POLITECNICO DI TORINO

Master's Degree in DATA SCIENCE and ENGINEERING

Master's Degree Thesis

Machine Learning for Student Performance Prediction

Supervisors

Prof. Paolo Giaccone

Prof. Paolo Manfredi

Candidate

Hamed Goldoust

Academic Year 2025-2026

Acknowledgements

I want to thank my supervisors, Professor Paolo Giaccone and Professor Paolo Manfredi, for all the help, patience, and knowledge they have given me during this research journey. They always supported me and gave me good advice, which all helped me a lot with writing this thesis.

Thank you to everyone at the Polytechnic of Turin who helped me with this research. In particular, I want to thank the Electronics and Telecommunications Department (DET).

Finally, I want to thank my family from the bottom of my heart for always being there for me, getting me, and supporting me while I trained.

Abstract

This thesis aims at studying student performance prediction for different kinds of course implementations and using machine learning techniques applied to behavioral data collected from a Moodle-based check-in/checkout system to support self-regulated learning. Modules 1 and 2 targeted students who first scored under 80, and Modules 3 and 4 targeted those who initially scored under 70 . This multi-module framework allowed rigorous examination of how predictive factors change across different performance cutoffs and instructional models operating in Moodle Learning Management System context. The technology infrastructure for this research is the Moodle-based check-in/checkout system, which becomes a structured two-phase assessment system in this work for recording diagnostic and summative outcomes together with the automatic logging of detailed behavioral data during learning. The check-in is given at the outset of each module to develop the level of knowledge of the underlying concepts and to identify students who need more focused help, with the check-out designed to track learning gains and intervention impact after students have interacted with materials, experienced lectures, and tried practice. This longitudinal model allows the computation of a measure of improvement efficiency, an indicator of the extent to which students are able to close the gap in early performance, which was found to be the strongest predictor of success for at-risk students in all four modules. Moodle platform's comprehensive data logging capabilities captured multiple dimensions of student engagement including normalized timing metrics relative to cohort behavior, lecture video viewing patterns, quiz attempt frequencies and time investments, resource access patterns, and task completion strategies.

This study has strong implications for educational data mining in that it presents how machine learning can be merged with everyday instruction to ensure interventions are proactive and supportive. The four-module design presented compelling evidence to support that behavioral predictors discovered by the Moodle-based Learning Analytics community are generalizable over different course adoptions and students, so that they are deserving to be adopted by an institution. That improvement efficiency is the strongest predictor for at-risk students has direct implications for applying the model in an educational context regarding intervention design, highlighting that the intervention should aim at promoting students' systematic way to fill in knowledge gaps rather than to insert more content or time.

A machine learning method was developed and tested with behavioral data from a moodle based check in/out system to support self-regulated learning. We organized systematic comparison of both regression and classification models. This study adds to the field of educational data mining to enable evidence based practice for achieving the maximum learning achievement in higher education students, the addition of machine learning in daily instructional practices. This study shows that machine learning models trained on

Moodle-derived behavioral data may represent viable, transferable methods for the prediction of student performance and the identification of where, when, and how evidence-based interventions should be implemented in a range of higher education settings to foster learning success.

Contents

Chapter 1

Introduction

1.1 Overview of the thesis

There has been a problem in engineering education for a long time: how to tell early on how well students will do so that effective interventions can be made. This thesis looks at that problem. Students have long had a problem with procrastination, which means putting off studying until right before tests. This makes it harder for them to learn and improve their skills. For students who are having trouble with the material at first, this problem tends to get worse as they go through school.

To solve this problem, this study comes up with a new way to use educational technology, data analytics, and machine learning together. Its main part is a Moodle-based check-in/checkout system that manages how students talk to each other during a course and collects a lot of behavioral data that can be used to guess how well students will do in school. A number of engineering classes at the Polytechnic of Turin have used and tested the framework. One of these classes focused on circuit theory as part of their biomedical engineering program.

This thesis is easiest to understand if you think of it as making a system that can tell when a student is having trouble in school. This is true whether you are a parent or an engineer reading it. You can use this system to find out which students might need help and also to find out why they're having trouble and what kind of help would work best for them. This system is meant to help education in the same way that modern healthcare does by using early detection to stop problems before they happen.

