

Understanding Parameters and Hyperparameters in Machine Learning

Let me explain these concepts clearly to help solidify your understanding.

Core Concepts

1. Learning Algorithm Basics

- **Goal:** Find a function (f) that minimizes error (loss) on your data
- **Process:** Adjusts parameters to make better predictions
- **Example:** Like tuning a radio - you adjust knobs (parameters) to get clearer sound (better predictions)

2. Parameters vs Hyperparameters

Parameters:

- Are learned automatically from the data during training
- Directly affect model predictions
- Examples:
 - Weights in neural networks
 - Coefficients in linear regression
 - Split points in decision trees

Hyperparameters:

- Are set before training begins
- Control how the model learns its parameters
- Examples:
 - Learning rate in neural networks
 - Tree depth in decision trees
 - Regularization strength

Model-Specific Details

Linear Regression

- **Parameters:** β coefficients ($\beta_0, \beta_1, \beta_2$, etc.)
- **Equation:** $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \epsilon$
- **Loss function:** RSS (Residual Sum of Squares)
- **Vanilla version has no hyperparameters**
- **Regularized versions** (Ridge/Lasso) add λ (lambda) as a hyperparameter that controls penalty strength

Decision Trees

- **Parameters:**
 - Which features to split on
 - Split thresholds
 - Tree structure
- **Hyperparameters:**
 - Maximum tree depth
 - Minimum samples per leaf
 - Splitting criterion (Gini impurity, entropy)

Neural Networks

- **Parameters:**
 - All the connection weights between neurons
 - Bias terms
- **Hyperparameters:**
 - Number of layers
 - Number of neurons per layer
 - Learning rate
 - Activation functions

Key Differences

Feature	Parameters	Hyperparameters
Set by	Learning algorithm	Data scientist
Learned from	Training data	Not learned - set before training
Purpose	Make predictions	Control learning process
Adjusted via	Optimization (e.g., gradient descent)	Manual tuning or automated search

Hyperparameters are the "knobs" and "dials" we adjust *before* training a model. They control:

- How the model learns
- The model's structure
- The training process itself

Key characteristics:

- **Not learned** from data (set manually)
- **Control model capacity** (complexity/flexibility)
- **Affect performance** significantly

Hyperparameters by Model Type

1. Linear Regression

- **Vanilla (basic) version:** No hyperparameters
- **Regularized versions:**
 - **λ (lambda):** Controls regularization strength
 - Ridge Regression: $\lambda \sum \beta_j^2$ (L2 penalty)
 - Lasso Regression: $\lambda \sum |\beta_j|$ (L1 penalty)
 - Elastic Net: Combination of L1 and L2

2. Decision Trees

- **Max depth:** How deep the tree can grow
- **Min samples split:** Minimum data points needed to split a node
- **Split criterion:** Gini impurity or entropy
- **Max features:** Number of features to consider at each split

3. Random Forests & Gradient Boosted Machines (GBMs)

- **Number of trees/estimators**
- **Max depth** (per tree)
- **Learning rate** (GBMs only): How quickly the model adapts
- **Subsample ratio:** Fraction of data used for each tree

4. Neural Networks

- **Architecture:**
 - Number of layers
 - Neurons per layer
- **Training:**
 - Learning rate
 - Batch size
- **Regularization:**
 - Dropout rate
 - Weight decay

5. Other Models

- **k-Nearest Neighbors:** Number of neighbors (k)
- **Support Vector Machines:**
 - Kernel type (linear, RBF, polynomial)
 - Regularization parameter (C)

Why Hyperparameters Matter

1. Performance Impact:

- Good hyperparameters → Better model accuracy
- Bad hyperparameters → Poor performance

2. Dataset Dependence:

- Optimal settings vary by dataset
- Must be tuned for each new problem

3. Overfitting Control:

- Help balance bias-variance tradeoff
- Example: Limiting tree depth prevents overly complex trees