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
In this study we investigate the correlation between student behavior and performance in an online course. We introduce high-level behavior features derived from the course syllabus and sequential patterns. We propose a random forest algorithm with cross-validation to find correlation between behavior features and student performance. Considering a course with 10 periods, our empirical results indicate that our models can reach 70% accuracy from behavior features in the first period and 90% from features in the first 5 periods. We identify both important individual behaviors and behavior combinations; our results indicate starting to study earlier is important in both types of behavior.

Keywords
Student behaviors, student performance, and machine learning.

1. INTRODUCTION
Student interactions with online class platforms can be collected and analyzed. Analyzing the interactions and performance allows guidance for individual students and improves the performance of students. The main goal of this study is to identify important student behaviors that are correlated to strong performance. Furthermore, we aim to identify important behaviors in the first half of the semester so that we can provide feedback and encourage more appropriate behavior.

Our approach first generates behavioral features from log files. Particularly, we introduce high-level behavior features derived from the course syllabus and SPAM (Sequential PAttern Mining algorithm) [11]. Some of high-level features are how early or how regular the student accesses study materials, and whether the student follows the recommended ordering of study materials. We use the random forest algorithm to correlate behavioral features from the first half of the term with student performance. Hence, we can intervene and encourage behavior that can improve the course performance earlier.

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. Cross-validation is used to find the appropriate forest size and feature subset size.
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 is the most important behavior. Also, behavior combinations that include studying early are also more important behavior combinations. Further, studying early is a high-level feature derived from the course syllabus, which indicates that our high-level features could be useful.

We discuss related work in Section 2. The details of our approach are in Section 3. Section 4 discusses the evaluation of our approach. Section 5 will sum up this study.

2. RELATED WORK
Some related studies [1-3] discuss Massive Open Online Courses (MOOCs) and what students are doing in online courses. They suggest analyzing learner behavior by participation level, instructional resource usage and time spending in the course, especially, total time spent and course component usage of students in a MOOC [2,3]. Many studies [2-5, 7-9, 14-15] 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 [11-13] propose sequence of activities as features that come from a sequential pattern mining algorithm [13,21]. Kinnebrew and Biswas [12] use SPAM to identify important behavior sequence in their study. Further, Kim and Yoon [8] measure the interval of login sessions to find the regularity of login interval. Coffrin et al. [10] 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, related studies [1, 4-7, 14] use a single learning algorithm to predict student performance. For example, Bunkar et al. [20] uses a decision tree algorithm [7]. Gökhan and Arif [5] attempt to use clustering analysis to identify difference of student learning behavior. Elbadrawy and Studham [18] propose using linear multi-regression, which is a weighted sum of multiple linear regression models. Our study uses an ensemble machine learning approach random forest to predict student performance.

Many related studies perform performance prediction or behavior analysis using student activities from the entire term, which does not allow intervention during the term. Some related studies [9, 18, 20] use non-behavior features such as quiz or assignment scores in their model. Though assignment and test scores are highly indicative of student performance, we cannot just ask students to perform better on assignments and tests. Our approach only uses behavioral features from the first half of the term so that we can intervene and encourage desirable behaviors.

A number of studies only analyze individual behaviors separately. However, some studies analyze behavior combinations. Elbadrawy and Studham [18] use a weighted sum of multiple linear regression models, each of which can be considered as a behavior combination. Kinnebrew and Biswas [12] use SPAM [13,21] to identify important sequence of learning behaviors. Our approach identifies both important single behaviors and behavior combinations.

3. APPROACH

Our overall system is depicted in Figure 1. In this study we focus on steps that are indicated in white boxes. The first step is to generate features that can represent students' behavior in the course. Features are composed of low-level features and high-level features from syllabus and pattern mining algorithm. 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.

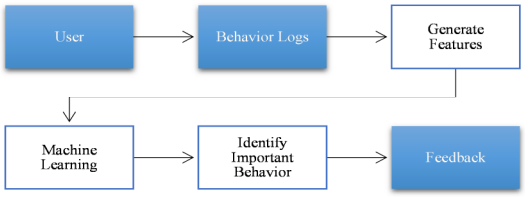

Figure 1 — Main components of overall system

3.1 Generating Features

We have two categories of features. The first category has low-level features, which are more typical features such as "how frequent" and "how much" a student accesses course materials. The second category has high-level features, which include features based on "how early/late" and "how regular" a student performs a certain activity. These features are derived from the course syllabus or a sequential pattern mining algorithm.

