# CS 150 Final Project

Ziqi Han
SIST
hanzq@shanghaitech.edu.cn

## Abstract

*In this project, we are asked to predict student performance on mathematical problems from the logs of students' interactions with a computer system, the "Intelligent Tutoring System". I used a simple mathematical method to test the correlation between different features and the final results, and selected and processed features. Finally, decision tree is used as base classifier and bagging is used as ensemble classifier to reduce variance. Finally, the optimal Super-parameters are obtained by GridSearchCV.*

## 1. Introduction

Proj provides train.csv as training data, which includes 'Anon Student Id', ' Problem Hierarchy', 'Problem Name', 'Problem View', 'Step Name', 'Step Start Time', 'First Transaction Time', 'Correct Transaction Time Step', 'End Time Duration (sec)', 'Correct Step Duration (sec)', 'Error Duration (sec)', 'Correct First Attempt', 'Incorrects', 'Hints', 'Corrects (Default)', 'Opportunity (Default)'.

Because some data category are not provided in test. csv, or too many data missing to be used directly, we finally extract 'Anon Student Id', ' Problem Hierarchy', 'Problem Name', 'Problem View', 'Step Name' as effective training data.

## 2. Feature engineering

### 2.1. Split data

Split data category 'Problem Hierarchy' in to two parts: 'Problem Unit', 'Problem Section'.

### 2.2. Label Encoding

In the train.csv, many data categories are all text, which is not acceptable in many machine learning model. And to convert this kind of categorical text data into model-understandable numerical data, there are two ways: one-hot encoding and label encoding. Since the application of onehot encoding in this proj will consume a lot of memory, resulting in insufficient memory error. So here we choose to use label encoding.

I use LabelEncoder class from the sklearn library, fit and transform the first column of the data, and then replace the existing text data with the new encoded data.

### 2.3. Data Analysis

Use heatmap to compute and visualize pairwise correlation of each feature, excluding NA/null values.

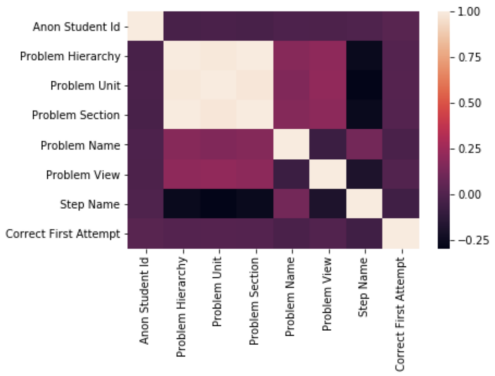


Figure 1. The heatmap of correlation between features and "Correct First Attempt"

## 3. Machine Learning Model

### 3.1. Random Forest Model

In this project, I use decision tree algorithm as the base estimator and random foreset model as the ensembl method, which is a a classification algorithm consisting of many decisions trees. The low correlation between models is the key. Uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions becasuse the trees protect each other from their individual errors.

And random forest model uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

### 3.2. Evaluating Performance

In this project, I import cross validation score ,confusion matrix, classification report and RMSE to evaluate the permormance of prediction model. The classification report shows the accuracy of your model; the roc auc scoring in the cross-validation model shows the area under the ROC curve; the RMSE shows differences between predicted values and observed values or the quadratic mean of these differences.

the ROC curve plots out the true positive rate versus the false positive rate at various thresholds and the roc auc scoring in the cross-validation model shows the area under the ROC curve;

- **Confusion matrix**

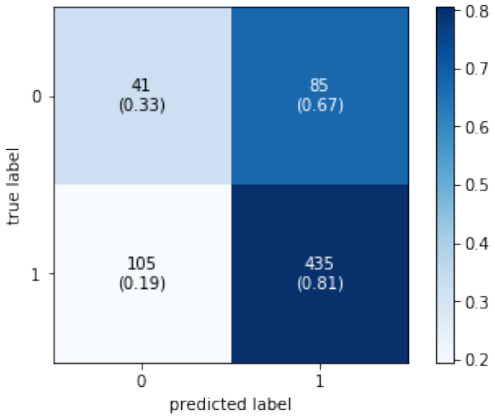


Figure 2. The confusion matrix of random forest model

From the confusion matrix, we see that the Random forest classifier got the following results:

  - Out of the 540 actual instances of 1 (Correct First Attempt), it predicted correctly 435 of them;
  - Out of the 126 actual instances of 0 (Correct First Attempt), it predicted correctly 41 of them.

