1. Random Forest

An ensemble learning method based on decision trees. A random forest builds multiple decision trees on random subsets of data and combines their outputs.Improve decision tree performance by creating many trees and averaging their results.

Random Forest is an ensemble learning method that constructs multiple decision trees using random subsets of the training data and random subsets of the features. Each tree makes an independent prediction, and for classification tasks, the final output is determined by a majority vote, while for regression tasks, the predictions are averaged. This randomness helps reduce overfitting and improves model accuracy by combining uncorrelated trees.

Prediction = **majority vote.**

How it works and uses:

Builds multiple decision trees on random subsets of data and features.

Each tree gives a vote; the majority vote (classification) or average (regression) is taken.

Reduces overfitting and increases accuracy.

This reduces overfitting and improves accuracy.

Strengths:

Reduces overfitting compared to single decision trees.

Works well for large datasets with many features.

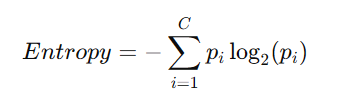
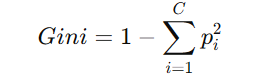
2. Decision Tree

A tree-like model used for decision making. It classifies data by splitting it into smaller and smaller groups based on feature values.

A Decision Tree is a supervised, non-parametric algorithm that represents decisions in a tree structure. Starting at a single root node, the algorithm splits the data based on feature values, creating branches and further subdivisions until the data is partitioned into homogeneous leaf nodes that represent outcomes.

How It Works:

Splits data at each node based on the feature that gives the best separation (e.g., using Gini Impurity, Entropy).



Simple, interpretable, but prone to overfitting.

3. Gradient Boosting:

An ensemble technique that builds models sequentially to correct previous errors.

Gradient Boosting is an ensemble technique that builds a strong predictive model by sequentially adding weak learners, typically small decision trees. At each iteration, the algorithm trains a new model to predict the residual errors (the gradient of the loss function) of the combined model from the previous iterations. This process of iteratively correcting mistakes continues until a predefined stopping criterion is met, such as a maximum number of iterations or minimal improvement in error reduction.

How it works:

1. Each new model is trained to predict the residual (error) of the previous model.
2. Uses gradient descent to minimize a loss function.
3. The final prediction is a combination of all models.
4. Combines weak learners (often shallow trees) to form a strong learner.



hm​(x) is the weak learner (a tree), and γ(m)​ is the step size (learning rate).

4. Support Vector Machine (SVM)

A supervised learning algorithm mainly used for classification.

Support Vector Machine (SVM) is a supervised learning algorithm used for both classification and regression tasks. It works by finding the optimal hyperplane that best separates the data into classes while maximizing the margin between different class boundaries. For datasets that are not linearly separable, SVM can use kernel functions to map the data into a higher-dimensional space where a clear separation is possible.

How it works:

1. Finds the optimal hyperplane that best separates different classes.
2. SVM finds the line (or plane in higher dimensions) that best separates the data into two classes.
3. It tries to maximize the margin (distance) between the closest points of

each class and the dividing line.

1. Maximizes the margin between support vectors (critical boundary points).

5. Logistic Regression

A statistical model used for binary (and multinomial) classification.

Logistic Regression is a statistical model used primarily for binary classification problems. Unlike linear regression, which predicts continuous outcomes, logistic regression models the probability that a given input belongs to a particular class by applying the logistic (sigmoid) function to a linear combination of the input features. The output is a probability value between 0 and 1, which can then be thresholded to assign a class label

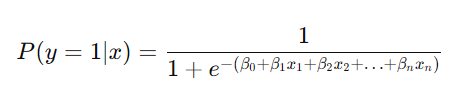
How it works :

Models the probability that an instance belongs to a class using the sigmoid function.

Outputs probabilities between 0 and 1.

Decision boundary usually set at 0.5.

Uses Maximum Likelihood Estimation (MLE) for training.



