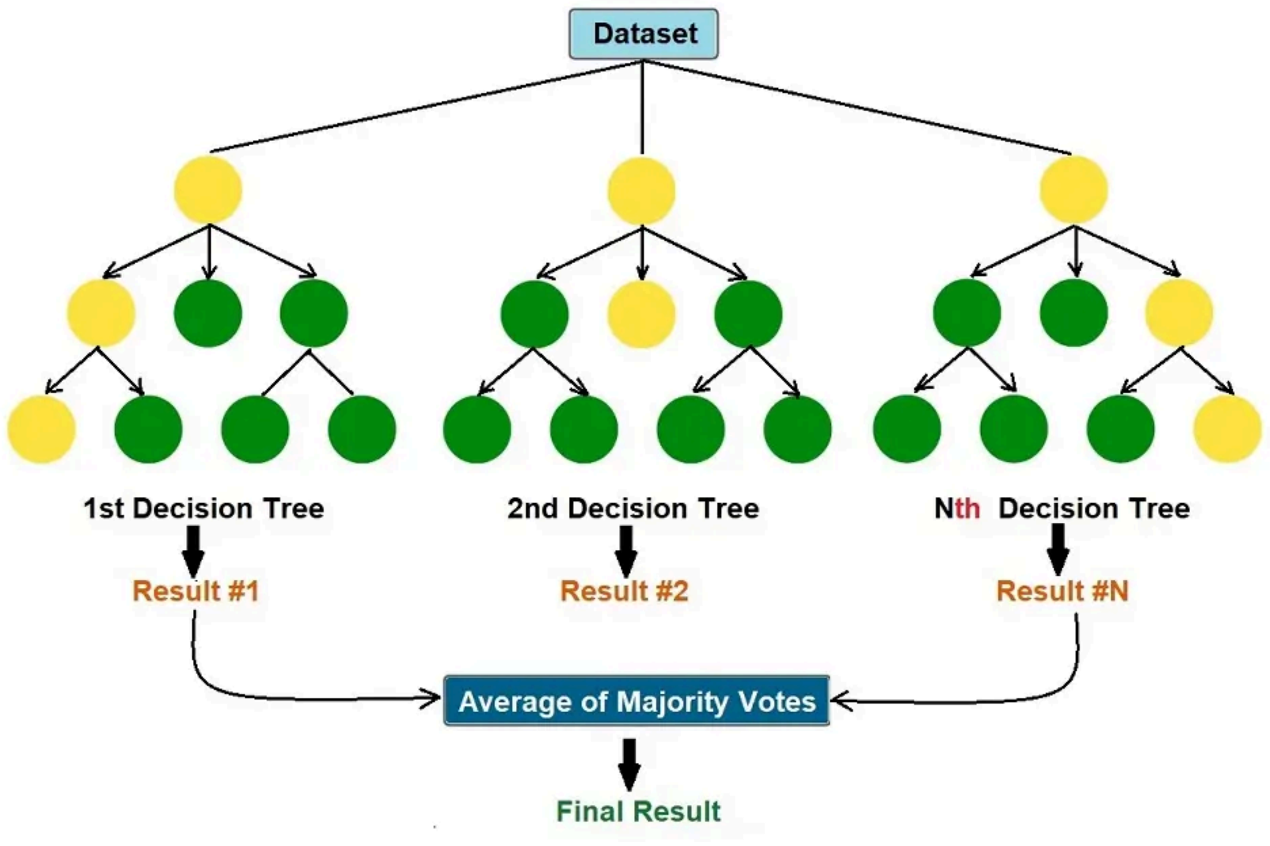
Having higher numbers of Decision Trees in a Random Forest ensures better accuracy and prevention or reduction of overfitting.

So, how does this algorithm work?

Random Forests Algorithm

The Random Forest algorithm works in 4 steps:

1. Select random samples from a given dataset.
2. Create a Decision Tree for each sample and get a prediction result.
3. Perform a vote on each of the predicted results.
4. Select the best prediction result with the most votes and use that as the final prediction.



Let’s apply this to a very simple example. Let’s say you want to go to the spa for the weekend. Your first step may be to go online and search for spas in the area. Moreover, to look at reviews, get feedback from friends, and more.

Let’s say you have a lot of family and friends who visit the spa often, so you collate different spas from their past experiences. Now you have a list of recommendations and want to dig a bit further to choose the right spa. So, you ask your friends and family various questions, for example, the pool size, massage quality, etc. Then, you ask them to vote on which is the best out of your recommended spa list.

