How Can Learning Analytics Improve a Course?

Bowen Hui
Department of Computer Science
University of British Columbia
bowen.hui@ubc.ca

Shannon Farvolden
Department of Computer Science
University of British Columbia
shannonfarvolden@gmail.com

ABSTRACT
Despite much excitement with learning analytics, there is still a lack of adoption in the classrooms. Possible reasons may include not having enough time to incorporate the use of analytics, not being familiar enough with specific techniques to readily apply them, or not knowing how data can help shape a curriculum or the classroom experience altogether. Learning analytics is a problem-driven research field, where the domain problem – the people involved, the subject matter, and the learning environment – drives the techniques and the solutions that are used. From this perspective, we propose a new framework with a suite of pedagogical questions that can be addressed using data to support decisions made about the curriculum or classroom structure. In addition, we present a case study with 69 participants in a CS1 course as a way to demonstrate how some of these questions are addressed. Our ultimate goal is to improve the quality of the students' learning experience using an evidence-based approach.

CCS Concepts
• Social and professional topics →Computing education →Computer science education CS1 • Information systems →Decision support systems →Data analytics.

Keywords
learning analytics; CS1; needs analysis; evidence-based course design

1. INTRODUCTION
Research and development in learning analytics has surged in the recent years. Over the last decade, a noticeable increase in learning analytics activities within computer science is also observed [6]. Despite this movement, there is still a lack of adoption in practice. Studies report the top 5 barriers to adoption in higher education are cost, proper use of data, regulations requiring the use of data, not knowing how to make decisions with data, and having inaccurate data [2]. Among these, three of them pertain to a lack of expertise with data. While novice data users might not know how analytics can help improve the learning environment, data-literate educators may choose not to adopt analytics available in existing learning management systems (LMS) because many of these systems do not clarify which algorithms or statistical models are used [9]. Moreover, in the field of analytics where it is littered with case studies, it can be quite overwhelming and difficult for novice data users to navigate the literature and synthesize the results.

WCCCE'17, May 05–06, 2017, Abbortsford, BC, Canada.

ACM ISBN 978-1-4503-5066-2/17/05…$15.00
http://dx.doi.org/10.1145/3085585.3085586

Our goal is to promote the use of learning analytics by instructors across various disciplines. As a start, we propose a framework in Section 2 to help individuals decide if and how they may want to use analytics to improve the learning environment. Although learning analytics may be broadly construed as an evidence-based approach to tackle problems with retention, course planning, and program effectiveness, we restrict our attention to the use of data in the context of a course in this paper. Since learning analytics is a problem-driven field, our framework is designed from the perspectives of the key stakeholders in the domain – the instructor and the student. Based on their needs, various categories surfaced for the two stakeholders. To delineate these categories, we provide an initial set of pedagogical questions that can be addressed using simple to complex learning analytics. For example, an instructor may wish to know what kind of mistakes students make on specific concepts, while a student may wish to know the strategies successful students use to prepare for exams. These questions serve as a starting point to help others ask and answer deeper questions about student learning. Where available, we include in Section 2 the relevant literature that addresses these pedagogical questions. Due to space limitations, we provide a representative sample of literature rather than an exhaustive review.

To demonstrate the utility of our framework, we present a case study completed in the spring of 2016 with 69 participants in a CS1 course. Section 3 describes an in-house LMS that was built and used as part of this project and Section 4 explains the study setup. In contrast to more intricate studies that employ detailed metrics (see [6] for a survey), the data we collected include site usage information in the LMS, self-reported responses from three short surveys, and scores that students obtained in various class activities. Using this data, we present analyses to address sample pedagogical questions in our framework, specifically those in the categories of knowledge assessment, student engagement, course design, and self-improvement for students.

Section 5 reports details of our analyses and results. By analyzing activity patterns, we were able to identify concepts that students spent more time on and their levels of engagement throughout the semester. The self-reported surveys and online activities such as participation levels in the discussion forum and assignment submission times provided us with behavioral data that we correlated with student performance. Of particular interest is an analysis where we investigated the predictive value that different course components have on student performance in order to gain a better understanding of how the design of a course can be improved. Overall, where literature exists, our results are consistent with existing findings.

