Supporting Quality Teaching using Educational Data Mining based on OpenEdX Platform

Wan Han[1,2], Ding Jun[1], Gao Xiaopeng[1,2], Liu Kangxu[1]
School of Computer Science and Engineering[1], Honors College of Beihang University[2]
Beihang University
Beijing, China
{wanhan, dingjun, gxp, liukangxu}@buaa.edu.cn

***Abstract*—Our lab-based small private online course (SPOC) combined online resources and technology with engagement between faculty and students based on OpenEdX platform. It worked with an auto-grading submission system which could reduce the instructors' burden of evaluation and provide better learners' experience. Different study behaviors were observed from the system tracking logs. Identifying at-risk students becomes timely important in SPOC, and the early prediction can help instructors provide proper supports. In this paper, we focused on extracting features from students' learning activities and study habits for building machine learning models to predict students' performance. We conducted experiments to compare feature importance, and the results showed that study habits related features had played more important role in predicting students' performance. 34 predictive features extracted from Computer Structure Course in Fall 2016, and our model achieved an ROC (Receiver Operating Characteristic Curve)-AUC (area under the curve) in the range of 0.927 - 0.984 when predicting the performance. Our evaluation showed that data mining is useful in education especially when examining students' learning behavior in online environment, and could support quality teaching. In the next course iteration, we will do A/B testing to determine efficacy for subsequent interventions in a SPOC.**

Keywords—educational data mining; quality education; study behavior; data analytics

I. Introduction

Learning observation for educational analysis and constructive appraisal purposes has been found relevant. It is important to develop and adpate tool to observe the learning acitivities inside and outside classrooms, especially for the blended education. Furthermore, it is a necessity to develop capacities for quantitative researches. But, what learning related data should we collect? How could we gather these educational data? How could we extract meaningful information from them? And finally, how could the measurement improve the teaching practices?

The ability of identifying students who are at risk of failing a course is important for teachers to take corrective actions. Some general approaches have been developed to solve this problem.

Since the dropout rate is unexpectedly high, how to identify at-risk students in Massive Open Online Course (MOOC) has received increasing attention. [1] viewed the dropout prediction problem as a sequence classification problem. They considered that dropout probability of a student at the current time step can be dependent on his/her engagement at the previous time step, and proposed a nonlinear state space model to solve the problem.

Hu [2] developed an early warning system based on learners' interaction with Learning Management System (LMS). Their studied interactions included information such as login, time duration, and metadata concerning homework assignments. But their course required students to watch videos in specific time periods. To build their early warning system, they generated three datasets to create different periods to study, and applied classification techniques.

He [3] explored an accurate early identification of students who are at risk of not completing courses. They build predictive models weekly and two transfer learning logistic regression algorithms which would be practical for deployment in MOOCs.

Srilekshmi created a model to identify learners who are at risk using data of students who have already completed the course [4]. Furthermore, they would find what type of the learner was once he is identified as at risk. The type may vary from a visual learner to auditory to kinesthetic learner. Finally, they suggest methods to improve learner performance according to the learner type.

Ren [5] formulated a personalized linear multiple regression model to predict the homework grades for a student participating within a MOOC. Their model is real-time and tracks the participaton of a student within a MOOC from click-stream server logs. And experiments showed this model helped in identify the key features that associated with the learning behaviors.

Previous studies mainly focused on predicting students' failure of online course. [6] showed that it is possible to predict students' performance of offline courses from the access records on general websites. Their feature set includes students' access records on websites, scores of another course delivered in last semester, and the time spent on online videos.

Some researchers used features related to the students' past course performance and their interactions with learning management system (LMS) to build a predictive model for all students. [7] showed it was important to understand students' behavior to be able to guide them through all the studies to graduate. They used collaborative filtering (CF) methods for student modeling, and confirmed the hypothesis that students' knowledge can be sufficiently characterized only by their previously passed courses. For each investigated student, they searched for most similar students who enrolled in the same course in the last years, and predicted the students' performance based on their study results.

