Identifying Student Behaviors Early in the Term for Improving Online Course Performance

Makoto Mori
Department of Computer Sciences
Florida Institute of Technology, USA
mmori2013@my.fit.edu

Philip K. Chan
Department of Computer Sciences
Florida Institute of Technology, USA
pkc@cs.fit.edu

ABSTRACT
To study the correlation between student behavior and performance, we propose using high-level behavior features and a random forest algorithm. Considering a course with 10 periods, our results indicate that our models can reach 70% accuracy in the first period and 90% in the first 5 periods and starting to study earlier is important in individual behaviors and behavior combinations.

1. INTRODUCTION
The main goal of this study is to identify student behaviors in the first half of the semester that are correlated to strong performance so that we can provide feedback and encourage more appropriate behavior. The contributions of our study include: (1) we introduce *high-level* behavioral features derived from the course syllabus and sequential patterns; (2) we propose a random forest algorithm with cross-validation; (3) considering a course with ten periods, our empirical results indicate that our models can reach at least 70% accuracy from behavior features in the first cumulative period and 90% from features in the fifth cumulative period; (4) our approach can identify both important single behavior and behavior combinations. Our empirical results indicate that starting to access course materials early (a *high-level* feature) is important in individual behaviors and behavior combinations.

2. RELATED WORK
Many studies, e.g. [5], generally use how frequent activities occur and how long activities take as main features in their models. We call such features low-level features. Besides low-level features, related studies [4, 6] propose sequence of activities as features that come from a sequential pattern mining algorithm [4]. Further, Jo et al. [2] measure the interval of login sessions to find the regularity of login interval. Coffrin et al. [2] analyze the ordering of materials used in a course. We call features that not only simply measuring frequency and duration of activities as *high-level* features. For learning algorithms, many related studies, e.g. [8], use a single learning algorithm to predict student performance. However, Elbadrawy and Studham [3] propose using linear multi-regression, which is a weighted sum of multiple linear regression models. Many related studies perform performance prediction based on analysis using student activities from the entire term, which does not allow intervention during the term. Some related studies, e.g. [3], use non-behavior features such as quiz or assignment scores in their model. A number of studies only analyze individual behaviors separately. However, some studies analyze behavior combinations. Elbadrawy and Studham [3] use a weighted sum of multiple linear regression models, each of which can be considered as a behavior combination. Kinnebrew and Biswas [6] use SPAM [4] to identify important sequence of learning behaviors. Our approach uses high and low-level behavior features early in the term with an ensemble learning algorithm to identify both important single behaviors and behavior combinations.

3. APPROACH
In this study we focus on three steps. The first step is to generate features that can represent students' behavior. The second step is to use a machine learning algorithm to find correlations between behavioral features and performance. The third step is to identify important behaviors from the learned models.

3.1 Generating Features
Based on our experience, we identify *low-level* features that characterize the amount of different activities. Activities include number of logins, number of videos watched, number of questions asked and so on. ASRs (Active Student Responding Exercises) are questions that are embedded in the instructional video and students enter their answers after watching the video.

For *high-level* features, we focus on measuring beyond just "how frequent" or "how much" from the log files. For example, a motivated student would likely schedule a regular study time. To measure how regular a student studies, we first identify the day of the week that the student studies the most. For example, if a student studies most on Wednesdays, the student is quite regular in using Wednesday for studying. We then divide the frequency of the most studied weekday (e.g. Wednesday) by the frequency of the weekday (e.g. Wednesday) in the behavior period. The course syllabus has due dates and test dates. We generate features of student behavior with respect to those dates. For example, number of days the student studies before a test, number of days to submit a test before it is due. The syllabus also specifies when materials are released. We generate features that measure how soon the student starts accessing the released materials. We use SPAM [4] to identify high-level features based on behavior sequences. SPAM finds sequential patterns that meet the minimum support and maximum gap constraints. Support is the count of a sequence, while gap is the number of "wide cards" between items in a sequence.

3.2 Random Forests with Cross Validation
To improve effectiveness, we propose using the random forest algorithm [16] which builds multiple less-correlated decision trees and combines the classifications from individual trees. The random forest algorithm has two key parameters: forest size (number of trees) and feature subset size (number of features that can be considered in each node). To find a suitable combination of forest size and feature subset size, we vary the two parameters, build a forest, estimate the quality of the forest via cross validation (by splitting the training set), and select the parameter combination that yields the most accurate forest.

3.3 Identifying Important Behaviors
Given a random forest, we identify the most frequent feature used in the root nodes as the most important single behavior. In a random forest, the root of each tree is selected from a random subset of all the features. Hence, the most frequent feature in the root nodes is most likely to be the most important behavior.

Considering a single behavior might not be sufficient, we desire to study behavior combinations that are correlated with higher performance. Consider a forest that has n trees, we calculate a quality score for each feature combination that appears in the top two levels of a tree. The score of feature combination f_i in tree r is the number of positive examples $P_r(f_i)$ divided by the total number of examples $T_r(f_i)$ for this combination. The score of a feature combination $S(f_i)$ in the forest is the sum of scores from the trees: $S(f_i) = \sum_{r=1}^{n} \frac{P_r(f_i)}{T_r(f_i)}$.

