Identifying Student Behavior for Improving Online Course Performance with Machine Learning

by

Mori, Makoto

A thesis submitted to the Graduate School of
Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Melbourne, Florida
December, 2015

We the undersigned committee hereby approve the attached thesis, “Identifying Student Behavior for Improving Online Course Performance with Machine Learning,” by Mori, Makoto.

Philip K Chan, Ph.D.
Associate Professor
Computer Sciences and Cybersecurity

Marius Silaghi, Ph.D.
Associate Professor
Computer Sciences and Cybersecurity

Joshua Pritchard, Ph.D.
Assistant Professor
School of Behavior Analysis

J. Richard Newman, Ph.D.
Professor and Department Head
Computer Sciences and Cybersecurity

Abstract

Title: Identifying Student Behavior for Improving Online Course Performance with Machine Learning

Author: Mori, Makoto

Advisor: Philip K. Chan, Ph. D.

In this study we investigate the correlation between student behavior and performance in online courses. Based on the web logs and syllabus of a course, we extract features that characterize student behavior. Using machine learning algorithms, we build models to predict performance at end of the period. Furthermore, we identify important behavior and behavior combinations in the models. The result of prediction in three tasks reach 87% accurate on average without using any score related features in the first half of the semester.

Table of Contents

List of Figures

List of Tables

List of Equations

Acknowledgement

Firstly, I want to sincerely thank my advisor Dr. Philip Chan. For the two years, I learnt so much from both his class and the meeting with him. He guides me as patiently as my father. The time of discussing algorithms with him is inspiring and enjoyable.

My sincere thanks also goes to Dr. Joshua Pritchard. He provide me such a good project that allows me to apply my machine learning acknowledges on the field of education.

My sincere thanks also goes to Dr. Marius Silaghi. His class improved my programming algorithm so much.

I also want to say thanks to my friends lingfeng, tiger as well as tiger. I learnt from them that a person can change so many things by hard working. I enjoy the time when we programmed and celebrated together.

Dedication

I dedicate this work to my father and mother, Qun Sun and Peiwen Yao. Your support to me is so warm and unconditional, but what I repay to you is loneliness. Studying aboard is both what I want and what you want, but when I get more and more mature, I feel more and more sad. If there is a chance going back to two years before, I think our decision would still be the same. I wish our lives could be better in the future.

Keywords

Educational data mining, Machine learning, Online course, student learning behavior, Random forest with K-fold cross-validation

Chapter 1
Introduction

Much attention has been paid to embedding sensors into products to collect operational data over the net, and taking advantage of the Internet of things (IoT) for operation and maintenance in the manufacturing industry, and to allow the industry to utilize Big Data. This wave has also extended to the education industry. Students' learning frequencies, results and problems with their degree progress are collected via online class platforms. This allows for optimum guidance for individual students, and it maximizes the performance of the students. This paper is also a scientific approach to improve the quality of education services.

The related papers simply predict the performance of students in final exams given the accumulated data from online classes. Some of papers focus on identifying the important sequential pattern of activities using data mining algorithms. To meet the needs of students, content recommendations can be made based on collating the different learning styles made in advance in the system. Other papers only can identify the learning style of students at the end of semester, which is too late to improve students' performance.

This paper creates more than 100 features with the help of data mining algorithms, in an attempt to identify the learning styles of individual student. Inspired by the research discussed above, this paper attempts to experimentally determine whether it is possible to predict students who pass or fail in the early stage of the semester.

As a result, after considering the syllabus and using pattern mining algorithm, our system has a good ability to identify the key features of the students' learning behavior. In addition, the random forest algorithm is able to detect important behavior for the students at different periods in a semester. This means that no need to wait until the end of the term, we can improve the course performance earlier. Further, the approach proposed is able to detect important behavior for each period. The average accuracy of predictions in three difficult tasks is able to reach 90%.

Discussions of related papers of this study will be made in Chapter 2. The key algorithms and major approaches that are used in this paper will be introduced in Chapter 3. The data we used, a method for evaluating our system and descriptions of the various parameters will be elucidated in Chapter 4. Chapter 5 will sum up this paper.

Chapter 2
Related Work

This chapter discusses previous work related to our study. According to articles [2] [3], they provide the general ideas of Massive Open Online Courses (MOOC), and talk about what students are doing in online courses. They suggest analyzing learner behavior by participation level, instructional resource usage and time allocation, especially, total time spent and course component usage of students in a MOOC [3]. They try to analyze and visualize the learning behavior of students, such as visualizing learning analytical data presented in their paper, and providing insights into student activities in MOOCs. However, there is more work that can be done in this field. Then, they [1] discuss a general idea of the history and application of data mining techniques in the educational fields, which are the traditional educational system, the web-based educational system, intelligent tutoring systems, and e-learning. Our system has the same motivation.

This paper focuses on both identifying students' profiles by their behavior in the online course and intelligently providing personalized assistance. We briefly review some significant attributes to measure students' behavior in online courses in Section 2.1, and discuss other similar studies on providing personalized recommendations to students need help in Section 2.2.

2.1 Mining of Attributes

First of all, some useful attributes to investigate students' interaction profiles and to measure students' performance are introduced [6]. They categorize the students' learning behavior first, then they compare the new student's behavior to the previous behavior. Moreover, the paper [8] mentions that although each online course has different structures and uses different components, we still can extract the low-level attributes, such as learning time range, clicking points, and login and logout times calculated from the dataset.

Particularly, deriving useful features from the user log and extracting the behavioral data embedded within logs, the paper [9] lists 26 different attributes for each student in their course. These 26 different attributes include high-level features and features combinations.

A unique pattern mining approach [11] has been proposed. One relatively new and effective Sequential PAttern Mining Algorithm Using a Bitmap (SPAM) makes use of a depth-first search strategy, an efficient candidate pruning mechanism as well as a two-dimensional bitmap data structure. Notwithstanding the space inefficiency, SPAM surpasses other sequential pattern mining algorithms in running time, especially on large databases with lengthy sequential patterns like a huge number of students' logs. SPAM has already been used in other papers and fields. For instance, one methodology [12] combines action abstraction, differentially frequent activity patterns, and piecewise linear segmentation of activity phases. The paper [12] makes it possible to identify and compare learning behaviors from students' learning interaction traces. Another successful example is the paper [13], which identifies the important learning behavior with an iterative approach using a variety of action features. The paper [13] presents results by analyzing the reading behavior of students in a learning-by-teaching environment. However, they couldn't help students increase their performance based on the existing information of students' behavior, which is one of the goals of our paper.

We introduce machine learning algorithms of related work in the following part. First, paper [5] introduces some useful attributes to investigate students' interaction profiles. They use clustering algorithm to measure students' performance. Similarly, the paper [4] aims to extract learning patterns of users to develop an Adaptive User Interface. Their attributes come from the web contents usage mining, and their measuring technique is Artificial Neural Network. They successfully gathered information about learning preferences. Furthermore, their approach can recognize changes in learning behavior. Then paper [18] adopted the decision tree, an classical but efficient machine learning technique, into their approach. Unlike other machine learning techniques, the decision tree is a fast classification algorithm without using any random parameters inside of the algorithm.

Additionally, the decision tree [7] is commonly used for gaining information for the purpose of decision-making. The decision tree algorithm was introduced in 1986 by Quinlan Ross [7], which is based on Hunt's algorithm. However, the decision tree algorithm can easily get over fitted on training examples, and this problem is not solved in the paper [7]. The paper [17] constructs two phases for growing the decision tree: tree building and tree pruning. At the pruning process, the paper [17] adopts a rule mining algorithm to improve the decision tree over fitting training examples shortage. Especially, it is possible to make the entire program strong against noisy data.

2.2 Recommendation System

The paper [14] and paper [15] share similar ideas with our paper. The paper [14] describes a personalized e-learning system that can automatically detect interests, habits and knowledge levels of learners. Differences between learners are determined by learner's previous knowledge, learning style, learning characteristics, preferences and goals. However, the paper [14] divides students into roughly eight groups in the system based on students' preferences. Their system determines whether student behavior belongs to any group in the system. Although the system increases students' satisfaction, it does not optimize students' performance [14]. The paper [15] also determines which students belong to which group, but they still do not try to help poor-performing students.

Separately from the above related work, we attempt to combine data mining strategies to find out high level attributes automatically. Unlike other papers, we are not using any features related to scores to predict a course performance. Moreover, our system attempts to output important behavior in every single period. We use not only single algorithm, but we also combine multiple machine learning algorithms to achieve our goals experimentally.

Then the first goal of this paper is to list and compare learning behaviors by extracting variables, aggregating attributes as well as using a feature-mining algorithm. Additionally, the second goal is to detect the significant behavior. Moreover, the third goal is that the entire system should be able to identify the important behavior in the early period without using any score-related features.

Chapter 3
Approach

3.1 Overall System

In this section we give an overall idea of the entire system (see Figure 1). Once we can get logs from course BEHP5000, the first step is to list all possible features that can successfully identify students' behavior in BEHP5000. Features are composed of aggregated features and pattern mining algorithm. The second step would be learning behavior and making decisions. This step tries to find the best solution for data, candidate solutions including single decision tree learning (with/without rule pruning) and random forest (with/without k-fold cross-validation) learning to learn more deeply about data. Especially, random forest works for these examples with weak features and bad result examples from single decision tree. The last step is to detect important behavior for students, system designer and educator. We list all important parts of the system in Figure 1 and this paper basically concentrates on the blue parts (see Figure 1). Since we aren't able to get the right to directly access to the university system, we only received the data and logs from the considerable sector. Further, giving feedback also require right of access, thus function of feedback is also marked as green.

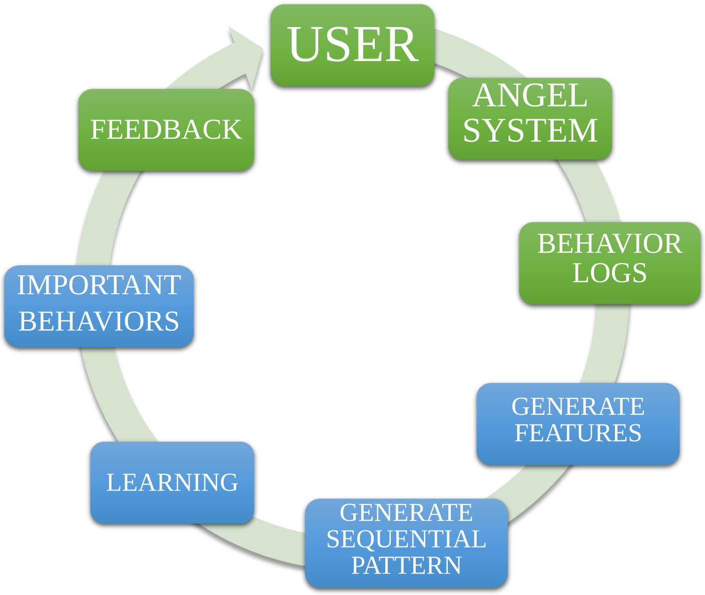


Figure 1 — Example of the overall system

3.2 Features

This section explains all of the features used in this paper. Feature has been mainly used for a particular student’s learning style. Feature is divided into primarily two types: one is learning habits of students that have been discovered in pattern mining algorithm, and the other is an artificially discovered features from the BEHP5000 log files and syllabus.

First, in order to profile the learning behavior recorded in the log file of BEHP5000, this paper carries out mining of features representing the learning behavior. To derive usefulness from the students' log and to extract the behavioral data embedded into it, the first group of the attribute values are aggregated (see Table 1).

Table 1 — 28 different attributes extracted/aggregated for students from the log file of the BEHP5000.

attribute	Description
totallogintimes	Login Frequency
totalmeettimes	Num. of meetings attended
totalASRtimes	Num. of ASRs attempted
totaldrilltimes	Num. of fluency drills attempted
totalppttimes	Num. of powerpoints watched
totalvdotimes	Num. of videos watched
totalunttimes	Num. of unit materials watched
totalsuptimes	Num. of supplemental materials attempted
totalstyguidetimes	Num. of study guides watched
totalcybtimes	Num. of cyberrats submitted
totalfaqtimes	Num. of questions asked
totalcybasgntimes	Num. of optional assignments submitted
totaldiscustimes	Num. of discussions attended
totalloginhours	Hours of login to logout
totalmeethours	Hours of meetings attended
totalASRhours	Hours of ASRs attempted
totaldrillhours	Hours of fluency drills attempted
totalppthours	Hours of powerpoints watched
totalvdohours	Hours of videos watched
totalunthours	Hours of unit materials watched
totalsuphours	Hours of supplemental materials attempted
totalstyguidehours	Hours of study guides watched
totalcybhours	Hours of cyberrats submitted
totalfaqhours	Hours of questions asked
totalcybasgnhours	Hours of optional assignments submitted
totaldiscuschours	Hours of discussions attended
meets	Evaluation of online meeting
ASRs	Evaluation of ASRs

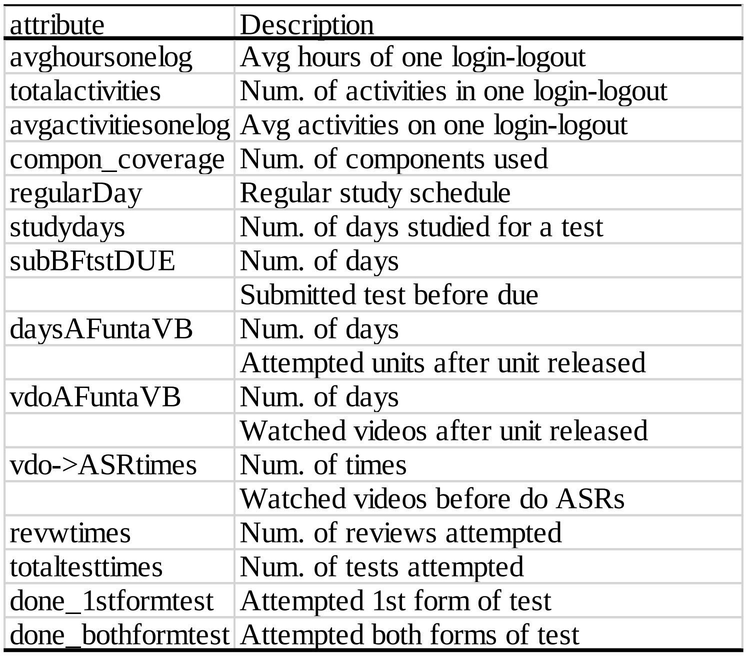
In order to measure students' learning behavior in a better way, the above 28 attributes in Table 1 will be measured in two ways: one is total score. For example, if we want to know students' performance in the 5th period, then the program will sum up scores of the previous 5 periods into one score, to give one total score. Another score is the minimum score (see Table 2). For instance, the minimum score in the program refers to the minimum score in the last particular period. In a more specific way, the 1st period score is 5, the 2nd period score is 4, the 3rd period score is 0 and the 4th period score is 10, then the total score of the 4th period is 19, and the minimum score of the 4th period is 0 in cumulative periods way, and the minimum score of the 4th period is 10 in individual period way.

Table 2 — 28 different attributes extracted/aggregated for students from the log file of the BEHP5000 and measured by minimum value.

attribute	Description
(min)totallogintimes	The minimum value of login Frequency
(min)totalmeettimes	The minimum value of Num. of meetings attended
(min)totalASRtimes	The minimum value of Num. of ASRs attempted
(min)totaldrilltimes	The minimum value of Num. of fluency drills attempted
(min)totalppttimes	The minimum value of Num. of powerpoints watched
(min)totalvdotimes	The minimum value of Num. of videos watched
(min)totalunttimes	The minimum value of Num. of unit materials watched
(min)totalsuptimes	The minimum value of Num. of supplemental materials attempted
(min)totalstyguidetimes	The minimum value of Num. of study guides watched
(min)totalcybtimes	The minimum value of Num. of cyberrats submitted
(min)totalfaqtimes	The minimum value of Num. of questions asked
(min)totalcybasgntimes	The minimum value of Num. of optional assignments submitted
(min)totaldiscustimes	The minimum value of Num. of discussions attended
(min)totalloginhours	The minimum value of Hours of login to logout
(min)totalmeethours	The minimum value of Hours of meetings attended
(min)totalASRhours	The minimum value of Hours of ASRs attempted
(min)totaldrillhours	The minimum value of Hours of fluency drills attempted
(min)totalppthours	The minimum value of Hours of powerpoints watched
(min)totalvdohours	The minimum value of Hours of videos watched
(min)totalunthours	The minimum value of Hours of unit materials watched
(min)totalsuphours	The minimum value of Hours of supplemental materials attempted
(min)totalstyguidehours	The minimum value of Hours of study guides watched
(min)totalcybhours	The minimum value of Hours of cyberrats submitted
(min)totalfaqhours	The minimum value of Hours of questions asked
(min)totalcybasgnhours	The minimum value of Hours of optional assignments submitted
(min)totaldiscuschours	The minimum value of Hours of discussions attended
(min)meets	The minimum value of Evaluation of online meeting
(min)ASRscore	The minimum value of Evaluation of ASRs

Besides using the activity logs, we also consider features that characterize how student behavior are related to the syllabus of the course. Table 3 lists 14 high-level attributes extracted/aggregated for students from the syllabus file of the BEHP5000. These features focus on the relationship between different events in the course and student behaviors. Furthermore, the syllabus provides recommendations, and we construct features that indicate if the students follows the recommendations.