As Elbadrawy mentioned in [8], previous methods estimate a single model for all students based on their past course performance and interactions with LMS, or models that do not consider LMS interactions. These methods failed to exploit fine-grain information related to a student's engagement and effort in a course. Compared to single linear regression model, [9] showed that their collaborative multi-regression model performs better prediction accuracy. Their results showed that the features related to viewing the course material and previous performance highly contribute to the predicted grades.

Conijn analyzed 17 blended courses with 4989 students using Moodle LMS [10]. Their predictive modeling results strongly vary across courses, thus the portability of the models across courses is low. They emphasize the need to include more specific theoretical argumentation and additional data sources other than just the LMS data.

Shaymaa [11] examined the impacts of teacher interventions on students' attitude by analyzing the comment data after every lesson. Their study proposed a method for building a prediction model that represents students' activities, situations and viewpoints. They classified words in the comments into six attribute types and indicated the most important types which affect the prediction results.

Laboratory-based Computer Structure is a second-year course offered in School of Computer Science and Engineering at Beihang University. In this course, we helped students to write a MIPS CPU using Verilog-HDL to deepen their understanding of the computer structure. We conducted a SPOC based on OpenEdx platform with on-campus students when they were doing the projects of Computer Structure course. This blended course maximized the opportunity for application of lessons through practice with other learners and instructors.

On the other hand, the instructors are enabled to tracking all the event (such as watching videos, working on homework, attending quizzes, and activities in the discussion forum) recorded in the OpenEdX's server log. Personal study differences including follow features: some students would attempt the quiz or exercise immediately after class, and someone may make the first attempt just prior to the deadline. Some students make subsequent submissions to the auto-grading submission system based upon the feedback received to improve the homework quality, and someone may just submit once without any further attempts. Furthermore, we can tell students into different types: attempt the homework soon after it release or not, submit homework in timely fashion VS at last minute fashion, and review if the submission is incorrect or not.

Prediction of students' academic performance is a worthwhile task, as it may provide necessary help to at-risk students as early as possible. In other words, we need to be able to predict whether a learner will fail when the course proceeds, and we want the grade their final performance.

In this paper, we elaborate how to build an external grader based on OpenEdX platform to integrate an auto-grading submission system. Our models capture personal student differences will be introduced to predict the learners' performance using the additional features we mentioned above. Then we presented which are the most important features when perform the prediction.

II. External Grader using XQueue

As shown in [12], we introduced the curriculum evolvement from Fall 2013 to Fall 2015. In Fall 2016, we elaborated the digital system related content using Logisim, MIPS assembly language using MARS, and Verilog-HDL.

There are 3 tutorials (from week 1 to week 6) helped students to grasp the fundamental knowledge points and aiding tools. And project started from week 7.

In the first 3 projects (P0, P1, and P2), learners need to build CPU components and learn how to code and debug assembly programs. In P3, they need to design a single cycle CPU supports the MIPS-Lite 1 instructions (addu, subu, ori, lw, beq, and lui) using Logisim. And then, they need to design a single cycle CPU supports the MIPS-Lite 2 instructions (MIPS-Lite1, j, jal, and jr) in P4. From P5 to P6, they need to design a pipelined processor with full hazard handling to support MIPS-Lite2 and MIPS-C3 instructions. For the students who passed the checkpoint of P6 could choose to finish P7, which need them to build a micro-system including MIPS-processor, bridge and timer based on MIPS-C4 instructions. Final project P8 is building a micro-system based on P7, 8 bit-LED segment, 32 bit-toggle switch and RS232-communicate-system.

The course was organized by 8 projects in Kung Fu-style from easiest to the hardest. Suppose the student starts project 3 at the beginning of week n, then he need to finish the design a single cycle CPU supports the MIPS-Lite 1 instructions before the checkpoint of week n, usually at the end of that week in class. During the in-class session, he need to add one more instruction based on the design he accomplished that week. If he/she succeed in the testing, then there will be a face-to-face discussion between the learner and instructor or TA. In the discussion, instructor need to decide whether the learner had mastered related content at corresponding project level, then he/she could step in the next project.

