ICS 435/635

Machine Learning

Spring 2025

Assignment 3 Report

ICS 435 Name: Ju-Bin Choi

1. Problem Introduction

The goal of this assignment is to build machine learning models that predicts the class of a sample of Higgs-Boson Detection dataset. It is started in 2012, and machine learning classifiers were trained on simulation and calibration data, then used to analyze real data from the experiment. Improved machine learning methods for this type of data could enable physicists to make new discoveries faster. (Zhang and sldsrt).

My Kaggle leaderboard name is **junc**, and the handle is **jubinchoi**.

2. Data Introduction

The dataset is sourced from Kaggle's "Higgs Boson Detection 2025". It consists of:

- **Training Set:** 50000 rows and 29 columns.
  - Include 'label' as a target column in training set
- **Test Set:** 50000 rows and 28 columns containing only feature data.

For model development, the training set was split into 80% for training and 20% for validation using scikit-learn library's *train_test_split* function with *random_state of 42*.

3. Model Introduction

Since higher accuracy is desired in this task, employed ensemble models because it better predicts through bagging, boosting, and stacking methods. Thus, three modeling approaches were explored for this classification task:

- **XGBoost:** A robust gradient boosting algorithm known for its high performance and scalability. Hyperparameter tuning was conducted using the Optuna library.
- **LightGBM:** An efficient gradient boosting method optimized for speed and memory usage on large datasets, also tuned with Optuna.
- **AutoGluon (Weighted_Ensemble_L3):** An automated machine learning framework that simplifies model selection, hyperparameter tuning, and ensembling. To achieve more accurate predictions during the model selection phase, ensemble models were used. In this assignment, AutoGluon built a Level 3 weighted ensemble (weighted_ensemble_l3) that aggregates predictions from various base models. This ensemble model consists of base models of:
  - **XGBoost_BAG_L1:** A bag of 8 XGBoost.
  - **XGBoost_BAG_L2:** A bagging ensemble that consists of 8 XGBoost base models, providing increased robustness by combining multiple XGBoost predictors.

4. Data Splitting

The original training set (50000 rows and 29 columns) was split into training and validation subsets using scikit-learn's train_test_split method with a random state of 42 for reproducibility:

- **Training Subset (80%):** Used for model training.
- **Validation Subset (20%):** Used for model evaluation and hyperparameter tuning.

The test set, comprising 50000 rows and 28 feature columns, remains unlabeled and is reserved for final evaluation.

5. Data Pre-processing

The data seemed it has been standardized in advance. Additional pre-processing steps were not applied, as tree based models like XGBoost and LightGBM are insensitive to feature scaling.

6. Hyperparameters

For both XGBoost and LightGBM, key hyperparameters include:

- **n_estimators:** The number of boosting rounds (trees).
- **learning_rate:** The shrinkage rate applied during updates.
- **max_depth:** Maximum depth of trees, controlling model complexity.
- **min_child_weight (XGBoost) / min_child_samples (LightGBM):** Minimum sum of instance weights (or number of data points) required in a child.
- **subsample:** Fraction of observations randomly sampled for each tree.
- **colsample_bytree:** Fraction of features randomly sampled for each tree.
- **reg_alpha / reg_lambda:** L1 and L2 regularization parameters.
- **gamma (XGBoost only):** Minimum loss reduction required to make a further partition.

AutoGluon (Weighted_Ensembles_L3) automates hyperparameter tuning and model ensembling, integrating these decision tree–based models into a robust weighted ensemble.

7. Hyperparameter Optimization

For LightGBM and XGBoost, the hyperparameter tuning was performed using the combination of RandomizedSearchCV and **Optuna** library, combined with 5-fold cross-validation on the training subset. I first restricted the range using RandomizedSearchCV, then optimized the hyperparameter using Optuna framework. For reproducibility, a random seed and random_seed of 42 were used when optimizing hyperparameters. The optimized hyperparameters are shown below.
LightGBM:

n_estimators:	1253
learning_rate:	0.011970747911503524
num_leaves:	189
max_depth:	14
min_child_samples:	13
subsample:	0.4675509736186637
colsample_bytree:	0.8839974860053773
reg_alpha:	0.6588762410125882
reg_lambda:	1.8038339364619203
early_stopping	100

