Key Hyperparameters of the Decision Tree Algorithm (Scikit-Learn)

The **DecisionTreeClassifier** in **Scikit-Learn** has several important hyperparameters that affect the model's performance, complexity, and generalization. Below are five key hyperparameters, their functionality, impact on the model, and how increasing/decreasing them affects performance.

1️⃣ max_depth → Controls the depth of the tree

- **Functionality:** Limits how deep the tree can grow. A deeper tree captures more patterns but may overfit.
- **Effect on Model:**
  - **Low max_depth (e.g., 3-5)** → Prevents overfitting, generalizes better.
  - **High max_depth (e.g., 10-20 or None)** → Captures more details but risks overfitting.
- **Performance Impact:**
  - Increasing `max_depth` **increases training accuracy** but can **reduce test accuracy** due to overfitting.

2️⃣ min_samples_split → Minimum number of samples required to split an internal node

- **Functionality:** Prevents the tree from making unnecessary splits when the data size is small.
- **Effect on Model:**
  - **Low values (e.g., 2-5)** → Creates a deep, complex tree (more overfitting risk).

- **High values (e.g., 10-20)** → Forces the tree to consider larger splits, reducing overfitting.

- **Performance Impact:**
  - Increasing `min_samples_split` **reduces variance**, making the model more stable.
  - Decreasing it allows **more flexibility** but can lead to overfitting.

3️⃣ min_samples_leaf → Minimum number of samples required in a leaf node

- **Functionality:** Ensures that leaf nodes contain a minimum number of samples, preventing very small, unreliable splits.
- **Effect on Model:**
  - **Low values (e.g., 1-5)** → Creates many small leaves (high variance, possible overfitting).
  - **High values (e.g., 10-50)** → Enforces larger leaves (smoother decision boundaries, better generalization).
- **Performance Impact:**
  - Increasing `min_samples_leaf` **reduces model complexity and overfitting** but might miss important patterns.

4️⃣ max_features → Number of features to consider for each split

- **Functionality:** Limits the number of features the model considers at each split, introducing randomness and reducing overfitting.
- **Effect on Model:**
  - **Low values (e.g., sqrt or log2 of total features)** → Encourages diversity, helps generalization (good for large datasets).
  - **High values (e.g., all features)** → Uses all features at every split, leading to overfitting.
- **Performance Impact:**

- Lower values help **reduce overfitting** and make the model **more robust**.

5️⃣ criterion → The function used to measure the quality of a split

- **Functionality:** Determines how splits are made by calculating impurity.
- **Options:**
  - `"gini"` (default) → Measures impurity using **Gini index** (faster).
  - `"entropy"` → Uses **information gain**, slightly slower but can be more precise in some cases.
- **Performance Impact:**
  - `"entropy"` may lead to **better splits in complex datasets**, but `"gini"` is computationally faster.
  - **Choosing the right one depends on the dataset** rather than tuning.

Summary of Hyperparameter Effects

Hyperparameter	Low Value Effect	High Value Effect
`max_depth`	Underfits (too simple)	Overfits (too complex)
`min_samples_split`	Overfits (too many splits)	Underfits (not enough splits)
`min_samples_leaf`	Overfits (small leaves)	Underfits (large leaves)
`max_features`	Better generalization	Overfits (too much reliance on all features)
`criterion`	`"gini"` is fast but may not always be optimal	`"entropy"` is slower but may create better splits

Would you like me to apply these concepts in an example with Titanic data? 🚀