Identifying Patterns of Learning: A Case Study of MIT's Introductory Programming Course (6.000x)

by

Helen Abadiotakis

B.S. Electrical Engineering and Computer Science, Massachusetts Institute of Technology (2018)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science and Molecular Biology

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2020

Author . . . .
Department of Electrical Engineering and Computer Science
August 21, 2020

Certified by . . . .
Ana Bell
Lecturer
Thesis Supervisor

Accepted by . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Identifying Patterns of Learning: A Case Study of MIT's Introductory Programming Course (6.000x)

by

Helen Abadiotakis

Submitted to the Department of Electrical Engineering and Computer Science on August 21, 2020, in partial fulfillment of the requirements for the degree of Master of Engineering in Computer Science and Molecular Biology

Abstract

The ever-increasingly relevant introductory programming course offered at MIT presents a unique opportunity to uncover student learning patterns and common behavioral motifs. The course 6.0001/6.0002 harbors a wealth of student interaction data on its companion MITx platform as well as associated grades. Although this course has been offered for the last twelve years, since 2008, little has been done to identify aspects of the course that best aid or hinder student success. This thesis will focus on finding various learner subpopulations to elucidate those materials that best aid certain students to allow for a more tailored teaching mode for future iterations of the course. In addition, this thesis will define an 'effort' statistic that encompasses the holistic engagement of a given student in order to provide an additional statistic to use when determining final grades.

I begin with a course specific analysis of enrollment demonstrating the significance of this type of analysis. Given enrollment numbers that rival a general institute requirement, this analysis could easily extend its finding to these other large courses. In addition, I show how this course, by utilizing the MITx platform, best leverages a way to facilitate student introduction to a programming language.

Second, I look to see how the introduction of an 'effort' statistic would positively affect grading outcomes for certain students near a letter grade border. I identify a possible mode to utilize this index during the determination of final grading as an additional measure in order to improve the issued letter grade.

Lastly, curious to see generalizability of results, sought out to survey past students to see if their computed effort statistic aligned with their personal view of effort into the course. I furthermore use these findings to tabulate resources that are most helpful to student subpopulations. Collectively, the inquiries in this thesis form a foundation for a more equitable way of teaching where students are best equipped for success.

Thesis Supervisor: Ana Bell
Title: Lecturer

Acknowledgments

I would like to express my deepest gratitude to community that has supported me throughout my MIT career. The help I have received from faculty in my department, as well as those I have had the good fortune have meeting have molded this cumulative work.

Most crucial to this process was my advisor Ana Bell, whom I have had the good fortune of knowing and working with throughout the past three years while TA'ing MIT's introductory programming course. Not only serving as the inspiration for this work, but also a motivating factor in my decision in wanting to be an educator, Ana has set the bar for the type of teacher I aspire to be. She demonstrated to me the same patience that she shows towards all of her students, and I cannot be more grateful to have worked with such an encouraging individual. Thanks to Ana, I was able to couple the disciplines I devoted my undergraduate career to pursuing, programming and education. She has taught me that all educators should strive to help all individuals, starting with the simple act of making all students feel welcome, regardless of their background. Working with support from the Office of Digital Learning, I was able to create a project that does just that, with hopes that thesis can help further improve an already deliberate course. Thank you Ana for all of your kindness and continuous understanding these past years!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The collection of courses popularly known as 6.00, an Introduction to Computer Science and Programming, has become a staple course for many Massachusetts Institute of Technology (MIT) students. The courses 6.00, 6.0001, and 6.0002 combine a traditional classroom environment with an online teaching platform deliver introductory programming content, and draws students from all departments and degree levels.

Due to the course's relevance to many other engineering and non-engineering fields, it has become a requirement for four departments outside of the department of Electrical Engineering and Computer Science (EECS). In addition, the course is cited as a pre-requisite course for 38 other MIT courses ranging from the Civil Engineering to Architecture departments. As a result, enrollment has swelled in recent years, with enrollment numbers easily stacking up to that of any GIR, giving rise to a dataset that may inform course staff on how material is utilized and how the course can better adapt to benefit its learners.

1.1 The Emergence of 6.00

6.00 was first introduced in Fall 2008 by Professor John Guttag. Starting out with a class size that could easily fit into small recitation classroom, it has now outgrown even the largest lecture hall at MIT. Currently, the course comfortably draws over 500 students a semester, inviting students from over 90% of MIT offered majors.

Students also include cross-registered students from Harvard as Wellesley as well as adult learners affiliated with the institution.

In addition to the listed courses available to MIT affiliates, a web-based version of the course arose in 2012 and is currently still offered on the platforms edX and MITx. edX is the second larger provider of Massive Open Online Courses (MOOCs), hosting thousands of courses from 130 academic institutions located throughout the world. MITx, a subsidiary version of edX available exclusively for MIT students, hosts accompanying material for current iterations of 6.0001 and 6.0002 (both half-semester courses), with 6.00 being the full-semester offering that comprises both halves. All offerings of the course boast exposure to computer programming as well as computational thinking, appealing to all students alike regardless of their background.

1.2 Limitations to Student Evaluation

Akin to other traditional courses, 6.00 has yet to perfect a curriculum that appeals to all learner types. The course is overwhelming delivered in a one-size-fits-all manner, despite over 10 years of refinement. This remains a fatal drawback for certain students that may have limited prior exposure to the field, as material is either unstimulating or ineffective at elucidating concepts. This problem is revealed during final grading where students are evaluated as a collective whole, since the large enrollment inhibits meticulous analysis of individual performance.

As it stands, students are evaluated according to a sliding scale at the end of the semester, with grades weighted according to the outlined syllabus. This embraces the idea of meritocracy, that higher marks deserve a higher overall grade. However, this method fails to acknowledge the varying degrees of "effort" that each student exerts, and fundamentally undercuts those students that may not have had previous exposure to the subject. Even worse, it may ultimately dissuade those students from ever pursuing the field if they receive poor marks even if they show potential for future mastery. In short, this type of student evaluation overlooks a complex set of factors that dictate a student's understanding of course material and is insufficient as the sole basis for grading.

1.3 Contributions of this thesis

There stands an unprecedented opportunity to analyze this wealth of information to better understand differences between students across years and backgrounds. Since course content and structure has been relatively homogeneous across the last two years, we can analyze recurring trends and student archetypes of student behavior that can be generalized to current and future iterations of the course.

Using data collected from at-home assignments, exams, the MITx platform, and office hours, we uncover patterns of student behavior and resulting subpopulations that arise. By studying meta-data extracted from the MITx platform, we use the captured click-stream data from the site to see which resources are best positioned to help our students understand the targeted concepts. In the following chapters, we present a thorough analysis of data from the three semesters spanning Fall 2017 to Fall 2018 for the courses 6.00, 6.0001, and 6.0002.

Chapter 2 introduces the disparate overall time invested into MITx content, and relation to overall grades. It demonstrates the immense potential that MITx platform has on achievement for those students outside of EECS and younger class years.

Chapter 3 delves into the specific ways students are interacting with the course can also affect performance. We show how distributed engagement is preferential over fewer but longer practice sessions. In particular, we show how this type of interaction best suits the subset of aformentioned students.

Chapter 2

Student Enrollment and Interaction

Prospective students who are considering an introductory programming course have the option to take either 6.00, or both 6.0001 and 6.0002 to receive equivalent credit. It is to note that the the full semester course 6.00 has been abandoned in favor of the 2 half-semester equivalent courses, 6.0001 and 6.0002. As a whole, the course aims to prepare students for higher level programming courses by equipping students with skills that can be generalized to all programming languages. Emphasis is not necessarily placed on specific Python coding practices but rather the development of a computational mindset for approaching complex problems. In this chapter, we delve into a diverse set of student outcomes in the course and the differential modes of interaction with the course.