A. Testing Requests

In Fall 2016, there were 377 students registered our course. As shown in Fig. 1, during our in-class checking session on Nov. 4th, there were 1807 testing requests that had been submitted to the platform during 13:30:39 to 15:43:20 on November 4th.

On the other side, the log record from OpenEdx showed that there were 15465 evaluation requirements in P0, 12068 evaluation requirements in P3, and 11993 evaluation requirements in P5 when the students prepared each project at home during Fall 2016 semester.

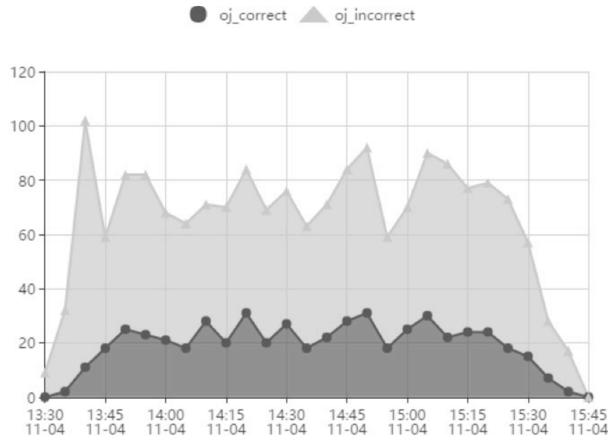

Fig. 1. Evaluation requirements during in-class checking session

We used auto-grading submission system to reduce the burden of teacher, and return feedback to the student immediately by assessing a student's submission automatically. Learners could resubmit their homework after the improvement until the final deadline. The most important is that this system is likely to better learner's experience when the student allocates time to resubmit and get feedback from the system. During last two years' practices, this flipped classroom could improve learning and student outcomes.

B. Building an External Grader

The external grader is a service provided by OpenEdX. It could receive learners' responses to a problem, process those responses, and return feedback and the grade to the OpenEdX platform.

As shown in Fig. 2, we used XQueue as the interface for LMS to communicate with external grader services. The learner either attaches a Verilog-HDL file or a MIPS assembly file, then selects 'Check'. The external grader pulls the code from XQueue and runs the judge scripts. After the judge process, the external grader returns the grade for the corresponding submission to XQueue, and XQueue delivers the results to the OpenEdX LMS. Finally, the learner will see the problem results and the grade.

The inputs to the external grader is a JavaScript Object Notation (JSON) object with following keys: student_info (user ID and submission timestamp), grader_payload (problem id and problem type), callback url for xqueue to link results to the corresponding submission, and file that were submitted by the learner.

After running the judge script, the grader will return information using a JSON response. The response contains the score and any message for each test case.

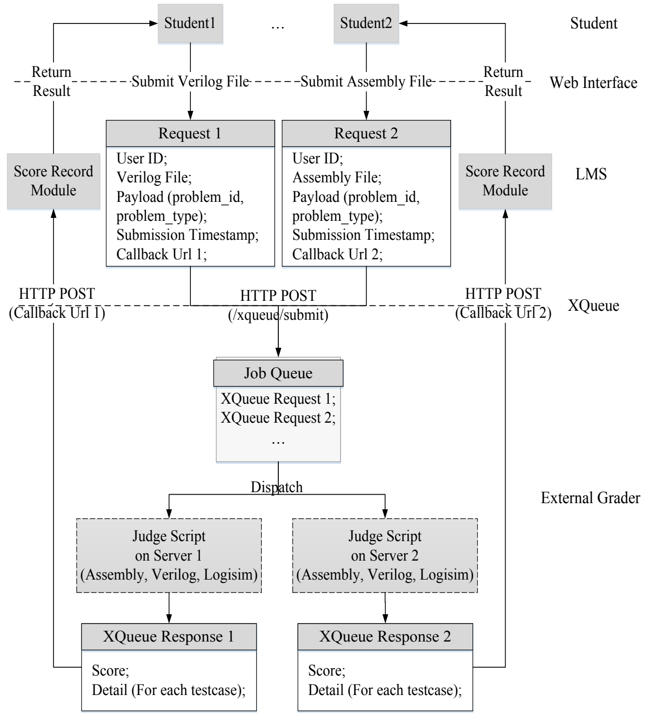

Fig. 2. The external grader workflow

III. FEATURES ENGINEERING

As shown in [13], there are different events initiated by learners, which are generated by their interactions with the OpenEdX platform. These events could be grouped into following types: resources interaction events (video, textbook, course navigation), problem interaction events, discussion forum events, survey events and so on.

