Random forest: It is the model we used in classification as well as in regression task, in both it has fantastic performance. The main difference between the two lies in the loss function and the output of the model.

The key differences between classification and regression in Random Forest are:

1. **Loss function**: The loss function used to evaluate the performance of the model. Cross-entropy loss is used for classification, while MSE or MAE is used for regression.

2. **Output**: The output of the model. For classification, the output is a class label or class probabilities, while for regression, the output is a continuous value.

3. **Tree construction:** Although the tree construction process is similar for both classification and regression, the splitting criterion used to split the nodes may differ. For classification, the Gini impurity or entropy is used, while for regression, the mean squared error or mean absolute error is used.

What makes Random Forest different for classification and regression?

In Random Forest there is different for classification and regression:

Criterion: The criterion used to split the nodes in the decision trees. For classification, the criterion is typically Gini impurity or entropy, while for regression, the criterion is typically mean squared error or mean absolute error.

Loss function: The loss function used to evaluate the performance of the model. For classification, the loss function is typically cross-entropy loss, while for regression, the loss function is typically mean squared error or mean absolute error.

Output: The output of the model. For classification, the output is a class label or class probabilities, while for regression, the output is a continuous value.

Hyperparameters used in Random Forest(classifier/Regressor):

n_estimators: The number of decision trees in the forest.

max_depth: The maximum depth of the decision trees.

min_samples_split: The minimum number of samples required to split an internal node.

min_samples_leaf: The minimum number of samples required to be at a leaf node.

The goal of a loss function is to calculate the error or difference between the model's output and the desired output. This allows us to evaluate the performance of the model.

Loss Function:

Loss functions are used to optimize the model's parameters during training. Loss functions provide a way to evaluate the performance of the model on a test dataset. A lower loss value indicates better performance. The loss function should be non-negative, meaning that the error is always positive or zero.

Common loss function used in below:

Supervised Learning:

a. Classification Type:

1. Cross-Entropy Loss: Used for multi-class classification tasks, cross-entropy loss measures the difference between predicted probabilities and true labels.

2. Binary Cross-Entropy Loss: Used for binary classification tasks, binary cross-entropy loss measures the difference between predicted probabilities and true labels.

3. Hinge Loss: Used for binary classification tasks, hinge loss measures the difference between predicted scores and true labels.

b. Regression Type:

Mean Squared Error (MSE): Used for regression tasks, MSE measures the average squared difference between predictions and true values.

Mean Absolute Error (MAE): Used for regression tasks, MAE measures the average absolute difference between predictions and true values.

Mean Absolute Percentage Error (MAPE): Used for regression tasks, MAPE measures the average absolute percentage difference between predictions and true values.

Unsupervised Learning:

K-Means Clustering Loss: Used for clustering tasks, K-means clustering loss measures the difference between cluster assignments and data points.

Auto encoder Loss: Used for dimensionality reduction tasks, auto encoder loss measures the difference between input data and reconstructed data.

Neural Networks:

Cross-Entropy Loss: Used for classification tasks, cross-entropy loss measures the difference between predicted probabilities and true labels.

Mean Squared Error (MSE): Used for regression tasks, MSE measures the average squared difference between predictions and true values.

Binary Cross-Entropy Loss: Used for binary classification tasks, binary cross-entropy loss measures the difference between predicted probabilities and true labels.

Optimizer: **Stochastic Gradient Descent (SGD), Adam,**

Activation function - Activation functions are used in neural networks to introduce non-linearity into the model. Without activation functions, neural networks would only be able to learn linear relationships between inputs and outputs.

Activation functions perform the following tasks:

Introduce non-linearity: Activation functions introduce non-linearity into the model, allowing it to learn more complex relationships between inputs and outputs.

Help with feature learning: Activation functions help the model learn features from the input data.

Increase expressiveness: Activation functions increase the expressiveness of the model, allowing it to learn more complex relationships.

Different Activation Functions:

Sigmoid: Used for binary classification tasks. Output range: [0, 1]

Tanh (Hyperbolic Tangent): Used for regression and classification tasks. Output range: [-1, 1]

ReLU (Rectified Linear Unit): Used for classification and regression tasks. Output range: [0, inf)

Leaky ReLU: leaky_relu(x) -Used for classification and regression tasks. Output range: (-inf, inf)

Softmax: softmax(x): Used for multi-class classification tasks. Output range: [0, 1]

Swish: swish(x) = x * sigmoid(x): Used for classification and regression tasks. Output range: (-inf, inf)

GELU (Gaussian Error Linear Unit): Used for classification and regression tasks. Output range: (-inf, inf)

Linear Equation and how its used in Neural network to make the non-linerity:

A linear equation is a mathematical equation in which the highest power of the variable(s) is 1. In the context of neural networks and machine learning, a linear equation can be represented as:

$y = w * x + b$ where y is the output, w is the weight, x is the input, and b is the bias.

Neural Network: A neural network is a complex model composed of multiple layers of interconnected nodes or "neurons". Each node applies a non-linear transformation to the input data, allowing the network to learn complex relationships between inputs and outputs.

Machine Learning: Machine learning is a subfield of artificial intelligence that involves training algorithms to make predictions or decisions based on data. Machine learning models can be linear or non-linear, and can include techniques such as regression, classification, clustering, and more.

Difference between Neural Network and Machine Learning: Neural network helps to solve complex patterns. It automatically learns the features required where as in machine learning it requires the domain expertise.

Linear Equation in Neural Networks:

In a neural network, the linear equation is used in the forward pass to compute the output of each node. The output of a node is computed as:

$$y = \sigma (w * x + b)$$ where σ is the activation function, w is the weight, x is the input, and b is the bias.

Non-Linearity in Neural Networks:

Neural networks introduce non-linearity through the use of activation functions, which allow the model to learn complex relationships between inputs and outputs. Common activation functions include:

Activation Function Graphs:

ReLU: Outputs zero for negative inputs and linear for positive ones.
Sigmoid: S-shaped curve between 0 and 1.
Tanh: S-shaped curve between -1 and 1.

Linear vs. Non-Linear Models:

Linear Model: Straight decision boundary. On-Linear Model it is Curved or complex decision boundaries due to activation functions like ReLU, Sigmoid, or Tanh.

Linear models, such as linear regression, assume a linear relationship between the inputs and outputs. Non-linear models, such as neural networks, can learn more complex relationships between inputs and outputs.