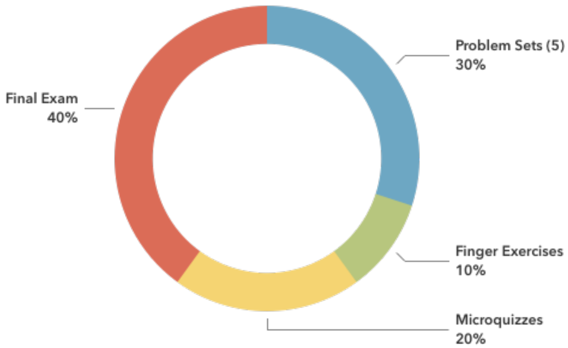
2.1 Structure of the Course

The course 6.00 was the earliest introductory programming course offered at MIT, introduced in Fall 2008 by Professor John Guttag. Starting out with a class of less than ten students, it has grown immensely, now with a consistent enrollment of over four hundred students per semester. Due to its popularity, 6.00 was eventually split into 2 half-semester courses, 6.0001 and 6.0002, to better manage the larger enrollment numbers. This subsequently made it feasible for freshman under the credit limit to have the option to have exposure to the field while still being able to complete their

GIR requirements in a timely manner, a suspected motivating factor in its growing popularity. The course is geared towards students with little to no programming experience, however students of all backgrounds are encouraged to enroll in the course. The first half (6.0001) focuses on skills to help students feel confident in writing small programs while the second half (6.0002) focuses on ways to develop a computational thinking mindset. Thus, students who solely wanted to focus on developing basic programming skills could elect to take just 6.0001. When taken together, the courseload is equivalent to a 12 unit class, with each half-semester course listed as a 6 unit course that outlines 3 hours of in-class lecture time in addition to 3 hours of expected work attributed to problem sets and other preparation. Two 1.5 hour lectures on Monday and Wednesday afternoons focus on core concepts of programming, with optional 1 hour recitations available on Fridays. Students are exposed to a variety of concepts including simple algorithms, data structures, testing and debugging, and algorithmic complexity. Students are required to complete problem sets that each test a different fundamental programming concept, such as classes or dynamic programming. Alongside each problem set, students are also expected to complete a checkoff to ensure they understand the key concept being studied. In general, content is curated so that students with little to no programming experience can easily understand and master the material. The course continues to be updated with advancements in the Python language, now updated to reflect Python 3. Although 6.00 offered a way for students to gain exposure to all course concepts, it was ultimately phased out in favor of the half semester offerings in order to decrease the administrative overhead that accompanied each individual course. As of Spring 2019, 6.00 has stopped being a regularly offered course.

2.1.1 Grade Breakdown

The course is broken into categories with weights that add to 100%. The heftiest portion comes from the final exam, weighing in at 40% of the overall grade. Next, problem sets are worth 30%, distributed evenly across the 5 problem sets for the half-semester courses or 10 problem sets for the 6.00 version. As of Fall 2017, 3 mandatory



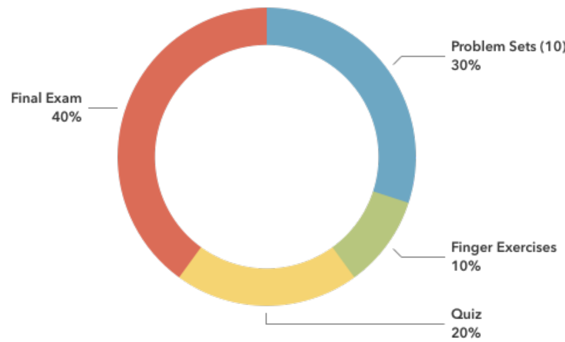


Figure 2-1: Grading Policy for 6.000x **Figure 2-2: Grading Policy for 6.00**

microquizzes were added to the half-semester courses, worth 20%. For 6.00, this 20% comes from a midterm exam instead of shorter microquizzes. For both classes, the remaining 10% comes from finger exercises hosted on MITx.

Grades are evaluated at the end of semester using the weights previously described. Cutoffs are adjusted, beginning at a traditional 90/80/70 border for A/B/C grades respectively, and adjusted downward to reflect semester dependent content difficulty. In the remainder of the chapter, we explore in detail how the course is structured, how students interact with the course as a result, and finally, how content can be improved to enhance student understanding of concepts.

2.1.2 Additional Course Resources

Although the course sets high expectations from its students, it offers an abundance of resources aimed to help students along the way. Students have access to videos hosted on the MITx site, offering a succinct summaries and review of key concepts from an associated lecture. In addition, students have access to 60 office hours hosted each week, with access to on 4 TAs or LAs on average per hour who are fluent in the course. There is also an actively monitored Piazza forum where students have the option to ask their fellow classmates or course staff for help. A document listing over a dozen of introductory programming tutorials is also posted in the case that students deem staff-offered resources is not comprehensive.

In addition to these resources offered by the course, MIT also offers support for its students. HKN, an EECS honor society that offers free tutoring for all department courses, is another way students can receive one-on-one tutoring with a knowledgeable student. Freshman additionally have the option to request additional tutoring from SeminarXL.

Given these extensive resources available to students, we sought to ask if our students effectively able to utilize them in order to facilitate learning?

2.2 Breakdown of Enrollment

Reviewing course enrollment for the 3 semesters spanning from Fall 2017 to Fall 2018, we find that a substantial number of students elect to take the course although it is not required for their specific major. It is suspected that over half of students, 486 out of a total of 958 students who took the course during academic year Fall 2017-Spring 2018, enrolled in the course in addition to graduation requirements. Fall 2018 was excluded due to a lack of information of freshman declared majors. This statistic, even when considering those students who may be pursuing a programming minor, still is quite large and indicates the increasing relevance of the course across all disciplines. If indeed this many students are electing to take the course outside of majors that have little dealing with programming, it may be useful to see whether these students are actually exit the course with a fundamental understanding of programming in Python.

In addition, we find that about one third of students taking the course are course 6 majors, or end up declaring course 6 if they take the course as a freshman. It may be beneficial to see how their activity may contrast or correlate to those students outside of the major, and how their performance ultimately differs as well. Overall, there are three main questions we would like to ask to address student enrollment:

1. Are a significant number of students enrolling across all MIT majors?
2. How does the effort and performance of students differ across those students who are required to take the course as opposed to those who are not?
3. Does the course offer enough material and support for the wide demographic of students?

From the 1000+ students that pre-registered for the course in Fall 2017 and Spring 2018, 958 students remained with the course to completion, earning a final letter grade of D or higher. Students who were freshmen, earning a "D" or "F" were omitted due to MIT's freshman no-record policy for grade below a C. In addition, students were additionally filtered out by majority completion of the course, including taking final exam, as course records are not always updated to reflect students who drop the course after the assigned "drop" date. These students were filtered out in order to avoid bias in later analysis of student interaction. Before delving into the specifics of these resources, we hoped to identify those students who took the course with intentions of learning the content and completing it in full.

2.2.1 Freshman Interest

Freshmen at MIT have a unique opportunity when selecting courses for their first semester. Although placed under a strict credit limit of four and a half courses (evaluated at 12 units per full class, or a total of 54 units), students take all classes under a "pass/no record" grading policy. Thus, a student can opt to take any course within the constraint of their pre-requisite portfolio with no risk of penalty on their transcript and GPA. Since students do not have to declare a major until their freshman spring, they can gauge overall interest a department through introductory courses offered by most majors. Although 6.00 is not a General Institute Requirement (GIR), a course that is required by all undergraduates, the course still draws a population as large as

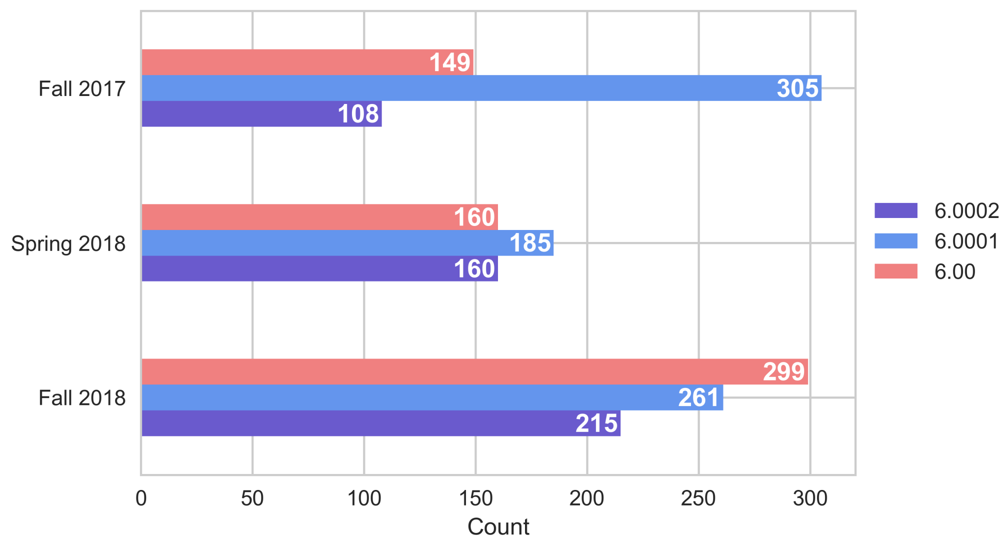


Figure 2-3: Aggregate Enrollment Numbers Histogram of enrollment for each course for the semesters Fall 2017, Spring 2018 and Fall 2018. These numbers include cross-registered students and students who did not finish the course however remained officially enrolled for the semester.

those mandatory courses. This student body is a representative demographic of students across all majors, becoming increasingly diverse when considering auditors and those students cross-registered from other institutions. This all-encompassing cross section of students provides informative data on how students of different backgrounds and prior expertise interact with the course.