In our model, we focus on these events to extract the features.

A. Events in the Tracking Logs

When users stream video, the following events will emit: 'load_video', 'pause_video', 'play_video', 'show_transcript', 'speed_change_video', 'stop_video' and so on.

Another import type we focus on is the problem interaction events, which are emmitted if interact with core problem types: 'problem_check', 'problem_show', 'problem_save', 'problem_reset', and 'problem_graded'.

The last part events we cared is discussion forum related events: 'comment.created', 'response.created', 'response.voted', 'searched', 'thread.created' and 'thread.voted'.

B. Feature Definition

As shown in TABLE I. , the first group of features are extracted from the OpenEdX's tracking logs, including time spent on the resources and problem interaction character.

TABLE I. FEATURES EXTRACTED FROM TRACKING LOGS

	NAME	Definition
x2	total_duration	Total time spent on all resources
x3	number_forum_posts	Number of forum posts
x4	average_length_forum_post	Average length of forum posts
x5	number_distinct_problems submitted	Number of distinct problems attempted
x6	number_submissions	Number of submissions
x7	number_distinct_problems_ submitted_correct	Number of distinct correct problems
x8	average_number_submissio ns	Average number of submissions per problem (x6/ x5)
x9	observed_event_duration_p er_correct_problem	Total time spent /number of distinct correct problems (x2 / x7)
x10	submissions_per_correct_pr oblem	Number of problems attempted / number of correct problems (x5 / x7)
x11	average_time_to_solve_pro blem	Average time between first and last problem submissions for each problem (average(max(submission.timestamp) - min(submission.timestamp) for each problem in a week))
x12	observed_event_variance	Variance of a student's observed event timestamps
x13	max_observed_event_durat ion	Duration of longest observed event
x14	total_video_duration	Total time spent on video resources
x15	total_etext_duration	Total time spent on etext resources

[14] showed complex predictive features from crowd proposed, which require relative comparison or temporal trends, is one important contributor to successful prediction.

As shown in TABLE II. , we extracted some features that proposed by crowd, such as ‘number_forum_responses’, ‘number_submitions_correct’, and ‘average_pre_deadline submission_time’. Furthermore, we could calculate the percentage of the total submissions that were correct (x207) using feature x6 and x7, which were described in TABLE I. .

TABLE II. CROWD PROPOSED FEATURES

	NAME	Definition
x201	number_forum_response s	Number of forum responses
x202	average_number_of_sub missions_percentile	A student's average number of submissions / the average of all the students' submission
x203	average_number_of_sub missions_percent	A student's average number of submissions / maximum average number of submissions
x204	pset_grade	Number of the week’s homework problems answered correctly / number of that week’s homework problems
x205	pset_grade_overtime	Difference in grade between current pset grade and average of student’s past pset grade
x206	number_submissions_fin al_correct_problems	Number of submissions for final correct problems
x207	correct_submissions_per cent	Percentage of the total submissions that were correct (x7 / x 6)
x208	average_pre_deadline_su bmission time	Average time between a problem submission and problem due date over each submission

