**1. max\_depth:**

**Functionality**: Controls the maximum depth of the decision tree.

**why pick them**: Increasing max\_depth allows the tree to make more complex decisions.

**2. min\_samples\_leaf:**

**Functionality**: the minimum number of samples required to create a leaf node.

**why pick them**: ensuring that each leaf node represents a sufficient amount of data. Increasing this parameter prevents the tree from creating nodes with fewer samples, which helps to prevent overfitting. Larger values of min\_samples\_leaf result in simpler trees with fewer nodes, reducing the risk of overfitting but potentially increasing bias.

**3.min\_samples\_split:**

**Functionality**: Specifies the minimum number of samples required to split an internal node.

**why pick them**: Increasing min\_samples\_split prevents the tree from splitting nodes that have fewer samples than the specified threshold. This helps to prevent overfitting.

**4. criterion:**

**Functionality**: Specifies the function to measure the quality of a split.

**why pick them**: The criterion parameter determines the metric used to evaluate the quality of a split at each node. Common choices include "gini" for the Gini impurity for faster, simpler model and "entropy" for the information gain for more balanced split

**5.max\_features:**

**Functionality**: Determines the maximum number of features to consider when looking for the best split.

**why pick them**: max\_features limits the number of features that the algorithm considers when determining the best split at each node. By reducing the number of features, it can help prevent overfitting .