Based on our experience, we identify features that characterize the amount of different activities. Activities include number of logins, number of videos watched, number of questions asked and so on. Table 1 lists low-level features based on our experience. ASRs (Active Student Responding Exercises) are questions that are embedded in the instructional video and students enter their answers after watching the video. Cyberat is an exercise for training a virtual rat.

Table 1 — Low-level features of student behavior.

Feature	Description
(total/mini)login(times/hours)	Login Frequency
(total/mini)meeting(times/hours)	Meetings attended
(total/mini)ASR(times/hours)	ASRs attempted
(total/mini)drill(times/hours)	Fluency drills attempted
(total/mini)ppt(times/hours)	Powerpoints accessed
(total/mini)video(times/hours)	Videos accessed
(total/mini)unit(times/hours)	Unit materials accessed
(total/mini)supplemental_materials(times/hours)	Supplemental materials accessed
(total/mini)study_guide(times/hours)	Study guides accessed
(total/mini)cyberrat(times/hours)	Cyberrat attempted
(total/mini)ask_question(times/hours)	Questions asked
(total/mini)cyberratasgn(times/hours)	Optional assignment submitted
(total/mini)discussion(times/hours)	Discussions participated
(total/avg)activities_one_log	Activities in one login-logout
(total/avg)hours_one_log	Hours spend on one login-logout
total_test_times	Tests attempted

Since the total amount of activities over a period of time might hide inactivity (or low activity) within that period. We also generate features that represent the minimum amount of activities in shorter intervals. For example, if we are generating a feature over 5 weeks, the total amount of activities would be the sum of activities over the 5 weeks and the minimum amount of activities would be the weekly amount that is the least within the 5 weeks. That way, we can capture inactivity (or low activity) in the student's behavior.

For high-level features in Table 2, 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. The syllabus recommends certain activities; for example, watching the videos before performing ASR and studying the review materials before a test. We identify features that indicate how well the student follows the recommendations.

Table 2 — High-level features of students behavior.

Feature	Description	Type
activity_coverage	Coverage of all types of activities	Syllabus
regular_day	Regular study schedule	Experience
study_days	Days studied for a test	Syllabus
test_submit_before_due	Days submitted test before due	Syllabus
(total/avg)days_after_unit_release	Days attempted units after unit released	Syllabus
(total/avg)video_after_unit_release	Days videos watched after unit released	Syllabus
(total/avg)video_before_asr	Times videos watched before ASRs	Syllabus
(total/avg)review_times	Correct materials reviewed before test	Syllabus
done_1st_form_test	Attempted 1st form of test	Syllabus
done_both_form_test	Attempted both forms of test	Syllabus

Features in Table 3 are generated by SPAM [13,21] (Sequential PAttern Mining). SPAM finds sequential patterns (frequent sequences) 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. SPAM uses a bitmap data structure and a depth-first search to efficiently find the patterns. Table 3 lists some of sequential patterns found by SPAM in our training set (activities from 70% of the students).

Table 3 — High-level features from sequential patterns.

Feature	Description
[assignment,assignment]	Two assignments occur sequentially
[test,unit,unit]	Access two units materials after test
[video,unit]	Access a unit material after video
[video,unit,unit]	Access two unit materials after video
[ASR,unit]	Access a unit material after ASR
[drill_submit,drill_review,unit]	Drill_review occurs after drill_submit, then unit occurs
[drill_review,unit]	Access a unit material after drill_review
[study_guide,unit]	Access a unit material after study_guide
[unit,study_guide,unit]	Access study_guide between two unit materials