2.3 Teaching Staff Structure

To support the facilitation of such a large class, there are two dedicated lecturers, Ana Bell and John Guttag, with Eric Grimson overseeing approximately 1/4 to 1/3 of lectures depending on the semester. At minimum, there is a staff of 12 TAs consisting of undergraduate and graduate students that help to update and refine course material. In addition, between 30-40 undergraduate LAs are recruited to staff office hours to help with in-person questions and problem set checkoffs. In general, staff is selected from those who have previously taken the course or those who have extensive programming experience and a passion for helping students. Staff can be accessed in a variety of ways, whether it be at lectures, office hours, Piazza, or over email. Given that about 350 students take the course at any given point in a semester, there is about a 1:6 ratio of staff to student at any given point, a generous figure considering less-staffed GIRs. Certain mediums like office hours provide an even favorable ratio, offering students an opportunity to get individual and tailored help with issues.

All staff has been specially selected across a wide pool of applicants, with preference for TAs given to upperclassmen programmers who have taken the course, or previous LAs of the course. In general, TAs are more likely to answer course logistic issues, however all staff is expected to know the course content and feel comfortable teaching students. Students can reach TAs and LAs in office hours, or through Piazza if they cannot make scheduled times. In the case they have specific questions for a course administrator, they are able to email professors and receive a response generally fairly quickly.

As the course is becoming more popular, we expect that the pool of staff applicants will become increasingly competitive. We also expect more of the staff to have previously taken the course in accordance to the recent EECS curriculum update in Fall 2016. Although the curriculum offers increased flexibility for computer science students, the fundamental programming requirement still mandates that students take 6.0001 and 6.0002.

2.4 Course Teaching Ideologies

The course has been designed to appeal to a variety of learning styles, supplementing teacher-centered instruction with student-centered activities. This dual style is crucial in closing the gap between expected learning of material and actual student understanding of material [3]. This, when combined with consistent feedback, help maximize the potential of effectively delivering content. The course follows suggestions outlined by Chen et. al. by integrating immediate corrective feedback in a blended learning environment. Taking the generally accepted definition of blended learning, where 30-80% of learning occurs through web-based tools [2], we see how the structure of 6.0001/2 tries to follow this theoretical framework. It with the intention students being able to practice these skills that web-based tools were added to the course. This platform also gives us increased resolution in exploring student interaction with the course to ensure students truly taking advantage of these resources and indeed benefiting.

2.4.1 Traditional Teaching Methods

As most classes at MIT, the course is didactically taught with in-class lectures. Each of the course instructors have preferentially taken over the half-semester courses. Lecturer Bell primarily runs the 6.0001 half, while Professor Guttag primarily teaches 6.0002. They each lecture for one and a half hours every Monday and Wednesday, elaborating over concepts using the aid of PowerPoint slides and relevant sample code. Students are additionally able to attend a one-hour recitation section each Friday, to go over key concepts presented in class. At recitation, a TA goes over a set of notes that aggregates applicable content along with helpful approaches when encountering related problems. This supplements recitation-specific code that also covers edge cases that may not be covered in lectures. In addition, lecture code is also reviewed and extended to ensure students have an opportunity to revisit concept that may have only been briefly touched on in class. These aspects, combined with assigned problem sets, are typical markers of traditional college courses.

A drawback of the avenues listed above is the lack of emphasis on individuals. It assumes that all students learn in the same manner (aurally in lectures) and does not provide extensive support for those more visually or kinesthetically inclined [6]. However, other aspects of the course seeks to address these shortcomings and ensure all students have a chance of success.

2.4.2 Innovative Teaching Methods

To counteract issues that traditional teaching methods usually give rise to, 6.00 also utilizes MITx to give students an additional resource in their programming journey. There, they are given an interactive platform to engage with and get immediate feedback on short programming exercises. In addition, 6.00 is structured such that all office hours are staffed with at least 1 TA to ensure proper and consistent course messaging is relayed to students. A strong community of teaching assistants, developed through weekly staff meetings, ensure that each TA feels increasingly invested in every student's success.

The main reason that MITx plays an important role in the course is due to its accessibility anytime, anywhere given online access. MITx pairs previously recorded lectures and specific content videos that can be replayed multiple times at a speed best suited for the learner. There, they can also find optional coding programs as well as mandatory finger exercises that test basic programming concepts. These help familiarize the students with the platform come time for microquizzes. This resource can have tremendous impact with student viewership, allowing students to easily revisit problem material and get quick feedback on coding performance. Within MITx, they are given access to an IDE that although does not provide feedback on un-compiled code, each coding questions does come alongside a comprehensive suite of tests that cover all edge cases. This, combined with the course-endorsed Spyder IDE gives students the ability to test locally on machines. This also gives students exposure to debugging when completing problem sets. The cross-talk between coursework platforms allow students to navigate the intricacies of programming and testing code, while growing as coders.

Strong TA involvement in the course promotes interaction between staff and the student body. This also reduces the barrier for students asking for help; students feel more comfortable asking for help, whether it be in-person at office hours or anonymously through the Piazza forum.

Massive Open Online Courses

MOOCs have been rising in prevalence across all learner levels and subjects. Given the incredibly accessible platform, the internet, learners from all parts of the world are able to come together to learn myriad skills that would otherwise be inaccessible to them in a traditional classroom environment. The virtual classroom setup allows participants to engage with the content at a pace that is right for them, eliminating the barrier for learners who might time restrictions. Most courses provide video recorded lectures as well as digital copies of course material, essentially offering all aspects of a traditional classroom from the comfort of their home. There, they can play, pause, and accelerate videos as they see fit while also getting real-time feedback on their actions.

The MITx Platform

The obvious goal of integrating web-based platforms is to foster student achievement. Numerous studies have supported this notion, showing that platforms with formative feedback help to motivate students alongside encouraging deep learning [5]. The MITx platform provides this lauded feedback mechanism, implemented as a "checkable answer feature" (CAF). Although the feature does is not always able to provide as detailed of an explanation as a staff member, it generally is able to address common mistakes. If students have any lingering questions, they are always able to reach out in any of the aforementioned routes, whether it be through Piazza or email.

The aim of the course by integrating this platform is to keep students engaged outside of scheduled lecture times. This includes building in elements of spaced learning, as well as providing a variety of exercises of varying difficultly to appeal to learners of all levels. The following chapter discusses in detail the specific ways that users are interacting with the site and how those patterns affect their overall performance in the course.

2.5 Conclusions

The course takes measures to ensure students are leaving with the appropriate skills to succeed in higher lever computational courses or when encountering any situation that would benefit from a computational mode of thinking. The union of traditional teaching methods with web-based approaches allow for maximal student retention of content. To support this fact, we more deeply investigate specific student interaction with the MITx site as well as performance on problem sets, exams, and attendance in office hours.

Through this analysis, we aim to refine course pedagogy and student support modes, either by adapting existing content or adopting novel teaching practices. As MOOCs are becoming more prevalent in foundational courses, we hope that through this case study, we can apply results to other introductory courses that may also want to investigate their effectiveness.

Chapter 3

The Relationship Between Learning and Deadlines

A tride tradition of pedagogy has reinforced the value of acquiring a multiplex of skills during a student's undergraduate career. Full-time students learn to balance at minimum three full courses that focus on disparate subject matters, all with varying assignment deadlines. Clashing of deadlines result in "hell weeks" that inevitably arise, typically causing students to temporarily set aside learning for courses that do not have imminent deadlines. Thus, a general phenomenon that educators find is the sub-linear learning curve of students over the course of a semester, with noticeable spikes occurring prior to important deadlines. In this model, students do not necessarily learn the material as it is presented in class or through assignments, however will cram large amounts of information before an important deadline. By investigating student interaction data on MITx as well as collected office hour ticket data, we attempt to observe possible benefits of those students who use a spaced practice model in the course. We show that interaction with the MITx over the course of many short sessions, effectively spacing out "studying" over time, is an important predictor of student achievement with regard to their final letter grade. Furthermore, we show that those students that elect to complete optional exercises, and utilizing the CAF feature of the platform, also tend to perform better in the course. We conclude the chapter by proposing relatively simple ways that students can modify behavior in order to leverage these observed benefits in student achievement.

3.1 Introduction

There is an abundance of research into traditional learning environments as well as pure web-based learning environments. However, there is relatively little insight to student interaction and performance with regards to a blended learning environments that 6.0001 and 6.0002 provide. However, despite a lack of research in this area, there are consistent patterns that characterize both learning environments: 1) formative feedback is most constructive for student understanding, and 2) increased participation is an indicator of higher grades [4].

Using these factors as guiding principles in analyzing student behavior, we sought out to observe the effect of learning as a function of study time. We aim to see how a student's interaction patterns coupled with MITx feedback mechanisms, affect a student's overall understanding of course content. It is important to note that the singular factor of time was not used as a basis for effort; an abundance of literature would actually suggest that aggregate study time does not correlate to academic benefits [7]. Instead, we seek to see how students interact with the course at specific time points, and how this spaced interaction correlates to student success, as measured by overall performance in the course. A noteworthy finding is that study time effectively partitioned over time leads to better performance, otherwise known as the spacing effect. In this chapter, we show evidence that supports these established learning theories even with only a small subset of student study patterns coming from MITx.