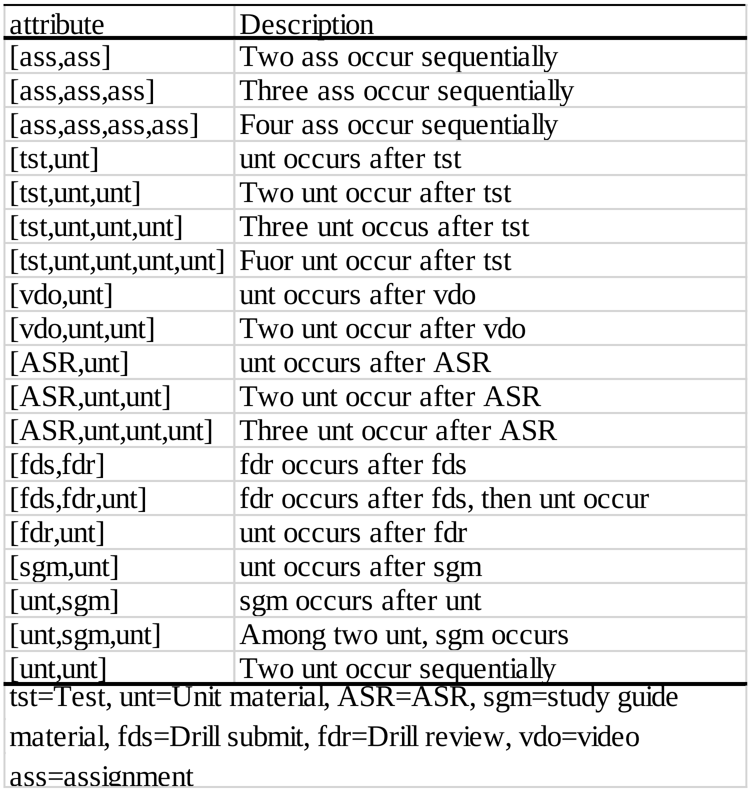
Table 3 — lists 14 high-level attributes extracted/aggregated for students from the syllabus file of the BEHP5000.



attribute	Description
avghoursonelog	Avg hours of one login-logout
totalactivities	Num. of activities in one login-logout
avgactivitiesonelog	Avg activities on one login-logout
compon_coverage	Num. of components used
regularDay	Regular study schedule
studydays	Num. of days studied for a test
subBFtstDUE	Num. of days
	Submitted test before due
daysAFuntaVB	Num. of days
	Attempted units after unit released
vdoAFuntaVB	Num. of days
	Watched videos after unit released
vdo->ASRtimes	Num. of times
	Watched videos before do ASRs
revwtimes	Num. of reviews attempted
totaltesttimes	Num. of tests attempted
done_1stformtest	Attempted 1st form of test
done_bothformtest	Attempted both forms of test

The last group of features that is mined by the sequential pattern mining algorithm. It allows us to extract more specific and behavior relevant attributes by the incorporation of maximum gaps and minimum frequency support constraints. These sequential patterns lists in Table 4 shows that 70% students' log files focus on different multiple periods. We discuss the SPAM algorithm at section 3.3 (see page 12).

Table 4 — lists 19 high-level sequential pattern attributes extracted for students from log files by the mining algorithm.



attribute	Description
[ass,ass]	Two ass occur sequentially
[ass,ass,ass]	Three ass occur sequentially
[ass,ass,ass,ass]	Four ass occur sequentially
[tst,unt]	unt occurs after tst
[tst,unt,unt]	Two unt occur after tst
[tst,unt,unt,unt]	Three unt occus after tst
[tst,unt,unt,unt,unt]	Fuor unt occur after tst
[vdo,unt]	unt occurs after vdo
[vdo,unt,unt]	Two unt occur after vdo
[ASR,unt]	unt occurs after ASR
[ASR,unt,unt]	Two unt occur after ASR
[ASR,unt,unt,unt]	Three unt occur after ASR
[fds,fdr]	fdr occurs after fds
[fds,fdr,unt]	fdr occurs after fds, then unt occur
[fdr,unt]	unt occurs after fdr
[sgm,unt]	unt occurs after sgm
[unt,sgm]	sgm occurs after unt
[unt,sgm,unt]	Among two unt, sgm occurs
[unt,unt]	Two unt occur sequentially

tst=Test, unt=Unit material, ASR=ASR, sgm=study guide material, fds=Drill submit, fdr=Drill review, vdo=video ass=assignment

3.3 Sequential Pattern Mining Algorithm (SPAM)

In this section we generalize a well-known sequential pattern mining algorithm [11], by combining gaps and frequency constraints along the instances. However, the advantage of using a depth-first algorithm like this algorithm is that it allows us to push the constraints deeper inside the mining process; it uses an easily understood bitmap data structure for counting and it is efficient for mining long patterns in huge data.

In order to adapt sequential pattern mining algorithm for this paper, the first step is to convert all raw student activity logs into the abbreviated behavioral codes as shown in Figure 2. Each short code represents an activity.

vdo unt unt unt unt asr unt unt unt tst unt unt tst unt sgm unt unt
vdo unt vdo unt vdo unt vdo unt unt vdo unt vdo unt vdo unt vdo
unt vdo cyb cyb cyi cyi unt vdo unt vdo unt vdo unt vdo unt vdo
unt vdo unt vdo unt vdo unt unt unt unt asr unt….

Figure 2 — Abbreviated behavioral codes represent student activities.

Table 5 — Example of a transaction database

	tid				
sid	1	2	3	4	5
1	{unt}	{unt}	{unt}	{sgm}	{cyb}
2	{unt}	{cyb}	{cyb}	{sgm}	{cyb}
3	{unt}	{cyb}	{sgm}	{cyb}	
4	{unt}	{sgm}	{sgm}	{cyb}	

For example, we have 4 students and their behavioral codes (see Table 5). In Table 5 "sid" refers to student ID, "tid" refers to transaction ID. The first and second students have 5 temporary actions. The third and fourth students have 4 temporary actions. We want to find the most frequent sequence of actions among these 4 students. Then we set minimum frequency support to be 4 and we allow 2 maximum gaps to exist among 2 activities at this particular point. In detail, when we want to find a sequence of actions {unt,sgm}. We recognise either {unt,xxx,yyy,sgm} or {unt,sgm} situations as the pattern that we want. Further, at least 4 students should have the pattern, which we call minimum support 4. In Figure 3 we show the depth-first algorithm starting from empty pattern and keep adding new actions, in order to create all possible sequence of actions that satisfy 2 gap constraints and minimum support 4. Otherwise, creating new patterns would be removed from the pattern list and this branch would be closed. Since if fewer pattern instances do not satisfy constraints, then longer pattern instances based on this pattern also do not satisfy constraints.

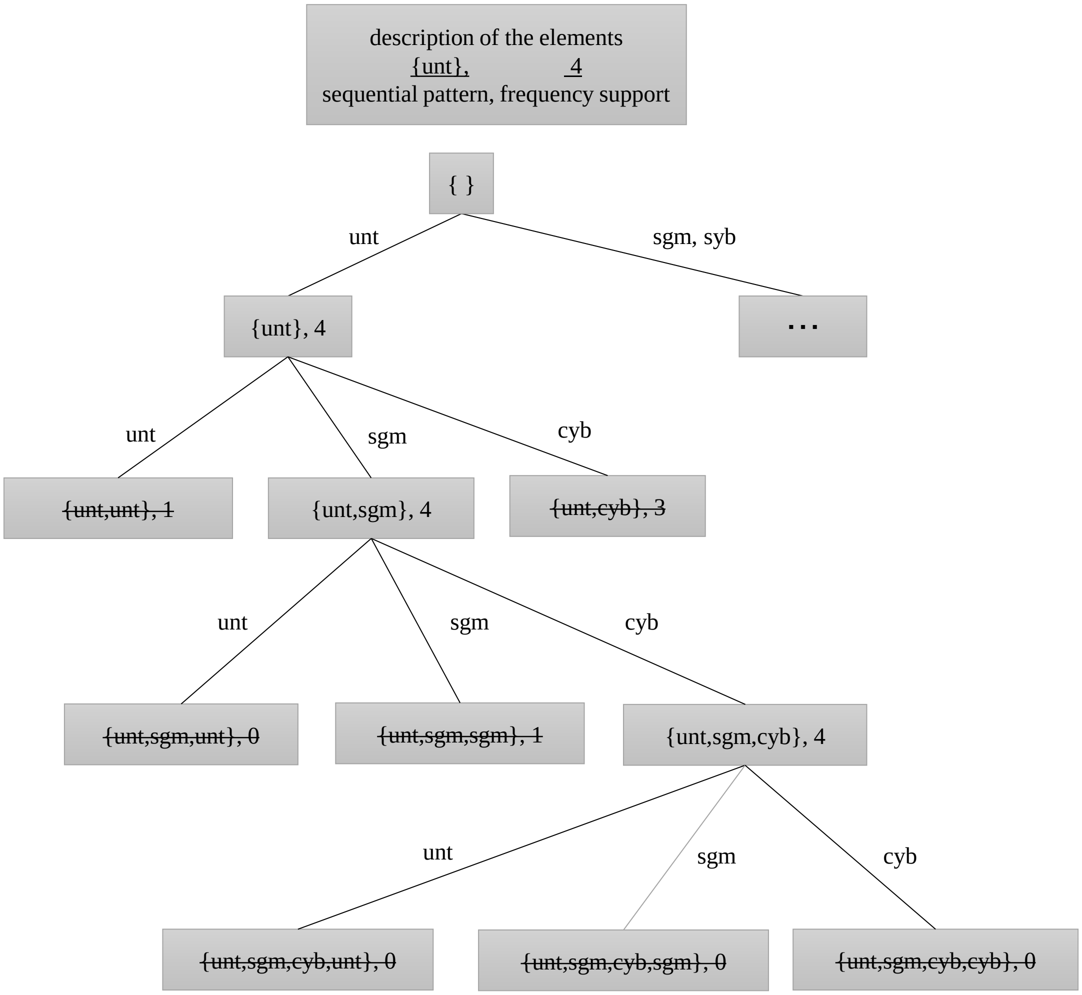
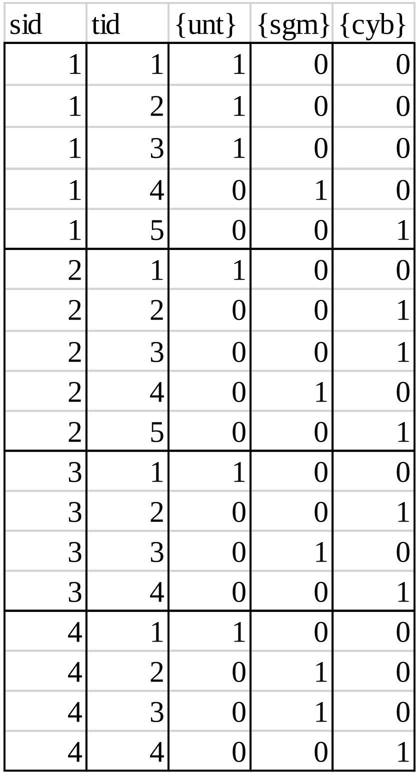


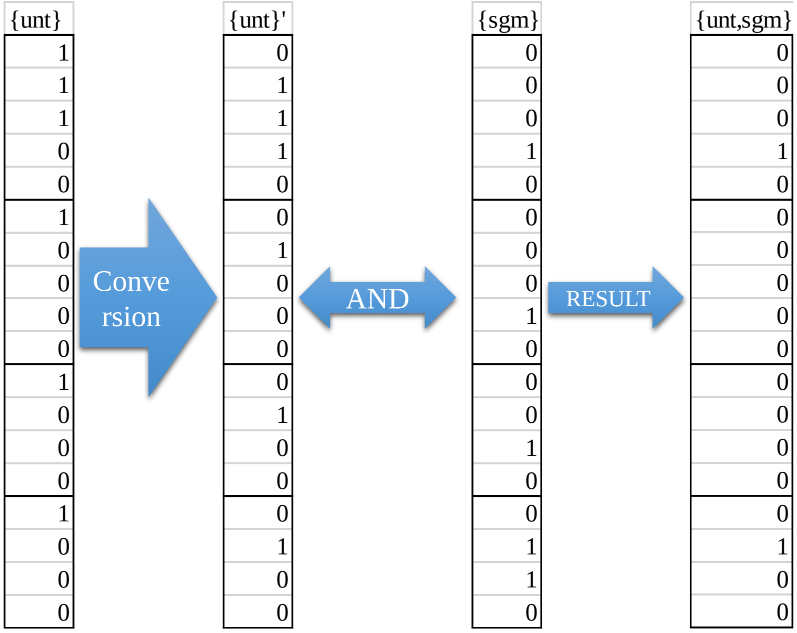
Figure 3 — Running algorithm with 2 gap constraints and minimum support 4.

Table 6 — The bitmap representation of the database



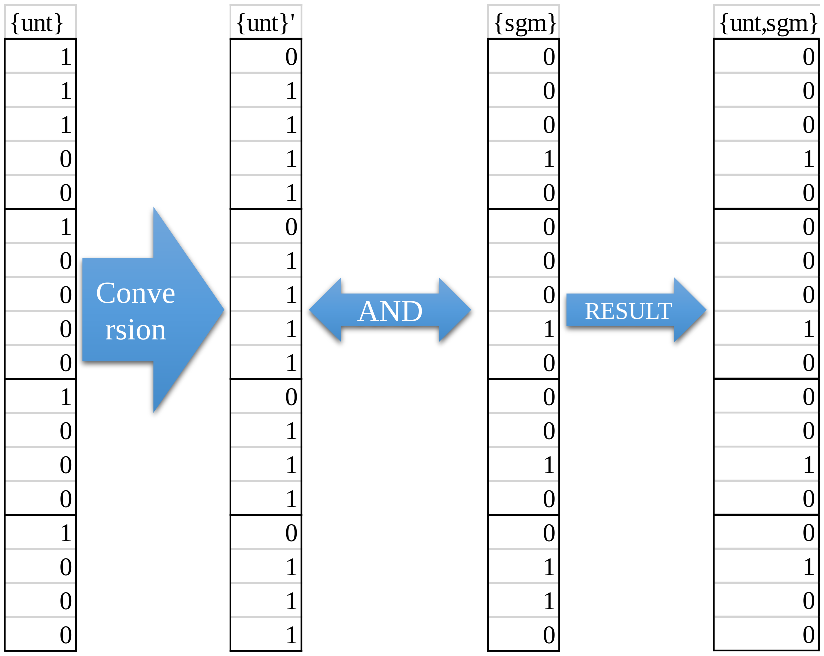
sid	tid	{unt}	{sgm}	{cyb}
1	1	1	0	0
1	2	1	0	0
1	3	1	0	0
1	4	0	1	0
1	5	0	0	1
2	1	1	0	0
2	2	0	0	1
2	3	0	0	1
2	4	0	1	0
2	5	0	0	1
3	1	1	0	0
3	2	0	0	1
3	3	0	1	0
3	4	0	0	1
4	1	1	0	0
4	2	0	1	0
4	3	0	1	0
4	4	0	0	1

To efficiently find if a patterns satisfies the constraints of minimum support and maximum gap, SPAM uses a bitmap data structure. At each node in Figure 3 the program would use a bitmap to figure out whether the newly created pattern satisfies constraints or not. The bitmap data structure for all activities would have an individual table like Table 6 to identify their transition information for each student. When we consider the sequence of actions {unt,sgm}, first of all we need to convert bitmap {unt} to bitmap {unt}' based on gap constraints. Basically, at this step we try to find the index of the bit that is "1", and then flip it to 0 and flip the following bits to "1". The number of following bits that we are going to flip is dependent on the number of gaps that we allow in constraints. {unt}' means we know {unt} occurs, then we and {unt}' with {sgm} to find if there is any activity {sgm} after {unt} satisfying gap constraints. Two different gap constraints and example {unt}' flipping are listed in Figure 4 and Figure 5. Bitmap {unt,sgm} shows the result. If there is any student having "1" in bitmap {unt,sgm}, it means this pattern occurs in this student's activities. If all students have "1" in bitmap {unt,sgm}, it means frequency support is 4 for pattern {unt,sgm}.