2. WAYS TO SHAPE A COURSE
The purpose of this discussion is to increase awareness of how data can be used to improve a course. Based on the results of a focus group with students and instructors, as well as a synthesis of the literature, we identified a list of pedagogical questions and

developed categories for them in a framework. As a result, we propose this framework as a way analyze the impact of the course delivery and the student's experience of a course. While the pedagogical questions in each category are motivated by computer science courses, they are relevant to courses in other domains. It is not our intention to offer an exhaustive list of questions; these questions serve as a starting point for individuals to consider whether learning analytics is the right tool to solve their problem.

Our framework begins by considering the users in the problem domain. In the context of an introductory computer science course, we note that there are two distinct user groups: the instructor and the student. We present our pedagogical questions based on the different needs that these two user groups have.

2.1 Instructor Needs

We identified several categories of questions that an instructor might be interested in assessing. Generally, an instructor might wish to know how the students are doing in the course at any point in time and how the course is going overall. For this, we identified four categories (see Sections 2.1.1 to 2.1.4). As well, we consider the case where an instructor might change the course. We identified two categories here (see Sections 2.1.5 and 2.1.6). The rest of this section explains these categories further.

2.1.1 Knowledge Assessment

The questions in this category help an instructor understand how well the students are doing as a whole. The specific questions are:

(a) What are the current grades for each course component?
(b) Which concepts are generally more difficult?
(c) Do I need to spend more time on this material?
(d) How well do students understand the material that was just presented? Or in a (pre-)reading?
(e) Are students following instructions correctly?
(f) Which students are doing well? Which are at risk of failing?
(g) What is the learning trajectory of the student?

Summary course grades are commonly integrated into instructor dashboards [13]. Code metrics can be used to identify concept difficulty [3], as well as the use of online resources (see an example in our case study in Section 5). Various response systems, such as iClickers, have been used to assess and visualize change in student understanding [8]. Systems that provide personalized feedback to students (e.g., [11]) can also be used to give instructors feedback on whether instructions are followed correctly. Predicting students at-risk have been a focus of many studies (e.g., [1,4]). Probabilistic student models have been used to identify students who may struggle with future material [10].

2.1.2 Types of Errors

An instructor might also wish to understand the kinds of mistakes that students tend to make so that precautions can be taken in the future or a detailed review of difficult questions can be provided. The questions in this category include:

(a) What kind of mistakes do students usually make on a particular concept?
(b) Which are the most common compilation errors students make for specific concepts?
(c) Which questions incur the most mistakes?
(d) Which coding behaviors do successful students exhibit?

Descriptive statistics from assignments and exams can provide a general overview of problematic areas. Using code metrics, predicting programming performance can also be achieved [14].

2.1.3 Engagement

The questions in this category help assess how an instructor is doing in the course. The assumption is that if the students are engaged, then the delivery of the course material as perceived by the students is positive. These questions include:

(a) Are students engaged in the course?
(b) How much participation is there in specific activities?
(c) Do students attempt optional assignments?
(d) How engaged are the students in the optional topics? Which optional topics do students find more interesting?

Using logged data that shows utilization of online materials, student's participation level and interest in specific topics can be tracked. Our case study results in Section 5 provides an example.

2.1.4 Expectations

Instructors who have taught a course before typically has expectations about how it would go in the next offering of the course. Example questions include:

(a) With regards to the tentative schedule, are we behind or ahead of schedule?
(b) What are the average scores on a particular exercise, relative to previous year's scores?

Schedule expectations can be easily determined by matching the current topic against a predetermined tentative schedule. Scores on various course components can be summarized using descriptive statistics that are commonly available in instructor dashboards [13].

2.1.5 Experimentation

Experimentation questions pertain to more purposeful initiatives where an instructor tweaks a particular aspect of the course and measures the impact of that change. Example questions include:

By changing a specific aspect of the course, what is the impact on:

(a) … individual performance within the course?
(b) … retention in comparison to a previous course offering?
(c) … student engagement in the class?
(d) … interest in the field of computer science?

Many controlled experiments have been reported. In particular, the use of Course Signals show an increase in retention and students were reported to be more proactive knowing how they were doing in the course [1].

2.1.6 Course Design

This category consists of questions that one might have about the effectiveness of different components of the course, typically during a post-assessment period.

(a) Is the lab/assignment too long or difficult?
(b) Are there any questions on the exam that is too easy or too hard?
(c) Does this assessment help students be better prepared for the final exam?

Reporting average completion times and success rates provide an indication on length and difficulty. Code metrics have also been used to summarize common errors [7,5] and to provide deeper insight into programming ability [14,12].