3.2 Random Forests with Cross Validation

Related studies mostly use a single model. To improve effectiveness, we propose using an ensemble approach which utilizes multiple models. The random forest algorithm [16] builds multiple decision trees and combines the classifications from individual trees. In order to gain performance over a single tree,

the trees in a forest needs to be less correlated. To decrease correlation, the random forest algorithm considers only a random subset of the available features at each node when a tree is built. That is, though the decision tree algorithm is deterministic in choosing the best feature at each node, the random forest algorithm restricts the decision tree algorithm to a random subset so that the resulting trees are less correlated. The final predicted class is based on the class predicted by the most 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). Instead of using some predetermined parameter values [16], we propose to use k-fold cross validation [17] to find the suitable values for our dataset. In k-fold cross validation, the original training dataset is divided into k subsets. In each iteration one subset form the validation set and the remaining subsets form the training set. The process is repeated k times, each time a different subset is selected as the validation set. To evaluate the quality of a forest, we calculate average accuracy of the forest on the validation sets. 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, and select the parameter combination that yields the most accurate forest.

3.3 Identifying Important Behavior

Given a random forest, we identify the most frequent feature used in the root nodes as the most important single behavior. We analyze the root node because the decision tree learning algorithm selects the most informative feature for classification as the root. 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. Since nodes near the root of a decision tree are more informative for classification, we consider feature combinations from the first two levels. Consider a forest that has n trees, we would like to 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: $S(f_i) = \sum_{r=1}^{n} \frac{P_r(f_i)}{T_r(f_i)}$. Note that when a tree does not have feature combination f_i, it is not part of the summation. Feature combinations with a higher score are considered more important behavior combinations.

3.4 Assessment and Behavior Periods

In order to identify important behavior for higher performance, we need to specify the assessment and when the behavior occurs. In this study we focus on two assessments: the first attempt of the final exam and the overall course score.

Our first task is to find important behaviors in the first half of the term that correlate with an above average score on the final exam. This task is challenging because we would like to identify the important behavior earlier in the term which allows less information about student behavior. Also, our features do not include scores of assignments or tests that are highly indicative of student performance in the final exam. We would like to identify behaviors that we can encourage later, instead of just asking students to perform better on assignments and tests. Furthermore, since many students pass the final exam, predicting passing or failing the final exam will be easier than achieving above or below the average score on the final exam.

Our second task is to find important behaviors in the first half of the term that correlate with an above average score in the overall course score. The overall score is a weighted sum of scores from all assessments. Again, we only use features from student behaviors.

Within the first half of the term, we would like to study how early we can identify important behaviors that estimate performance in the final exam or overall score 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. In other words, we would like to identify important behaviors in the first k periods that correlate with performance on the final exam (or overall score). We call such periods as "cumulative" periods.

4. EXPERIMENTAL EVALUATION

4.1 Experimental Data

This study analyzes BEHP5000 "Concepts and Principles of Behavior Analysis" that was offered from January 2013 to April 2013 at Florida Institute of Technology. The course includes 7 instructional units, covering a total of 11 study guides. Each unit takes one to two periods to complete. Further, each unit has interactive videos as well as Acquisition ASRs (Active Student Responding Exercises) which are questions provided by the instructor during instructional videos. Students write down their responses and later enter them into the online system. Moreover, 10 one-hour interactive online sessions with co-instructors were scheduled. 6 optional CyberRat exercises involve the training of a virtual rat. The course has 8 online tests and one online proctored final exam. We obtained data for 110 students from the course.

4.2 Evaluation Criteria and Procedures

Our evaluation criteria are prediction accuracy, true positive and false positive rates on the test set. In this study, positive denotes stronger student performance and negative denotes weaker performance. Two thirds of students are randomly selected as the training examples, and the rest of students are the test examples. 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 pruning [17]. We also choose the original random forest algorithm [16] 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.

4.3 Predicting Performance

4.3.1 Predicting Performance on Final Exam

According to Figure 2, random forest with k-fold cross-validation is the most accurate among the four algorithms. Generally, random forest based models are more accurate than decision tree based models. Using cross-validation to find the appropriate forest size and feature subset size improves accuracy of random forest based algorithms. Our approach reaches 74% of accuracy in the first cumulative period, and 90% of accuracy in the fifth cumulative period. In the second half of the term, our approach continues to be more accurate than the other approaches.