TABLE III. LEARNING HABITS RELATED FEATURES

	NAME	Definition
x301	total_etext_time_after_i ncorrect_submit	Etext access time after the problem submit incorrect
x302	total_video_time_after_i ncorrect_submit	Video access time after the problem submit incorrect
x303	total_etext_time_before _submit	Etext access time before the problem submit
x304	total_video_time_before _submit	Video access time before the problem submit
x305	average_time_between_ problem_submission	Average time between problem submissions(average((max(submissi on.timestamp) - min(submission.timestamp)/(proble m submissions)) for each problem in a week))
x306	time_on_problem_atomi c	Sum of all time intervals dedicated to the problem
x307	time_on_problem_mole cular	Time between first problem_get to last problem_check
x308	problem_finish_time_pr e_start96h	Average(project issue time + 96h - problem finished time) for each problem learner finished correctly in the first 96h after the problem issued
x309	problem_finish_time_pr e_deadline48h	Average(problem due time - problem finished time) for each problem learner finished correctly in the last 48h before the problem due
x310	time_first_visit	Min(first problem_get time , first etext access time) - project issue time
x311	average_time_till_first_ check	Average of all problem the time between Problem_first_check and problem_first_get
x312	discussion_duration_aft er_incorrect_submit	Total discussion duration after incorrect submitssion

We also extracted students’ learning behavior related features as shown in TABLE III. :

- Feature x301 to x302 indicate whether the learner would review resources if submitted a wrong answer;
- Feature x303 to x304 demonstrate the time that a learner spent on the resources before submit a problem;
- Feature x305 shows whether the student just resubmit another answer quickly when the former one is incorrect.

On one hand, we calculated the time a learner spend on one problem:

- The atomic time (x306) is the sum of all time intervals dedicated to the problem;
- The molecular time (x307) counts the time a learner spend on related resources to fix the incorrect answer together with the time spend on just solving the problem. It is the time between first problem_get to last problem_check.

On the other hand, we used some features to partition students into different types:

- Feature x308 and x309 reveal students who submit homework in timely fashion or at last minute fashion;
- Feature x310 and x311 show students who attempt to do the homework soon after the project has been released or not;
- Feature x312 indicate how long the learner spent time in discussion forum if the submission is incorrect.

Additional, we also used the GPA of freshman year to present the ability of learner at the beginning of the week 06, since our first checking point is at week 06.

C. Feature Assembling

As shown in Fig. 3, we assembled the features from $week_1$ to $week_n$ to form the feature of $week_n$. And then predict the performance at the end of week 16 using these features.

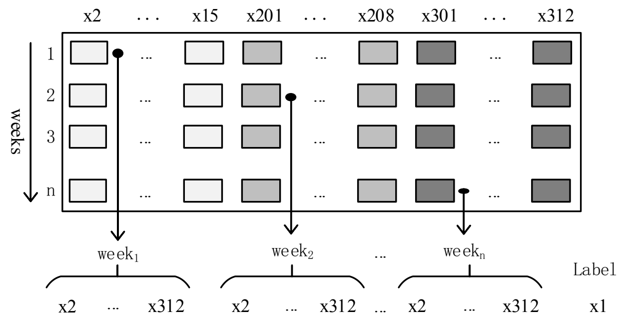

Fig. 3. Feature flattening process

IV. PREDICITING PERFORMANCE

In our predicting model, we defined feature x1 as learner's final performance at the end of week 16. As shown in section II, there were 9 projects in Fall 2016 semester, and if students pass the in-class checking session of project 5, their final performance is succeeded (x1 equals to 1, otherwise x1 equals to 0).

A. Model Selection

We used gradient boosting decision tree (GBDT) classifier to build the predicting mode based on an open source machine learning library scikit-learn, which includes an easy interface for cross validation and feature normalization.

The reason why we chose GBDT algorithm has two aspects:

- Compared to Regression/ Bayes/ SVM model, GBDT allows the combination of different features to have different discriminant.
- GBDT has inherited the characteristics of the boosting algorithm with low degree of overfitting.

B. Model Evaluation

In classification problems, using terms 'true positives (TP)', 'true negatives (TN)', 'false positives (FP)', and 'false negatives (FN)' to compare the results of the classifier under test with trusted external judgements. Here terms 'positive' and 'negative' refer to the classifier's prediction. And terms 'true' and 'false' refer to whether the prediction corresponds to the external judgement (observation).

Receiver operating characteristic (ROC) is a widely-used metric of classifier performance. The ROC curve represents the range of possible probability of False Alarm (pFA) and probability of Detection values (pD), which are given by confusion matrix quantities.