2.2 Student Needs

We identified three categories of questions that a student might have while taking a course (see Sections 2.2.1 to 2.2.3). We elaborate on these categories below.

2.2.1 Planning

This category pertains to how a student might plan out their activities and set goals for the course. Better planning may motivate students to start their assignments earlier – a factor that is correlated with high performance [12]. These questions include:

(a) What and when is my next deadline?

(b) What are my upcoming deadlines? Alternatively, what are all the deadlines for the course?

(c) What do I need to get on my final exam in order to get a certain grade for the course?

A simple calendar of events showing deadlines clearly and an up-to-date calculation of the target scores to obtain can be easily incorporated into a student dashboard.

2.2.2 Monitoring

Students always want to know where they stand on a particular deliverable or in the course overall. These questions include:

(a) How am I doing in the course?

(b) How am I doing in comparison to others in the class?

(c) Am I spending too long or not enough time on this question or assignment?

(d) Am I following the right steps in completing this exercise?

Descriptive statistics can be used to give students an overview of how they are doing in a course or in specific exercises. Predictive analytics can also be used to warn students of potential failures [1]. In order to know whether students are taking the right steps to solve a problem or not, domain-specific approaches to provide informative feedback are needed (see [11] for example).

2.2.3 Improvement

Some students may wish to know how they can avoid making the same mistakes and what other problem solving strategies are effective. Specifically, they may ask the following questions:

(a) What can I do to improve my grade?

(b) What kind of mistakes do I usually make?

(c) Where do I usually lose marks on exams?

(d) What kinds of programming strategies work best for other successful students?

(e) Which study habits are more effective for exams?

(f) What kind of study habits do successful students have?

A personalized version of common mistakes can provide valuable feedback to students (e.g., [7,5]). Study habits that are correlated with good performance can also be used (e.g., see [12] and results discussed in our case study in Section 5).

3. DEVELOPING COURSE CANVAS

We developed a light-weight LMS called Course Canvas that hosts all the course content, provides gradebook and reporting functionality, and displays basic analytics. This system employs a minimalist design in order to maximize system performance and end-user usability. It is developed using modern web technologies (specifically, PHP, HTML, Sass, Boostrap, Laravel, and jQuery) so that it can be responsive across different devices and easily maintained and extended for future development.

Course Canvas was setup solely for a single CS1 course. Once the user enters the site on a web browser, the course homepage appears showing the course syllabus, evaluation criteria, and a week-by-week tentative schedule. Additional functionality is displayed as menu options via the top navigation bar on the site.

The features provided as part of Course Canvas include:

- **User Authentication:** All content requires a login to enable event tracking.
- **Slide Carousel:** Lecture slides are organized by topic and by week. Each topic has a set of thumbnail slides. A carousel is available for users to navigate the slides.
- **Discussion Forum:** An online forum was implemented to enable asynchronous discussions on course related topics. This forum supports anonymous thread posts and content filtering via a set of predefined labels.
- **Quizzes:** Weekly multiple-choice quizzes were available. Each quiz has a start and end time with an unlimited number of attempts. In addition to randomizing the order of the questions and answers, a 24-hour gap is also placed between each re-take so to avoid trial-and-error attempts.
- **Online Submission and Gradebook:** Students can submit their work and view their grades and feedback online.
- **Student Dashboard:** A simple dashboard shows graphs of a student's own grades relative to the class average for each submission. This feature enables students to view their own performance and compare it to their peers as a group. For each course component, the student also sees their marks to date and a danger, warning, or success flag depending on their performance.
- **Administrator Dashboard:** This interface enables an administrator to monitor activities on Course Canvas and manage user accounts. Only very simple summary analytics were available at the time of the study.

One notable feature in our system is that we keep track of the time spent on each set of slides topic, so that we can subsequently estimate the difficulty of a topic based on slides usage.

4. CASE STUDY

During the winter of 2016, we investigated the use of Course Canvas in a CS1 course that teaches introductory Java programming. In this section, we describe the details of the study. Next, we present the exploratory findings from the project as they relate to the pedagogical questions in our framework.

4.1 Participants

Sixty-nine students participated in this study. Among them, 51 were male and 18 were female. These students came from a variety of backgrounds: 40 were pursuing a Science degree, 13 were pursuing an Arts degree, 8 were pursuing a Management degree, 4 from Engineering, 1 from Human Kinetics, 1 from Visual Arts, and 2 were undecided. In addition, 47 of these participants were in their first year of university study, 12 in their second year, 5 in their third year, and 5 in fourth year or more.