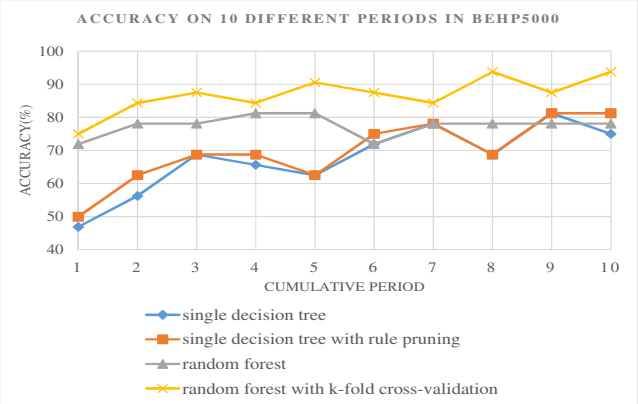

Figure 2 — Accuracy of 4 algorithms with features from 10 cumulative periods.

Table 4 — True positive and false positive rates of 4 algorithms with features from 5 cumulative periods.

	True positive rate (higher is better)					False positive rate (lower is better)			
	DT	DTwP	RF	RFwCV		DT	DTwP	RF	RFwCV
period1	0.388	0.444	**0.722**	**0.722**	period1	0.428	0.428	0.286	**0.214**
period2	0.555	0.555	0.722	**0.778**	period2	0.428	0.285	0.143	**0.071**
period3	0.778	0.778	0.778	**1.000**	period3	0.428	0.428	**0.214**	0.286
period4	0.722	0.722	0.778	**0.944**	period4	0.428	0.357	**0.143**	0.286
period5	0.722	0.722	0.778	**0.889**	period5	0.500	0.500	0.143	**0.071**
DT=single decision tree					RF=random forest				
DTwP=single tree with pruning					RFwCV=random forest with cross-validation				

True positive and false positive rates of the four approaches are in Table 4. Our approach random forest with cross validation approach is more effective than the other approaches, except for the false-positive rates for cumulative periods 3 and 4. Generally, random forest approaches are more effective than decision tree approaches. The random forest approach is able to reveal the importance of weak attributes. Rule pruning can generally increases the performance of a single decision tree.

4.3.2 Predicting Performance of Overall Score

According to Figure 3, our approach is more accurate than the other approaches. Particularly, our random forest approach with cross validation is more accurate than without cross validation. Our approach achieves 69% in accuracy in the first cumulative period and 92% in the fifth cumulative period. With more data from the second half of the term, our approach continues to increase in accuracy and more accurate than the other approaches.

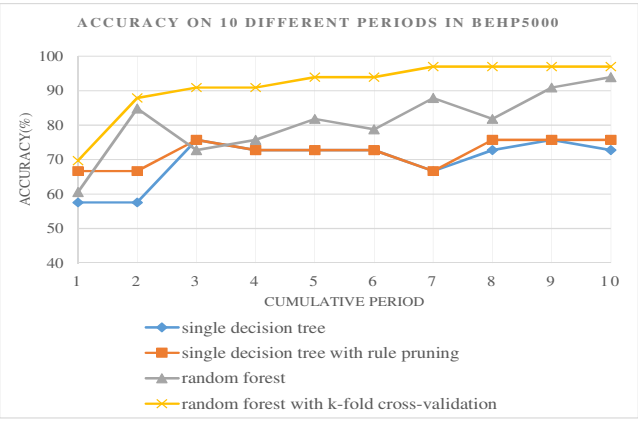

Figure 3 — Accuracy of 4 algorithms with features from 10 cumulative periods.

Table 5 — True positive and false positive rates of 4 algorithms with features from 5 cumulative periods.

	True positive rate (higher is better)					False positive rate (lower is better)			
	DT	DTwP	RF	RFwCV		DT	DTwP	RF	RFwCV
period1	0.650	**0.750**	0.650	**0.750**	period1	0.538	0.461	0.462	**0.385**
period2	0.650	0.800	0.850	**0.900**	period2	0.538	0.538	**0.154**	**0.154**
period3	0.750	0.750	0.750	**1.000**	period3	**0.230**	**0.230**	0.308	0.231
period4	0.750	0.750	0.750	**0.900**	period4	0.307	0.307	0.231	**0.077**
period5	0.800	0.800	0.800	**0.950**	period5	0.384	0.384	0.154	**0.077**
DT=single decision tree					RF=random forest				
DTwP=single tree with pruning					RFwCV=random forest with cross-validation				

True positive and false positive rates of the four approaches are in Table 5. Our random forest with cross validation approach is more effective than the other approaches, except for the false-positive rates for cumulative periods 2 and 3. Generally, random forest approaches are more effective than decision tree approaches.