{unt}	{unt}'	{sgm}	{unt,sgm}
1	0	0	0
1	1	0	0
1	1	0	0
0	1	1	1
0	0	0	0
1	0	0	0
0	1	0	0
0	0	0	0
0	0	1	0
0	0	0	0
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	0
1	0	0	0
0	1	1	1
0	0	1	0
0	0	0	0

Figure 4 — Adding instance (activity) with no gap constraints



{unt}	{unt}'	{sgm}	{unt,sgm}
1	0	0	0
1	1	0	0
1	1	0	0
0	1	1	1
0	1	0	0
1	0	0	0
0	1	0	0
0	1	0	0
0	1	1	1
0	1	0	0
1	0	0	0
0	1	0	0
0	1	1	1
0	1	0	0
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	0

Figure 5 — Adding instance (activity) with any gap constraints

3.4 Decision Trees

A decision tree [7] is a tree structure classification algorithm (see Figure 6), where each internal node represents attributes, and leaf nodes represent predictive class. All nodes have two or more child nodes. Arcs from an internal node to its children are labelled with a distinct value of the feature. Each leaf node has a class label representing the prediction result. The decision tree is commonly used for decision-making. Grow a decision tree is grown by picking the best attribute at each node and splitting the node to child node recursively is commonly dependent on the information theory.

Information theory is basically used as a measure of purity of classes' entropy, indicating that high entropy is high randomness of the classes of examples in the current node. Low entropy indicates the majority class of examples in the current node that one of the classes is occupied. That when the entropy is zero, which example of classes in the current node is composed of single class. Calculation of the entropy of the decision tree is represented by Equation 1.

$$\text{Entropy: } \mathrm{I}\big(\mathrm{P}(\mathrm{v}_1), \ldots \mathrm{P}(\mathrm{v}_\mathrm{n})\big) = \sum_{\mathrm{i}=1}^{\mathrm{n}} -\mathrm{P}(\mathrm{v}_\mathrm{i}) \log_2 \mathrm{P}(\mathrm{v}_\mathrm{i})$$

Equation 1 — Entropy

In Equation 1, vi refers to class, P(vi) refers to the number of examples in class vi. In order to pick the best attribute as the current node value, we have to estimate entropy for all child nodes if we pick one of the candidate attributes. All of possible candidate attributes would be tested. We will pick the attribute with the lowest entropy that is summed up from the entropy of all of its child nodes as current node.

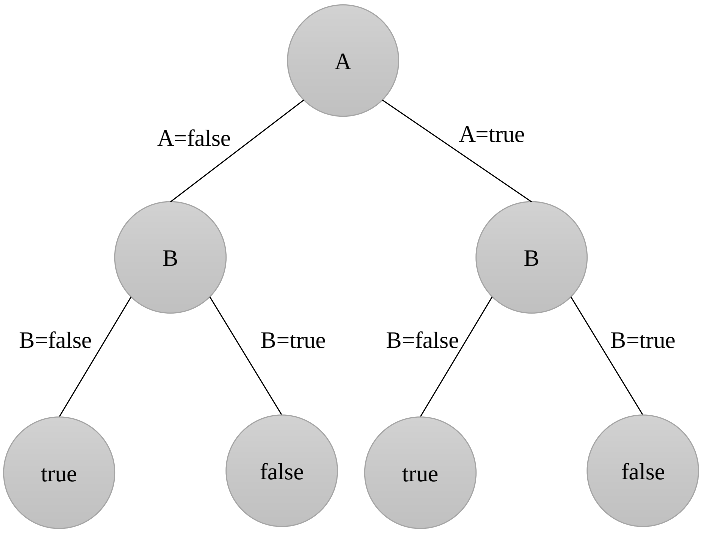


Figure 6 — Example of the decision tree

3.5 Decision Trees and Rule Pruning

A decision tree is an algorithm that is earlier drop into over-fitting (over-learning) training examples problem. Over-fitting, in statistics and machine learning, has been learned from the training data, and is not fit for the unknown data (test data) well, and it means a state that is unable to generalize due to a lack of the generalization ability. One way to solve over-learning is pruning the decision tree directly (see Figure 7). A direct decision tree pruning usually prunes nodes close to leave node at first; root node and upper nodes usually would not be removed. This is the limitation of this method. Therefore, this paper proposes another possibility: rule pruning [17]. First of all, we have to convert the decision tree to rules (rule set) (see Figure 8). The principle of the rule pruning is intended to improve the accuracy by removing the conditions in the rule. For example, once we remove one condition from a rule, we check the accuracy of entire rule set on the validation examples. If the accuracy increases, we move to next condition, otherwise, we move that condition back. We apply this process to all rules as well as all conditions. It is more likely to prune root node and upper nodes, compared to direct pruning.

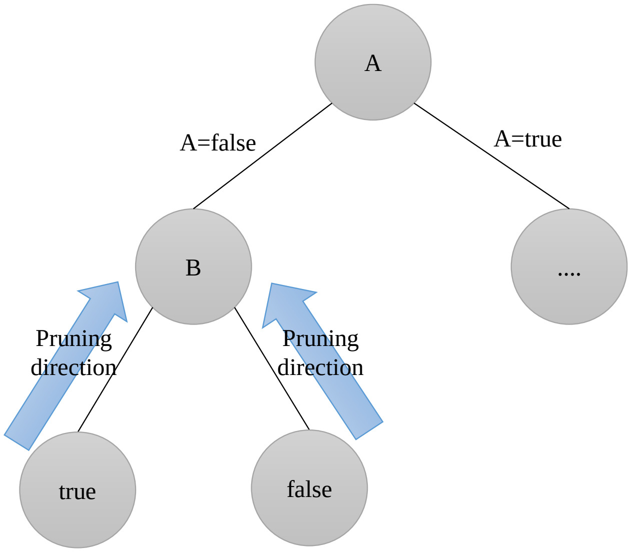


Figure 7 — Example of pruning on the decision tree

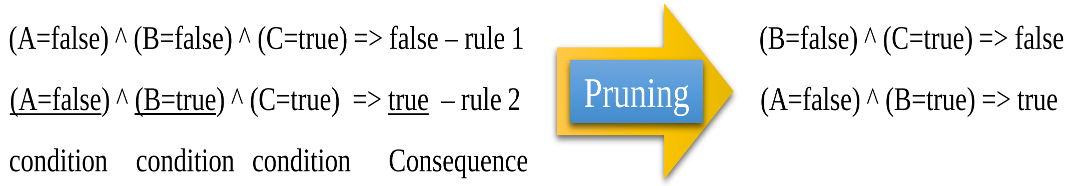

(A=false) ^ (B=false) ^ (C=true) => false – rule 1
(A=false) ^ (B=true) ^ (C=true) => true – rule 2
condition
condition
condition
Consequence
Pruning
(B=false) ^ (C=true) => false
(A=false) ^ (B=true) => true

Figure 8 — Example of rules that are converted from the decision tree

3.6 Random Forests

Random forest is an algorithm based on the decision tree algorithm. Random forests are a combination of multiple predictors such that each tree depends on an independently random parameter and there is the same distribution for all trees in the forest from the original data set. Parameter limits tree can only have a subset of all possible attributes at each node in it. Node only can pick the best attributes from the subset. That is why random forest is an effective tool in prediction, and the Law of Large Numbers is not over fitting training examples. After a large number of trees are created, they vote for the majority class to get a prediction. This paper calls these procedures as random forests. The advantages of random forest, one is strong on the noisy data, it also works better on weak attributes data compares to the single decision tree.

3.7 Random Forests with K-Fold Cross-Validation

Since forest size and attribute size in each tree is a changeable parameter in the random forest algorithm, this paper tries to use k-fold cross-validation [17] (see Figure 9) to solve this limitation as much as we can. The goal is to find out the best forest size and attribute size in each tree to substitute changeable parameters. The original training data are divided to K pieces, K-1 pieces is going to be new training data. The remaining one would be treated as validation examples. Validation examples exchange with other k-1 pieces without overlapping of examples until all examples became validation example once. In order to obtain an average estimate, each validation examples would give a score. Each possible combination of forest size and attribute size would get a score through k-fold cross-validation method, and the program would return the best combination at each period throughout the entire semester.

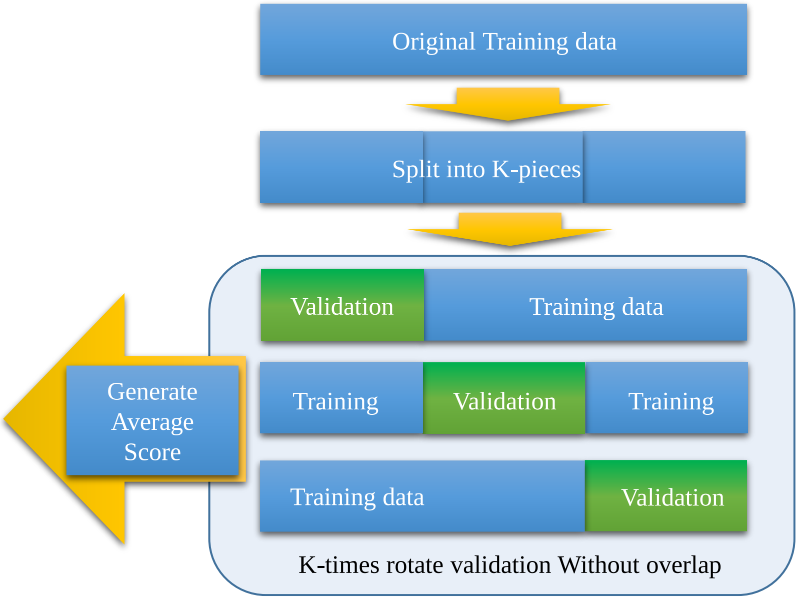


Figure 9 — Example of k-fold cross-validation

3.8 Behaviors in Individual periods versus Cumulative Periods

We consider two types of periods in which we measure the behavioral features discussed above. The first type is cumulative periods. The cumulative measurement refers amount of data is accumulating with the time increasing. This approach allows us to detect important behavior so far. This approach can be expected to alleviate the unevenness of student behavior occurs among a short period. Thereby it can be expected to increase the accuracy of detecting poor performance student (see section 4.4, section 4.5 and section 4.6). The second type is individual periods. Unlike the first type, the data is not cumulative with time increasing. This approach allows us to detect important behavior for a particular period. We can know important behavior in current period (see section 4.7 page 44).

Chapter 4
Experimental Evaluation

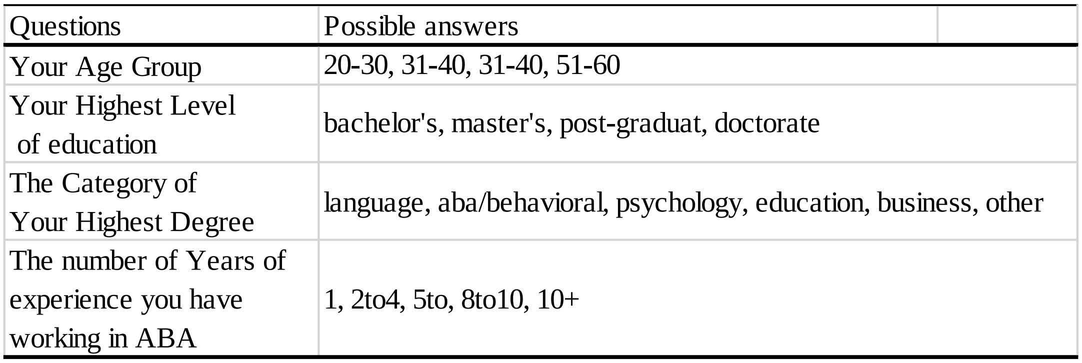
4.1 Experimental Data

This paper investigates one course: BEHP5000, which is “Concepts and Principles of Behavior Analysis”. This graduate-level 45-hour course covers concepts, principles and processes derived from the experimental analysis of behavior, how they are related to Applied Behavior Analysis, and definitions and characteristics of ABA. The period of this course is from January 2013 to April 2013, the spring semester of 2013 in Florida Institute of Technology. BEHP5000 is offered by Jose A. Martinez-Diaz, Ph.D., BCBA.

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. ASRs is short for Active Student Responding Exercises; questions provided by the instructor during instructional videos. Students will write down their responses and later enter them into ANGEL. See Acquisition ASRs below for more information. Moreover, 10 1-hour interactive online sessions with co-instructors are scheduled in the entire semester. 6 CyberRat Assignments (optional) involve the application of the principles that you will learn in class to the training of a virtual rat. In addition to being entertaining, these assignments are designed to give you a more complete understanding of the concepts and principles taught throughout the course. BEHP5000 is also planned to have 8 online tests (two forms available) and one online proctored final exam.

We use 110 students from BEHP5000. The passing criterion of BEHP5000 is to earn 70% (101 points) or above on the final exam as well as above 800/1000 (80%) overall score for the course. This paper uses three log files from the course: one is a student’s activity log file including 110 students’ activities in course. This file comprises information of single activity starting date (moth, day and year), time (hour, minute and second), activity and object title of activity at this point. The next file is a grade information file including 110 students' grade information. Moreover, prior to access to the course materials students will first need to complete the Student Information Survey. Basically, all questions will be answered to better serve students' needs. Student Information Survey log file includes information shown in Table 7 and Figure 10.

Table 7 — These questions have been used to build student profile and possible answers of questions in Student Information Survey log file.



Questions	Possible answers
Your Age Group	20-30, 31-40, 31-40, 51-60
Your Highest Level of education	bachelor's, master's, post-graduat, doctorate
The Category of Your Highest Degree	language, aba/behavioral, psychology, education, business, other
The number of Years of experience you have working in ABA	1, 2to4, 5to, 8to10, 10+

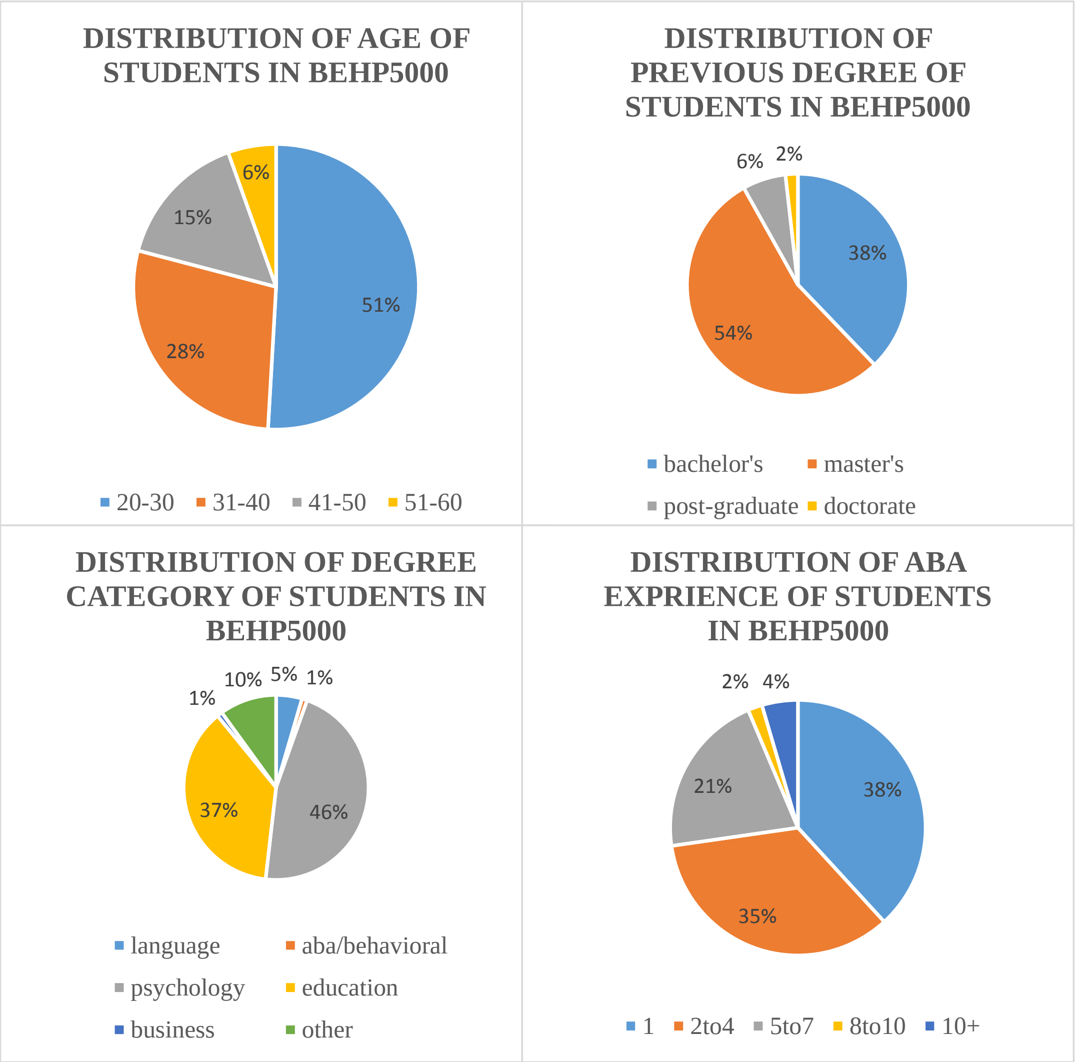


Figure 10 — The features and distributions of previous experience of these 110 students in BEHP5000.