- pD is equal to TP / (TP+FN), which represents how good a classifier is at finding the positive examples.
- pFA is equal to FP / (FP+TN), which represents how likely a classifier is to mistakenly classify a negative example as a positive one.

The area under the curve (AUC) is produced by the integral of the ROC curve. And it is the primary metric we used to evaluate the predictive model.

Fig. 4. The 10-fold cross validation

As shown in Fig. 4, the dataset is randomly divided into 10 partitions (folds). Cross validation then builds 10 models, with each model is constructed using 9 folds, and the model is evaluated using the unused fold. This cross validation helps to protect against over-fitting, and provides an alternative and complimentary evaluation metric to the test dataset.

Here we employed 10-fold cross validation and use the average ROC-AUC as the evaluation metric.

C. Experimental Setup

We executed the following steps for every week to perform GBDT analysis:

- Assembling the features as described in section III.C;
- Dividing the data into 10 folds;
- As shown in Fig. 4, training a GBDT model for turn i, and evaluating the model using AUC score ti;
- Evaluating the model using the mean of ti (i = 1..10)

D. Experimental Results

As shown in Fig. 5, the x-axis indicates which data (from week 01 to week x) are used to do the prediction, and y-axis shows the ROC-AUC value.

When the tutorials parts finished in week 06, we used the data from week 01 to week 06 with their freshman year's GPA to predict the performance, and received the ROC-AUC metric is 0.936.

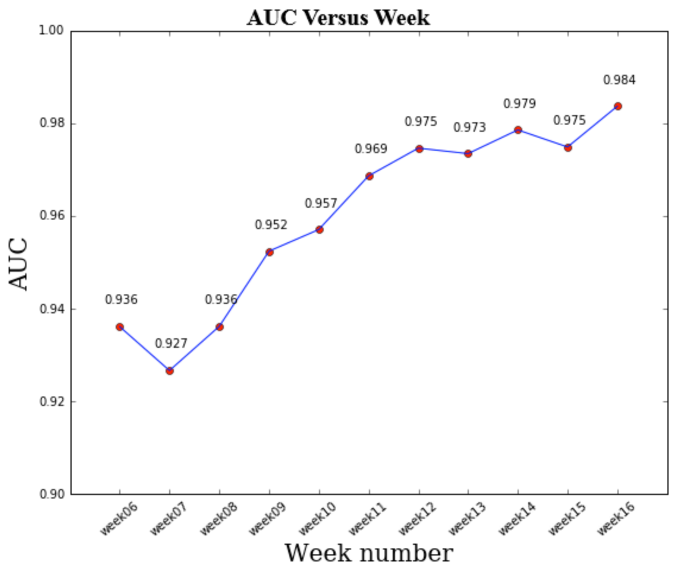

Fig. 5. ROC-AUC results for predict stduent performance

We used data from week 06 to week 12 to predict whether the learner succeed in the final week 16, and the ROC-AUC metric is 0.975 using 10-fold cross validation.

Some student passed the in-class checking session of project 5 and decided not to continue other projects after week 12. From week 12, the ROC-AUC value still maintains an upward trend, but the increasing rate declined.

We can surmise that the extracted features are capable of predicting performance in each week timely, especially when the prediction week is near the final week 16.

E. Feature Importance

Assume that importance of feature i in week j is F_{ij}, N is the number of week we used to predict student final score. Then WF_i is the importance of feature i in the experiment as in (1):

$$WF_i = \frac{\sum_{j=1}^{N} F_{ij}}{\sum_{i=1}^{34} \sum_{j=1}^{N} F_{ij}} \quad (1)$$

After quantifying importance for the feature of each experiment, we use the mean number as the week-invariant feature importance.

The feature importance score in our predictive model is shown in Fig. 6.

The higher the score indicates that the feature is relatively more important in the predictive model.

We took features in TABLE I. into considerations, and found that freshman year's GPA, feature x2 and feature x12-x15 extracted from the tracking logs got higher feature importance score:

- Here freshman year's GPA could represent the learner's ability when they enter the course.
- Feature x2 measures the time a learner spent on all the resources.
- This was the first year that we transferred all the lab to the OpenEdX platform. We prepared videos for the tutorials part, and e-texts were the main parts of resources when learners entered the project parts from week 07. So here feature x15's score is higher than feature x14.

