CS150A Database Course Project

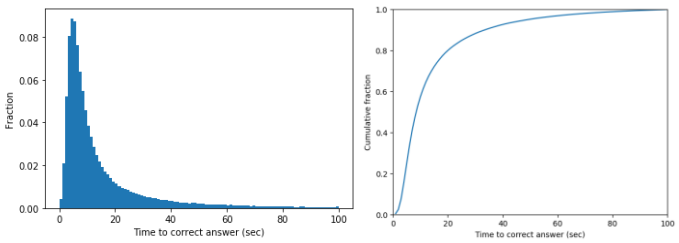
Chen Boke
ID: 2020533035
chenbk@shanghaitech.edu.cn

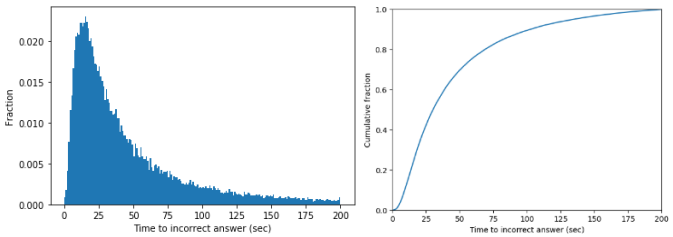
Guideline

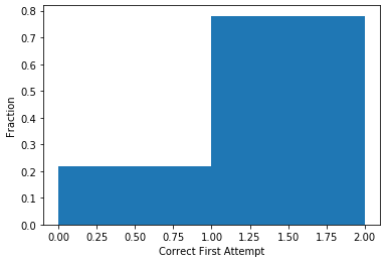
Compared with developing a novel machine learning algorithm, building a machine learning system is less theoretical but more engineering, so it is important to get your hands dirty. To build an entire machine learning system, you have to go through some essential steps. We have listed 5 steps which we hope you to go through. Read the instructions of each section before you fill in. You are free to add more sections.
If you use PySpark to implement the algorithms and want to earn some additional points, you should also report your implementation briefly in the last section.

1 Explore the dataset

As we can presume from the dataset, different students have different rate of progress when it comes to their encountered issues. This will lead to the result that students' endeavors affecting their first attempt. So, the pressure should be put on their previous try-outs. And before we train the models, we need to focus on the analysis of the dataset. There are about a dozen of columns which can be divided into two parts, classification feature and numerical characteristics. Due to the massiveness of this dataset, I mainly focus on the following columns: Correct step duration (which has been well introduced in the data exploration file), Error step duration and correct first attempt. The correct step duration has a mean value of 17.9, a minimum of 0 and a maximum of 1067, while the error step duration has a mean value of 60.5, a minimum of 0 and a maximum of 1888. The mean of the correct first attempt is 0.8, and it has a minimum of 0 and a maximum of 1 of course. The distribution of correct answer, incorrect answer and the first correct attempt is as follows:







Obviously, there're some missing values, mainly in the form of columns, such as step start time, first transaction time, correct transaction time, step end time, step duration, correct step duration, error step duration etc.

2 Data cleaning

According to data exploration, there're certain missing columns which needs further insights. I plan to inspect on columns such as student id, problem name, problem view and correct first attempt etc. The missing values, namely the NaN values, in the columns I inspect, do not exist for no reason, they means the output should be none. And there're some other missing values need me to fill in. As for the outliers, I inspect on all the columns with numerical characteristics, almost all the data are within the proper range.

3 Feature engineering

As mentioned above, the missing columns in testing data such as step start time, step end time, step duration, corrects, hints, incorrects are not useful for the training process, so I decided to abandon these columns. After that, I can divide the column problem hierarchy and calculating into problem unit; problem section and feature compression; feature extraction for convenience. For all the remaining categorical feature columns, we need to encode them and generate the data matrix, so I encode them follow the order of natural growth by 1, which is the simplest way to encode. As for the feature compression part and feature extraction part, I calculate the number of separated knowledge components and the average of separated opportunities, and generate more features for predicting respectively. To be more specific, I chose to calculate the features like personal CFAC, personal CFAR, problem CFAR, KC CFAR and so on (CFAR stands for correct first attempt).

The whole list of features are as follows: Anon student ID, problem name, problem view, problem unit, problem section, step name, KC count, average opportunity, CFA, personal CFAC, personal CFAR, problem CFAR, unit CFAR, section CFAR, step CFAR and KC CFAR.

4 Learning algorithm

After searching for the info, I focus on the following algorithms: Basic Decision Tree, Gradient Boosting Decision Tree, AdaBoost, XGBoost, Random Forest, KNN, Bagging and Logistic Regression. Despite the truth of 'the more the better', I have to choose the algorithm of best performance and fits. And in order to decide the best algorithm, I test them by the default parameters, and the result is shown below.

Name	RMSE
Basic decision tree	0.5327
Gradient decision tree	0.4101
AdaBoost	0.3914
XGBoost	0.4209
Random forest	0.3736
KNN	0.4488
Bagging	0.3943

While the Random Forest may seem to outrun all the other algorithms, I decide to compare them under different parameters in the next part to reach a better result.

5 Hyperparameter selection and model performance

I use the GridSearchCV to decide the parameter, the result is as follows:

Parameters	Range	Best Value
n_estimators	(10, 200, 10)	190
max_depth	(5, 21)	15
max_leaf_nodes	(100, 1000, 10)	900
min_samples_split	(2, 52, 2)	22

The chart above is the hyperparameter choice for Random Forest, and the RMSE of applying the hyperparameters is 0.3535, which clearly outperforms the sub-parameters (original parameters) of 0.3642.

6 PySpark implementation (optional)

PySpark is applied in the process of feature engineering in the form of distributed processing methods (SparkContext & SQLContext), which bounce the efficiency enormously.