4.4 Important Individual Student Behavior

4.4.1 Individual Behavior Correlated with Final Exam Performance

According to Table 6, in the first half of the semester the most frequent feature is *days_after_unit_release,* which is marked in blue. This behavior appears in every cumulative period. The behavior *days_after_unit_release* means that 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. This behavior is a high-level feature from the syllabus, which includes the dates of when materials were released.

The second most frequent feature is *total(asr_times)* which appears 3 times and is marked in pink. The behavior *total(asr_times)* means the total number of times a student attempts ASR. ASR intends to improve student engagement and the understanding of concepts presented in videos. More ASR attempts indicate a student is more engaged and yields deeper understanding. This behavior is a low-level feature from our own experience.

Table 6 — Top 3 important behaviors in 5 cumulative periods

		Detected important student behavior	
Period	NO.	final exam	overall score
period1	top. 1	avg(days_after_unit_release)	total(cyberrat_times)
	top. 2	total(discussion_times)	total(discussion_times)
	top. 3	total_activities_one_log	total_activities_one_log
period2	top. 1	test_submit_before_due	total(login_times)
	top. 2	total(login_times)	min(ppt_hours)
	top. 3	avg(days_after_unit_release)	min(login_times)
period3	top. 1	total(days_after_unit_release)	total(days_after_unit_release)
	top. 2	total(asr_times)	total(unit_times)
	top. 3	test_submit_before_due	total(unit_hours)
period4	top. 1	total(discussion_times)	total(days_after_unit_release)
	top. 2	total(asr_times)	test_submit_before_due
	top. 3	total(days_after_unit_release)	min(meetinging)
period5	top. 1	total(days_after_unit_release)	total(days_after_unit_release)
	top. 2	total(asr__times)	total(login_times)
	top. 3	total(study_guide__times)	total(study__guide_times)

4.4.2 Individual Behavior Correlated with Overall Score Performance

According to Table 6, in the first half of the semester, the most frequent features are *days_after_unit_release* which is marked in blue, and *total(login_times)* which is marked in pink. Both features appear 3 times. The behavior *total(login_times)* means that the number of login-logout sessions. This behavior is a low level feature from our own experience.

The feature *days_after_unit_release* is the most frequent behavior for both the final exam and the overall score. Hence, starting to study earlier is quite important to achieve higher performance at the end of the course. This is reasonable because studying earlier indicates higher motivation of the student and allows more time to study and ask questions. Therefore, providing incentives to encourage students to start studying earlier can improve their course performance.

4.5 Important Student Behavior Combinations

4.5.1 Behavior Combinations Correlated with Final Exam Performance

In Table 7, 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—the release day is from the syllabus. *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.

In addition to how frequent a behavior combination appears, we can identify important combinations based on the scores. Since the scores partially depend on the number of trees in the forest and different cumulative periods have different forests and different number of trees, we cannot compare scores across cumulative periods. However, we can compare scores within the same cumulative period. We calculate the difference in scores between the top 2 combinations—the difference is in the "delta" column of Table 7. A larger delta indicates the top combination is less likely to be at the top by chance and hence more reliable. The behavior combination with largest delta is *test_submit_before_due<=9.5 and total(discussion_times)>1,* which is marked in pink. The behavior combination means a student submits a test closer to the due date and participates more in the discussions. The first behavior in the combination is a high-level feature derived from the syllabus and the second is a low-level feature. Submitting a test closer to the due date allows more time to study and more discussions increase understanding.

Table 7 — Top 2 most important behavior combinations for final exam in 5 different cumulative periods. Delta is the score difference between the top 2 behavior combinations.