4.2 Evaluation Criteria

This section discusses the method for verifying whether students' learning profiles are valid or not. Further, in the identification problem, if our system has 1% of positive examples and 99% of negative examples, our prediction would ignore the minor class. For such data, the false positive rate is greatly increased.

Table 8 — Vertical actual classes, lateral was predicted classes, the number of samples in that the upper left and lower right is hit the prediction, the number that the upper right and lower left is off the prediction, and give the true and false, respectively, positive, negative represents whether prediction is true and false.

		predicted class	
		postitive	negative
real class	positive	true positive(TP)	false negative (FN)
	negative	false positive (FP)	true negative (TN)

This paper suggests that the main evaluation criteria are the prediction accuracy, the true positive rate of the prediction and false positive rate of prediction. Positive examples refer to those students who exceeded the average score or passed the first attempt in the final test. Negative examples refer to those students who couldn't exceed the average score or couldn't pass the first attempt in the final test. The accuracy is calculated by adding TP (True Positive) and TN (True Negative), then dividing by the total sum. Basically, the higher accuracy the better (see Equation 2).

$$(\text{Accuracy}) = (\text{TP} + \text{TN}) / (\text{TP} + \text{FP} + \text{TN} + \text{FN})$$

Equation 2 — Accuracy

Basically, the true positive rate refers to the performance of predicting true examples, the higher true positive rate the better. The false positive rate refers to how many negative examples that we have predicted wrongly, the lower FP rate the better. True positive rate can be expressed by Equation 3. False positive rate can be expressed by Equation 4.

$$\text{(True Positive Rate)} = \text{(TP)} / \text{(TP + FN)}$$

Equation 3 — True positive rate

$$\text{(False Positive Rate)} = \text{(FP)} / \text{(FP + TN)}$$

Equation 4 — False positive rate

4.3 Experimental Procedure

Firstly, this section will introduce various parameters used in this paper, the experimental data and the main steps of the program. Two thirds of students are randomly selected as the training examples, and the rest of students are the test examples. We use training examples to build profiles of students' learning behavior, and test examples are used to deperiodine the accuracy of the constructed profiles. Additionally, we have a validation set split from the training set to prevent over-fitting problem of decision tree learning. Validation examples are randomly picked from training examples; the number of validation examples is 20% of entire training examples. However, random forest does not require validation examples.

Secondly, the most important task in this paper is using features to construct learning behavior profiles of students. All the activities of the students in the BEHP5000 are listed in Table 1, Table 2, Table 3 and Table 4. Besides, these activities are extracted from syllabus and logs manually; this paper also applies the sequential pattern mining algorithm to extract more important learning sequences from the log file. Two main parameters have been used in the sequential pattern mining algorithm, which are the maximum gap and the minimum support. For example, in this paper the maximum gap is two, we allow less than two irrelevant activities existing between two activities that we want to extract. The minimum support refers to that the least number of activity sequences occurring among different students. In the paper, the minimum support is 70% of all students.

Thirdly, three predicting tasks of our paper are: whether the overall score of a student is greater than the average score or not, whether the final exam score of a student is greater than the average score or not, and whether a student passes the final exam on the first try or not. Further, each task will be predicted at different periods of a semester, in order to capture students' profiles in the early period of a semester. Besides the above work, we also predict three events in different conditions. Although the Score related features have great impacts on identifying student performance, we predict three events without using score relevant features. Because one of our goal is to detect the important learning behavior.

Finally, besides above work, we also use random forest [16] to learn student's behavior. In a random forest, a number of decision trees are generated, and the forest votes for the most popular class. The forest size is 399 in the paper [16], and randomly picked subset of attributes size is (log2*M); M is total attributes number. We use k-fold cross-validation to generate the best combination of forest size and attribute set size. The size of random forest is from 99 to 999, and the size of randomly selected subset of attributes is from 8 to 55. The best combination of attributes size and forest size aims to maintain the low correlation between each trees in the random forest; otherwise we are just building a lot of the similar trees at a time, and the random forest algorithm has no difference from the decision tree algorithm.

4.4 Results and Discussions

All below predictions in this section use the cumulative data in the experiments. In other words, the data increases and accumulates with the time increasing. This approach can alleviate the unevenness of student behavior occurring among short periods. Thereby, it can be expected to increase the accuracy of detecting poor performance students. We also discuss the result of the individual period data prediction in section 4.7 (see page 44).

4.4.1 Predicting Performance on Final Exam with Cumulative Periods

The first task is to measure the method of predicting performance on the final exam in different periods throughout the entire semester. We measure whether student can exceed the average score in the final test or not. Classes of prediction are above the average score and below the average score. 77 students are randomly selected as training data, and 33 students as test data. Further, 40 students are above the average score in 77 students. This task is very challenging because there are not direct facts that have an effect on this task. We do not predict whether a student will pass or fail in final test because most students will pass.

According to following Figure 11 and Table 9, the accuracy of random forest with k-fold cross-validation is the best among four algorithms on the average. On the average, True Positive and False Positive rate of random forest with k-fold cross-validation algorithm is the best among four algorithms. The accuracy of this algorithm reaches 84% after period two, and further, the accuracy still slowly increases. However, the training of random forest with k-fold cross-validation takes the most time (see Figure 12).

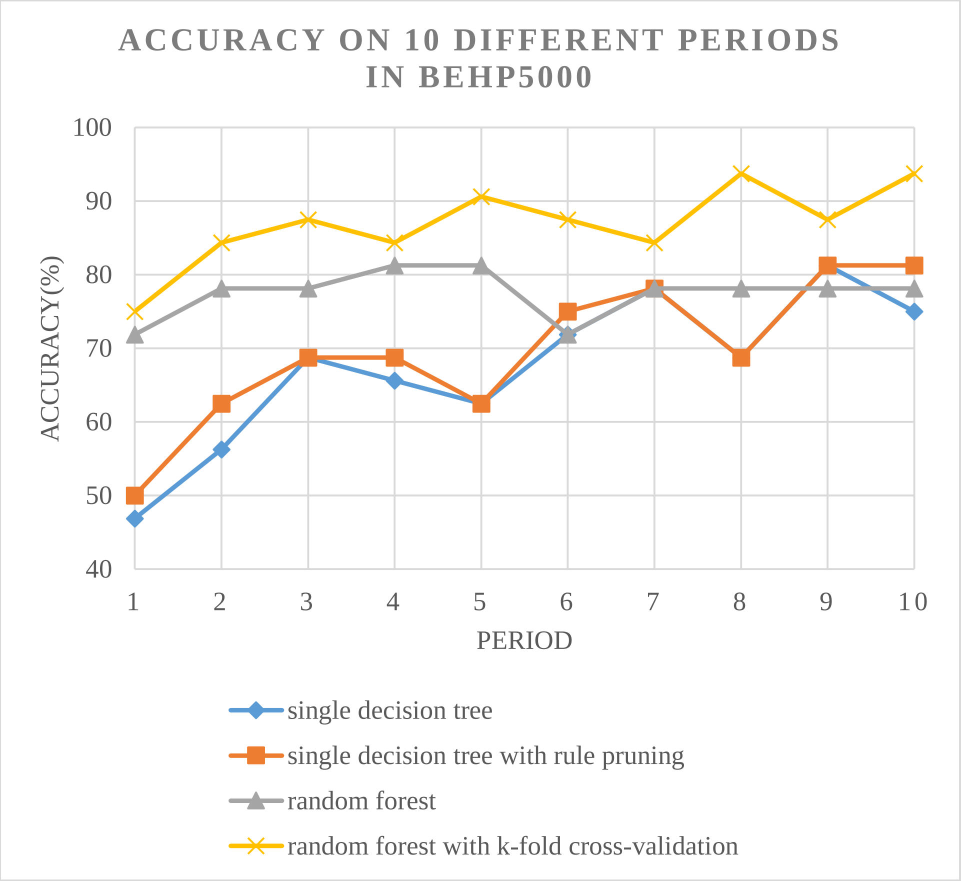
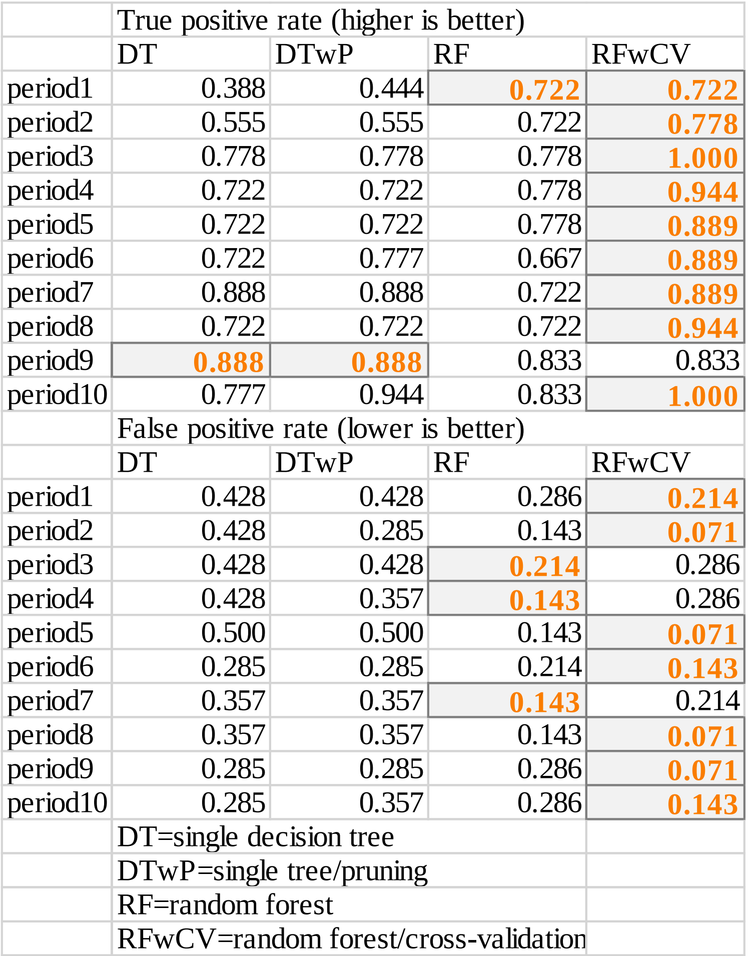


Figure 11 — Accuracy on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)

Table 9 — True positive and false positive rates on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)



	True positive rate (higher is better)			
	DT	DTwP	RF	RFwCV
period1	0.388	0.444	**0.722**	**0.722**
period2	0.555	0.555	0.722	**0.778**
period3	0.778	0.778	0.778	**1.000**
period4	0.722	0.722	0.778	**0.944**
period5	0.722	0.722	0.778	**0.889**
period6	0.722	0.777	0.667	**0.889**
period7	0.888	0.888	0.722	**0.889**
period8	0.722	0.722	0.722	**0.944**
period9	**0.888**	**0.888**	0.833	0.833
period10	0.777	0.944	0.833	**1.000**
	False positive rate (lower is better)			
	DT	DTwP	RF	RFwCV
period1	0.428	0.428	0.286	**0.214**
period2	0.428	0.285	0.143	**0.071**
period3	0.428	0.428	**0.214**	0.286
period4	0.428	0.357	**0.143**	0.286
period5	0.500	0.500	0.143	**0.071**
period6	0.285	0.285	0.214	**0.143**
period7	0.357	0.357	**0.143**	0.214
period8	0.357	0.357	0.143	**0.071**
period9	0.285	0.285	0.286	**0.071**
period10	0.285	0.357	0.286	**0.143**
	DT=single decision tree			
	DTwP=single tree/pruning			
	RF=random forest			
	RFwCV=random forest/cross-validation			

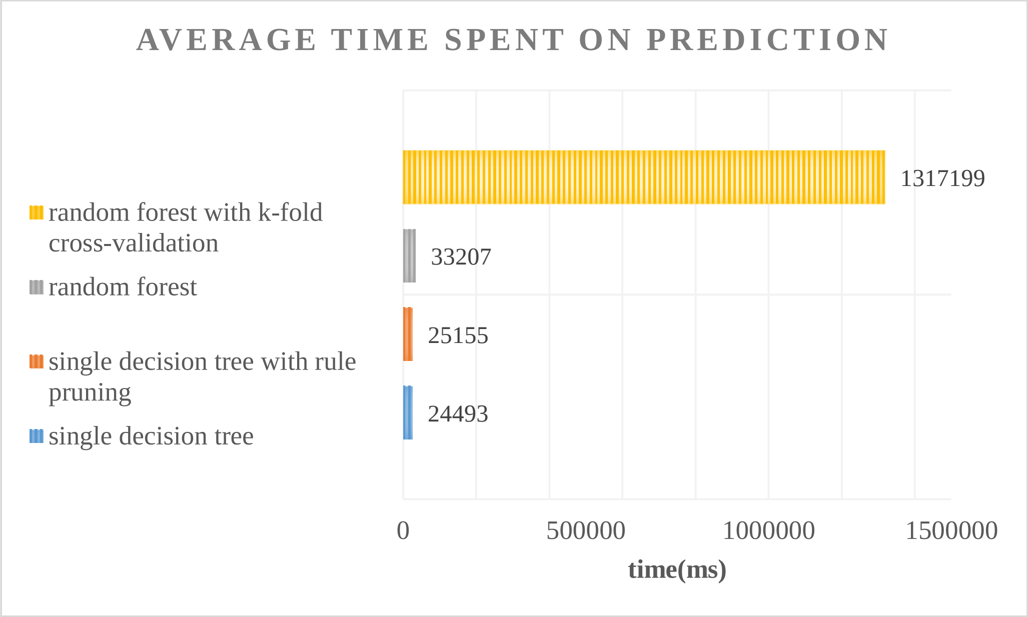


Figure 12 — Average time spent on predicting performance on final exam by decision tree (with/without pruning) and random forest (with/without cross validation)

4.4.2 Predicting Success or Failure in the First Attempt of Final Exam with Cumulative Periods

We measure whether student can pass the final exam in the first attempt. Compared to the last task, this task is more difficult to predict. The first reason is that the prediction is highly related to students' learning behavior. The second reason is that we do not use any attributes has direct impact on this prediction. This task highly requires to capture important behavior successfully. Classes of prediction are success and failure in the first attempt in the final test. 77 students are randomly selected as training data, and 33 students as test data. Further, 48 students pass the first attempt of the final exam in training data.

According to following Figure 13 and Table 10, the accuracy of random forest with k-fold cross-validation is the best among the four algorithms on the average accuracy. On the average, True Positive and False Positive rate of random forest with k-fold cross-validation algorithm is the best among four algorithms. The accuracy of prediction of this algorithm reaches 90% after period two, and further, the accuracy still slowly increases. However, the training of random forest with k-fold cross-validation takes the most time (see Figure 14).