No compensation was given. Note that participation in the study is not mandatory; opting out of the study will result in flagging the student's data to be excluded from research reports and does not preclude the student from using Course Canvas in any way.

4.2 Materials

The following teaching materials were placed on Course Canvas: a course syllabus, a regularly updated week-by-week schedule, 13 weeks of lecture slides, 9 multiple-choice quizzes, 9 programming labs, and 3 programming assignments. The actual week-by-week content, such as slides, quizzes, labs, and assignments were released gradually over the course of the semester.

In addition, the LMS hosted a survey with 11 multiple-choice questions that address the students' academic intent and strengths. These questions include the student's programming experience, score in a prerequisite mathematics course, goals in the current course, ability to do well on exams, and intent on doing all the assigned homework.

The LMS also hosted a survey on study habits with 9 multiple-choice questions that explore the methods used to study exams. These questions include the total time spent studying, the type of materials used to prepare for the exam, the amount of reading and programming practice students got, the amount of sleep they had the night before the exam.

4.3 Procedure

At the beginning of the semester, students were shown how to create an account in Course Canvas and to indicate whether they would voluntarily opt into the study or not, and where this preference can be changed in their account settings. A two-minute overview of the site was given to show the students where the various course content was located. Thereafter, the students could use Course Canvas as much or as little as they desired.

For the academic background survey, participants were asked to complete it within the first two weeks of the semester. As an incentive, one bonus lab mark was given for completing this survey. For the survey on study habits, students were asked to complete it within two weeks of each of the two midterms in the course. Two bonus lab marks were given for each survey. In total, a student could get 5 bonus lab marks (worth about 0.3% of the course) by completing all three surveys.

4.4 Measures

In combination with Google Analytics, we gathered anonymous data on the following: survey responses (see Section 4.1), scores on various class activities, and LMS usage (by pageviews, number of sessions, and time spent in seconds).

5. RESULTS

Data was collected throughout an entire semester lasting 16 weeks in total. During this period, there was approximately 13 weeks of lectures, with 1 week of midterm break and 2 weeks between the last day of classes and the final exam date.

5.1 Instructor's Need: Knowledge Assessment

As a first step to estimating concept difficulty, we assumed a positive correlation between the time a student spend reading lecture slides and the difficulty of the topic. Note that this assumption excludes data involving the time students spend studying using offline resources.

Figure 1 presents a summary of the website usage for an individual student on average[1]. It is interesting to note that among all the slides, students spent the most time on summary slides. Likewise, slides that acted like references (e.g., cheatsheet of methods) were used heavily. We believe these slides were accessed more often for studying and problem solving purposes.

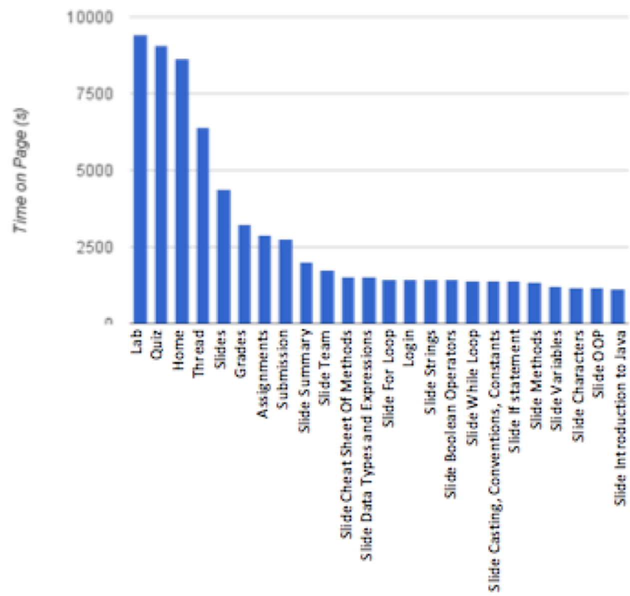

Figure 1: Partial results showing average time spent per page per student.

In order of difficulty, the slides usage pattern also suggests that loops, conditionals, and methods were the most challenging concepts. This is consistent with existing findings where conditionals and loops were found to be particularly problematic for CS1 students [3].