3.1.1 The Psychology of Learning

It is easy to find mounds of conflicting literature expounding on the intricacies of student learning and comprehension. As there is no accepted universal standard for teaching and learning, students enter higher education having been exposed to a myriad learning environments. Thus, each student carries their own unique style of learning that does not always neatly fit into a generally rigid college course setting.

As research into the psychology of learning is ever expanding, key similarities across styles have been elucidated. Namely, spaced learning, formative assessments, and immediate corrective feedback prove most influential in a typical classroom.

The spacing effect was first characterized by the German psychologist Hermann Ebbinghaus in 1885. He demonstrated a learning curve marks the increased learning time that accompanies increased material. This, coupled with the proven difficulty of initial learning as opposed to relearning clearly shows the drawbacks of typical marathon study sessions [1]. One might consider a situation with two students A and B; student A might study the material for one hour a day over the course of two weeks while student B studies for fourteen hours over two days preceding an exam. Not only does the research suggest the student A is more likely to perform well, student A is more likely to retain the material if retested at a later date. This early work by Ebbinghaus has laid the foundation for the development of new pedagogical frameworks aimed to support students.

The Spacing Effect in Practice

Knowing that distributed studying is superior to monumental studying, the 6.00 course seeks to best put it to practice. The MITx portion of the course seeks to mitigate the pitfalls of long study periods by adding finger exercises to the course requirements. This study only extends to the spacing effect with regards to interaction with MITx, as we have not collected data on study patterns with regard to other outside resources. Students come from a variety of backgrounds, all having unique learning styles. There is no unifying "learning" style that is taught across schools, rather the emphasis has tended to be on passing standardized exams [1]. Thus, the concept of learning has lost its underlying value of being an asset to reconcile with the outside world. Instead, formal education has conditioned students to focus on a specific number that appears on a report at the end of a semester. This occurrence bleeds into higher education, where a lot of the focus is still on grades and GPA.

3.1.2 Blended Learning Approaches

One of the first analyses on blended learning environment arose from investigation into another MITx introductory physics course. Specifically, research focused on the benefits of applying elements of immediate corrective feedback in reinforcing content and addressing conceptual misunderstandings sooner rather than later. Since using a MOOCs platform such as MITx gives students flexibility in completing content, essentially acting as an asynchronous platform for learning, we sought to define measurable statistics of student interaction and performance metrics.

Adapting metrics first proposed by Chen et al, we defined the principle elements we wanted to understand regarding student engagement [3]. We sought to find those patterns that are both positively and negatively correlated with performance in order to explain performance. The measures that were especially of interest included 1) frequency of interaction with the site along with associated events that occurred including completing mandatory exercises as well as interaction with optional coding question or videos, 2) interval between unique sessions, and 3) content that was frequently revisited by students.

All of data utilized in this paper comes from the Fall 2017 and Spring 2018 semesters. Extensions of the findings were applied to the Fall 2018 semester.

3.2 Data Collection

This thesis focused on using data aggregated from course records, MITx, and office hours to explore the relationship between student performance and recorded student effort. We examined collective information of all students enrolled in the 2017-2018 academic year who passed the course (earning a D or above) to find indicators of student completion and success in the course. Only data from the most recent offerings of the course were used due to the high degree of similarity in content, course structure, and overlap in teaching staff.

To select those unique users who interacted with the course, we cross-checked those students who were officially enrolled according to the MIT Registrar and data from

MITx. In this analysis, we focus first on measures that predict course completion, then on criterion that predict performance.

3.2.1 Measures

We obtain a series of measures that predict completion of the course: (1) MITx interaction patterns, (2) assignment submissions time-stamps, and (3) the number of unique MITx and office hour sessions. A unique session in this context was defined as a log of click events separated by at least 30 minutes of inactivity for MITx, and a personal ticket request that is separated from other tickets by 30 minutes or more for office hours.

3.3 Results

3.3.1 Enrollment Number Analysis

The course begins each semester with a high volume of students showing initially interest, as demonstrated by enrollment and activity on MITx. However, we find that generally there is a 10% difference in those students who interact with MITx and those to are officially on the course roster at the semester's end. Most of this difference is attributed to MIT policies that allow individuals to drop a class past a due date, as shown by the drop-off in MITx activity around MIT's official "drop date" for courses.

This discrepancy sparked an interest in uncovering the relationship between a student's engagement with the course and their ultimate decision to stick with a course to completion. Thus, we looked at patterns of interaction with MITx to uncover commonalities in students who dropped the course. The average student who dropped the course with nonzero MITx activity or office hour visits logged about 33% less minutes less watching videos, and 42% less office hour visits. Thus, the interaction with the MITx videos, finger exercises, and office hour visits vary widely between students who drop the course as opposed to complete the course.

Semester	Course	Unique MITx Users	Enrolled	Passed Course
Fall 2017	6.00	285	150	122
	6.0001	393	300	253
	6.0002	94	106	91
Spring 2018	6.00	379	160	138
	6.0001	382	185	169
	6.0002	209	160	140
Total		**1742**	**1061**	**913**

Table 3.1: Student Enrollment Numbers Over a Semester Data from Fall 2017 and Spring 2018 for students enrolled in 6.00, 6.0001, and 6.0002.

Letter Grade	6.00	6.0001	6.0002
A	30.0%	39.8%	35.2%
B	40.7%	35.8%	41.7%
C	11.3%	13.3%	9.3%
D	4%	3.9%	8.3%
F	11.3%	7.0%	5.5%
O	2.7%	0.2%	0.0%

Table 3.2: Percentage of Students Receiving Letter Grades Shows the percent of students obtaining a given letter grade A-F across Fall 2017. A letter grade 'O' reflects an incomplete.

We first outline the distribution of final grades to get a high level overview of how students perform, and the breakdown on how students in every letter grade group generally performed on course subsets. As shown in Table 3.2, many students end up receiving A's and B's as final grades, with typically more A's granted than B's. However within these groups, as shown in later investigation, we demonstrate different levels of effort put into the course as determined from the above statistics. Later in the chapter, we show typical archetypes of students within each letter grade bracket. We subsequently propose ways to acknowledge this differential effort and reward students for engagement, with an accompanying scoring system.

3.3.2 MITx Trends

In this section, we specifically look at data solely gathering from the MITx platform for 6.0001 and 6.0002. Students who were registered for the full semester version (6.00) were automatically enrolled in both MITx courses; their interaction data was aggregated to compose their overall interaction with the site. Only those students who completed the course were analyzed so to avoid outliers from students with limited data points. In addition, auditors are omitted as they are not required to complete course material, and generally have sporadic interactions with the site.

To outline the immense disparity between those who interact with the course in some capacity and those who ultimately who cease to continue, Table 3.1 shows the immense attrition that occurs throughout a semester. Various reasons for such attrition is due to 1) discrepancies between those who pre-register for the course and those who ultimately take the course, 2) students who drop the course at any point during the semester, 3) students, namely freshman, who are able to fail a course during their first year without any penalty on their record, and 4) students, typically upperclassmen, who are able to use a Pass/Fail grading scheme for the course. Since we use completion as the primary filter in the ensuing analysis, we only analyze those students for which we have ample click data on and also those students who sought out to learn the course material as a whole.

For the semesters analyzed, the content posted on the site did not change, barring minor clarifications and changes in word choice. The composition of videos, the mandatory finger exercises, and practice exercises remained the constant. The only differences between semester offerings were addition of microquizzes that that were hosted on the site, however this content will not be considered until later analysis.

3.3.3 Aggregate Interaction Patterns

The first attempts at analyzing MITx data was focused on aggregate interaction patterns of students with the site. This included number of visits to the site, average site interaction time, total time spent on watching videos, time spent on finger exercises, as well as other information from click data. Although this method yielded information revealing three main user interaction patterns, it ultimately fell short in explaining the holistic differences in student performance. Thus, the bulk of later analysis discards the idea that greater total time spent is influential, and rather takes into consideration additional metadata regarding the specific ways students interact

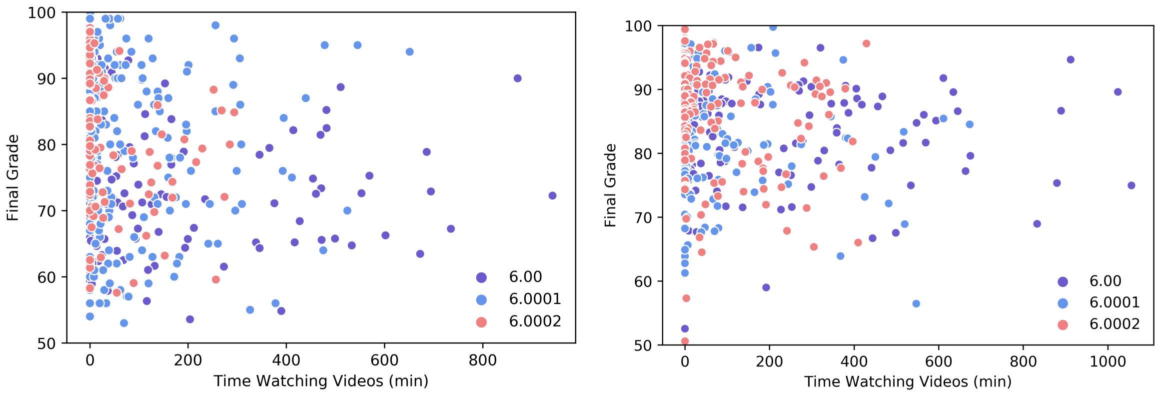


Figure 3-1: Time Spent Watching Videos by Final Grades Scatterplot of MITx video interaction by final grade of students who completed the course in Fall 2017 (**A**) and Spring 2018 (**B**). **A** 53.7% of students watched 5 minutes or less of video content. **B** 59.3% of students watched 5 minutes or less of videos.

with the course over time.