Period	Rank	Detected important behavior	Score	Delta
period1	top. 1	total(days_after_unit_release)>4.5^total(ask_question_times)<=3.5	291	60.2
	top. 2	total(days_after_unit_release)<=4.5^total(video_times)>3.5	231	
period2	top. 1	test_submit_before_due<=9.5^total(discussion_times)>1.0	217	62.6
	top. 2	total(days_after_unit_release)>17.0^test_submit_before_due<=9.5	155	
period3	top. 1	total(days_after_unit_release)>24.5^test_submit_before_due<=12.5	41.3	10.9
	top. 2	total(days_after_unit_release)<=24.5^total(asr_hours)>4.5	30.3	
period4	top. 1	total(discussion_times)>5.5^activity_coverage<=37.5	36.3	12.8
	top. 2	total(asr_times)<=4.5^[unit,study_guide,unit]=false	23.5	
period5	top. 1	total(days_after_unit_release)<=48.5^study_days<=17.0	59.6	15
	top. 2	total(days_after_unit_release)>48.5^total_activities_one_log>63.5	44.6	

4.5.2 Behavior Combinations Correlated with Overall Score Performance

Table 8 has the top 2 behavior combinations from each of the first 5 cumulative periods. We observe that none of the behavior combinations appear more than once so we do not have the most frequent behavior combination. However, multiple behavior combinations include feature *total(days_after_unit_release)*. For example, *total(days_after_unit_release)<=40.0 and test_submit_before_due>=8.5,* which is marked in blue. *test_submit_before_due* represents how early the student submits a test before the due date. Both of the features are high-level features derived from the syllabus. This behavior combination represents a student accessing unit materials soon after they have been released and submitting a test early before its due date. The student who has such behavior combination is considered as highly motivated.

The behavior combination with the largest delta in Table 8 is *total(login_times)>10.5 and min(ppt_hours)>0.5,*which is marked in pink. The combination means a student has more login sessions and spends more time on the powerpoint materials during each access. The behaviors in the combination are low-level features. The combination indicates the student is diligent in engaging with the course and studying the powerpoint materials.

Table 8 — Top 2 most important behavior combinations for overall score in 5 different cumulative periods. Delta is the score difference between the top 2 behavior combinations.

Period	Rank	Detected important behavior	Score	Delta
period1	top. 1	total(cyberrat_times)<=1.5^total(drill_times)<=2.5	50.6	7.25
	top. 2	total(discussion_times)<=8.5^total(study_guidehours)<=118.0	43.4	
period2	top. 1	total(login_times)>10.5^min(ppt_hours)>0.5	119	15.5
	top. 2	min(ppt_hours)>3.5^total(meeting_hours)<=110.5	104	
period3	top. 1	total(unit_times)>9.5^test_submit_before_due>7.5	26.5	2.72
	top. 2	total(days_after_unit_release)>22.5^total(discussion_times)>3.0	23.8	
period4	top. 1	total(days_after_unit_release)<=40.0^test_submit_before_due>=8.5	64.2	9.04
	top. 2	min(meetinging)<=2.5^done_both_form_test>5.0	55.2	
period5	top. 1	total(days_after_unit_release)<=48.5^	66.7	9.89
		total(Supplemental_materials_hours)<=20.5		
	top. 2	total(login_times)>3.5^total(video_hours)<=62.5	56.8	

5. CONCLUSION

This study attempts to identify student behaviors in the first half of term that are correlated to student performance. To represent student behavior, we identify low-level and high-level features from the logs files, the course syllabus and SPAM (Sequential PAttern Mining) algorithm. We propose the random forest algorithm with cross-validation to correlate our features and student performance. Cross-validation is used to identify the appropriate forest size and feature subset size. Our empirical results indicate that our models can achieve 70% accuracy from behavior in the first cumulative period and 90% accuracy from behavior in the fifth cumulative period. We identify *total(days_after_unit_release)* as the most important single behavior, and it also appears in the important behavior combinations. This feature behavior indicates how early the student starts to study, which is a high level feature. An important behavior combination for the final exam is that a student submits a test closer to the due date and participates more in the discussions. For overall score, an important combination is that a student has more login sessions and spends more time on the powerpoint materials during each access.