The spa that has the most votes will be your final result.

The Random Forest model uses two key concepts that give it the name random:

1. A random sampling of training data points when building trees
2. Random subsets of features are considered when splitting nodes

Random Sampling of Training Data

Each Decision Tree in a Random Forest learns from a random sample of a given dataset during the training phase. The samples are ‘drawn with replacement,’ also known as bootstrapping.

To recap, Bootstrapping is a sampling method where a sample is chosen out of a set using the replacement method, making the selection procedure completely random. The overall aim is that by training each tree on different samples, the Forest will have lower variance, but it does not increase the bias.

During the testing phase, the generation of predictions is done by averaging the predictions of each of the Decision Trees. Here, we Train the model on different bootstrapped subsets of the data and then generate the average through the predictions, i.e., by Bagging. Bagging aims to reduce the complexity of models that overfit the training data.

Random Subsets of features for splitting nodes

Each tree gets the full set of features, but only a random subset of features is considered at each node.

For classification tasks, this is generally set to "max_features=sqrt(n_features)". For example, if you have 16 features at each node in each tree, only 4 random features will be considered.

We use this hyperparameter in Random Forest because if you allow all the features for each split, we will end up with the same trees in the entire Random Forest. Considering only a random subset of features helps us to overcome this.

Advantages of a Random Forest

- We can use Random Forests for both classification and regression problems.
- The predictions by random forests are easily understandable.
- It can handle large datasets efficiently.
- A highly accurate and robust method, as there are several Decision Trees in the process.
- It eliminates or reduces over-fitting of data science model by taking the average of all the predictions, canceling the biases.
- It can handle missing values by using median values to replace continuous variables and computing the proximity-weighted average of missing values.
- Random Forest uses the application of feature importance, helping to determine the most contributing features.

Disadvantages of a Random Forest

- The whole process of Random Forest is very slow at generating predictions due to the multiple trees. Moreover, it is a time-consuming algorithm.
- Compared to Decision Trees, the Random Forest model is difficult to interpret and make quick decisions.
- The process also takes more time than Decision Trees.

Applications Where Random Forest is Preferable

The Random Forest algorithm excels in various applications where it is considered among the best approaches. This is due to its versatility, ease of use, and robust performance. Here are a few scenarios where Random Forest can be particularly effective:

1. **Biomedical Fields**: For medical diagnoses, we can use the Random Forest algorithm to classify the patient's disease as malignant or benign based on their medical records.
2. **Banking Industry**: It is useful in detecting customers who are more likely to be safe or risky for loans or to detect fraudulent activity based on patterns of transactions.
3. **E-Commerce**: Random Forest helps recommend products to users based on their past purchasing history and behavior on the site.
4. **Stock Market Analysis**: It can predict stock price movements and trends, aiding investment decision-making.
5. **Quality Control**: In manufacturing, it can predict the failure of a product or a manufacturing process.
6. **Remote Sensing**: The algorithm is adept at classifying different land cover types in satellite images.

The key reasons why Random Forest is a good choice for these applications include its ability to handle large datasets with higher dimensionality, its robustness to noise, and its capability to run in parallel, which speeds up the training process. Moreover, it provides a good estimate of the importance of features, which helps understand the factors driving the prediction, making it valuable for real-world applications.

Applications to Avoid Random Forest

While the Random Forest algorithm is highly versatile, there are certain scenarios where it may not be the best choice:

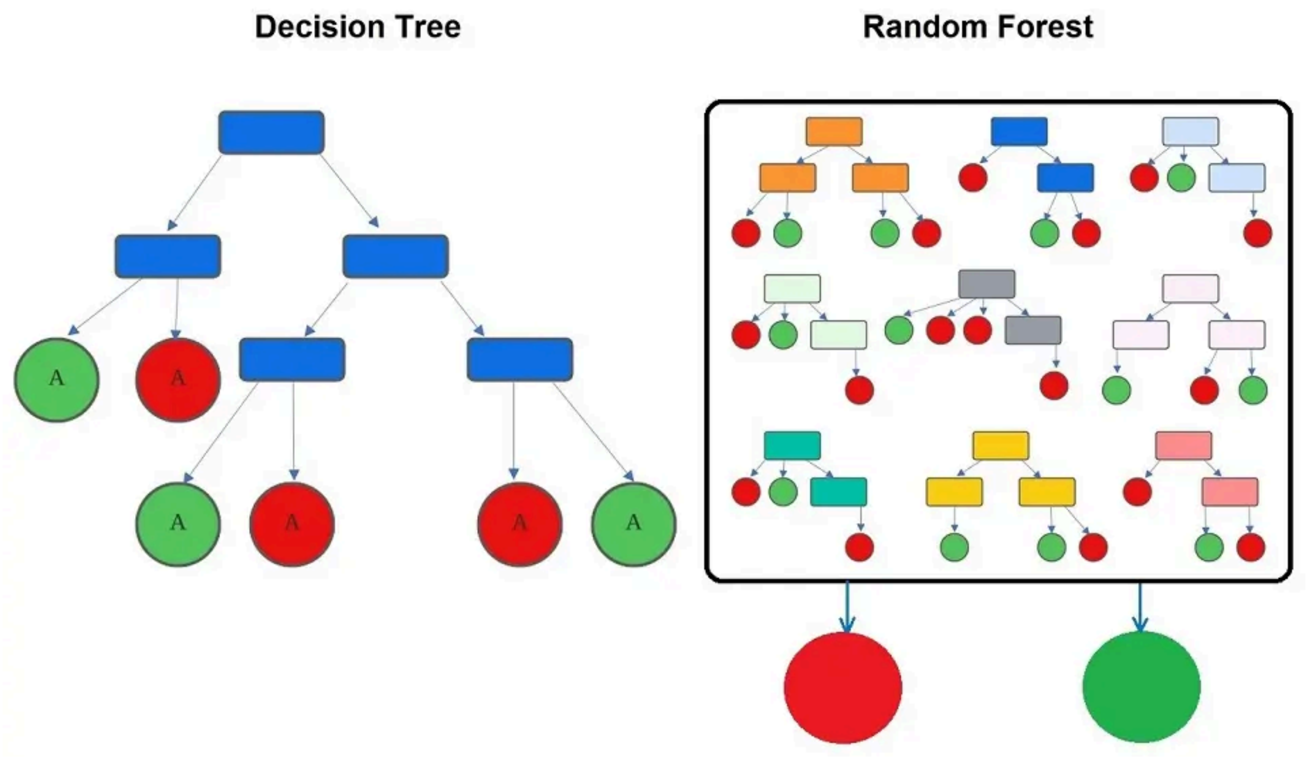
1. **Interpretability Requirements**: If the model needs to be highly interpretable, like in the case of regulatory or business decisions

where explaining the reasoning behind a decision is crucial, Random Forest can be less suitable due to its ensemble nature, which makes it more of a black box compared to simpler models like linear regression or decision trees.

2. **Real-time Predictions**: Random Forest can be slow to make predictions compared to other algorithms because it requires aggregating results across many trees. This can be problematic for applications that require real-time decision-making.
3. **Very Large Data Sets**: Training a Random Forest on a very large dataset can be computationally expensive and time-consuming because it requires building and combining hundreds or even thousands of trees.
4. **High Dimensionality with Many Sparse Features**: Although Random Forest handles high dimensional data quite well, when the features are sparse (like in text classification), models specifically designed for high-dimensional sparse data, like Naive Bayes or Support Vector Machines with linear kernel, might perform better.
5. **Limited Data**: If the available dataset is very small, the Random Forest model may not perform well because the model's strength comes from learning patterns across many trees. In such cases, simpler models or Bayesian approaches might be more effective.
6. **Complex Relationships in Data**: For data with complex, non-linear relationships that vary on a very fine scale, deep learning models may be able to capture these patterns more effectively than Random Forests.

It's always crucial to consider the specific requirements and constraints of the application before choosing Random Forest or any other machine learning algorithm.

Conclusion: Random Forests vs. Decision Trees



- Random Forests consist of a set of multiple Decision Trees.
- Decision Trees are more likely to suffer from overfitting, but Random Forests prevent overfitting by creating trees on random subsets.
- Decision Trees are computationally faster than Random Forests.
- Random Forests are much more challenging to interpret in comparison to Decision Trees.

Nisha Arya

Nisha Arya is a Data Scientist and Technical writer from London.

Having worked in the world of Data Science, she is particularly interested in providing Data Science career advice or tutorials and theory-based knowledge around Data Science. She is a keen learner seeking to broaden her tech knowledge and writing skills while helping guide others.

← Previous post Next post →

Need help?

Let us know about your question or problem and we will reach out to you.

Contact Us