Looking at Fall Fig. 3-1 displays information regarding the total time students spend watching MITx videos plotted against their final grade. For both Fall 2017 and Spring 2018, over fifty percent of students watched less than five minutes of total video content. Out of a possible 11 hours of videos for 6.0001, and 8 hours of videos for 6.0002, only 14 students watched 80% or more of posted video content, a mere 1.5% of total students who completed the course.

Taking in consideration the dataset as a whole, there is no clear correlation between the amount of time spent watching videos and the resulting grade. In fact, the calculated R^2 value for both semesters is negative (-0.17 and -0.13 respectively), suggesting that increased time spent on watching videos correlates with a decreased final grade. Even when omitting those students who watched less than 10 minutes of MITx videos, this correlation value still remains negative, at -0.07 for Fall 2017 and -0.14 for Spring 2018 (note how the R^2 value is now larger for the Spring 2018 semester. This goes contrary to the view that increased time spent on material is beneficial for learning however is consistent with notions of spaced learning effects. In short, increased interactions with the site only partially explained final grades; thus, alternate modes of analysis were employed to better make sense of the contradictory results.

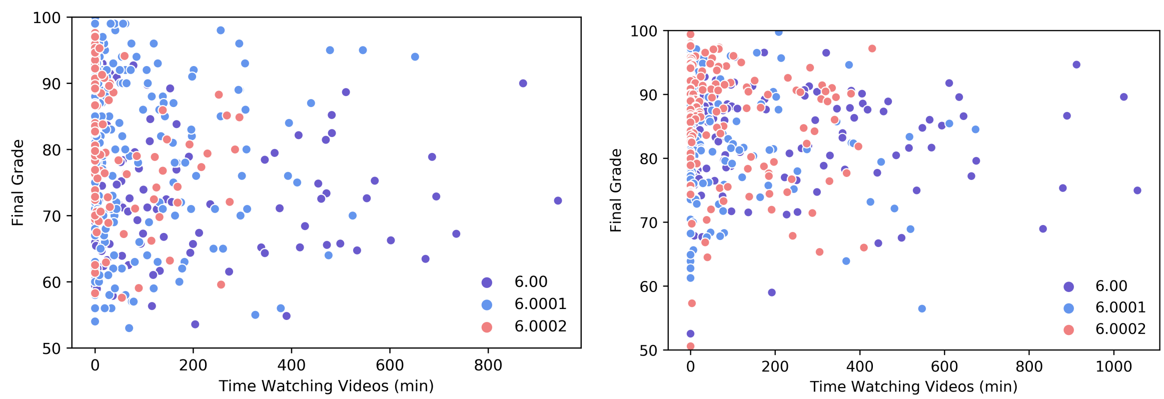


Figure 3-2: Analysis of Number of MITx Sessions by Final Grades Scatterplot of unique MITx sessions by final grade for students who completed the course in Fall 2017 (**A**) and Spring 2018 (**B**).

3.3.4 Timelapse Patterns

Expanding our analysis to include timestamps of unique MITx sessions, we sought to compare the effect of video session count while controlling for time on-site. By looking into how frequently a student interacts with the site, and if students are typically watching videos before a lecture as opposed to after a lecture, we hope to detect slight correlations that may be attributed to spaced learning in practice. We want to find those students who are putting in more than the minimum required work as determined by the finger exercise deadlines.

Accordingly, we looked at the number of a student's unique sessions during which they watched any video for more than 10 seconds. This threshold was to ensure students spent ample time looking at the content and filter out actions of deciding to watch a video. The minimum number of sessions was 8 for 6.0001 and 5 for 6.0002. These thresholds in consideration, we plot students' total number of unique sessions against their final grade as shown in Fig. 3-2. We find that this ends up being a slightly better indicator of performance as opposed to overall time spent watching.

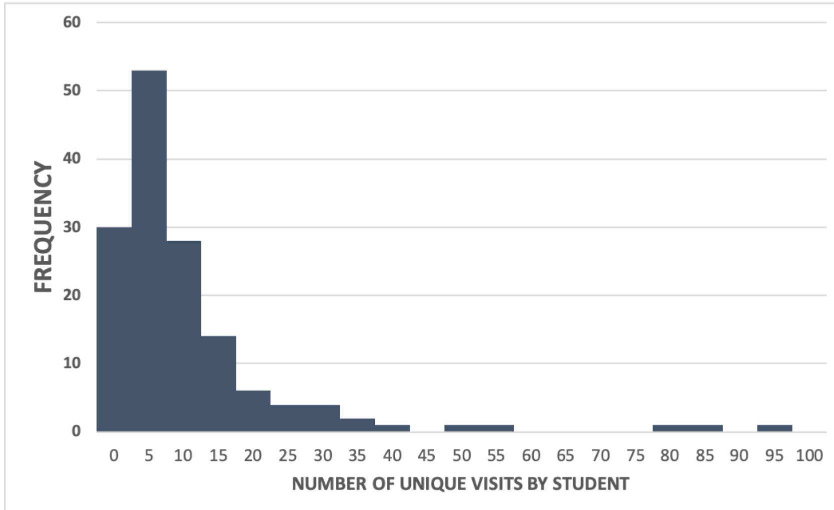
3.3.5 Timelapse Patterns

Taking into consideration the principles of space learning, we reanalyzed the same MITx data, this time including temporal aspects. Using time stamps from unique interactions with the site, we explored how student achievement varied based on consistency of attendance.

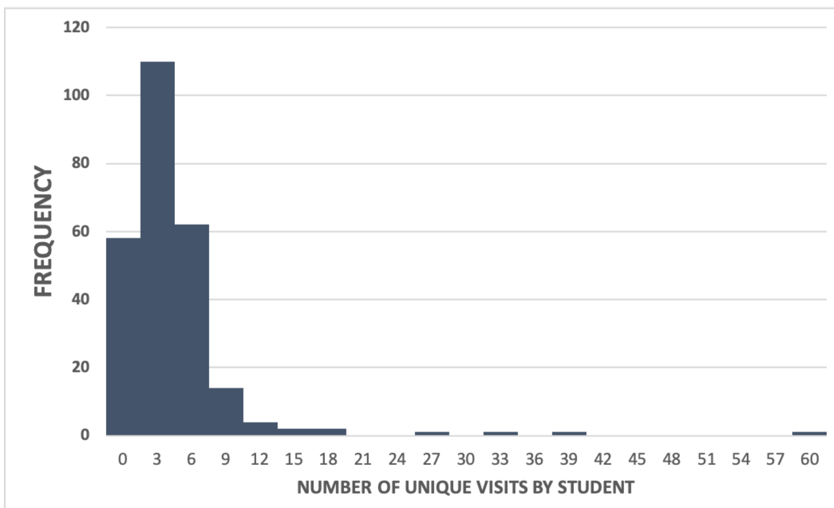
At minimum, we expect that students visit the MITx platform the same number of times as the total number of exercises. As for office hours, we would hope to expect at minimum the number of visits as the number of problems sets for the course (either 5 or 10), however we adjust this number based on ticket number as students can opt to complete multiple checkoffs in one setting provided that the time window overlaps. The histograms shown in Fig. 3-3 do show relatively modes for students with 6.000x, with a higher average for 6.00 students as the course runs through a full semester.