A limitation of our work is that, the datasets (log files) are not designed for educational data mining. If the logs include more detailed information on students' activities such as page numbers of PowerPoint files and segments in video files. The course is part of the certification process. Some students might have prior knowledge of the field and others might be switching into the field. If we have information on how familiar the students are with the field, we can identify important behaviors for students with different background.

6. REFERENCES

[1] Sachin, R.B. and Vijay, M.S., 2012. A survey and future vision of data mining in educational field. In *Advanced Comp. & Comm. Technologies (ACCT),* Second Intl.Conf. on (pp. 96-100).

[2] Champaign, J., Colvin, K.F., Liu, A., Fredericks, C., Seaton, D. and Pritchard, D.E., 2014. Correlating skill and improvement in 2 MOOCs with a student's time on tasks. In *Proc. of the first ACM Conf. on Learning@ scale.* (pp. 11-20). ACM.

[3] 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.

[4] Kolekar, S.V., Sanjeevi, S.G. and Bormane, D.S., 2010, December. Learning style recognition using artificial neural network for adaptive user interface in e-learning. In *Comp. intelligence and Comp. research (ICCIC)*, (pp. 1-5).

[5] Akcapynar, G., Altun, A. and Cosgun, E., 2014, July. Investigating Students' Interaction Profile in an Online Learning Environment with Clustering. In *Advanced Learning Technologies (ICALT),* (pp. 109-111).

[6] Wolff, A., Zdrahal, Z., Nikolov, A. and Pantucek, M., 2013, April. Improving retention: predicting at-risk students by analysing clicking behaviour in a virtual learning environment. In *Proc. of the third Intl.Conf. on learning analytics and knowledge* (pp. 145-149).

[7] Quinlan, J.R., 1990. Decision trees and decision-making. *Systems, Man and Cybernetics, IEEE Transactions* on, 20(2), pp.339-346.

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

[9] Singh, S. and Lal, S.P., 2013. Using feature selection and association rule mining to evaluate digital courseware. In *ICT and Knowledge Engineering (ICT&KE)*, (pp. 1-7).

[10] Coffrin, C., Corrin, L., de Barba, P. and Kennedy, G., 2014. Visualizing patterns of student engagement and performance in MOOCs. In *Proc. of the fourth Intl.Conf. on learning analytics and knowledge* (pp. 83-92).

[11] Kinnebrew, J.S., Loretz, K.M. and Biswas, G., 2013. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *JEDM-J. of Educational Data Mining*, 5(1), pp.190-219.

[12] Kinnebrew, J.S. and Biswas, G., 2012. Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution. *Intl.Educational Data Mining Society*.

[13] Ho, J., Lukov, L., & Chawla, S. 2005. Sequential pattern mining with constraints on large protein databases. In *Proc. of the 12th Intl.Conf. on Management of Data (COMAD)* (pp. 89-100).

[14] Klašnja-Milićević, A., Vesin, B., Ivanović, M. and Budimac, Z., 2011. E-Learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Educatio*n, 56(3), pp.885-899.

[15] Schiaffino, S., Garcia, P. and Amandi, A., 2008. eTeacher: Providing personalized assistance to e-learning students. *Computers & Education*, 51(4), pp.1744-1754.

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

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

[18] Elbadrawy, A., Studham, R.S. and Karypis, G., 2015. Collaborative multi-regression models for predicting students' performance in course activities. In *Proc. of the 5th Intl.Conf. on Learning Analytics and Know.* (pp. 103-107).

[19] Luo, L., Koprinska, I. and Liu, W., Discrimination-Aware Classifiers for Student Performance Prediction.

[20] Bunkar, K., Singh, U.K., Pandya, B. and Bunkar, R., 2012. Data mining: Prediction for performance improvement of graduate students using classification. In *Wireless and Optical Comm.s Networks (WOCN)*, (pp. 1-5).

[21] Ayres, J., Flannick, J., Gehrke, J. and Yiu, T., 2002. Sequential pattern mining using a bitmap representation. In *Proc. of the eighth ACM SIGKDD Intl.Conf. On Knowledge discovery and data mining* (pp. 429-435). ACM.