



# Predicting Heart Disease (Playground Series S6E2)

CS3111 - Introduction to Machine Learning

Date: February 26, 2026

Suriyabandara S.M.M. - 210626L

## 1. Final Kaggle Public Leaderboard Score

**Kaggle Username:** UOM\_210626L  
**Score:** 0.95305

## 2. Framing & Data Preprocessing

The goal of this competition is to predict the presence (1) or absence (0) of heart disease given patient data. The evaluation metric is **ROC AUC**, requiring the model to output *probabilities* rather than absolute classes.

- **Preventing Data Leakage:** Features from the training and test sets were strictly separated. No cross-contamination occurred during the preprocessing stages.
- **Target Interpretation:** The target string labels `Presence` and `Absence` were mapped to numeric labels 1 and 0 to allow the algorithms to calculate ROC AUC probabilities effectively.
- **Scaling & Normalization:** Continuous numerical features (`Age`, `BP`, `Cholesterol`, `Max HR`, `ST depression`) have widely different ranges. To ensure distance calculations and optimization steps aren't biased, `StandardScaler` was applied to put them on an equal playing field (Mean 0, Variance 1).
- **Categorical Handling (Feature Engineering):** Nominal features like `Chest pain type`, `Thallium`, and `EKG results` are represented as numbers in the data, but they lack mathematical meaning. `OrdinalEncoder` was utilized to explicitly treat them as categories, passing them into an algorithm natively built for categorical data.

## 3. Designing Experiments & Evaluation

To robustly test model improvements before submitting to the Kaggle Leaderboard, **Stratified K-Fold Cross-Validation (K=5)** was implemented.

- Since creating a random 80/20 train-test split risks placing all hard-to-predict outliers in a single fold, Stratified K-Fold ensured the ratio of Heart Disease `Presence` vs `Absence` was identical across all 5 slices of the dataset.
- The model's predictions were averaged directly on the **ROC AUC** metric, yielding a more reliable correlation to the hidden public leaderboard.

## 4. Selection and Improvement of Models

Instead of relying on a single algorithm, **Ensemble Methods** were used to combine the strengths of different models and reduce variance:

1. **Random Forest Classifier (Bagging):** Averages many deep decision trees. To enforce the *Decision Tree Tip for Pruning*, the `max_depth` was restricted to 10 and `min_samples_split` to 20, forcing the trees to generalize rather than memorize the training data.
2. **Histogram-Based Gradient Boosting (Boosting):** This algorithm iteratively builds trees, where each new tree specifically tries to minimize the errors (residuals) of the previous trees.

- **Hyperparameter Tuning:** Using `RandomizedSearchCV`, optimal parameters were identified to prevent overfitting: lowering learning rate to 0.05, restricting `max_leaf_nodes` to 20, and introducing strong **L2 Regularization** (`l2_regularization=1.0`). L2 heavily penalizes massive feature weights, creating a more robust boundary.
3. **Soft Voting Classifier:** Finally, both algorithms were combined using a Soft-Voting mechanism (weighted 85% to Gradient Boosting and 15% to Random Forest). Soft voting averages the predicted *probabilities* of both models, which perfectly aligns with the required ROC AUC Kaggle metric.