3.4 Discussion

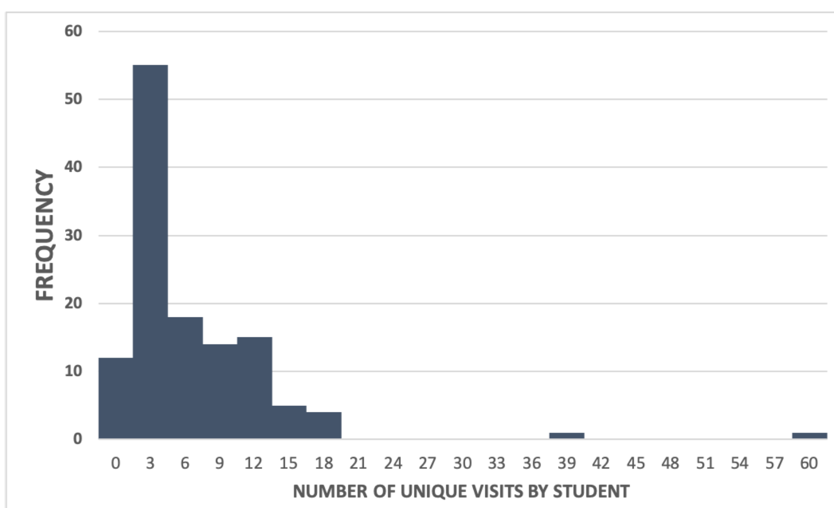
Motivated by literature on pedagogy and learning styles, we aimed to find specific student archetypes that exemplified mastery of different aspects of course material. More accurately, we hoped to identify those students that fit into one of four groups: 1) students who demonstrate the ability to code and understand the underlying concepts, 2) those who are comfortable in coding but have not learned the core notions, 3) those who have learned the material but lack proficiency in writing code, and 4) students who have neither mastered concepts nor feel comfortable writing simple programs. Our findings arose from in-depth analysis of student interaction data, agnostic of previous experience and background. Agglomerating results, these results suggest a mode of interpreting student grades and appropriately scaling up issued final grades. In addition, through this analysis, we also identify concepts that students generally have trouble understanding as well as resources that is less effective in teaching students or is confusing for learners. It is important to note that coding ability was determined by performance on problem sets (autograde score), finger exercises, microquizzes (from coding problems), and exam scores (the coding portion), with no information assumed about a student's background. Conceptual understanding was assessed from checkoff scores, microquizzes (from theory problems), and exam scores (the written portion). Using these markers as starting points, we appropriately weigh



(a) 6.00



(b) 6.0001



(c) 6.0002

Figure 3-3: Histogram of Office Hour Visits for Fall 2017

these different portions to best represent student understanding.

3.4.1 Measuring Understanding

Using all aspects of a student's performance on assignments and combining it with data collected from MITx and office hours, we have developed a scoring metric to assess participation, or more specifically an "effort" statistic. This "effort" index seeks to explain the engagement of a student throughout a semester, scaled to range from 0-100% to be consistent with other calculated performance scores. This, coupled with a student's assessed understanding of material (any combination of concepts or coding), would help course staff to evaluate performance. Recall that semester-dependent normalization was performed to account for semester-dependent differences in difficulty due to variation in content. Thus we can appropriately determine these statistics each semester to accurately gauge how added or removed content affect student mastery as a whole.

The "effort" index was constructed in the following way: accounting for time and engagement with the MITx platform (with videos, content, and optional exercises), number of visits and time spent at office hours beyond checkoff visits. Time spent at office hours was only considered for tickets that were not logged as checkoffs, as these visits were performed additionally upon mandated office hour visits by students. It is to note that this index does not account for additional time students spent on or with third party resources/platforms to better increase their understanding of course topics.

Thus, a matrix was created taking into account the following "effort" metrics for students who completed the course:

1. Number of visits to MITx site (for periods >10min)

2. Percentage of videos watched (instances with >80% video completion)

3. Average number of times videos were watched

4. Number of optional exercises completed

Variables	# MITx Visits	% of Videos Watched	Avg Video Views	# Optional Exercises Completed	# OH Visits	Avg OH Visit Time (min)
# MITx Visits	1	0.580	0.517	0.388	0.340	0.322
% of Videos Watched	0.580	1	0.720	0.700	0.588	0.612
Avg Video Views	0.517	0.720	1	0.450	0.482	0.509
# Optional Exercises Completed	0.388	0.700	0.450	1	0.530	0.499
# OH Visits	0.340	0.588	0.482	0.530	1	0.639
Avg OH Visit Time (min)	0.322	0.612	0.509	0.499	0.639	1

Table 3.3: Correlation Matrix of Effort Factors for Fall 2017. Variables related to MITx are more closely correlated while variables related to Office Hours tend to show greater correlation.

5. Number of office hour visits beyond checkoffs (identified by the removal of tickets referring to "checkoff" or variations of the term)

6. Average office hour ticket time beyond checkoffs (similarly identified by the removal of tickets referring to "checkoff" or variations of the term)

To analyze this data, a factor analysis was conducted in favor of a principle component analysis due to the few amount of variables as well as the interrelationship of these items. The associated correlation matrix for Fall 2017 students is shown in 3.3. Upon initial glance, variables related to MITx seem to be closely related whereas variables related to office hour visits are strongly correlated, indicating presence in either platform show increased engagement with related activities.

Inspecting factor loadings with 2 factors, it appears that increased engagement on either platform (MITx or Office Hours) are tightly linked, with higher engagement on one platform typically associated with higher engagement on the other as shown in 3-4. This makes sense as students who utilize these resources typically exhaust available course resources where possible. Similar patterns arose in Spring 2018, indicating similar patterns of student behavior with available resources. For the purposes of this analysis, only Fall 2017 was analyzed completely due to a lack of complete MITx data for the latter portion of the semester. However, for both datasets, as the eigenvalue was greater than 1 for F1, the data suggests that there is one factor that can explain a reasonable amount of variance, all which relate to "effort" that was previously defined. This singular factor (which consists of a linear combination

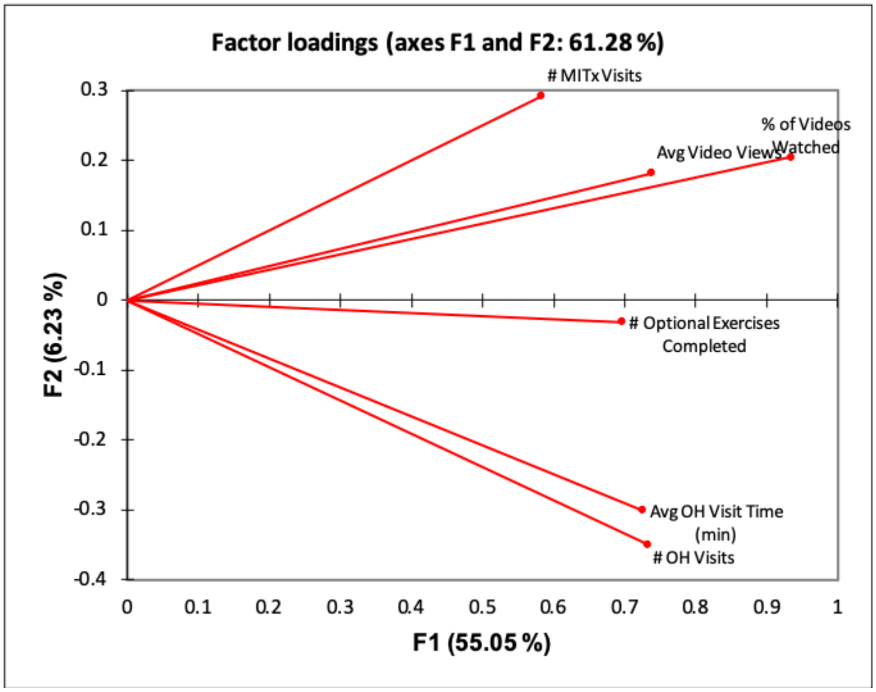


Figure 3-4: Factor Loadings for Fall 2017 Plot of factor loadings across 2 axes, indicating closely linked MITx variables juxtaposed with closely linked OH variables.

including MITx and OH engagement) is thus used to determine factor scores for each student to determine an "effort" score that could subsequently be used in grading consideration. In this analysis, loading scores were then normalized and scaled to 0-100% to determine appropriate "effort" scores for each student that could then be used in grading determination in a similar manner that finger exercise completion would be used.

	F1	Initial Communality	Final Communality	Specific Variance
# MITx visits	0.572	0.361	0.328	0.672
% of videos watched	0.932	0.745	0.869	0.131
# times videos watched	0.738	0.551	0.544	0.456
# optional exercises completed	0.707	0.522	0.500	0.500
# oh visits	0.702	0.489	0.493	0.507
time at OH	0.707	0.501	0.499	0.501

Table 3.4: Factor Loadings for Fall 2017

3.5 Identifying Student Effort

The identification of portions of courses that students are learning particularly well, as well their interaction with additional resources, could perhaps give rise to a way better schema of performance assessment. Given that most didactic modes rely performance on assignments, and not on the time invested or initiative in understand the material, final grading can be skewed in favor of those students who have had previous exposure to the concepts. However, since this is an introductory course, we do not want to skew grading in favor of students who were fortunate enough to have been previously introduced to concepts and thus more likely to exceed in the course. This coupling of variables hopes to recognize those beginner programmers and laud them for exemplary effort in engaging with other portions of the course with an "effort" index. In this way, students who do well would still receive the grade they earned; however students who are on a grade border and not higher, simply because they have had less opportunity to internalize the concepts, are still given a way to receive a higher grade due to their effort.

Based on this "effort" index, 11% of students were identified to have displayed exemplary effort, scoring 80% or higher based on demonstrated engagement in MITx and Office Hours. When cross-examining these students with grading, many of students end up falling between grade borders, and may benefit if "effort" was taken into account into grading schema. To show the vast differences in effort displayed by these students as opposed to the lowest "effort" input students, these students utilize the MITx platform over 5 times as much as their lower "effort" peers, logging over 6hrs on average more at office hours.

To apply these "effort" scores in a grading setting, scores can be appropriately considered in final grading sessions. When compiling student final grades for the year, student final grades are tabulated across assignments according to the syllabus to arrive to a final score, which is then sorted and grouped into letter grades. To utilize this effort statistic in a grading setting, these "effort" scores can be simplified into a categorical variable and bucketed into 3 main groups: (1) High Effort, (2)

"Effort"	% Students	# MITx Visits	% Videos Watched	Avg Video Views	# Optional Exercises	# OH Visits	Avg OH Visit (min)
80-100%	11%	25	94%	2.7	4.9	8.2	46.6
60-80%	16%	21	83%	2.3	4.4	4.6	38.6
40-60%	8%	13	46%	2.0	3.3	5.2	35.3
20-40%	37%	14	27%	1.4	0.8	1.6	22.1
0-20%	28%	6	5%	0.3	0.7	0.7	8.2

Table 3.5: "Effort" Scoring for Students in Fall 2017 Semester. About one quarter of students show exemplary effort (>60%) in their engagement with course resources, utilizing MITx significantly more than their peers with lower "effort" scores.

Medium Effort, (3) Low Effort. This additional statistic can then be considered for students who straddle grade letter borders in an effort to "bump" them up to a higher final grade letter. One possible mode of application is to consider the highest scoring 10 students of a grade letter and analyzing their "effort" scores, improving their letter grade only if they were determined to have displayed "High Effort."

It is to note that thresholds for these groups should be developed based on specific interaction data of the current semester, based on decisions by course staff. In this analysis, we group "High Effort" students as those who scored between 60-100%, "Medium Effort" students who scored 20-40%, and "Low Effort" students as those who scored 0-20%. Once determined, we examine final grading to determine if students, if any, would be positively affected by this additional statistic. Through this, we find that the final grades of 28 students would be positively affected through the consideration of this effort statistic, with the greatest impact on grades being upgraded from a B to an A.

This type of analysis could forego grading session discussions that call for TA input. In this manner, students would have the potential to be recognized for effort, even if unbeknownst by a TA or other course staff. This index provides another measure to consider in a way that positively impacts students and recognizes them for their apparent effort that would otherwise go unnoticed.

Final Grade	Letter Grade	Effort
77.5	B	High Effort
77.47	B	High Effort
77.12	B	Medium Effort
76.95	B	Low Effort
76.14	B	Medium Effort
75.27	B	High Effort
75.2	B	Medium Effort
75.07	B	High Effort
74.84	B	High Effort
74.69	B	High Effort

Table 3.6: "Effort" Consideration of 10 highest scoring students along the A/B Border for 6.00 Fall 2017. Of these top students in the B range, 6 students displayed "High Effort," and could have potentially be given an A as their final grade.

	6.00		6.0001		6.0002	
Letter Grade	Effort	Students Adjusted	Effort	Students Adjusted	Effort	Students Adjusted
A	34.0%	6	40.1%	6	38.9%	4
B	38.7%	3	35.7%	4	40.8%	3
C	9.3%	0	13.2%	1	7.4%	1
D	4.0%	0	3.8%	0	7.4%	0
F	11.3%	0	7.0%	0	5.5%	0
O	2.7%	0	0.2%	0	0.0%	0

Table 3.7: Average Percentage of Students Receiving Letter Grades given High Effort Adjustments. Shows the percentage of students receiving final letter grades by course in Fall 2017 along with the number of students "bumped" to the level due to the consideration of the effort statistic. A letter grade 'O' reflects an incomplete.

3.6 Student Groups

As this course attracts a wide range of students with varying backgrounds, the introduction of the "effort" index would provide a statistic that aid those students who may not have been exposed to course content prior to enrolling. Although effort does not directly translate into mastery over programming and related concepts, the goal of introducing this measure is to appropriately credit individuals near grade borders to reward them for engagement indicated by their MITx and OH history. Through this, we attempt to characterize the main types of students often seen in the course.

Mastering Concepts and Coding

The first group of students identified demonstrate mastery of the majority of course content. They feel comfortable approaching a computational problem and know how to approach writing a simple program. They have a mean autograder score of 9.85/10, and similarly high averages across other coding exercises. The finger exercise average was slightly lower, at 8.3/10, assumed to be lowered by those students with previous exposure to programming who neglected to complete all assignments due to confidence in scoring well in other aspects of the course. This theory is supported by the fact that missed points in this section were from problems that students never accessed, rather than attempted but did not successfully complete.

Mastering Concepts

This second group demonstrate primarily mastery over theoretical concepts and general grasp of lecture material. These students tend to do equally as well on problem sets, with a mean autograder score of 9.47/10, however do worse in all other coding portions. Despite this fact, they tend to have consistently high checkoff scores, and do above average on the paper portion of the exam. One noticeable characteristic of this group is their frequent attendance in office hours, attending at least 1.4x more often than any other group. They have lower scores on coding portions; upon closer inspection of coding problems, this is usually due to incomplete implementation of programs from a defect of time. These students typically display higher "effort" scores due to engagement on platforms and would benefit the most from the addition of the "effort" index.

Mastering Coding

This group is characterized by high scores across all coding components of the course. Similar to the first group, finger exercise grades are lower. This phenomenon is even more exaggerated in this group, attributed to the high number of upperclassmen and computer science majors in the course. We believe that these students primarily take the course to fulfill a requirement, and less so to learn all concepts. Thus, students will sometimes overlook deadlines and forget to complete these exercises in addition to checkoffs. These same students show lower scores on the paper portion of the exam, typically showing deficits in concepts spanning 6.0002, specifically inferential statistics.

Lack of Mastery

The last group exhibited fundamental misunderstanding across most aspects of course content. Even after various filtering methods, including removing students who showed diminishing involvement in the course (perhaps due to their registered grading policy), these students showed displayed shortcomings in even simple programming exercises on MITx. There was a wide variation in the effort exerted, spanning from no time spent watching videos to 10+ hours. Despite various methods employed to, it was difficult to ascertain the factors driving these differences.

Chapter 4

Applications Beyond General Trends

MIT typically offers over 5000 subjects per semester. Over 100 of the courses have affiliated MITx resources that accompany coursework, providing both additional resources for students as well as a platform for hosting assignments. This chapter discusses the possibility of applying the discoveries of learning trends to these other dual platform courses. We show how our general framework can be adapted to fit the specific syllabi of other courses and also be used to uncover rich patterns and trends in a given semester or across semester offerings.

4.1 Introduction

Advances in technology have allowed for the ed-tech space to grow. As open access platforms continue to reach more learners, there is a tremendous opportunity to analyze this high resolution behavior of users all around the world. Although this introductory programming class has only a small subset of the overall demographic of all learners, the benefits of developing a metric to test the efficacy of the course and its ability to engage its learners can be great implications. The ability to parse through such rich data to find certain course material that may be aiding or hindering student success is crucial for rapid refinement of a course. This newfound capability for a course to evolve along with its students ensures that truly no student is left behind.

4.2 Longitudinal Study

To test whether the assumptions of this study preserved the integrity of the data, we chose to follow up with a representative student from each of the groups described in the previous chapter. Below, we expound the specific experiences of Anna, Bailey, Charlie, and David (anonymized) to see if our classifications were correct, and specifically how their particular interactions with the course has influenced their opinions of the programming field.

Given that we had no prior information of each student's background in the course, a general survey was sent out to obtain relevant past coursework and exposure to the field. Each of the students was enrolled during a semester between Fall 2017 and Fall 2018. Using their responses, we hoped to glean useful information that is not always captured through other avenues such as end-of-semester course evaluations. In this section, we question each student in order to address the following questions:

1. What aspects of the course encouraged or discouraged engagement with the material?

2. Were there any materials that clarified concepts? Which materials caused confusion?

4.2.1 Student A: Anna

Anna, scoring a 56 on effort, had reported previously taking AP Computer Science as a senior in high school. Having taken 6.00 her freshman spring, she did exceptionally well and has since declared computer programming as her major. She had scored in the 85th percentile in the final exam, and with consistently high marks across all assignments, received an A in the class. She attributed her grades due to prior exposure to the field, and only having to focus on studying the concepts. She was able to calmly excel at microquizzes and the coding portions of the exams since she was comfortable with foundational concepts. Although she struggled with certain areas in regard to content presented 6.0002, she used visits to office hours to help her to elucidate concepts with help from course staff.