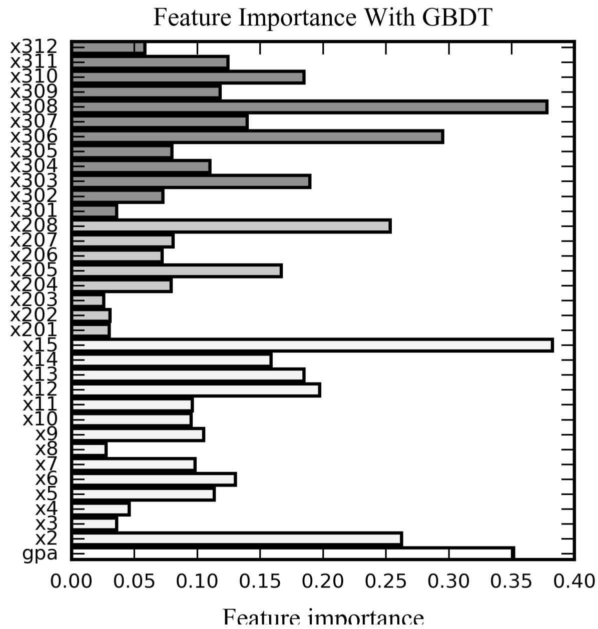

Fig. 6. Feature importance when predicting stduent performance

- The smaller value of feature x12 reflects the learner may access the course at a fixed period, and they may have a good study plan. And feature x13 means the frequency of interaction.

On one hand, complex predictive features from crowd proposed in TABLE II. , which require relative comparison or temporal trends, got higher importance score:

- Feature x205 and feature x204 are all related to the pset grade. But feature x205 imports temporal trends, and it became more important contributor to successful prediction;
- Feature x208 and feature x309 are all related to the finish time before deadline. But feature x208 is a relative comparison, and it received higher importance score.

On the other hand, we found that some learning habits related features in TABLE III. got higher scores than some simple features extracted from the tracking logs and those proposed by crowd in TABLE I. and TABLE II. :

- The features that reveal the learning enthusiasm got higher score, such as feature x308 and feature x310.
- The features that indicate study method are more important, such as feature x303 and feature x311.

- The features that show the time spent on the course are play more important role in the predicting, such as feature x306, feature x307, and feature x309.

V. Conlusions

Educational data mining is an emerging method to extract knowledge from different data within academia context. In this paper, we showed how to identify at-risk students in a SPOC using educational data mining method.

Our experiments on Fall 2016 course data showed that the GBDT predictive model with learning habits features to predicate the final performance could achieve ROC-AUC in the range of 0.927- 0.984.

The contributions compared to other studies could summarized as follows:

- We extended the OpenEdX platform to auto-judge the submission of Verilog-HDL, Logisim, and MIPS assembly projects with an external grader using XQueue scheme.
- Based on the click stream logging data from the OpenEdX, we extracted the complicated features which could represent the students study habits.
- Using 10-fold cross validation showed that we could help instructors design interventions towards the learners who are at-risk.

In the future, we will deploy our predictive models in the next course iteration, and do A/B testing to determine efficacy for subsequent interventions in a SPOC. Furthermore, we will focus on how to help student improve their performance, including following aspects:

- Improve the online resources, refine the knowledge granularity for each project.
- Improve the external grader to identify the error reason and response more useful information back to students.
- Learning from data about the error information, recommend related resource for student to read and improve their next submission.

Acknowledgment

This work is supported by the Teaching Research Funding in Honors College of Beihang University (2017) and China Scholarship Council (No.201406025114).

We thank PCO (Principles of Computer Organization) teaching staff members Liu Xudong, Xiao Limin, Niu Jianwei, Luan Zhongzhi, Zhang Liang, Fu Cuijiao and Li Huiyong. And we acknowledge Wu Wenjun and Huang Jiankun for their work in course platform based on OpenEdX.