5.2 Instructor's Need: Engagement

To get a general sense of engagement, we examined the number of web sessions over the span of a week and throughout the entire semester. Figure 2 plots the average number of sessions over the week for an individual student. This pattern indicates that students use the site most during lectures (held on Mondays) and labs (two sections on Wednesdays and one section on Fridays), with less frequent usage on the remaining days.

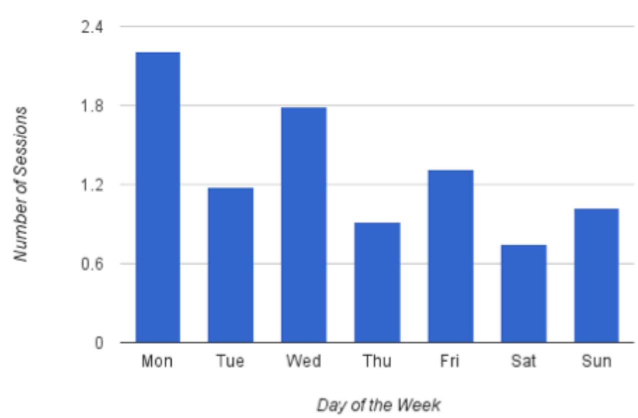

Figure 2: Average usage per individual student.

Next, we investigated on the engagement level throughout the semester as shown in Figure 3. Not surprisingly, we see the site hits peak usage on due dates and exam dates (specifically, weeks

[1] Unfortunately, the data collected by Google Analytics was incomplete. Specifically, the amount of time spent on a page is only logged if a user continues onto another page within the site before the session ends. This means, if a user opens up Course Canvas, reads the content of the page, and leaves, the time spent on that page is not collected. Thus, the time tracked may be less than the actual total amount of time spent on the LMS.

3, 7, 9, 11, 14, and 16). As well, the site gets low usage during study breaks and in between due dates.

The graph in Figure 3 also shows a decline in engagement after the second midterm. We suspect this decline is due to a combination of a common end-of-semester crunch time and many students being discouraged from getting poor results on their second midterm (class average was 55%). In order to encourage students who did not do well on the second midterm, this course implemented a rule so that a final exam mark that is higher than both midterm marks would replace both midterm marks. In future course offerings, appropriate intervention and remedial actions are needed to minimize failure rates.

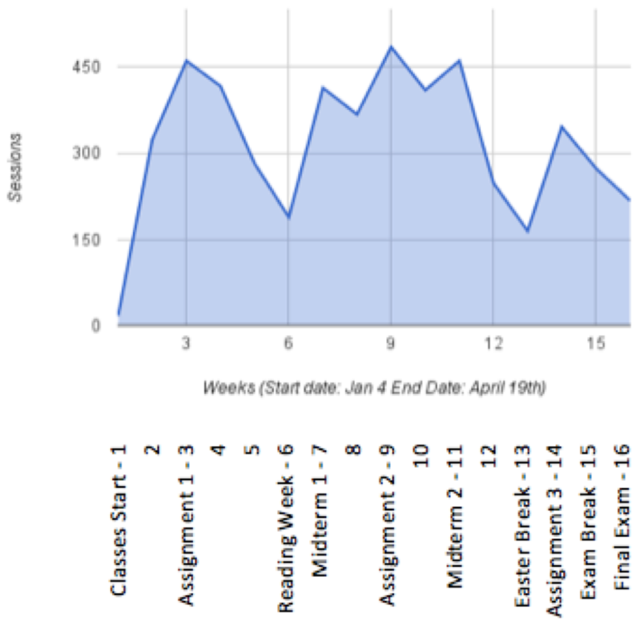

Figure 3: Total usage across the semester.

5.3 Instructor's Need: Course Design

In Section 5.1, we found that summary and reference slides were accessed more frequently than other slides. A design implication drawn from this finding is to position these slides in a more prominent location to facilitate easy access for students, rather than having them be embedded among other weekly slides.

We also found from Section 5.1 that loops, conditionals, and methods were the most difficult concepts. In response to this finding, extra care is needed to redesign the course so that students are provided with more intuitive examples and hands-on practice with these concepts.

Additionally, we also explored how different course components contribute to student success. The various course components are labs, assignments, in-class activities, quizzes, midterms, and final exam. Our goal is to ensure that the various course components help prepare students be successful in the final exam as well as the course overall.

First, we focus on the final exam. A stepwise multiple linear regression was calculated to predict final exam score based on scores from the other course components mentioned above. A significant regression equation was found with the assignments, in-class activities, and the two midterm marks being the predictors ($F(4,64)=73.827$, $p = 2.877E-23$), with an R^2 of 0.811.

As expected, the midterm marks are strong predictors of the final exam mark since they all have similar format. However, there is a split among the remaining course components – assignments and in-class activities are independent variables in the model but labs and quizzes are not. We suspect that while in-class activities and assignments may be done collaboratively, the exercises involved resemble aspects of the final exam – in-class activities have a strict time constraint and assignments require strong problem solving skills because the exercises are worded in an open-ended manner. On the other hand, lab exercises are all broken down into detailed steps for students to follow, which is not available on the final exam. Likewise, because quizzes consist of purely multiple-choice questions that can be attempted repeatedly throughout the semester, they do not predict final exam performance.

Next, we focus on the course overall. A stepwise multiple linear regression was calculated to predict overall course grade based on scores from all the course components mentioned above plus the final exam. A significant regression equation was found with all the variables except assignment marks being the predictors ($F(6,62)=493.755$, $p = 2.339E-50$), with an R^2 of 0.978.

Not surprisingly, all the exams are strong predictors of the overall grade since they make up 60% of the overall mark. It is unclear why assignment marks is not significant in predicting the overall grade while other similar course components (e.g., labs) with similar weighting in the course grade do. Further investigation is needed to better understand this result.

Altogether, we see that the overall course grade is predicted based on a combination of factors that measures an individual's capability as well as the overall effort exhibited during the whole learning process.

5.4 Student Needs: Improvement

In terms of self-improvement, we report results that speak to two different questions. First, we address which study habits are more effective for getting good grades on exams. Since we were not able to obtain survey data after the final exam, our data is drawn from the study habits survey which were completed a week after each of the two midterms. Overall, 54 respondents completed the survey after the first midterm and 33 respondents completed the survey after the second midterm.

Both midterms were designed and administered in a similar manner. Students had 75 minutes to complete a written exam, each worth a total of 40 points with ten multiple-choice questions, three short answer questions (e.g., identify errors in code, write program output, write Java statements, write a method that works with given Java code), and one long answer question (write a complete Java program). The topics covered for the first midterm included: basic Java programs, conditional statements, data types, special classes such as Math, String, and Random. The second midterm was cumulative and included additional topics such as loops, methods, and arrays.

A stepwise multiple linear regression was calculated to predict midterm 1 score based on the variables in the study habits survey. A significant regression equation was found with the amount of sleep being the only predictor ($F(1,52)=4.999$, $p = .03$), with an R^2 of 0.070[2]. Our experience indicates that students tend to do well on this exam if they put enough effort into the course since the

[2] While this low R^2 value indicates that only 7% of the variation was explained, the significance of it shows that a reliable relationship was found.

majority of the topics tested are fairly basic at this point. It is not surprising that if the students get enough sleep and are not cramming the night before that they can do well on it.

On the other hand, a stepwise multiple linear regression was calculated to predict midterm 2 score based on the variables in the study habits survey. A significant regression equation was found with the number of hours the student spent studying for the exam being the only predictor ($F(1,31)=6.506$, $p = .01$), with an R^2 of 0.147. We suspect the change in predictors in this case is due to the fact that harder programming concepts are tested on this exam. Furthermore, if students do not have a good grasp of the concepts from the first midterm, there would be an cumulative negative effect on their performance on future exams.

Surprisingly, other variables in the study habits survey did not surface as predictors of exam performance. Further investigation and more data are needed to explore the impact that the type of materials used to prepare for exams (e.g., textbook vs. slides) and the type of practice students did (e.g., reading vs. doing programming exercises) have on student performance.

Next, we looked at the general online behavior that students have as they progressed through the course and correlated them with performance. For this purpose, we gathered a set of behavioral variables such as number of discussion forum replies, number of submission attempts (for labs and assignments), submission time before due date, total number of online sessions, time spent on discussion threads, when quizzes were completed relative to when they were first released, pageviews on slides, time spent on the site, pageviews on the site, time spent reviewing grades, and the number of discussion posts created.