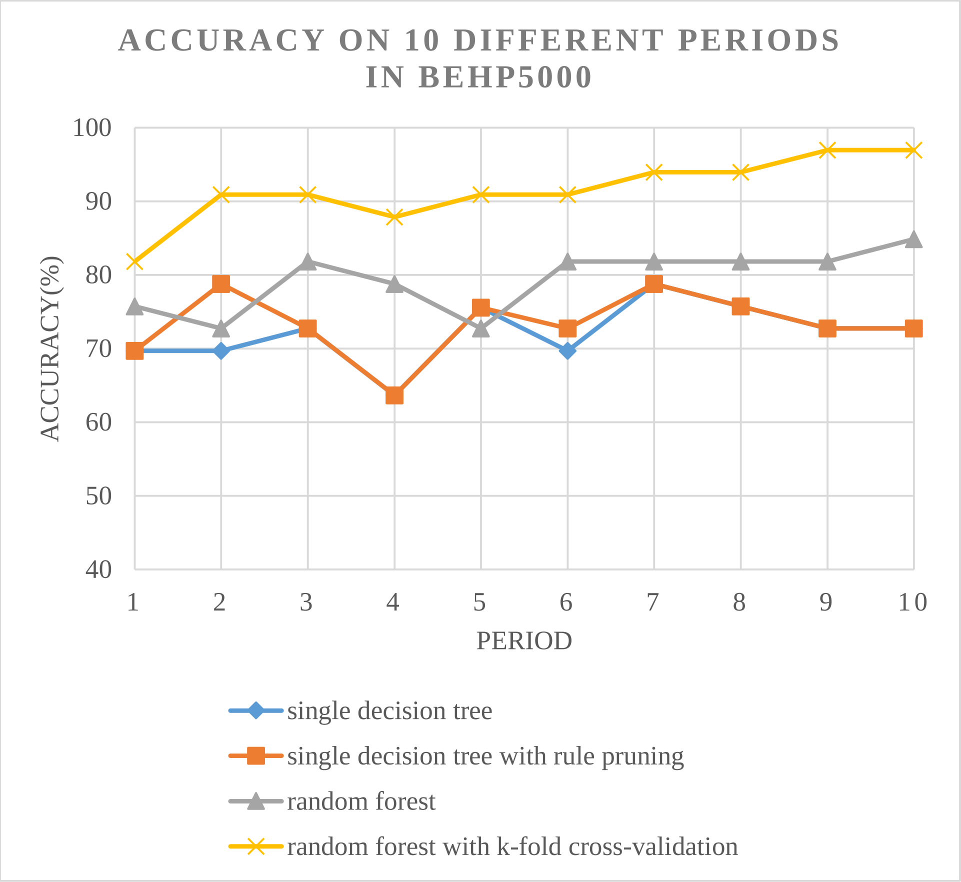
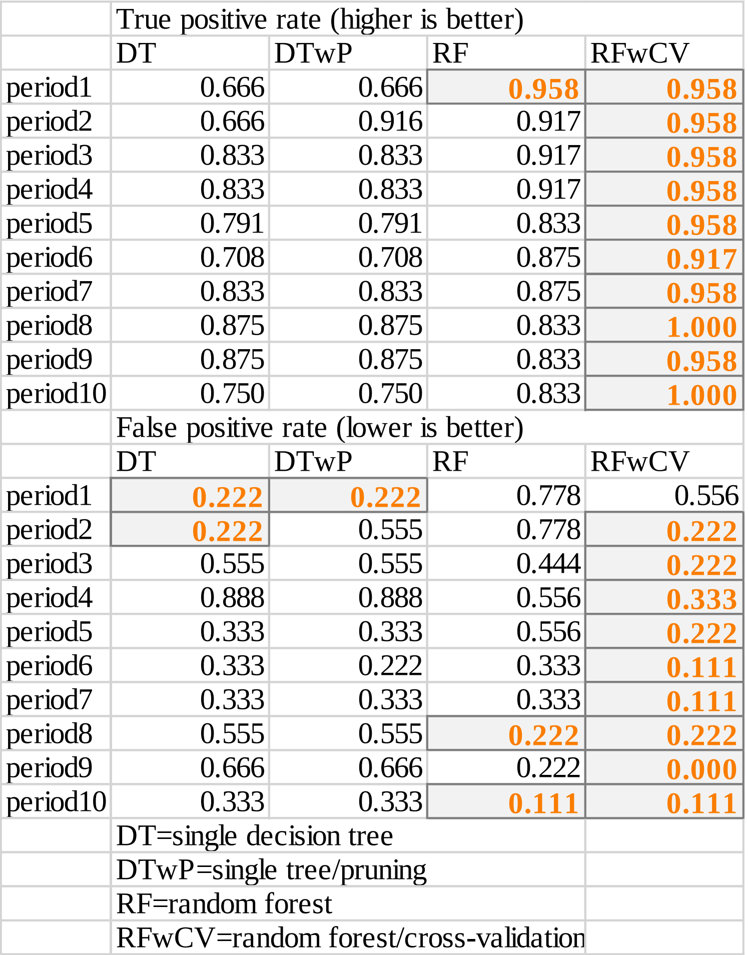


Figure 13 — Accuracy on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)

Table 10 — True positive and false positive rates on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)



	True positive rate (higher is better)			
	DT	DTwP	RF	RFwCV
period1	0.666	0.666	**0.958**	**0.958**
period2	0.666	0.916	0.917	**0.958**
period3	0.833	0.833	0.917	**0.958**
period4	0.833	0.833	0.917	**0.958**
period5	0.791	0.791	0.833	**0.958**
period6	0.708	0.708	0.875	**0.917**
period7	0.833	0.833	0.875	**0.958**
period8	0.875	0.875	0.833	**1.000**
period9	0.875	0.875	0.833	**0.958**
period10	0.750	0.750	0.833	**1.000**
	False positive rate (lower is better)			
	DT	DTwP	RF	RFwCV
period1	**0.222**	**0.222**	0.778	0.556
period2	**0.222**	0.555	0.778	**0.222**
period3	0.555	0.555	0.444	**0.222**
period4	0.888	0.888	0.556	**0.333**
period5	0.333	0.333	0.556	**0.222**
period6	0.333	0.222	0.333	**0.111**
period7	0.333	0.333	0.333	**0.111**
period8	0.555	0.555	**0.222**	**0.222**
period9	0.666	0.666	0.222	**0.000**
period10	0.333	0.333	**0.111**	**0.111**
	DT=single decision tree			
	DTwP=single tree/pruning			
	RF=random forest			
	RFwCV=random forest/cross-validation			

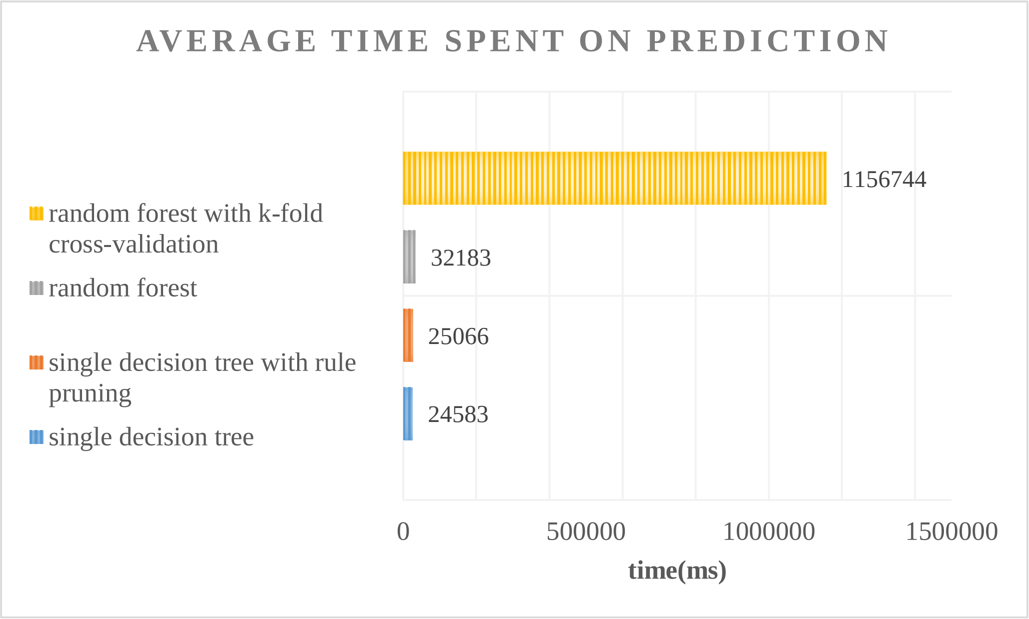


Figure 14 — Average time spent on predicting success and failure in the first attempt by decision tree (with/without pruning) and random forest (with/without cross validation)

4.4.3 Predicting Performance of Overall Score in BEHP5000 with Cumulative Periods

Since the success and failure of students depend on overall score, predicting the overall score is another task that must be measured. The overall score is calculated by test scores and other material scores, but this paper only uses students' behavior that happened and is measurable in BEHP5000 to do prediction in this task because this paper aims to identify the significant behavior as early as possible. Since most students pass this course, we predict whether students can exceed the average score in overall score or not. Classes of prediction are above the average and below the average. 77 students are randomly selected as training data, and 33 students as test data. Further, 42 students are above the average score in training example.

According to following Figure 15 and Table 11, the accuracy of random forest with k-fold cross-validation is the best among the four algorithms on average. On average, True Positive and False Positive rate of random forest with k-fold cross-validation algorithm is the best among the four algorithms. The accuracy of prediction of this algorithm reaches 87% after period two, and further, the accuracy still slowly increases. However, the training of random forest with k-fold cross-validation takes the most time (see Figure 16).

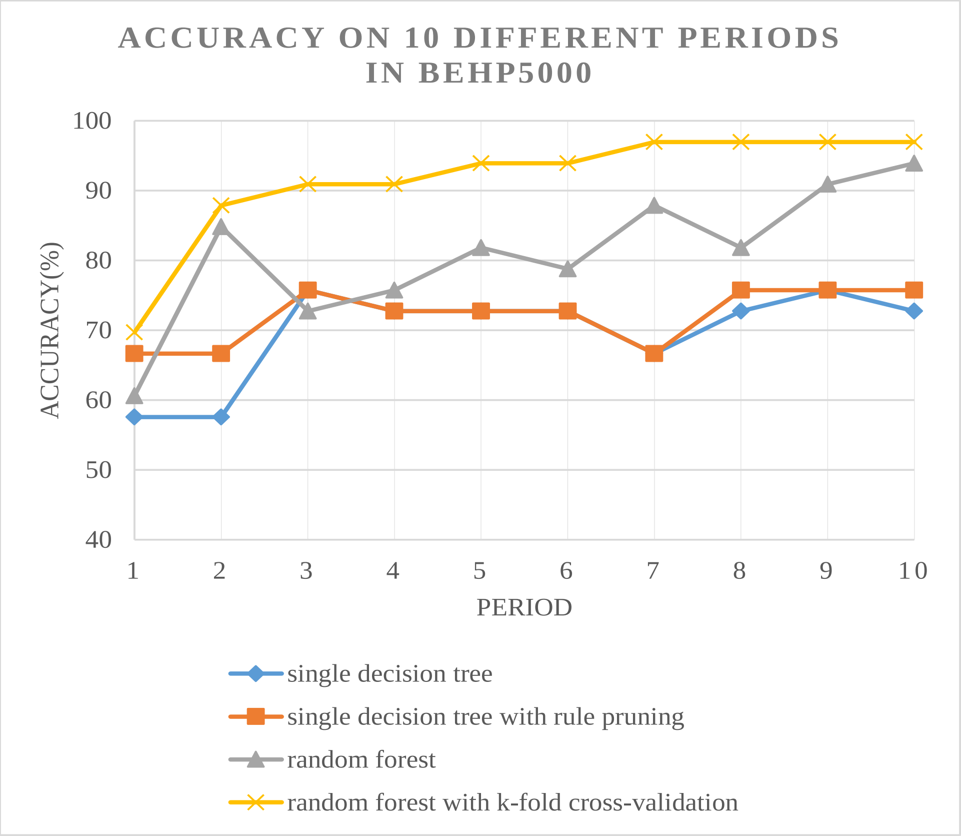


Figure 15 — Accuracy on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)

Table 11 — True positive and false positive rates on 10 different periods in BEHP5000 by decision tree (with/without pruning) and random forest (with/without cross validation)

	True positive rate (higher is better)			
	DT	DTwP	RF	RFwCV
period1	0.650	0.750	0.650	0.750
period2	0.650	0.800	0.850	0.900
period3	0.750	0.750	0.750	1.000
period4	0.750	0.750	0.750	0.900
period5	0.800	0.800	0.800	0.950
period6	0.600	0.600	0.750	0.950
period7	0.650	0.650	0.850	0.950
period8	0.750	0.750	0.800	0.950
period9	0.800	0.800	0.900	1.000
period10	0.850	0.850	0.900	1.000
	False positive rate (lower is better)			
	DT	DTwP	RF	RFwCV
period1	0.538	0.461	0.462	0.385
period2	0.538	0.538	0.154	0.154
period3	0.230	0.230	0.308	0.231
period4	0.307	0.307	0.231	0.077
period5	0.384	0.384	0.154	0.077
period6	0.077	0.077	0.154	0.077
period7	0.307	0.307	0.077	0.000
period8	0.307	0.230	0.154	0.000
period9	0.307	0.307	0.077	0.077
period10	0.461	0.384	0.077	0.077
	DT=single decision tree			
	DTwP=single tree/pruning			
	RF=random forest			
	RFwCV=random forest/cross-validation			

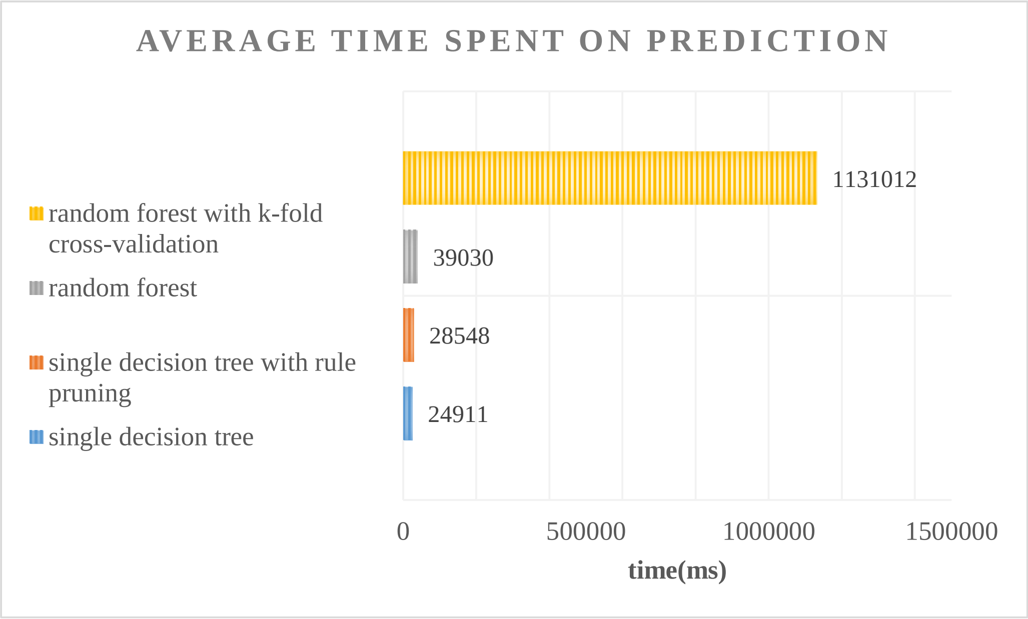


Figure 16 — Average time spent on predicting performance on final exam by decision tree (with/without pruning) and random forest (with/without cross validation)

4.4.4 Discussions of Results

According to Figure 11, Figure 13 and Figure 15, random forest with k-fold cross-validation has the highest effectiveness on BEHP5000 because it has advantages on robust to outliers (noisy data). Further, the random forest is able to reveal the importance of weak attributes. Rule pruning increases the accuracy of single decision tree effectively. Another important observation is that combining random forest with k-fold cross-validation is possible to improve usability and performance greatly. Further, the low false positive rate in Table 9, Table 10 and Table 11 proves the reliability of this combination method. Although random forest with k-fold cross-validation method shows outstanding result, this combination method takes a long time to train, according to Figure 12, Figure 14 and Figure 16.

The main contribution of this paper is that we are able to predict difficult tasks in the period as early as possible, which means it is not necessary to wait until the end of the semester. Further, we are able to identify important learning behavior at early time, and use these detected behavior to give suggestions to improve the performance for the entire course. Additionally, the instructor is able to leverage detected behavior to improve teaching. The features and learning methods proposed in this paper which capture learning profile as much as possible are also the great contribution.

4.5 Important Student Behavior with Cumulative Periods

In this section we list all the most important behavior we detected. In order to show students' behavior changing with time transition in BEHP5000, we study the top 3 important behavior for each period in each task. The detected behavior comes from root nodes of random forest. We count the number of occurrences of each feature in root nodes. Since root node has the most important meaning for decision tree to do the classification, the more frequency of appearance the more important. In the random forest, each root node of a tree comes from a random chosen subset of all attributes, the attribute repeatedly appearing at most root nodes must bring more important information than others.