- **Classification report**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.28 | 0.33 | 0.30 | 126 |
| 1 | 0.84 | 0.81 | 0.82 | 540 |
| accuracy | | | 0.71 | 666 |
| macro avg | 0.56 | 0.57 | 0.56 | 666 |
| weighted avg | 0.73 | 0.71 | 0.72 | 666 |

  - **Precision :**Prediction accuracy when the model predicts the positive result.
  - **Recall :**Prediction accuracy when it is actually the positive result.
  - **f1-score :**harmoonic mean of presion and recall.

- **AUC score**

  Applying cross validation on the model with cv = 5, AUC scores are 0.67175239, 0.66350747, 0.59698572, 0.65232918, 0.66437975 and Mean AUC Score is 0.6497909025808409.

- **RMSE**

  RMSE = 0.534121039920059

## 4. Improved Model

### 4.1. Tune hyperparameters

The way to improve the performance of the model is to tuned the hyperparameters by conducting an exhaustive grid search with GridSearchCV from sklearn. An exhaustive grid search takes in each hyperparameters and tries every single possible combination of the hyperparameters with optional cross-validations.

And in this project, I focus on 3 hyperparameters: n_estimators, max_features, and max_depth. Compared to Randomized search, exhaustive grid search is more time-consuming but the result is more accurate. The resulting best hyperparameters are as follows: n_estimators = 200, max_depth = 11, , n_estimators = 200.

### 4.2. Evaluating Performance

- **Confusion matrix**

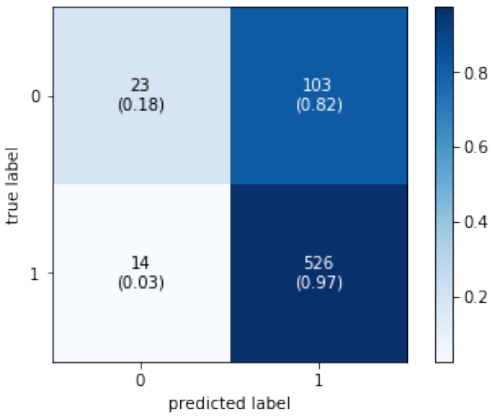


Figure 3. The confusion matrix of improved random forest model

- **Classification report**
- **AUC score**

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.62 | 0.18 | 0.28 | 126 |
| 1.0 | 0.84 | 0.97 | 0.90 | 540 |
| accuracy | | | 0.82 | 666 |
| macro avg | 0.73 | 0.58 | 0.59 | 666 |
| weighted avg | 0.80 | 0.82 | 0.78 | 666 |

Applying cross validation on the model with cv = 5, AUC scores are 0.72871266, 0.72405481, 0.60117052, 0.70803609, 0.73211901 and Mean AUC Score is 0.6988186171563655.

- **RMSE**

  RMSE = 0.4191368221424547

## 5. Model Analysis

Hyperparameter tuning help the model improve with classification. Compared to original model, improved model has lower RMSE, higher accuracy and AUC score. However, by observing the confusion matrix, we can find that though the total accuracy raised but the predict accuracy for 0 declined.

### 5.1. Unbalanced Dataset

Through analysis, I think the reason is the classes of traindata is unbalanced. The classes are either '1' or '0' where the ratio of class '1':'0' is 3.6:1.

The unbalanced classes create a problem due to two main reasons:

- Since the model never gets sufficient look at the underlying class, can not get optimized results for the class which is unbalanced in real time.
- A test sample is to have representation across classes since the number of observation for few classes is extremely less.

## 6. Future work

### 6.1. Feature Engineering

There are two more features 'KC' and 'Oppurtunity' I haved used and I can use feature combination to optimize the model.

### 6.2. Balance Dataset

There are three main approaches to solve the unbalanced dataset problem:

- **Undersampling**
- **Oversampling**
- **Synthetic sampling(SMOTE)**

## References


- https://medium.com/@hjhuney/implementing-a-random-forest-classification-model-in-python-583891c99652
- https://medium.com/all-things-ai/in-depth-parameter-tuning-for-random-forest-d67bb7e920d
- https://towardsdatascience.com/optimizing-hyperparameters-in-random-forest-classification-ec7741f9d3f6
- https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridS