Decision Tree Hyperparameters, Overfitting, and Underfitting

1. Introduction

Decision Trees are powerful but can easily become too simple (underfitting) or too complex (overfitting). Hyperparameters are the *“guardrails”* that control this balance.

2. Depth of Tree

- **What it means:** The maximum number of splits from the root to the deepest leaf.
- **Small depth:** Tree is shallow → can’t capture complex patterns → **underfitting**.
- **Large depth:** Tree is very deep → captures noise → **overfitting**.
- Depth control = controlling model complexity.

3. Geometrical Intuition of Overfitting

- Imagine plotting data points and letting the tree split until it perfectly separates each training point.
- This creates **very wiggly, irregular boundaries** — perfect fit for training, but poor generalization on new data.
- Overfitting = the model “memorizes” instead of “learning patterns”.

4. Geometric Intuition of Underfitting

- If the tree is too shallow, its decision boundaries are **too broad and crude**.
- This misses important structure in the data, leading to poor performance on both training and test data.
- Underfitting = the model is too simplistic to capture relationships.

5. Decision Tree Hyperparameter Tuning

Key hyperparameters to control complexity and improve generalization:

1. **max_depth**
    1. Restricts how deep the tree can grow.
    2. Prevents overfitting by avoiding too many detailed splits.
2. **min_samples_split**
    1. Minimum number of samples required to split an internal node.
    2. Larger values → fewer splits → simpler model.
3. **min_samples_leaf**
    1. Minimum number of samples allowed in a leaf node.

    2. Ensures leaves represent enough data to be meaningful.

4. **max_features** (not in all implementations)
    - ❖ Limits the number of features considered at each split → adds randomness → reduces overfitting.
5. **max_leaf_nodes**
    - ❖ Restricts the total number of leaves in the tree.
6. **min_impurity_decrease**
    - ❖ Splits only occur if impurity reduction is at least this value.

Tuning Strategy:

- Start with default → Check train/test accuracy → Adjust one hyperparameter at a time to balance bias (underfitting) and variance (overfitting).

6. Balancing Overfitting & Underfitting

- **If overfitting:**
    1. Reduce max_depth
    2. Increase min_samples_split or min_samples_leaf
    3. Reduce max_leaf_nodes
- **If underfitting:**
    1. Increase max_depth
    2. Reduce min_samples_split
    3. Allow more features per split

7. Key Takeaway

- Decision Trees are flexible but **too much freedom = memorization**, **too little freedom = ignorance**.
- Hyperparameters are your *knobs* to dial in the right amount of complexity.
- Always validate performance on unseen data to ensure the chosen settings generalize.