According to Table 12 below, in the first half semester we detected two frequent attributes which are *ASRs* and *how early students get to start materials* (blue marks and green marks in Table 12), so the feature type of components usage and positive attitude related behavior in course seem to be highly important. In the rest of semester *reviewing course materials* is another important behavior. The important behavior changes among each period because of following three reasons. One is constructive facts of random forest. The important behavior comes from root nodes of random forest, and root nodes are from randomly chosen subset of attributes. The second reason is that the amount of data in each period is different. Last, different periods could have different priority tasks.

The more specific description of following important behavior is referred in the Table 1, Table 2, Table 3 and Table 4.

Table 12 — Detected top 3 of the most important single behavior on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in three tasks

		Detected important student behavior		
Period	NO.	task1	task2	task3
period1	top. 1	avg(days.af.unt.avb)	(totalppthours)	(totalcybtimes)
	top. 2	(totaldiscustimes)	min(totalppthours)	(totaldiscustimes)
	top. 3	(totalactivities)	(totalstyguidetimes)	(totalactivities)
period2	top. 1	(sub.bf.tst.due)	(sub.bf.tst.due)	(totallogintimes)
	top. 2	(totallogintimes)	min(totalppthours)	min(totalppthours)
	top. 3	avg(days.af.unt.avb)	min(totalstyguidetimes)	min(totallogintimes)
period3	top. 1	sum(days.af.unt.avb)	(totalactivities)	sum(days.af.unt.avb)
	top. 2	(totalasrtimes)	(sub.bf.tst.due)	(totalunttimes)
	top. 3	(sub.bf.tst.due)	(regularday)	(totalunthours)
period4	top. 1	(totaldiscustimes)	(sub.bf.tst.due)	sum(days.af.unt.avb)
	top. 2	(totalasrtimes)	(regularday)	(sub.bf.tst.due)
	top. 3	sum(days.af.unt.avb)	(totaldiscustimes)	min(meeting)
period5	top. 1	sum(days.af.unt.avb)	(totalactivities)	sum(days.af.unt.avb)
	top. 2	(totalasrtimes)	sum(days.af.unt.avb)	(totallogintimes)
	top. 3	(totalstyguidetimes)	avg(days.af.unt.avb)	(totalstyguidetimes)
period6	top. 1	total_asr	(sub.bf.tst.due)	sum(days.af.unt.avb)
	top. 2	sum(nubofperiod)	(regularday)	(sub.bf.tst.due)
	top. 3	min(meeting)	avg(days.af.unt.avb)	(totallogintimes)
period7	top. 1	(sub.bf.tst.due)	(sub.bf.tst.due)	(totalactivities)
	top. 2	(totalstyguidetimes)	sum(days.af.unt.avb)	(sub.bf.tst.due)
	top. 3	(totalactivities)	avg(days.af.unt.avb)	(totalunthours)
period8	top. 1	(totalasrtimes)	(regularday)	(sub.bf.tst.due)
	top. 2	(sub.bf.tst.due)	sum(days.af.unt.avb)	sum(days.af.unt.avb)
	top. 3	avg(reviewtimes)	(sub.bf.tst.due)	(totalunthours)
period9	top. 1	(totaldiscustimes)	(sub.bf.tst.due)	sum(days.af.unt.avb)
	top. 2	(totalactivities)	sum(days.af.unt.avb)	(totalunthours)
	top. 3	(totalsuptimes)	avg(days.af.unt.avb)	(sub.bf.tst.due)
period10	top. 1	sum(reviewtimes)	(totaldiscustimes)	sum(days.af.unt.avb)
	top. 2	(sub.bf.tst.due)	sum(days.af.unt.avb)	(sub.bf.tst.due)
	top. 3	(totalasrtimes)	(sub.bf.tst.due)	(totalunthours)

task1=prediction of exceeding the average in the final test

task2=prediction of success and failure in the first attempt in the final test

task3=prediction of exceeding the average in overall score

4.6 Behavior Combinations with Cumulative Periods

We discuss behavior combinations in this section. Since upper level nodes of a decision tree has the most important meaning to do the classification, behavior combinations comes from root node combined with second level attributes. We attempt to find the behavior combinations of students which most likely lead to pass BEHP5000. Therefore, a score of a behavior combination that is computed by dividing the number of positive examples by the total number of examples in this combination. A higher score behavior combination is more likely to be successful in the course potentially.

According to following tables, in the root nodes, *discussion* repeatedly appears through the entire semester, and especially, it plays important role in the first half of semester. Further, *review* also repeatedly appears in the second half of semester. When we expand to the second level of tree, we can see sequential pattern attributes also seem to be important for our paper. Basically, these feature type of positive attitude related behavior is important.

For example, behavior combinations *sum(days.af.unt.avb)>4.5^(totalvdotimes)>3.5* has the highest score in Table 13. It refers to student get to start study unit materials 4.5 days after unit materials released combining with total video times are above 3.4 times in period 1. If you get to study materials later that 4.5 days and watch more videos, you seems to success. *sum(days.af.unt.avb)<=9.5^(totaldiscustimes)>1 is* another high score behavior combination in Table 13. It refers to that a student should study unit materials early and do discussion more than one time. The behavior combinations with the highest score in the first half semester are marked blue Table 13, Table 14 as well as Table 15.

The more specific description of following important behavior is referred in the Table 1, Table 2, Table 3 and Table 4.

Table 13 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of exceeding the average in the final test

Period	NO.	Detected important behavior	Score
period1	top. 1	sum(days.af.unt.avb)>4.5^(totalfaqtimes)<=3.5	291.1
	top. 2	sum(days.af.unt.avb)<=4.5^(totalvdotimes)>3.5	230.9
	top. 3	totaldiscustimes)>0.5^(totaldrilltimes)<=0.5	106.6
period2	top. 1	(sub.bf.tst.due)<=9.5^(totaldiscustimes)>1.0	217.3
	top. 2	sum(days.af.unt.avb)>17.0^(sub.bf.tst.due)<=9.5	154.7
	top. 3	sum(days.af.unt.avb)<=17.0^(totaldiscustimes)>1.0	116.1
period3	top. 1	sum(days.af.unt.avb)>24.5^(sub.bf.tst.due)<=12.5	41.26
	top. 2	sum(days.af.unt.avb)<=24.5^(totalasrhours)>4.5	30.34
	top. 3	(totalasrtimes)>4.5^(totalloginhours)>1306.0	17.61
period4	top. 1	(totaldiscustimes)>5.5^(components_coverage)<=37.5	36.25
	top. 2	(totalasrtimes)<=4.5^[unt,sgm,unt]=false	23.5
	top. 3	(totallogintimes)>3.5^[vdo,unt]=false	19.16
period5	top. 1	sum(days.af.unt.avb)<=48.5^(studydays)<=17.0	59.56
	top. 2	sum(days.af.unt.avb)>48.5^(totalactivities)>63.5	44.6
	top. 3	sum(days.af.unt.avb)<=48.5^(components_coverage)<=25.0	19.16
period6	top. 1	(sub.bf.tst.due)>9.5^(totalsuptimes)<=5.0	101.5
	top. 2	(sub.bf.tst.due)>9.5^(totallogintimes)>4.5	70.49
	top. 3	(totalstyguidetimes)>0.5^[unt,sgm]=false	48.83
period7	top. 1	(totalasrtimes)>2.5^(totalfaqtimes)<=7.0	17.98
	top. 2	(totalasrtimes)<=2.5^sum(reviewtimes)>18.0	10.86
	top. 3	(totalactivities)>22.0^(totalasrtimes)>2.5	10.03
period8	top. 1	(totalasrtimes)>2.5^(totalloginhours)>370.0	7.69
	top. 2	(sub.bf.tst.due)>17.5^(totalvdohours)<=49.5	7.57
	top. 3	sum(vdo.af.unt.avb)>15.5^(totalasrtimes)>2.5	6.31
period9	top. 1	(totalsuptimes)<=0.5^(sub.bf.tst.due)>21.5	106.3
	top. 2	(totalactivities)>15.5^(totalloginhours)>160.5	105.6
	top. 3	(totalsuptimes)>0.5^(totaldischours)>7.0	84.69
period10	top. 1	(sub.bf.tst.due)<=36.5^sum(reviewtimes)<=39.5	17.59
	top. 2	sum(reviewtimes)<=68.0^(components_coverage)<=14.5	13
	top. 3	(sub.bf.tst.due)<=36.5^(totaldiscustimes)>5.5	9.5

Table 14 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of success and failure in the first attempt in the final test

Period	NO.	Detected important behavior	Score
period1	top. 1	(totalppthours)>0.5^(totalsuptimes)<=2.5	42.49
	top. 2	(totalppthours)<=0.5^(totalloginhours)<=1941.0	29.38
	top. 3	(totalstyguidetimes)>2.5^(totalsuptimes)<=3.5	20.14
period2	top. 1	(sub.bf.tst.due)>0.5^min(totaldisctimes)>0.5	67.23
	top. 2	min(totalppthours)>0.5^(totaldrillhours)<=5.5	40.34
	top. 3	min(totalstyguidetimes)>0.5^min(totaldischours)<=281.5	23.3
period3	top. 1	(totalactivities)>52.5^(totalvdohours)<=174.0	29.44
	top. 2	(sub.bf.tst.due)<=11.5^(totalvdohours)<=174.0	10.69
	top. 3	(regularday)>2.5^(sub.bf.tst.due)>6.5	6.97
period4	top. 1	(sub.bf.tst.due)>7.5^(totaldrilltimes)<=2.5	55.56
	top. 2	(regularday)>2.5^(sub.bf.tst.due)>7.5	37.83
	top. 3	(sub.bf.tst.due)<=7.5^(totaldiscustimes)>1.5	36.41
period5	top. 1	(totalactivities)>26.0^(totalvdohours)<=104.5	99.23
	top. 2	sum(days.af.unt.avb)<=48.5^(totaldiscustimes)>1.5	71.92
	top. 3	avg(days.af.unt.avb)>4.5^(totalvdohours)<=104.5	46.44
period6	top. 1	(regularday)>2.5^(sub.bf.tst.due)>7.5	37.3
	top. 2	avg(days.af.unt.avb)>5.5^(totalvdohours)<=86.5	28.3
	top. 3	(sub.bf.tst.due)<=7.5^(totaldiscustimes)>1.5	26.05
period7	top. 1	sum(days.af.unt.avb)>67.5^(components_coverage)<=13.5	25.95
	top. 2	(sub.bf.tst.due)>6.5^(totalasrtimes)<=3.5	23.07
	top. 3	(components_coverage)>12.0^(sub.bf.tst.due)>12.0	22.37
period8	top. 1	(regularday)>2.5^(totaldrillhours)<=1.5	69.16
	top. 2	sum(days.af.unt.avb)>74.5^[asr,unt,unt]=false	62.75
	top. 3	avg(days.af.unt.avb)<=9.5^(regularday)>2.5	47.23
period9	top. 1	(sub.bf.tst.due)>7.5^(totalvdohours)<=70.0	61.21
	top. 2	(sub.bf.tst.due)<=7.5^(components_coverage)>9.5	38
	top. 3	sum(days.af.unt.avb)<=83.5^(totalvdohours)<=58.0	29.16
period10	top. 1	(totaldiscustimes)>1.5^(sub.bf.tst.due)<=17.0	23.97
	top. 2	(totaldiscustimes)>1.5^(totalunthours)<=12.5	17.89
	top. 3	(sub.bf.tst.due)<=17.5^(totalvdohours)>17.0	17.83

Table 15 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of exceeding the average in overall score

Period	NO.	Detected important behavior	Score
period1	top. 1	(totalcybtimes)<=1.5^(totaldrilltimes)<=2.5	50.62
	top. 2	(totaldiscustimes)<=8.5^(totalstyguidehours)<=118.0	43.37
	top. 3	(totalactivities)>82.5^(totalasrhours)<=16.5	37.78
period2	top. 1	(totallogintimes)>10.5^min(totalppthours)>0.5	119.5
	top. 2	min(totalppthours)>3.5^(totalmeethours)<=110.5	104
	top. 3	min(totallogintimes)>10.5^min(totalstyguidehours)<=122.5	92.84
period3	top. 1	(totalunttimes)>9.5^(sub.bf.tst.due)>7.5	26.49
	top. 2	sum(days.af.unt.avb)>22.5^(totaldiscustimes)>3.0	23.77
	top. 3	sum(days.af.unt.avb)<=22.5^(totalunthours)>3.5	17.05
period4	top. 1	sum(days.af.unt.avb)<=40.0^(sub.bf.tst.due)<=8.5	64.2
	top. 2	min(meeting)<=2.5^done_bothformperiod>5.0	55.16
	top. 3	(components_coverage)>29.0^(sub.bf.tst.due)>7.5	50.82
period5	top. 1	sum(days.af.unt.avb)<=48.5^(totalsuphours)<=20.5	66.66
	top. 2	(totallogintimes)>3.5^(totalvdohours)<=62.5	56.77
	top. 3	(totalstyguidetimes)>0.5^sum(days.af.unt.avb)<=48.5	28.56
period6	top. 1	sum(days.af.unt.avb)<=67.5^done_bothformperiod>3.0	44.49
	top. 2	(totallogintimes)<=3.5^(totalasrtimes)>2.5	40.25
	top. 3	(totaldiscustimes)>8.5^(totalppthours)<=1.5	19.87
period7	top. 1	(totalactivities)>22.5^sum(days.af.unt.avb)<=60.5	148.4
	top. 2	(totalactivities)>22.5^(totalvdohours)<=52.5	131.6
	top. 3	(sub.bf.tst.due)>15.5^(components_coverage)<=14.5	86.61
period8	top. 1	(sub.bf.tst.due)<=9.5^done_bothformperiod>11.0	81.33
	top. 2	(sub.bf.tst.due)>9.5^(totalvdohours)<=61.0	60.54
	top. 3	sum(days.af.unt.avb)>71.5^(totalasrhours)>1.5	36.49
period9	top. 1	sum(days.af.unt.avb)<=93.0^done_bothformperiod<=13.0	24.84
	top. 2	(totalunthours)>0.5^(totalvdohours)<=38.0	13.76
	top. 3	(sub.bf.tst.due)>10.5^(totallogintimes)<=2.5	11.83
period10	top. 1	sum(days.af.unt.avb)<=102.5^(totalactivities)>21.5	29.02
	top. 2	(sub.bf.tst.due)>11.5^(totallogintimes)>1.5	23.81
	top. 3	(sub.bf.tst.due)>11.5^done_bothformperiod<=11.0	18.5

4.7 Behavior with Individual Periods

Besides prediction using the cumulative data in section 4.4, we use the individual period data to predict the same tasks as previous section. Since random forest with k-fold cross-validation has the best performance among other 4 approaches, we only show results by this method.

According to Figure 17 and Table 16, the accuracy of task 2 reaches 87% after period two. However the accuracy repeats increasing and decreasing over the time. Compared to the accuracy in Section 4.4, the accuracy of this section is quite varied. On the average, the true positive and false positive rate are also in the similar situation. The accuracy of the individual period is not as good as the cumulative way. The reason is that the cumulative period measurement uses accumulative period of data to do the prediction, so the accuracy increases whenever periods increase.

