Lab 4

Ruoxin Jia
rj3u25@soton.ac.uk

November 5, 2025

1 Part 1: k-Nearest Neighbors and the Curse of Dimensionality

1.1 Experimental Results

The kNN classifier is evaluated on both original and high-dimensional noisy data:

Dataset	Feature Dimensions	Accuracy
Original Data	64	0.9926
Noisy High-Dimensional Data	10064	0.5278

Table 1: kNN Performance Comparison

1.2 Analysis

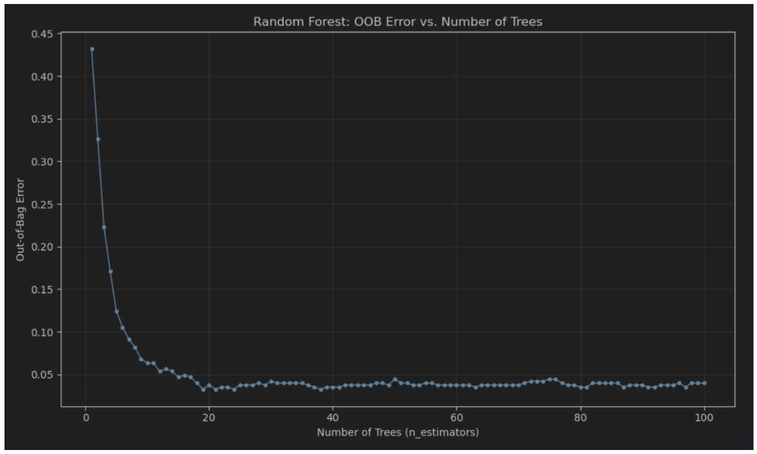
The results clearly demonstrate the **curse of dimensionality**. The accuracy dropped dramatically from 99.26% to 52.78% when 10,000 irrelevant noise features are added. This phenomenon occurs because:

- **Distance Concentration**: In high-dimensional spaces, Euclidean distances become less meaningful as all pairwise distances converge to similar values
- **Data Sparsity**: The feature space becomes exponentially sparse, making it difficult for kNN to find meaningful neighbors
- **Noise Dominance**: The 10,000 noise features overwhelmed the 64 meaningful features, making the signal-to-noise ratio very poor
- **Nearest Neighbor Instability**: In high dimensions, the concept of "nearest neighbors" become unstable and unreliable

2 Part 2: Random Forest - Variance Reduction and Feature Importance

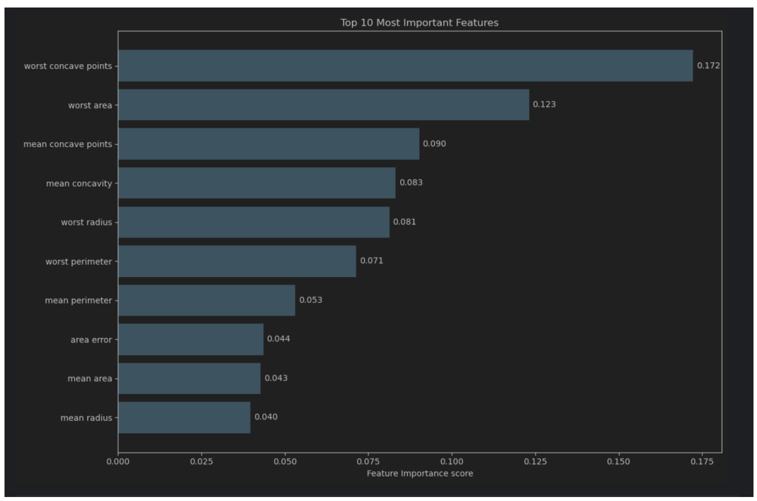
2.1 OOB Error Analysis

The Out-of-Bag (OOB) error was measured across different numbers of trees:



2.2 Feature Importance

Top 10 most important features:



2.3 Analysis

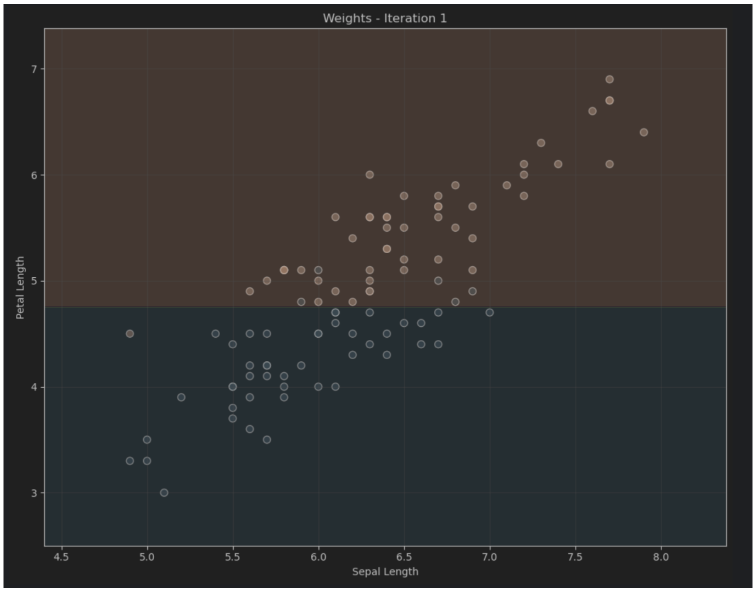
The OOB error demonstrates the **variance reduction** principle of ensemble methods:

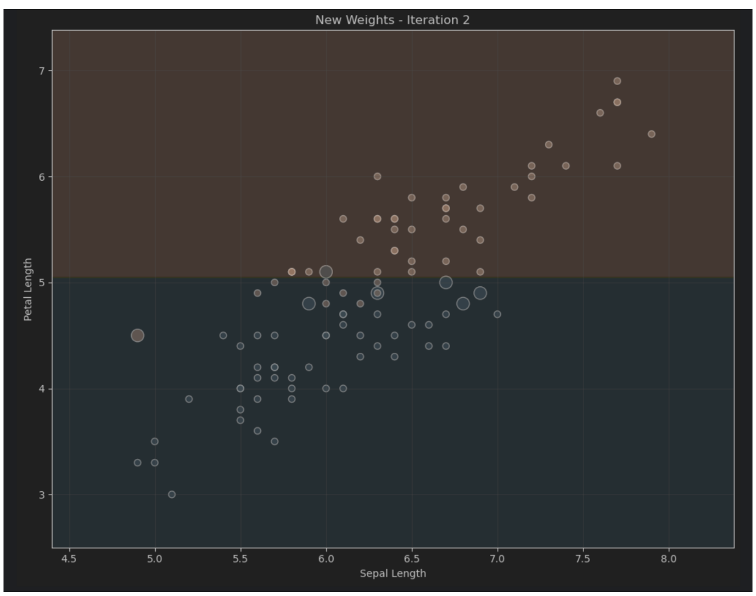
- **Rapid Improvement**: The error decreased quickly from 12.44% with 5 trees to 3.76% with 20 trees
- **Convergence**: After approximately 20 trees, the OOB error stabilize
- **Variance Reduction**: Each additional tree reduces variance by averaging predictions across multiple models
- **Stability**: The ensemble becomes more robust and stable as more trees are added

3 Part 3: Visualizing the AdaBoost Re-weighting Process

3.1 Experimental Results

The AdaBoost algorithm was run for two iterations:





3.2 Analysis

The two iterations clearly illustrate the AdaBoost's re-weighting mechanism:
The algorithm identified the 7 errors from Stump 1 and boosted their weights. This forced Stump 2 to change its strategy and find a new rule (shifting 4.75 to 5.05) to specifically correct those errors. Even though Stump 2 made more unweighted errors (7 to 10), its job was to be a specialist that complements Stump 1. The final strong classifier is a weighted vote where the high-α Stump

1 (1.29) make the main decision, and Stump 2 (0.95) provid a powerful correction for the cases Stump 1 failed on.

4 Part 4: Gradient Boosting with XGBoost

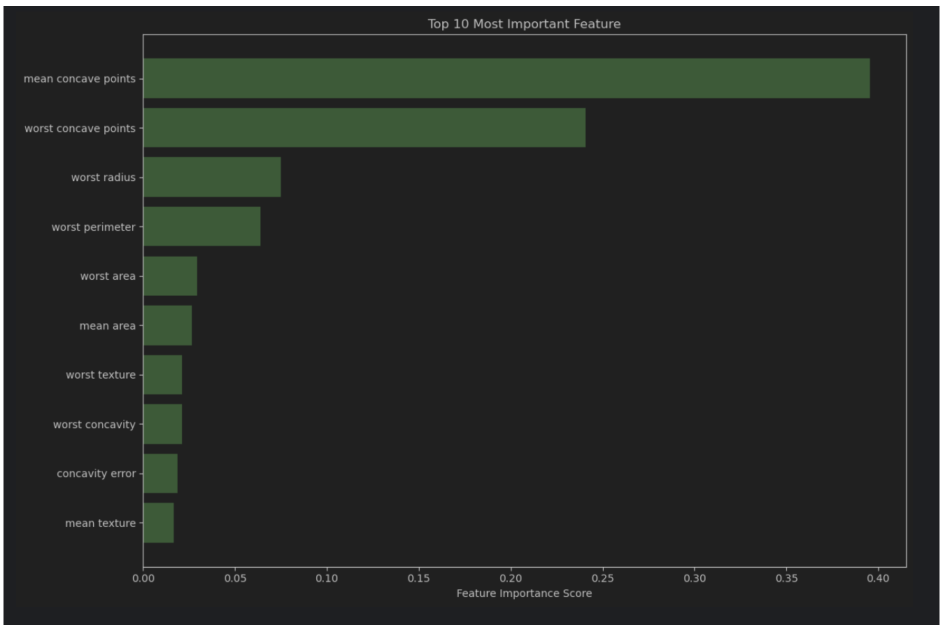
4.1 Performance Results

Model	Accuracy
Random Forest	0.9601
XGBoost	0.9580

Table 2: Model Performance Comparison

4.2 XGBoost Feature Importance

Top 10 features in XGBoost:



4.3 Analysis

The comparison between Random Forest and XGBoost:

- **Performance**: Random Forest slightly outperformed XGBoost (96.50% vs 95.80%), although both achieved excellent performance
- **Feature Importance Similarity**: Both models identified similar important feature, with 4 common features within the top 5 features
- **Different Prioritie**: While both models emphasized concave points, Random Forest prioritized worst measurements, while XGBoost gave more importance to mean measurements
- **Algorithm Difference**:

- Random Forest uses bagging and parallel tree building
- XGBoost uses boosting and sequential error correction
- XGBoost typically provides more granular feature importance scores

- **Complementary Strengths**: The high agreement on important features increases confidence in their biological relevance for breast cancer diagnosis