CSEDM 2019 CHALLENGE

Abhinay Natti
abhinay@iitk.ac.in

Dheeraj Athrey
dheerajb@iitk.ac.in

February 17, 2019

ABSTRACT

The Challenge in CSEDM Challenge is predicting students' future performance based on their previous data. We have extracted features both subject based and problem based, example, based on their history of making syntax and semantic errors. We have tried several models to improve the accuracy.We tried models using features extracted from the code based on syntax errors, semantic errors, etc. We also used user and problem performance statistics. We used several popular Machine learning models and the one that gave us the best results was a Logistic Regression classifier. We achieved an accuracy of 75.96 % and an f1-score of 0.77 .

1 Features Used

We have used the features that were given in the example problem like :
A subjects' performance in the previous questions can be used as features.
We considered subject specific features

- **pPriorCorrect** - % of the previous problems the user has done correct,
- **pPriorCompleted** - % of the previous problems the user has successfully completed and
- **priorAttempts** - the total number of prior attempts.

We also chose problem specific features

- **pCorrectProblem** - mean of subjects who did the problem correct in first attempt,
- **pMedian** - median number of attempts taken to solve the problem.

Then we tried features based on whether the code contained conditionals like if and also loops like while and if. But since the problems used were basic problems , this didn't help because most of the codes have same structure. Next we tried by checking the length of the code written as a feature. The latter one also failed because of the same reason that most of the codes were similiar. Further, features that we extraced from the code and gave some good results are :

- **pSubjectSyntaxErrors** - % of the problems which had syntax error committed by the subject,
- **pSubjectSemanticErrors** - % of the problems which had semantic error committed by the subject
- **pProblemSyntaxError** - % of the subjects who had committed syntax errors on that problem
- **pProblemSemanticError** - % of the subjects who had committed semantic errors on that problem

2 Implementation

We merged the files MainFile.csv and CodeState.csv so that we can do operations on the code. The problem is essentially a binary classification. We had to predict whether a user would be able to solve a question in his first attempt. Coming to the features, we used the python module ast to check the code for syntax errors. The codes which already have syntax errors are deemed to be failing in the first attempt. We trained the model by giving those codes where there won't be any syntax errors in hopes that the model can learn the semantic features rather than the syntactic features. Then features like prior performance statistics per user and per problem were used along with data about syntax and logical errors. Using all these 9 features , we generated a feature vector and used models like Logistic Regression, Decision Tree Classifier and an MLP Classifier. The accuracies obtained were boosted by 1 % using bagging classifier of sklearn. The best accuracy obtained was using Logistic Regression and is equal to 75.96 %

3 Results

.	Logistic Regression	MLP Classifier	Decision Tree Classifier
Accuracy	0.7596	0.7513532842306	0.756189060684801
f1-score	0.7709	0.764259163869007	0.761797482530667
Precision	0.771691	0.75651412295434	0.779397629213054
Recall	0.511809	0.780464327899491	0.751160103740436
Kappa	0.7993	0.491773763582876	0.502617349349608

4 Acknowledgements

We were guided by Prof. Arnab Bhattacharya, IITK.