AMNIL TECHNOLOGIES

Presentation on Random Forest and its Implementation

Prepared by: Sejal Panta
25th Sept, 2025

What is Random Forest?

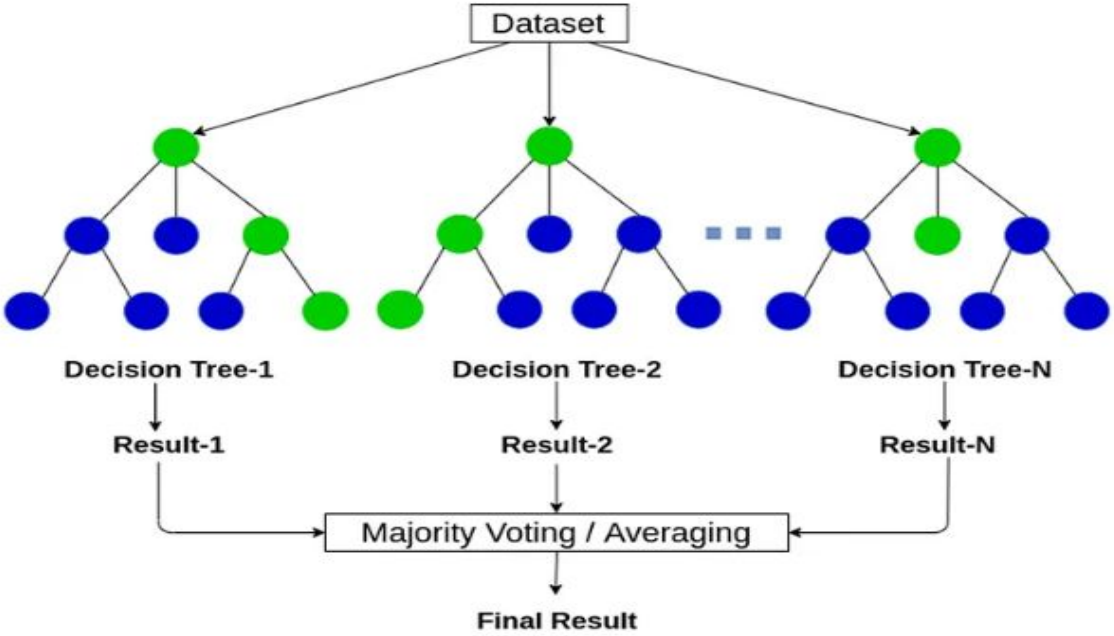
- Random Forest is an ensemble learning method that builds many decision trees and combines their results (by voting for classification or averaging for regression) to improve accuracy and reduce overfitting.

- Think of it as a “forest of decision trees” where each tree gives an answer, and the forest chooses the aggregate answer.

- Each tree in the forest is trained independently, and the final prediction is made by aggregating the predictions of all trees, either by taking the majority vote in classification or averaging the predictions in regression.

- The core idea behind random forests is to combine multiple models to produce a more robust and accurate final model. This process helps mitigate the tendency of individual decision trees to overfit the training data.

Random Forest



Random forest — workflow

Example:

Suppose you want to predict whether a transaction is fraud or not.

- A single decision tree might give biased results.
- Random Forest builds many trees on random samples and features.
- Each tree predicts "fraud" or "not fraud".
- The forest takes the entire votes into consideration, making it more accurate and stable.

Key Hyperparameters of Random Forest:

- n_estimators (Number of Trees): More trees improve accuracy but increase computation.
- max_features (Features per Split): Controls how many features are considered at each split (sqrt, log2, or a fixed number).
- max_depth (Tree Depth): Limits how deep trees can grow; prevents overfitting/underfitting.
- min_samples_split: Minimum samples needed to split a node; higher values reduce overfitting.
- min_samples_leaf: Minimum samples in each leaf; avoids overly specific tiny leaves.
- bootstrap: Whether to use random sampling with replacement when building trees.

How Random Forests Work?

1. Bootstrap Sampling (Bagging)
   - Create many random datasets from the original data (with replacement).
   - Each decision tree is trained on a different sample.

2. Random Feature Selection
   - At each split in a tree, only a random subset of features is considered.
   - This randomness makes trees less correlated and improves generalization.

3. Grow Decision Trees
   - Each tree grows fully (deep) on its bootstrap sample without pruning.
   - Individual trees may overfit, but together they balance out.

4. Aggregation (Final Prediction)
   - Classification → Trees vote, and the majority class is chosen.
   - Regression → Take the average of all tree predictions.

Feature Importance for Fraud Detection

Goal: Detect if a transaction is fraud (1) or not (0).

- Suppose your dataset has features like:
  amount, time_of_day, merchant_type, location, device_type.
- After training a model (like Random Forest or XGBoost), you can check feature importance.

  amount: 0.50
  time_of_day: 0.20
  merchant_type: 0.15
  location: 0.10
  device_type: 0.05

Interpretation:

- amount is the most important: large unusual amounts are more likely fraud.
- time_of_day also matters: transactions at odd hours may signal fraud.
- device_type has very little effect.

So, feature importance tells you what the model looks at most when deciding fraud.

SHAP Values for Fraud Detection

Goal: Understand why a specific transaction is flagged as fraud.

- SHAP can explain for one transaction why the model predicted fraud.

Example:

- Transaction: $5,000 at 2 AM from a new device.
- Model predicts fraud = 1.

SHAP values might show:

amount: +0.6 (large amount increases fraud risk)
time_of_day: +0.3 (odd hour increases risk)
device_type: +0.1 (new device slightly increases risk)
merchant_type: -0.2 (merchant is usually safe, lowers risk)
location: -0.1 (usual location lowers risk)

Total adds up to predicted probability of fraud = 0.7 (70%).

Interpretation: amount and time_of_day are the main contributors pushing this transaction toward fraud.

Advantages of Random Forests:

- Robustness: Random forests are resistant to overfitting, even with noisy datasets, because they aggregate many decision trees.
- Versatility: They can be used for both classification and regression problems.
- Handles Missing Data: Random forests can maintain good performance even when some data is missing.
- Feature Importance: Random forests can estimate the importance of each feature, making them valuable for feature selection.
- Scalability: Random forests are easily parallelizable, allowing them to scale to large datasets.

Disadvantages of Random Forests:

- Interpretability: While decision trees are easy to interpret, random forests are harder to interpret since they involve multiple trees. It can be difficult to understand the model as a whole.
- Computationally Intensive: Building a large number of trees can be computationally expensive, especially for large datasets.
- Memory Usage: Random forests require significant memory, as each tree needs to be stored.
- Slow Predictions: Making predictions with random forests can be slower than with a single decision tree, as it requires querying each tree in the forest.

Uses of Random Forest:

- Fraud detection (banking, e-commerce).
- Medical diagnosis (predicting disease from patient data).
- Stock market analysis (predicting price movement).
- Customer segmentation (classifying users for marketing).
- Recommendation systems.

Random Forest

- Definition: Bagging method using many decision trees and majority voting/averaging.
- Robust, reduces overfitting, handles noisy data.
- Slower with many trees, less interpretable.

XGBoost

- Definition: Boosting method building trees sequentially, each fixing previous errors.
- High accuracy, handles missing data, regularization.
- Complex, needs careful tuning, slower than LightGBM.

LightGBM

- Definition: Fast gradient boosting with leaf-wise tree growth.
- Very fast, efficient on large datasets, low memory use.
- Can overfit small data, sensitive to hyperparameters.

Thank you!