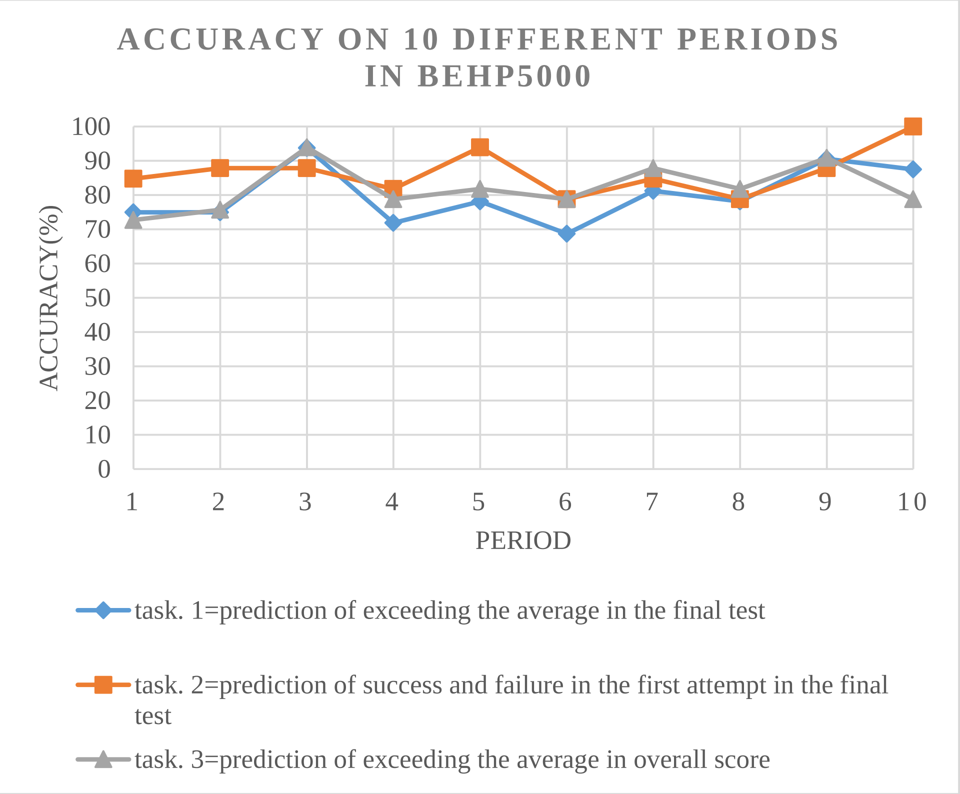
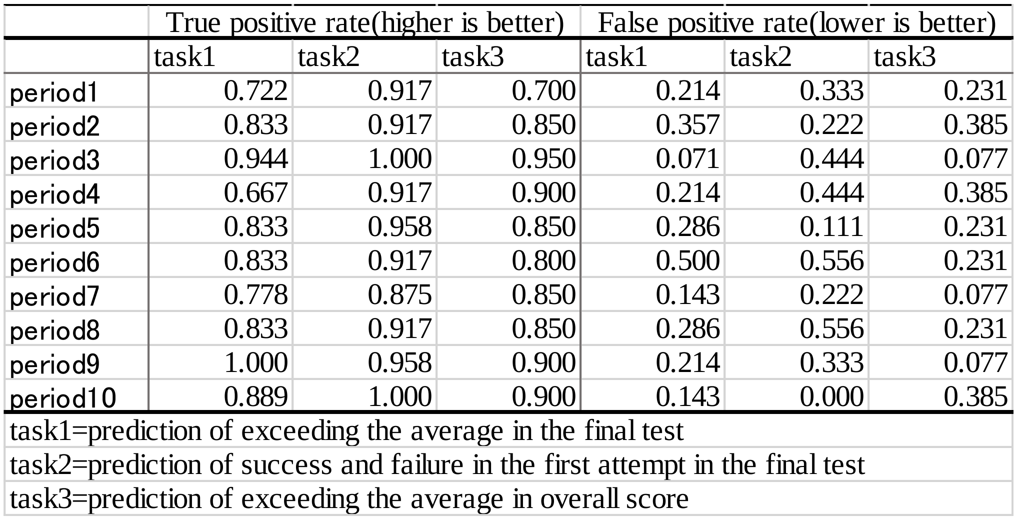


Figure 17 — Accuracy on 10 different periods in BEHP5000 by random forest (with/without cross validation) in three tasks

Table 16 — True positive and false positive rates on 10 different periods in BEHP5000 by random forest (with/without cross validation) in three tasks



	True positive rate(higher is better)			False positive rate(lower is better)		
	task1	task2	task3	task1	task2	task3
period1	0.722	0.917	0.700	0.214	0.333	0.231
period2	0.833	0.917	0.850	0.357	0.222	0.385
period3	0.944	1.000	0.950	0.071	0.444	0.077
period4	0.667	0.917	0.900	0.214	0.444	0.385
period5	0.833	0.958	0.850	0.286	0.111	0.231
period6	0.833	0.917	0.800	0.500	0.556	0.231
period7	0.778	0.875	0.850	0.143	0.222	0.077
period8	0.833	0.917	0.850	0.286	0.556	0.231
period9	1.000	0.958	0.900	0.214	0.333	0.077
period10	0.889	1.000	0.900	0.143	0.000	0.385
task1=prediction of exceeding the average in the final test						
task2=prediction of success and failure in the first attempt in the final test						
task3=prediction of exceeding the average in overall score						

According to Table 17, the single important behavior repeatedly appearing in the first half semester is component usage related features, such as *login times, unit hours*, and *total activities*. Similarly, *how early students get to start unit materials* and *how early they submit test* show frequently through the entire semester. Basically, the positive attitude related behavior seems to be highly important in each period (see blue mark and green mark in Table 17).

According to Table 18, Table 19 and Table 20, not only in the first half semester, *how early students submit test* and *how early they get to start material* seem to be highly important through the whole semester. Additionally, *regular study schedule* and *review times* also play important roles in behavior combinations. The behavior combinations with the highest score in the first half semester are marked blue Table 18, Table 19 as well as Table 20.

Table 17 — Detected top 3 of the most important single behavior on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in three tasks

		Detected important student behavior		
Period	NO.	task1	task2	task3
period1	top. 1	(days.af.unt.avb)	(totalactivities)	(components_coverage)
	top. 2	(totallogintimes)	(totalunttimes)	(totalstyguidehours)
	top. 3	(totalactivities)	(totalppthours)	(totalsuptimes)
period2	top. 1	(totalasrtimes)	(totaldiscustimes)	(totaldiscustimes)
	top. 2	(totallogintimes)	(components_coverage)	(totalunthours)
	top. 3	(totaldiscustimes)	(sub.tst.bf.due)	(totallogintimes)
period3	top. 1	(vdo.af.unt.avb)	(sub.tst.bf.due)	(days.af.unt.avb)
	top. 2	(days.af.unt.avb)	(regularday)	(sub.tst.bf.due)
	top. 3	(sub.tst.bf.due)	(vdo.af.unt.avb)	(vdo.af.unt.avb)
period4	top. 1	(sub.tst.bf.due)	(sub.tst.bf.due)	(days.af.unt.avb)
	top. 2	(reviewtimes)	(reviewtimes)	min(meeting)
	top. 3	total(asr)	(regularday)	(sub.tst.bf.due)
period5	top. 1	(sub.tst.bf.due)	(sub.tst.bf.due)	(sub.tst.bf.due)
	top. 2	(reviewtimes)	(regularday)	min(meeting)
	top. 3	(days.af.unt.avb)	(sub.tst.bf.due)	(reviewtimes)
period6	top. 1	(vdo.af.unt.avb)	(reviewtimes)	(sub.tst.bf.due)
	top. 2	(reviewtimes)	(days.af.unt.avb)	min(asr)
	top. 3	(sub.tst.bf.due)	(days.af.unt.avb)	min(meeting)
period7	top. 1	(sub.tst.bf.due)	(regularday)	(sub.tst.bf.due)
	top. 2	(days.af.unt.avb)	(sub.tst.bf.due)	(days.af.unt.avb)
	top. 3	(vdo.af.unt.avb)	(days.af.unt.avb)	min(asr)
period8	top. 1	(reviewtimes)	total(asr)	(days.af.unt.avb)
	top. 2	(sub.tst.bf.due)	(totalperiodtimes)	(sub.tst.bf.due)
	top. 3	(vdo.af.unt.avb)	(regularday)	(reviewtimes)
period9	top. 1	(sub.tst.bf.due)	(days.af.unt.avb)	(sub.tst.bf.due)
	top. 2	(days.af.unt.avb)	(reviewtimes)	min(asr)
	top. 3	(regularday)	total(asr)	(days.af.unt.avb)
period10	top. 1	done_bothformperiod	done_bothformperiod	done_bothformperiod
	top. 2	(sub.tst.bf.due)	(totalperiodtimes)	(totalperiodtimes)
	top. 3	(totalperiodtimes)	(regularday)	(sub.tst.bf.due)
task1=prediction of exceeding the average in the final test				
task2=prediction of success and failure in the first attempt in the final test				
task3=prediction of exceeding the average in overall score				

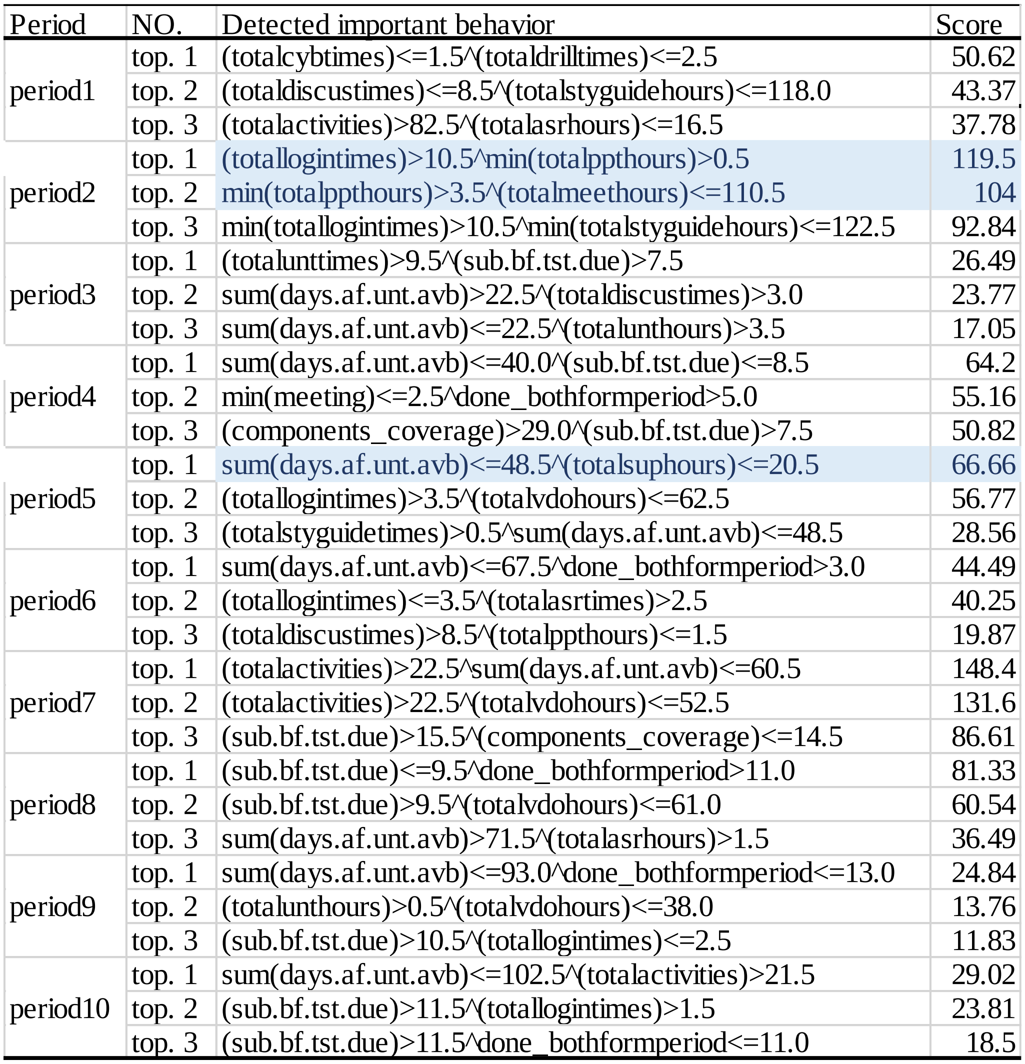
Table 18 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of exceeding the average in the final test

Period	NO.	Detected important behavior	Score
period1	top. 1	sum(days.af.unt.avb)>4.5^(totalfaqtimes)<=3.5	291.1
	top. 2	sum(days.af.unt.avb)<=4.5^(totalvdotimes)>3.5	230.9
	top. 3	totaldiscustimes)>0.5^(totaldrilltimes)<=0.5	106.6
period2	top. 1	(sub.bf.tst.due)<=9.5^(totaldiscustimes)>1.0	217.3
	top. 2	sum(days.af.unt.avb)>17.0^(sub.bf.tst.due)<=9.5	154.7
	top. 3	sum(days.af.unt.avb)<=17.0^(totaldiscustimes)>1.0	116.1
period3	top. 1	sum(days.af.unt.avb)>24.5^(sub.bf.tst.due)<=12.5	41.26
	top. 2	sum(days.af.unt.avb)<=24.5^(totalasrhours)>4.5	30.34
	top. 3	(totalasrtimes)>4.5^(totalloginhours)>1306.0	17.61
period4	top. 1	(totaldiscustimes)>5.5^(components_coverage)<=37.5	36.25
	top. 2	(totalasrtimes)<=4.5^[unt,sgm,unt]=false	23.5
	top. 3	(totallogintimes)>3.5^[vdo,unt]=false	19.16
period5	top. 1	sum(days.af.unt.avb)<=48.5^(studydays)<=17.0	59.56
	top. 2	sum(days.af.unt.avb)>48.5^(totalactivities)>63.5	44.6
	top. 3	sum(days.af.unt.avb)<=48.5^(components_coverage)<=25.0	19.16
period6	top. 1	(sub.bf.tst.due)>9.5^(totalsuptimes)<=5.0	101.5
	top. 2	(sub.bf.tst.due)>9.5^(totallogintimes)>4.5	70.49
	top. 3	(totalstyguidetimes)>0.5^[unt,sgm]=false	48.83
period7	top. 1	(totalasrtimes)>2.5^(totalfaqtimes)<=7.0	17.98
	top. 2	(totalasrtimes)<=2.5^sum(reviewtimes)>18.0	10.86
	top. 3	(totalactivities)>22.0^(totalasrtimes)>2.5	10.03
period8	top. 1	(totalasrtimes)>2.5^(totalloginhours)>370.0	7.69
	top. 2	(sub.bf.tst.due)>17.5^(totalvdohours)<=49.5	7.57
	top. 3	sum(vdo.af.unt.avb)>15.5^(totalasrtimes)>2.5	6.31
period9	top. 1	(totalsuptimes)<=0.5^(sub.bf.tst.due)>21.5	106.3
	top. 2	(totalactivities)>15.5^(totalloginhours)>160.5	105.6
	top. 3	(totalsuptimes)>0.5^(totaldischours)>7.0	84.69
period10	top. 1	(sub.bf.tst.due)<=36.5^sum(reviewtimes)<=39.5	17.59
	top. 2	sum(reviewtimes)<=68.0^(components_coverage)<=14.5	13
	top. 3	(sub.bf.tst.due)<=36.5^(totaldiscustimes)>5.5	9.5

Table 19 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of success and failure in the first attempt in the final test

Period	NO.	Detected important behavior	Score
period1	top. 1	(totalppthours)>0.5^(totalsuptimes)<=2.5	42.49
	top. 2	(totalppthours)<=0.5^(totalloginhours)<=1941.0	29.38
	top. 3	(totalstyguidetimes)>2.5^(totalsuptimes)<=3.5	20.14
period2	top. 1	(sub.bf.tst.due)>0.5^min(totaldisctimes)>0.5	67.23
	top. 2	min(totalppthours)>0.5^(totaldrillhours)<=5.5	40.34
	top. 3	min(totalstyguidetimes)>0.5^min(totaldischours)<=281.5	23.3
period3	top. 1	(totalactivities)>52.5^(totalvdohours)<=174.0	29.44
	top. 2	(sub.bf.tst.due)<=11.5^(totalvdohours)<=174.0	10.69
	top. 3	(regularday)>2.5^(sub.bf.tst.due)>6.5	6.97
period4	top. 1	(sub.bf.tst.due)>7.5^(totaldrilltimes)<=2.5	55.56
	top. 2	(regularday)>2.5^(sub.bf.tst.due)>7.5	37.83
	top. 3	(sub.bf.tst.due)<=7.5^(totaldiscustimes)>1.5	36.41
period5	top. 1	(totalactivities)>26.0^(totalvdohours)<=104.5	99.23
	top. 2	sum(days.af.unt.avb)<=48.5^(totaldiscustimes)>1.5	71.92
	top. 3	avg(days.af.unt.avb)>4.5^(totalvdohours)<=104.5	46.44
period6	top. 1	(regularday)>2.5^(sub.bf.tst.due)>7.5	37.3
	top. 2	avg(days.af.unt.avb)>5.5^(totalvdohours)<=86.5	28.3
	top. 3	(sub.bf.tst.due)<=7.5^(totaldiscustimes)>1.5	26.05
period7	top. 1	sum(days.af.unt.avb)>67.5^(components_coverage)<=13.5	25.95
	top. 2	(sub.bf.tst.due)>6.5^(totalasrtimes)<=3.5	23.07
	top. 3	(components_coverage)>12.0^(sub.bf.tst.due)>12.0	22.37
period8	top. 1	(regularday)>2.5^(totaldrillhours)<=1.5	69.16
	top. 2	sum(days.af.unt.avb)>74.5^[asr,unt,unt]=false	62.75
	top. 3	avg(days.af.unt.avb)<=9.5^(regularday)>2.5	47.23
period9	top. 1	(sub.bf.tst.due)>7.5^(totalvdohours)<=70.0	61.21
	top. 2	(sub.bf.tst.due)<=7.5^(components_coverage)>9.5	38
	top. 3	sum(days.af.unt.avb)<=83.5^(totalvdohours)<=58.0	29.16
period10	top. 1	(totaldiscustimes)>1.5^(sub.bf.tst.due)<=17.0	23.97
	top. 2	(totaldiscustimes)>1.5^(totalunthours)<=12.5	17.89
	top. 3	(sub.bf.tst.due)<=17.5^(totalvdohours)>17.0	17.83