Anna did not visit office hours frequently, only visiting office hours 8 times, with 5 being the minimum needed to fulfill checkoffs for the course. Despite the fact that she logged fewer office hour tickets than many of her peers, her overall ticket times were much longer than average. She stated that she would utilize her time there to ask lingering questions about concepts in addition to receiving checkoffs on her problem sets. Although she logged longer times in office hours, her ticket times were not considered due to the primary categorization as being used for checkoffs. Thus her additional "effort" in utilizing office hours to learn more about concepts was essentially uncounted despite spending 1.3x longer on average at office hours. Although she performed well in the class, this case where a student combines visits for checkoffs with additional help is discounted and can hurt their effort statistic. This gives rise to an opportunity to improve office hour tickets to include a closing survey for TAs/LAs to reflect the help they provided (e.g., checkoff, concept clarification, problem set help, etc.) to better track the services provided by course staff and track effort.

4.2.2 Student B: Bailey

Bailey, scoring a 72 on effort, had no extensive exposure to programming before taking 6.00 her sophomore fall. She took this course since it fulfilled a requirement for her major, however did not categorize herself as someone was initially interested in programming. She spent a lot of time on MITx working on practice problems, however did not spend time to watch the provided videos. Instead, she would go to YouTube to watch the exact video, or a video on the same topic. She did not perform exceedingly well on microquizzes, doing average or below on all three, however demonstrated an understanding of concepts through her checkoff scores as well as her grade on the paper portion of the exams.

Although she would regard herself as someone who understands the programming concepts gone over in class, she gets flustered when having to code under a time crunch. Although she did not spend as much time on MITx, she did attend office hours as frequently as higher "effort" scoring peers for help on problem sets and class concepts. She has since taken additional coding classes since taking this course and has proceeded to do well, attributing her success to her ability to retain core and fundamental coding concepts. She believes that additional exposure to coding resources and exercises may have helped her become more comfortable with timed assignments. Specifically, additional MITx exercises with a timed component would have been a useful additional resource that she believes she would have taken advantage of to help her in this area and would have contributed to a higher "effort" score as she resorted to other third party sites to expose her to additional coding problems.

4.2.3 Student C: Charles

Charles, scoring a 22 on effort, was a unique case as he was a Computer Science major who took the course as a senior due to the curriculum update. He generally put minimal time into completing exercises, forgetting to complete a portion of the mandatory finger exercises and never attempting the optional problems. Being more than comfortable with the programming concepts in the course, he would only show up for lectures for microquizzes, and for only the first two since he received perfect scores (and could subsequently "drop" the third). He admits that he struggled during the paper portion of the exam on questions that tested statistics, machine learning, and other concepts that were not directly tied to programming.

He recognized he could have performed a lot better in the course had he been more attentive to the mandatory parts of the course, such as completing all the finger exercises and getting checkoff scores for each problem set. As he failed to complete portions of the course that took place outside of the classroom, namely pset checkoffs and finger exercises, and was on the border of a letter grade, an "effort" statistic would have validated his lack of engagement with the course, although contrary to his high marks in certain areas of the course.

Student	Effort	Avg. Sessions/Week	Total MITx Session Time (min)	Office Hour Tickets
Anna	56	2.3	108	8
Bailey	72	2.7	219	11
Charles	22	0.8	39	7
David	18	0.6	149	9

Table 4.1: Sample Student Archetype Data Shows data from four students who took the course between Fall 2017 and Fall 2018. Shows differences in their overall "effort" put into the course segregated by student archetype.

4.2.4 Student D: David

David, scoring a 18 on effort, was a freshman who opted to explore computer programming his first semester enrolled at MIT. He ended up completing the course, however accepted a "No Record" grade on his transcript. In our discussion, I learned that he had wanted to take the course to gain exposure to the field so solidify his intention in pursuing a major in the field. However, halfway through the course when he realized that his grades were not up to passing standards, still chose to complete the course to have exposure to the material while intending to take the course again the following semester. He acknowledges that he did not put enough effort into the course to learn the content the first time around, which he attributes to adjusting to MIT, both socially and academically. In this case, his low "effort" statistic would not warrant additional consideration in final grading.

4.2.5 Newfound Discoveries

After interviewing these students, we had greater insight into the specific experiences of these students and their admirations and grievances. Their comments about the course structure were overall consistent, citing certain lectures being less practical in its application to completing problem sets. In general, they said there was a slight disconnect in the concepts covered and the questions asked in microquizzes and exams. However, only one of these four students regularly went to hosted recitations, or checked associated notes, where they could find content more relevant to those aspects of the course. Bailey stated that although recitation code was helpful to review, the overall notes were poorly organized and did not always pertain to the associated code.

4.3 Suggested Improvements

Below, we outline specific aspects of the course that if updated, would present the greatest potential to help students. The primary motivation factor in this report was to identify few ways to improve the course and its grading schema.

4.3.1 Centrality of Content

Students were generally frustrated by the abundance of relevant links and platforms for the course, and lack of a centralized location to access these tools. Although Stellar contained the links to all of the sites, students had know the specific platform to navigate to to find their content of interest.

For example, MITx is the platform with finger exercises and videos going over concepts for every lecture. However, the specific lecture material (PowerPoint and accompanying code), is found under the Materials section of Stellar, with no link to the MITx or vice versa. In addition, students also expressed a desire for additional resources developed by the course rather than linkage to third party sites to ensure development of skills directly pertinent to the course.

Implementation

For future iterations of the course, it may be easier to post in-class lecture content directly on MITx for students to download. There, they could find an amalgamate of all relevant material, as opposed to having to navigate various platforms to find the fullness of resources. In addition, recitation notes could similarly added to MITx so that students can find those materials while studying material from lecture. 3 of the 4 students surveyed were unaware of the fact that recitation content was posted, and even if aware, would not know where to begin looking for it. The simple transition of moving the Stellar Material content to MITx, students would be able to find everything pertaining to lectures under one umbrella.

4.3.2 Videos

The students request additional videos that showed a practical application of concepts in writing code. Specifically, given some problem, showing the computational thinking behind crafting an implementation. In addition, they cited the usefulness of examples going over common mistakes and edge cases. In addition, since certain students have trouble following the lecture during scheduled times (if the content is too fast-paced), they would like a greater degree of consistency between lecture slides and the associated videos.

Implementation

Given the low engagement of students with current videos (whether it be due to the availability of similar videos on the web, or other reasons), it may be more feasible to instead incorporate this suggestion into the development of optional coding exercises. In addition, instead of a typical problem that has students draft up a solution addressing all cases, it would be useful to break up a problem into reasonable sub-problems that tackle specific cases. In this way, students are introduced a method to decompose a problem and tackle it using a clear, directed approach. It seems that students are mostly concerned with the discrepancy between taught concepts and its relevant application, which is best addressed through feedback from problems. Since this would be done on MITx, which has a corrective feedback mechanism, it would ensure that students get immediate feedback on fundamental misunderstandings of certain concepts.

4.4 Future Applications

We hope that the "effort" statistic can be used to assess students who are on the border when assigning final grades. This would minimize bias in final assessment, so that students who are specially considered are not just those personally known by a current TA. In this way, we can see if a student's exerted work into the course could justify a better grade; we thus move away from a meritocracy schema of grading and allow for some leniency in certain misunderstandings of concepts as long as they have generally demonstrated improvement and basic literacy in programming.

As we expect the number of students electing to take computer science courses with the newly endowed Schwarzman College of Computing, the relevance for improving the introductory programming course. The ability to parse through student interaction data to output students who show deserve a second consideration in final grade due to their effort would demonstrate the course's dedication to student engagement and subsequent acknowledgement of said work.

Bibliography

[1] Hermann ebbinghaus, Apr 2018.

[2] Elaine Allen, Jeff Seaman, and Richard Garrett. Blending in: The extent and promise of blended education in the united states. *The Sloan Consortium*, 2007.

[3] Xin Chen, Lori Breslow, and Jennifer Deboer. Analyzing productive learning behaviors for students using immediate corrective feedback in a blended learning environment. *Computers Education*, 117:59âĂŞ74, 2018.

[4] Elaine D Collins. The research planning group for california community colleges. *The Research Planning Group for California Community Colleges*, 2013.

[5] Victoria Crisp and Christine Ward. The development of a formative scenario-based computer assisted assessment tool in psychology for teachers: The pepcaa project. 2008.

[6] Iam Jaebi. Disadvantages of traditional classroom training. *Classroom*, 2019.

[7] Sarath Nonis and Gail Hudson. Academic performance of college students: Influence of time spent studying and working. *The Journal of Education for Business*, 81:151–159, 01 2006.