XGBoost:

n_estimators:	1693
max_depth:	10
early_stopping:	100
learning_rate:	0.010440974841363403
reg_alpha:	1.7257123586024765
reg_lambda:	1.192831196972003
gamma:	0.9795703486993589
subsample:	0.8267842957638102
min_child_weight:	6
colsample_bytree:	0.9621918234465445

AutoGluon (Weighted_Ensemble_L3):
AutoGluon was set to run for a maximum of 10,800 seconds (3 hours). The Level 3 weighted ensemble (weighted_ensemble_l3) has the following settings:

- ensemble_size: 25
- subsample_size: 1,000,000
- random_state**:** 3
- model_weights**:**
  - XGBoost_BAG_L1: 0.777778
  - XGBoost_BAG_L2: 0.222222

The base models for the ensemble are configured as follows:

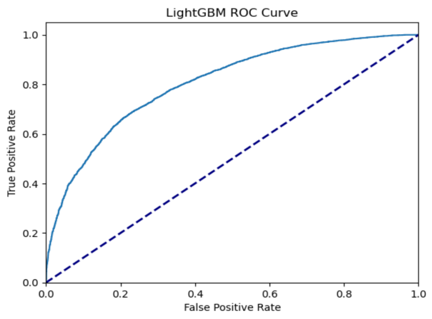
- XGBoost_BAG_L1:
  - random_state: 1
  - use_orig_features: True
    - Base Hyperparameters (XGBoost models in this bag):
      - n_estimators: 10,000
      - learning_rate: 0.1
      - n_jobs: -1
      - proc.max_category_levels: 100
      - objective: "binary:logistic"
      - booster: "gbtree"
- XGBoost_BAG_L2:
  - random_state: 2
  - use_orig_features: True
    - Uses the same base hyperparameters as above for its 8 XGBoost models.

8. Performance on Clean Validation and Test

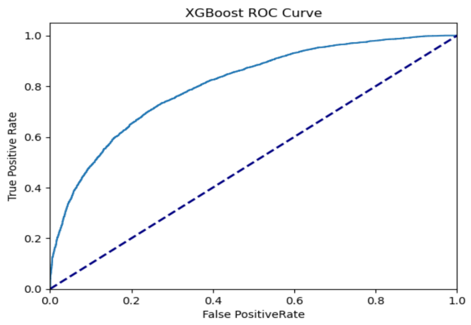
This project is evaluated based on the Area Under the ROC Curve (AUROC) metric. On the validation set, the performances of the tuned models are:

Model	AUROC
LightGBM	0.805
XGBoost	0.806
AutoGluon (Weighted_Ensemble_L3)	0.814

Validation AUROC Scores (Rounded to three decimal places)



LightGBM ROC Curve



XGBoost ROC Curve

For the actual test set, the AUROC scores are evaluated on Kaggle's test labels. The AUROC scores are based on the public leaderboard of the competition:

Model	**AUROC**
LightGBM	0.80346
XGBoost	0.80547
AutoGluon (Weighted_Ensemble_L3)	0.82024

Test AUROC Scores (Full Digits)

The code can be found in GitHub repository: https://github.com/jubinc0911/ics435_higgs_boson

9. Discussion on Generalization

Our models are trained on sample from original data, and a key challenge is ensuring that the feature distribution in the training set remains representative of the actual data encountered during testing. A feature shift may occur when the distribution of input features in the test set differs from that in the training set. Such a shift can adversely affect model performance because the models may not have learned the appropriate relationships if they encounter feature patterns that are significantly different from those seen during training.

Also, a higher volume of data might reveal feature patterns that were not apparent in the initial training set. With more data, the distribution of features might change, and new correlations could emerge. This could lead to:

- The new data introduces a significant feature shift, the performance of the existing models may degrade because they were optimized for the original feature space.
- An increase in dataset may require re-optimization of hyperparameters and possibly re-train models to adapt effectively to the new feature distribution.

In addition to that, the AutoGluon framework may be influenced by the local machine's resources and wall-clock time limitations. Since AutoGluon is limited by the available computational power and the time and the machine's inspection, its ensemble performance may vary on different machines. Thus, this can impact the overall generalization of the ensemble, and it should be considered when deploying the model in different environments.

In summary, while our current models demonstrate robust performance on the available simulated dataset, vigilance is needed for any feature shifts that might occur with new or larger volumes of data. Continuous monitoring and potential model updates will be essential to maintain optimal performance