4. EXPERIMENTAL EVALUATION

Our main task is to find important behaviors in the first half of the term that correlate with an above average score on the final exam. Also, we identify behaviors that we can encourage later, instead of just asking students to perform better on assignments and tests. Within the first half of the term, we would like to study how early we can identify important behaviors that estimate performance accurately. We divide the first half of the term into multiple periods (e.g. weeks). Features are generated from behavior in period 1 through *k*. We call such periods as "cumulative" periods.

This study analyzes BEHP5000 "Concepts and Principles of Behavior Analysis" that was offered in 2013 at Florida Institute of Technology. We obtained data for 110 students from the course. Our evaluation criterion is prediction accuracy on the test set. Two thirds of students are randomly selected to form the training set and the rest of students are in the test set. To generate sequential patterns with the SPAM algorithm, we use 70% as the minimum support and 2 as the maximum gap.

To compare the effectiveness of our proposed approach with existing approaches, we select a decision tree learning algorithm without and with rule post-pruning [7]. We also choose the original random forest algorithm [1] that uses 100 as the forest size, and log2**M* as the feature subset size, where *M* is the number of features. We use k=5 in the k-fold cross-validation for our random forest algorithm. For each k-fold cross-validation, we vary the forest size from 99 to 999 and the feature subset size from log2**M* to 55.

4.1 Predicting Performance on Final Exam

According to Figure 1, random forest with k-fold cross-validation is the most accurate among the four algorithms. Random forest based models are more accurate than other algorithms. Our approach reaches 74% of accuracy in the first cumulative period, and 90% of accuracy in the fifth cumulative period.

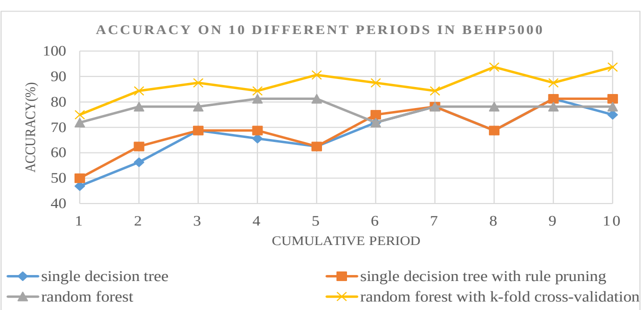

Fig. 1. Accuracy of 4 algorithms from 10 cumulative periods.

4.2 Important Student Behaviors

In the first half of the semester the most frequent feature is *days_after_unit_release* and appears in every cumulative period. This behavior measures, after the unit materials have been released, how many days the student takes to start accessing the materials. The behavior indicates how early a student starts to study, and hence, how motivated the student is. The second most frequent feature is *total(asr_times)* which appears 3 times. This behavior measures the number of times a student attempts ASR, which tries to improve student engagement and understanding of concepts presented in videos. More ASR attempts indicate a student is more engaged and yields deeper understanding.

The most frequent behavior combination is *total(days_after_unit_release)*>x and *test_submit_before_due* <=y which is marked in blue. Both features are high-level features. *total(days_after_unit_release)* represents how early the student starts to access to the unit material after it has been released. *test_submit_before_due* represents how early students submit test before the due date that is stated in the syllabus. Both features are highly related to study motivation of students. Smaller x and larger y values indicate higher motivation. That is, we expect *total(days_after_unit_release)* "<" x and *test_submit_before_due* ">=" y would indicate a highly motivated student. However, we found *total(days_after_unit_release)* ">" x and *test_submit_before_due* "<=" y is the most frequent. In other words, the student begins accessing the materials later and submits the test later, which is counter intuitive. One possible reason is that the behavior combination identifies a small group of students who are smart, therefore, they start studying later and submit test later. Another reason is that the behavior combination appears in cumulative periods 2 and 3, which include less data for the student behavior, therefore, the behavior combination might be less reliable.

Due to space limitation, further details can be found at: `cs.fit.edu/~pkc/papers/edm16long.pdf`.

5. REFERENCES

[1] Breiman, L., 2001. Random forests. *Machine Learning*, 45(1), pp.5-32.

[2] Coffrin, C., Corrin, L., de Barba, P. & Kennedy, G., 2014. Visualizing patterns of student engagement and performance in MOOCs. In *Proc. Intl. Conf. on Learning Analytics & Knowledge* (pp. 83-92).

[3] Elbadrawy, A., Studham, R.S. and Karypis, G., 2015. Collaborative multi-regression models for predicting students' performance in course activities. In *Proc. Intl. Conf. on Learning Analytics & Knowledge* (pp. 103-107).

[4] Ho, J., Lukov, L., & Chawla, S. 2005. Sequential pattern mining with constraints on large protein databases. In *Proc. Intl. Conf. on Management of Data (COMAD)* (pp. 89-100).

[5] Jo, I., Kim, D. & Yoon, M., 2014. Analyzing the log patterns of adult learners in LMS using learning analytics. In *Proc. Intl. Conf. Learning Analytics & Knowledge* (pp. 183-187).

[6] Kinnebrew, J. & Biswas, G., 2012. Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution. Proc. *Intl. Conf. Educational Data Mining* (pp. 57-64).

[7] Mitchell, T., 1997. *Machine learning*. McGraw-Hill.

[8] Seaton, D.T., Bergner, Y., Chuang, I., Mitros, P. and Pritchard, D.E., 2014. Who does what in a massive open online course? *Comm. of the ACM*, 57(4), pp.58-65.