We appreciate the aid of the TA group that helped the course improvements, including Wang Shuo, Liu Qian, Feng Weitao, Yang Zichen, Huang Qiuyu, Jin Huiming, Chen Hongchao, Lin Ziyi, Liu Hengrui, Wang Zhengdong, Huang Hongming and Liu Ying.

References

[1] Wang, F., and Chen, L. 2016. "A Nonlinear State Space Model for Identifying At-Risk Students in Open Online Courses". In Proceedings of the 9th International Conference on Educational Data Mining (June 29 - July 2, 2016, Raleigh, NC, USA), pp. 527-532.

[2] Y.-H. Hu, C.-L. Lo, and S.-P. Shih. "Developing early warning systems to predict students' online learning performance". Computers in Human Behavior, 36:469–478, 2014..

[3] He, J., Bailey, J., Rubinstein, B. I., & Zhang, R. 2015. Identifying At-Risk Students in Massive Open Online Courses. In Proceeding of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 1749-1755.

[4] Srilekshmi, M., Sindhumol, S., Chatterjee, S., and Bijlani, K. 2016. Learning Analytics to Identify Students At-risk in MOOCs. In 2016 IEEE Eighth International Conference on Technology for Education (T4E) (2 - 4 Dec. 2016), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, pp. 194-199. DOI= https://doi.org/10.1109/T4E.2016.048.

[5] Ren, Z., Rangwala, H., and Johri, A. 2016. Predicting Performance on MOOC Assessments using Multi-Regression Models. arXiv preprint arXiv:1605.02269.

[6] Zhou, Q., Zheng, Y., and Mou, C. 2015. Predicting students' performance of an offline course from their online behaviors. In 2015 Fifth International Conference on Digital Information and Communication Technology and its Applications (DICTAP) (29 April - 1 May 2015), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, pp. 70-73. DOI= https://doi.org/10.1109/DICTAP.2015.7113173.

[7] Bydžovská, H. 2015. "Are Collaborative Filtering Methods Suitable for Student Performance Prediction? " Progress in Artificial Intelligence - 17th Portuguese Conference on Artificial Intelligence, pp. 425-430.

[8] Elbadrawy, A., Studham, S., and Karypis, G. 2014. Personalized multi-regression models for predicting students' performance in course activities. Technical Report 14-011. University of Minnesota.

[9] Elbadrawy, A., Studham, R. S., and Karypis, G. (2015, March). "Collaborative multi-regression models for predicting students' performance in course activities". In Proceedings of the Fifth International Conference on Learning Analytics And Knowledge (Marist College, Poughkeepsie, NY, March 16-20, 2015). LAK '15. ACM New York, NY, USA, pp. 103-107. DOI= http://dx.doi.org/10.1145/2723576.2723590.

[10] Conijn, R., Snijders, C., Kleingeld, A., and Matzat, U. 2016. Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle LMS. In IEEE Transactions on Learning Technologies (12 October 2016), IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, 1-1, DOI= https://doi.org/10.1109/TLT.2016.2616312.

[11] Shaymaa E. Sorour, Shaymaa Abd El Rahman, Tsunenori Mine: Teacher interventions to enhance the quality of student comments and their effect on prediction performance. he 46th annual Frontiers in Education (FIE) Conference. Oct. 12-15, 2016 Erie, Pennsylvania, USA, pp. 1-9.

[12] Wan Han, Gao Xiaopeng, and Liu Qian. "Hybrid Teaching Mode for Laboratory-based Remote Education of Computer Structure Course", the 46th annual Frontiers in Education (FIE) Conference. Oct. 12-15, 2016 Erie, Pennsylvania, USA, pp. 1-8.

[13] Events in the tracking log, DOI = http://edx.readthedocs.io/projects/devdata/en/latest/internal_data_formats/tracking_logs.html

[14] Colin Taylor, Kalyan V., and Una-May O., 2014. Likely to stop? Predicting Stopout in Massive Open Online Courses. DOI = http://arxiv.org/pdf/1408.3382v1.pdf.