Table 20 — Detected top 3 of the most important behavior combinations on 10 different periods in BEHP5000 by random forest with k-fold cross-validation in prediction of exceeding the average in overall score



Period	NO.	Detected important behavior	Score
period1	top. 1	(totalcybtimes)<=1.5^(totaldrilltimes)<=2.5	50.62
	top. 2	(totaldiscustimes)<=8.5^(totalstyguidehours)<=118.0	43.37
	top. 3	(totalactivities)>82.5^(totalasrhours)<=16.5	37.78
period2	top. 1	(totallogintimes)>10.5^min(totalppthours)>0.5	119.5
	top. 2	min(totalppthours)>3.5^(totalmeethours)<=110.5	104
	top. 3	min(totallogintimes)>10.5^min(totalstyguidehours)<=122.5	92.84
period3	top. 1	(totalunttimes)>9.5^(sub.bf.tst.due)>7.5	26.49
	top. 2	sum(days.af.unt.avb)>22.5^(totaldiscustimes)>3.0	23.77
	top. 3	sum(days.af.unt.avb)<=22.5^(totalunthours)>3.5	17.05
period4	top. 1	sum(days.af.unt.avb)<=40.0^(sub.bf.tst.due)<=8.5	64.2
	top. 2	min(meeting)<=2.5^done_bothformperiod>5.0	55.16
	top. 3	(components_coverage)>29.0^(sub.bf.tst.due)>7.5	50.82
period5	top. 1	sum(days.af.unt.avb)<=48.5^(totalsuphours)<=20.5	66.66
	top. 2	(totallogintimes)>3.5^(totalvdohours)<=62.5	56.77
	top. 3	(totalstyguidetimes)>0.5^sum(days.af.unt.avb)<=48.5	28.56
period6	top. 1	sum(days.af.unt.avb)<=67.5^done_bothformperiod>3.0	44.49
	top. 2	(totallogintimes)<=3.5^(totalasrtimes)>2.5	40.25
	top. 3	(totaldiscustimes)>8.5^(totalppthours)<=1.5	19.87
period7	top. 1	(totalactivities)>22.5^sum(days.af.unt.avb)<=60.5	148.4
	top. 2	(totalactivities)>22.5^(totalvdohours)<=52.5	131.6
	top. 3	(sub.bf.tst.due)>15.5^(components_coverage)<=14.5	86.61
period8	top. 1	(sub.bf.tst.due)<=9.5^done_bothformperiod>11.0	81.33
	top. 2	(sub.bf.tst.due)>9.5^(totalvdohours)<=61.0	60.54
	top. 3	sum(days.af.unt.avb)>71.5^(totalasrhours)>1.5	36.49
period9	top. 1	sum(days.af.unt.avb)<=93.0^done_bothformperiod<=13.0	24.84
	top. 2	(totalunthours)>0.5^(totalvdohours)<=38.0	13.76
	top. 3	(sub.bf.tst.due)>10.5^(totallogintimes)<=2.5	11.83
period10	top. 1	sum(days.af.unt.avb)<=102.5^(totalactivities)>21.5	29.02
	top. 2	(sub.bf.tst.due)>11.5^(totallogintimes)>1.5	23.81
	top. 3	(sub.bf.tst.due)>11.5^done_bothformperiod<=11.0	18.5

4.8 Comparing Important Behavior Identified from Decision Trees and Random Forests

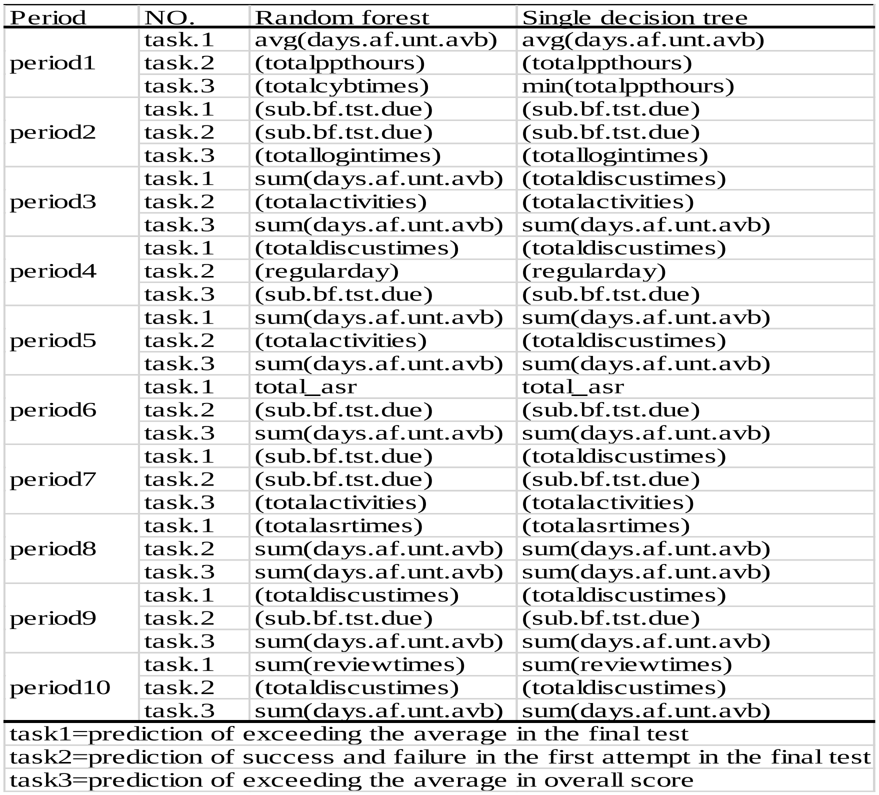
So far we have discussed how to identify important behavior from the random forests. However, we can also identify important behavior from a single decision tree that is learned from all the data. In this section, we compare the identified behavior from two different methods. Firstly, the single decision tree's root node is an attribute with the highest information gain among all attributes. It means that this attribute can classify the positive class examples and the negative class examples most correctly among all attributes. Hence, this root node attribute of the single decision tree can be recognized as the most important attribute (behavior in this paper). Secondly, we detect the single important behavior from random forest by counting the most frequent appeared root nodes.

The one fact that we observed from detected single important behavior between decision trees and random forests is that the single important behavior is basically the same between them (see Table 21), regardless of how periods are measured. The main reason is based on following observations.

In the random forests, to select each tree's root node, first we randomly choose an attributes subset from all attributes, and secondly we select the attribute with the highest information gain as the root node. Let say we assume that the subset size is 50%, we expect each attribute appears in half of the subsets. For example, the forest has ten trees, and A, B and C are the attributes. A (or B or C) would appear in 5 subsets to be consider as the root node. We assume attribute A has a higher information gain than B, and B has a higher information gain than C. If we only use the single decision tree, not random forest, A would be selected as the root node. In the random forest, A would be selected as the root node in five trees. B can be selected in the other five trees, if A and B do not appear in the same subsets. However, A and B would appear in 25% of the same subsets statistically. If A and B appear in the same subset, A will be selected. Therefore, B would be selected in 25% of subsets as the root node in the forest. Since attribute A appears more frequently than B, A would be identified as the most important behavior in the random forest. Therefore, A would be identified as the most important behavior in both the single decision tree and the random forest.

In a word, the single decision tree can detect the same most important behavior much faster than random forests. This is an advantage of using the single decision tree. However, one of our goal is to correctly identify the poorly performed students in the early period. The random forests’ accuracy of predicting the poorly performed students is much better than the single decision tree. That is why we prefer to use the random forest, even if it has a relatively long execution time.

Table 21 — Detected the most important single behavior on 10 different periods in BEHP5000 by random forests with k-fold cross-validation as well as single decision tree in three tasks (cumulative periods counting)



Period	NO.	Random forest	Single decision tree
period1	task.1	avg(days.af.unt.avb)	avg(days.af.unt.avb)
	task.2	(totalppthours)	(totalppthours)
	task.3	(totalcybtimes)	min(totalppthours)
period2	task.1	(sub.bf.tst.due)	(sub.bf.tst.due)
	task.2	(sub.bf.tst.due)	(sub.bf.tst.due)
	task.3	(totallogintimes)	(totallogintimes)
period3	task.1	sum(days.af.unt.avb)	(totaldiscustimes)
	task.2	(totalactivities)	(totalactivities)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)
period4	task.1	(totaldiscustimes)	(totaldiscustimes)
	task.2	(regularday)	(regularday)
	task.3	(sub.bf.tst.due)	(sub.bf.tst.due)
period5	task.1	sum(days.af.unt.avb)	sum(days.af.unt.avb)
	task.2	(totalactivities)	(totaldiscustimes)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)
period6	task.1	total_asr	total_asr
	task.2	(sub.bf.tst.due)	(sub.bf.tst.due)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)
period7	task.1	(sub.bf.tst.due)	(totaldiscustimes)
	task.2	(sub.bf.tst.due)	(sub.bf.tst.due)
	task.3	(totalactivities)	(totalactivities)
period8	task.1	(totalasrtimes)	(totalasrtimes)
	task.2	sum(days.af.unt.avb)	sum(days.af.unt.avb)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)
period9	task.1	(totaldiscustimes)	(totaldiscustimes)
	task.2	(sub.bf.tst.due)	(sub.bf.tst.due)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)
period10	task.1	sum(reviewtimes)	sum(reviewtimes)
	task.2	(totaldiscustimes)	(totaldiscustimes)
	task.3	sum(days.af.unt.avb)	sum(days.af.unt.avb)

task1=prediction of exceeding the average in the final test

task2=prediction of success and failure in the first attempt in the final test

task3=prediction of exceeding the average in overall score

4.9 Comparing Important Behavior Combinations Identified from Decision Trees and Random Forests

We are also able to identify important behavior combinations from a single decision tree that is constructed from the first level root node attributes and the second level attributes. In this section, we compare the behavior combinations from the two different methods.

Firstly, the behavior combinations in a single decision tree is very simple: one root node combined with next level attributes. The number of the important combinations are based on the number of the second level attributes. Secondly, the behavior combinations in random forests are also constructed from the root node attribute and the second level attributes. However, the number of combinations is based on the number of the second level attributes as well as the forest size. The random forest can extract more behavior combinations than a single decision tree. The main difference in selecting the important behavior combinations is that we measure the quality of the combinations. To calculate the quality score of the combinations in the random forest, first we calculate the percentage of the well performed students in a combination. Then, all the scores of the same behavior combinations will be summed up together to get a single score.

The one fact that we observed from identified important behavior combinations between decision trees and random forests is that the important behavior combinations are different, regardless of how periods are measured.

Compare to the important behavior combinations in the single decision tree, each behavior combination can have its own quality score in the random forest. The quality score can tell us which behavior combination can highly lead students to be success in course, rather than only information gain in the single decision tree. In the decision tree, there is only one root node and its child attributes can construct the behavior combinations. However, in the random forest, it make us to have more options to extract more behavior combinations with different root nodes. That is why we use random forests instead of the decision tree in this paper.

Chapter 5
Conclusion

This paper attempts to capture the learning profile of students in BEHP5000 as much as possible. We are able to identify the important single behavior and behavior combinations in early time of the semester by using of the machine learning algorithms. Especially, the random forest combined with k-fold cross-validation used in this paper works well on our particular tasks. Further, our approach is also able to identify the important behavior for each period, which allows us to observe the important behavioral changes along with time transition. Another contribution of this paper is that we can predict poorly performed students correctly early in the course. The accuracy of prediction poorly performed students could exceed 87% on average after period 1. Therefore, we are able to know who exactly needs the help as soon as possible.

The limitation of our work is that, firstly, the datasets (log files) are not really designed for educational data mining. If logs could include more detailed students' activity information inside itself, the performance of this system still has a room for improvement. Secondly, random forest combined with k-fold cross-validation has a high time complexity and a high accuracy on this task.

In the future, we would like to measure reliability of detected important behavior by receiving the feedback from both students and instructors. Further, we still work on changing the algorithm structure to obtain both a short running time and a high accuracy at same time.

Reference and Citation

[1] Barahate Sachin R. and Shelake Vijay M., "A Survey and Future Vision of Data mining in Educational Field", 2012 Second International Conference on Advanced Computing & Communication Technologies

[2] John Champaign, "Correlating Algorithm and Improvement in 2 MOOCs with a Student’s Time on Tasks", L@S 2014 Student Algorithms and Behavior, March 4–5, 2014, Atlanta, Georgia, USA

[3] Daniel T. Seaton, Yoav Bergner, Isaac Chuang, Piotr Mitros, and David E. Pritchard, "Who Does What in a Massive Open Online Course?”, Accepted by Communications of the ACM, Vol. 57 No. 4, Pages 58-65.

[4] Sucheta V. Kolekar, Dr. S. G. Sanjeevi and Dr. D. S. Bormane, "Learning Style Recognition using Artificial Neural Network for Adaptive User Interface in E-learning", Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference, page 1-5, 28-29 Dec. 2010.

[5] Gökhan Akcapinar, Arif Altun and Erdal Cogun"Investigating Students’ Interaction Profile in an Online Learning Environment with Clustering", Advanced Learning Technologies (ICALT), 2014 IEEE 14th International Conference, page 109-111, 7-10 July 2014.

[6] Annika Wolff, Zdenek Zdrahal, Andriy Nikolov and Michael Pantucek "Improving retention: predicting at-risk students by analyzing clicking behavior in a virtual learning environment".

[7] J. R. Quinlan, “Introduction of decision tree”, Journal of Machine learning”, pp. 81-106, 1986.

[8] Il-Hyun Jo, Dongho Kim and Meehyun Yoon, "Analyzing the Log Patterns of Adult Learners in LMS Using Learning Analytics", LAK '14 Proceedings of the Fourth International Conference on Learning Analytics And Knowledge, Pages 183-187.

[9] Shaveen Singh and Sunil Pranit Lal "Using Feature Selection and Association Rule Mining to Evaluate Digital Courseware" 2013 Eleventh International Conference on ICT and Knowledge Engineering.

[10] Carleton Coffrin, Linda Corrin, Paula de Barba and Gregor Kennedy "Visualizing patterns of student engagement and performance in MOOCs", LAK '14 Proceedings of the Fourth International Conference on Learning Analytics And Knowledge, Pages 83-92

[11] JOHN S. KINNEBREW, KIRK M. LORETZ, and GAUTAM BISWAS "A Contextualized, Differential Sequence Mining Method to Derive Students' Learning Behavior Patterns", Vanderbilt University.

[12] John S. Kinnebrew, Gautam Biswas, "Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution", G. (2012). Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution. In the 5th International Conference on Educational Data Mining (pp. 57–64).

[13] Joshua Ho, Lior Lukov and Sanjay Chawla "Sequential Pattern Mining with Constraints on Large Protein Databases", Sydney NSW 2006 Australia, School of Information Technologies, University of Sydney.

[14] Aleksandra Klasnja-Milicevic, Boban Vesin, Mirjana Ivanovic, Zoran Budimac "E-Learning personalization based on hybrid recommendation strategy and learning style identification",Computers & Education, Volume 56, Issue 3, April 2011, Pages 885–899.

[15] Silvia Schiaffino, Patricio Garcia, Analia Amandi "eTeacher: Providing personalized assistance to e-learning students",Computers & Education, Volume 51, Issue 4, December 2008, Pages 1744–1754.

[16] Leo Breiman "RANDOM FORESTS", machine learning, 45, 5-32, January 2001,2001 kluwer Academic Publishers, Manufactured in The Netherlands ,V University of California.

[17] Tom Mitchell, McGraw-Hill, “Machine Learning”, 1997.

[18] Kamal Bunkar "Data Mining: Prediction for Performance Improvement of Graduate Students using Classification", machine learning, 2012