6. Naive Bayes

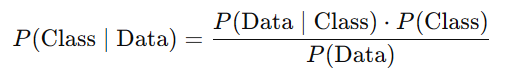
A probabilistic classifier based on Bayes’ theorem.

Naive Bayes is a probabilistic classifier based on Bayes’ Theorem. It assumes that all features in the dataset contribute independently to the probability of an outcome, which greatly simplifies the computation. Despite this strong independence assumption, Naive Bayes often performs well, especially in high-dimensional data, by calculating the posterior probability for each class and choosing the class with the highest probability.

How it works:

Assumes feature independence.which isn’t true for all cases.

Calculates posterior probability:



Fast and effective, especially for text classification like spam filtering.

7. K-Nearest Neighbors (KNN)

K-Nearest Neighbour (KNN) is an instance-based learning algorithm that classifies new data points by comparing them to the 'k' closest examples in the training data. The algorithm calculates the distance between the new data point and all examples in the dataset using a predefined metric (such as Euclidean distance). For classification, the most frequent class among the nearest neighbours is assigned, while for regression, the average of the neighbours’ values is used.

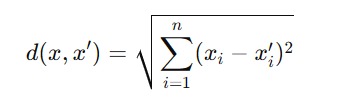
How It Works:

For a new data point, k-NN looks at the k nearest points and assigns the

most common class (for classification) or the average (for regression).

The "distance" between points is often measured by how far apart they

are (Euclidean distance).

Sensitive to distance metric and feature scaling.

Classify based on the **majority class among k-nearest neighbors**.

**Why Random Forest Was Chosen**

1. Highest Accuracy:

Random Forest consistently outperformed other models with an accuracy of 100% during baseline testing.

2. F1-Score:

It achieved the highest F1-Score among all classifiers, ensuring balanced precision and recall.

3. Robustness to Overfitting:

The ensemble nature of RF makes it resistant to overfitting compared to single decision trees.

4. Feature Importance

RF can rank features by importance, aiding in feature selection for a lightweight model

5. Generalization Capability:

It showed better adaptability to unseen malware subtypes during adaptive testing.

6. Computational Efficiency:

While other models like Gradient Boosting are accurate, RF strikes a better balance between speed and accuracy, suitable for real-time applications.

By combining high accuracy, interpretability, and efficiency, **Random Forest proved to be the best fit for the malware detection system.**

**SHAP (SHapley Additive exPlanations):**

SHAP is a powerful framework for explaining machine learning model predictions. It is based on game theory, particularly the concept of Shapley values, which were originally developed to fairly distribute payouts among players in a cooperative game. In the context of machine learning, SHAP assigns an importance value to each feature for a particular prediction.

Key Concepts in SHAP

1. Shapley Values

A Shapley value represents the marginal contribution of a feature to the model’s prediction.

It calculates how much the prediction changes when a specific feature is included or excluded from the model.

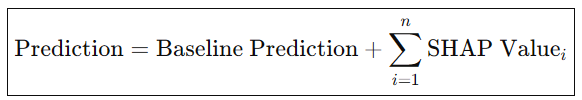
2. Additivity

SHAP ensures that the sum of the contributions of all features equals the difference between the model’s prediction and the average prediction of the dataset.

3. Global vs. Local Interpretations

Global Interpretation: Explains the overall impact of each feature on the model across all predictions.

Local Interpretation: Provides insights into the role of features for an individual prediction.

Final prediction as: 

This makes the model’s decision-making process transparent.

Visualizations in SHAP

1. Beeswarm Plot:

Visualizes the global importance of features.

Each feature is plotted on the y-axis, and its SHAP values for all samples are shown on the x-axis.

Red dots represent higher feature values, and blue dots represent lower feature values.

2. Force Plot:

Explains individual predictions (local interpretation).

Shows how each feature “pushes” the prediction towards a specific label (e.g., benign or malware).

Red arrows represent features pushing towards malware classification, while blue arrows push towards benign classification.

3. Summary Plot:

Combines the feature importance (global) with their distribution.

Helps identify the most influential features and how they interact with predictions.