Using stepwise multiple linear regression to predict overall course grade based on variables in the online behavioral variables above, a significant regression equation was found with the following predictors: number of discussion forum replies, number of submission attempts, submission time before due date, and total number of online sessions ($F(4,64)=25.369$, $p = 1.29E-12$), with an R^2 of 0.589. Consistent with literature findings, this model suggests that more active students who put more effort into the course and do not leave assignments to the last minute are more likely to do well in the course overall [12].

6. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a framework of pedagogical questions to help individuals decide how learning analytics can improve their courses. Our framework is centered on the stakeholders in the domain – the instructor and the student. In this context, a discussion of how data is used to address these pedagogical questions and related literature is presented. We also described a case study of using learning analytics in a CS1 course to illustrate how some of the issues in knowledge assessment, engagement, course design, and self-improvement for students can be addressed.

Our next step is to conduct more focus groups to broaden this applicability of this framework and to evaluate it by conducting an in-depth needs analysis study with instructors and students across different disciplines. Our ultimate goal is to promote the use of data in (re)designing a course and to help instructors understand how learning analytics can shape a course.

7. REFERENCES

[1] Arnold, K.E., and Pistilli, M.D. 2012. Course signals at Purdue: using learning analytics to increase student success. In *Proc. of the 2nd international conference on learning analytics and knowledge* (LAK'12), pp. 267-270.

[2] Bichsel, J. 2012. Analytics in higher education: Benefits, barriers, progress, and recommendations: *EDUCAUSE Center for Applied Research.*

[3] Y. Cherenkova, D. Zingaro, and A. Petersen. Identifying Challenging CS1 Concepts in a Large Problem Dataset. 2014. In *Proc. of the 45th ACM Technical Symposium on Computer Science Education* (SIGCSE'14), 695–700.

[4] Essa, A. and Ayad, H. 2012. Student success system: risk analytics and data visualization using ensembles of predictive models. In *Proc. of the 2nd International conference on learning analytics and knowledge* (LAK'12), 158-161.

[5] J. B. Fenwick, C. Norris, F. E. Barry, J. Rountree, C. J. Spicer, and S. D. Cheek. 2009. Another Look at the Behaviors of Novice Programmers. *SIGCSE Bulletin*, 41(1):296–300, Mar. 2009.

[6] Ihantola, P., Vihavainen, A., Ahadi, A., Butler, M., Borstler, J. Edwards, S., Isohanni, E., Korhonen, A., Petersen, A., Rivers, K., Rubio, M.A., Sheard, J., Skupas, B., Spacco, J., Szabo, C., and Toll, D. 2016. Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies. In *Proc. of the 20th Annual Conference on Innovation and Technology in Computer Science Education on Working Group Reports* (ITiCSE WGR'16), 41-62.

[7] M. C. Jadud. 2006. Methods and Tools for Exploring Novice Compilation Behaviour. In *Proc. of the 2nd International Workshop on Computing Education Research* (ICER'06), 73–84.

[8] Majumdar, R. and Iyer, S. 2015. Beyond Clickers: Tracing Patterns in Students' Response through iSAT. In *Proc. of the 23rd International Conference on Computers in Education* (ICCE).

[9] Nixon, A.L. 2016. Reflecting on Learning Analytics: *EDUCAUSE Review*, (September/October 2016), 62-63.

[10] Piech, C., Sahami, M., Koller, D., Cooper, S., and Blikstein, P. 2012. Modeling how students learn to program. In *Proc. of the 43rd ACM technical symposium on Computer Science Education* (SIGCSE'12), 153-158.

[11] Serral, E., De Weerdt, J., Sedrakyan, G., and Snoeck, M. 2016. Automating Immediate and Personalized Feedback. In *Proc. of the 10th IEEE International Conference on Research Challenges in Information Science* (RCIS).

[12] J. Spacco, D. Fossati, J. Stamper, and K. Rivers. 2013. Towards Improving Programming Habits to Create Better Computer Science Course Outcomes. In *Proc. of the 18th ACM Conference on Innovation and Technology in Computer Science Education* (ITiCSE'13), 243–248.

[13] Verbert, K., Duval, E., Klerkx, J., Govaerts, S., and Santos, J.L. 2013. Learning analytics dashboard applications. *American Behavioral Scientist*, 57(10): 1500-1509.

[14] Watson, C., Li, F. W., and Godwin, J. L. 2013. Predicting performance in an introductory programming course by logging and analyzing student programming behavior. In *Proc. of the 13th IEEE International Conference on Advanced Learning Technologies* (ICALT), 319-323.