IST 557

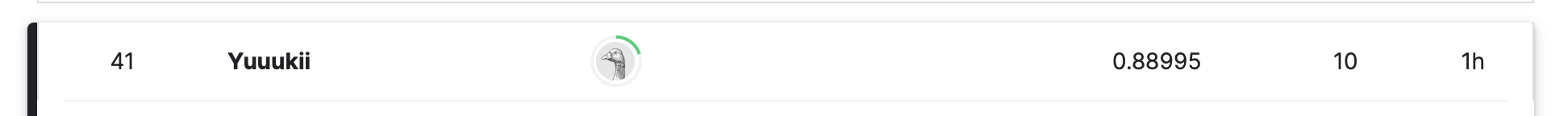
Individual Assignment 1

09/22/2023

**Yuqi Shen**

**yms5393@psu.edu**

**Kaggle account: Yuuukii**

****

Codes:

**Data preprocessing**

Although similar tasks can be handled by python using Numpy and Pandas, data preprocessing was performed in R this time (code was attached).

One-hot encoding was performed for all categorical variables. For example, sex was encoded as [1,0] for male, and [0,1] for female. Resting ECG was encoded as [1, 0, 0], [0, 1, 0], [0, 0, 1].

Distributions of the continuous variables were checked via descriptive statistics (e.g., range, mean, median, z-score), and visualizations (histogram). Within the training dataset, among all continuous variables, variable *cholesterol* was found to have 0 values, which is abnormal based on the nature of this variable. I recoded all 0 values in *cholesterol* as NA. Several outliers (absolute value of z-score greater than or equal to 3) were found in variables *cholesterol* and *resting BP*, and those values were winsorized to the value observed at z-score = ±2.8 respectively.

**Feature engineering**

Before the actual model training, a logistic-regression model was fit to explore the general linear trend between the predictors and the outcome variables using the *glm*() function in r. Observations with missing data (*cholesterol*) was removed when fitting the model. The results suggested that *sex, ChestPain, FastingBS, ExerciseAngina, Oldpeak*, and *ST\_slope* were the strongest predictors, thus these variables were included in the models.

When training the models, I also tried to include all variables, and the performance (F1 score around 0.87) was not as good as the reduced variable version (F1 score around 0.89).

**Hyperparameter tuning**

I used the *hyperopt* package to perform hyperparameter tuning.

**Performance evaluation**

Cross-validation was performed using the *cross\_val\_score* function. Specifically, “f1” was set as the scoring method to evaluate the performance of the model.

**Model building and selection**

For this assignment, I first tried XGBoost using the selected, preprocessed variables. After several attempts of hyperparameter tuning, the performance improved from the initial F1 score of around 0.86 to around 0.895. Using the best model from all the XGBoost attempts, the F1 score calculated based on test data prediction indicated okay performance (F1 score = 0.89 as shown on Kaggle, but not ideal. We compared the objective hist vs, approx., and sampling method uniform vs. gradient base, and the F1-scores suggested that for this dataset, hist performs a bit better than the approx, and uniform performed a bit better than gradient base.

Another approach that I tried was splitting the participants with and without missing data, and trained separate models for them. The rationale for doing this was that there might be underlying patterns for those who have missing data in *cholesterol*, thus training two models may have a better fit. As most observations who had missing data in *cholesterol* had *heartdisease* (=1), this model was trained based on imbalanced outcomes, thus *scale\_pos\_weight*  was added as another parameter. However, the performance did not outperform the model trained based on the full training dataset (F1 score around 0.88).

I also tried Random Forest. However, given the time limit, I was not able to fully tune the model. The performance of the random forest model did not outperform the XGBoost model, with the best F1-score observed during training around 0.88. When tuning the max\_features, I noticed that for this dataset, using the sqrt performed better than “none”.

**Future directions and ideas**

As the first time comparing XGBoost and Random Forest, I noticed the significant differences in training time, where XGBoost took around 10x time less than Random Forest. When performing hyperparameter tuning, modifying parameters seemed to have a greater impact on random forest models than XGBoost models, where random forest models kept improving even after 400 iterations. It would be interesting to keep exploring the random forest tuning, as well as some other methods such as adaboost.