Tuning XGBoost with Optuna: From Single to Multi-Objective Optimization

# 1. Introduction

Why hyperparameter tuning matters:  
- XGBoost has many knobs (learning rate, max depth, subsample, etc.).  
- Default parameters ≠ optimal for every dataset.  
  
Optuna’s role:  
- Automatic search for better hyperparameters.  
- Efficient (Tree-structured Parzen Estimator, pruning, etc.).

# 2. Optuna Basics

Key concepts:  
- Study: optimization run.  
- Trial: one set of parameters.  
- Objective function: defines what to optimize.  
  
Diagram idea: Optuna loop → suggest params → train model → evaluate → record metric → repeat.

# 3. Single-Objective Optimization with XGBoost

Setup:  
- Dataset: UCI Adult Income (classification).  
- Metric: AUC (or accuracy, but AUC is better for imbalanced datasets).

# 4. Transition: Why Multi-Objective?

Real-world trade-offs:  
- Accuracy vs model complexity (training time, depth).  
- Precision vs recall.  
- False positives vs false negatives (e.g., in medical tasks).  
  
Single metric hides trade-offs.  
Multi-objective lets us see the Pareto front.

# 5. Bi-Objective Optimization with Optuna

Example 1: AUC vs Inference Time  
- Objective 1: Maximize AUC.  
- Objective 2: Minimize prediction latency or number of trees.

# 6. Demo / Results

Show two sets of results:  
- Single-objective: “We got AUC = 0.89.”  
- Bi-objective: “Here’s the Pareto front; pick your preferred trade-off depending on constraints.”

# 7. Takeaways

- Optuna makes XGBoost tuning easy.  
- Single-objective: straightforward, good starting point.  
- Multi-objective: essential for real-world trade-offs.  
- Visual tools (optimization history, Pareto front) help in decision-making