DecisionTreeClassifier Parameters

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(
    criterion='gini',
    splitter='best',
    max_depth=None,
    min_samples_split=2,
    min_samples_leaf=1,
    max_features=None,
    random_state=None,
    max_leaf_nodes=None,
    min_impurity_decrease=0.0,
    class_weight=None
)
```

1. 🔍 criterion
Definition: The function used to measure the quality of a split.
Valid values: 'gini' (default), 'entropy', 'log_loss' (since scikit-learn 1.1)

'gini': Measures Gini Impurity. Lower is better. Faster to compute.

'entropy': Uses Information Gain from entropy. May yield better accuracy on some datasets.

'log_loss': Like entropy but with probabilities, used in probabilistic classifiers.

Use case:

Use 'gini' for general-purpose classification.

Use 'entropy' when you want more accurate splits based on information theory.

2. 🔀 splitter
Definition: Strategy used to choose the split at each node.
Valid values: 'best' (default), 'random'

'best': Chooses the best split based on the criterion.

'random': Randomly chooses the feature to split—adds randomness, often used in ensembles.

Tip:
Use 'random' with Random Forests or when you want to introduce randomness and reduce correlation between trees.

3. 🌲 max_depth
Definition: The maximum depth of the tree.
Valid values: int or None

None: Expands the tree until all leaves are pure or contain fewer than min_samples_split samples.

int: Limits the tree to a specified number of levels.

Impact:

Lower depth → less overfitting, but possibly underfitting.

Higher depth → can capture complex patterns, but may overfit.

Best practice:
Use cross-validation to find an optimal value.

4. ✂️ min_samples_split
Definition: The minimum number of samples required to split an internal node.
Valid values:

int: e.g., 2 means a node must have at least 2 samples to split.

float: Fraction of total samples (e.g., 0.1 = 10% of dataset).

Effect:

Higher value prevents the tree from learning too fine-grained patterns.

Helps reduce overfitting.

Tip:
Use larger values for noisy datasets.

5. 🍂 min_samples_leaf
Definition: The minimum number of samples required to be at a leaf node.
Valid values:

int: e.g., 1 (default)

float: Fraction of total samples

Why it matters:

Prevents the tree from creating leaves with very few samples, which may not generalize well.

Example:
If set to 5, then each leaf will have at least 5 samples, even if it sacrifices some purity.

6. 🔢 max_features
Definition: The number of features to consider when looking for the best split.
Valid values:

None: All features.

int: Specific number of features.

float: Fraction of total features.

'sqrt': Square root of total features (used in Random Forests).

'log2': Log base 2 of total features.

Use case:

Random Forests use 'sqrt' by default for classifiers.

Reduces training time and adds regularization.

7. 🎲 random_state
Definition: Controls randomness used in the algorithm.
Valid values: int or None

Purpose:

Ensures reproducibility of results.

Important when using splitter='random'.

Example:
Set random_state=42 to always get the same results on reruns.

8. 🍃 max_leaf_nodes
Definition: Limits the number of leaf nodes in the tree.
Valid values: int or None

Controls model complexity and acts as a pruning strategy.

Overrides max_depth if both are set.

Use case:
Useful when you want a tree with interpretable structure or fixed number of outcomes.

9. 🧪 min_impurity_decrease
Definition: A node will be split only if the impurity decrease is at least this threshold.
Valid values: float (default = 0.0)

Purpose:

Helps prevent unnecessary splits.
Acts as a form of regularization.
Formula:
Split is done if:

impurity(parent) - [weighted impurity(left) + weighted impurity(right)] ≥ min_impurity_decrease
10. ⚖️ class_weight
Definition: Weights associated with each class.
Valid values: dict, 'balanced', or None

'balanced': Adjusts weights inversely proportional to class frequencies.

dict: {0: 1, 1: 3} → class 1 has 3x weight.

Use case:

Vital for imbalanced datasets (e.g., fraud detection, rare diseases).

Helps the classifier pay more attention to the minority class.

✅ Summary Table

Parameter	Controls	Default	Use It When…
criterion	Quality of split	'gini'	Want 'entropy' for more accurate splits
splitter	Splitting strategy	'best'	You need randomness in splitting
max_depth	Tree depth	None	You want to avoid overfitting
min_samples_split	Minimum split size	2	You want to reduce overfitting
min_samples_leaf	Minimum leaf size	1	You want smoother decision boundaries
max_features	Features per split	None	Use 'sqrt' in Random Forests
random_state	Reproducibility	None	You want consistent results
max_leaf_nodes	Prune tree by leaves	None	You want simpler, interpretable trees
min_impurity_decrease	Split threshold	0.0	You want to prune small, meaningless splits
class_weight	Handling class imbalance	None	Your dataset is unbalanced