1.2 The Addressed scenario and problem

This dissertation looks at how hard it is to learn engineering as an undergraduate, especially in courses that use sequential, cumulative learning, where small problems can get worse over time. When this happens, a lot of problems come up: students put off studying until right before tests, which means they don't learn much and don't improve

their skills; teachers can't always tell which students are having trouble early enough to help them properly; educational support usually uses one-size-fits-all methods that don't work for each student; and even though research into education has come a long way, we still don't know which behavioral indicators are the best at predicting success in online and blended learning environments. These issues are especially bad in engineering education because ideas build on top of each other and don't understand things at the start can make them harder to understand later on. A structured learning environment is used to get students to participate regularly, and advanced machine learning methods are used to find the most important factors that show whether a student will do well or poorly in school. This thesis takes a new approach to these problems. By making this integrated system, the researchers hope to change how teachers find students who are at risk and plan how to help them. Instead of waiting for problems to happen, teachers will be able to offer proactive, individualized help based on early behavioral indicators that can accurately predict how well a student will do in school.

1.3 Important ideas and how they can be used

There are many good reasons to believe that the study is interesting and useful.

What this means for teaching and learning: Previous research found that 70–95% of college students use bad study methods, and about half of those students said their grades went down as a result. This study has the potential to make a big difference in education by fixing these problems.

The study's ability to predict things helps teachers find students who are at risk with a high degree of accuracy (R^2 values above 0.98 for students who are having trouble). This makes it possible to start targeted interventions early in the learning process, when they work best.

Advances in learning analytics: Until recently, researchers in the field focused more on engagement metrics, like how often students logged in, than on efficiency and time-related aspects of student behavior. The results help us understand how timing strategies, better efficiency, and school success are all connected.

What the results mean for education: They challenge the common belief that having natural talent is the most important thing for doing well in school. But they make evidence-based suggestions for how to teach by stressing the importance of metacognitive skills and the learning process.

This method is easy to copy and use in other schools because it is based on Moodle. Thousands of schools around the world use Moodle, a free and open-source learning management system.

The results have big effects on how higher education assesses, gives feedback, and supports students, not just on a course-by-course basis.

1.4 Main Points of View

These are the main points of this thesis:

1. **Setting up a structured way to check in and out:** Moodle uses a full educational framework to get students to participate often, give quick feedback, and collect detailed data on their behavior that can be used for analysis.
2. **A machine learning framework for educational data** is a planned way to handle, study, and learn from educational data. It includes feature engineering that is unique to educational settings, strict preprocessing methods, and checking out how well different prediction methods work.
3. **Finding the most important performance predictors:** Studies have shown that improvement efficiency is the most important factor for students who are having a hard time, while strategies for timing assessments work for all student groups.
4. **Comparative analysis of prediction methods:** When you look closely at regression and classification methods across a number of datasets, you can see that classification is better for making decisions in education while regression may give slightly more accurate numbers.
5. **Resubstitution analysis is used to prove generalization.** This is the first time resubstitution analysis has been used to make educational predictions. It shows that linear models are always better at generalizing than more complicated ones, even if they aren't as good at training.
6. **Step-by-step instructions for educational interventions:** These are suggestions for targeted interventions based on specific behavioral indicators. They show how predictive analytics can be used in the real world of education.

All of these additions help us learn more about how to use learning analytics in higher education and how to predict how well students will do in school.

1.5 Methodology

To figure out how to predict how well students will do, this study takes a thorough, multifaceted approach that includes designing educational systems, collecting data, machine learning, and testing the predictions in the real world. The first step of the project is to carefully plan and build a check-in and check-out system based on Moodle that uses educational ideas such as formative assessment, targeted remediation, immediate feedback, distributed practice, mastery learning, and self-regulation. Regular tests are part of this educational framework, which not only helps students learn but also builds a strong system for collecting data that tracks detailed behavioral metrics such as time factors, engagement measures, and performance indicators. The process of targeted feature engineering turns raw behavioral data into useful educational metrics. It does this by creating features that show how well students are doing, how they study, and how much better they could do. After that, a well-organized machine learning pipeline was

made. To make sure the results were correct, the data had to be preprocessed (to get rid of outliers and scale features), the model had to be trained with hyperparameter optimization, and the results had to be strictly evaluated using complementary metrics. For the tests, we used four different course datasets. This was a different check-in/checkout system for each dataset. This gave us a chance to check our results with students from a range of school types and age groups. A lot of different analysis methods were used. For example, regression and classification were compared to see which one was better at predicting performance. Different algorithms, from linear models to tree-based methods, were tested to see how well the model could generalize. Finally, feature importance analysis was used to find the most important predictors. Last but not least, the real-life results were linked to educational theory. This helped create a conceptual framework that explains the prediction patterns seen and what they mean for teaching and learning. This method makes sure that the results are useful in the classroom and valid from a scientific point of view. The results make sense from a statistical point of view and can be used in real classrooms.

1.6 Thesis organization

The thesis has five parts. There are different parts of the research in each one:

- **Chapter 1: Introduction:** This chapter gives an overview of the thesis by describing the scenario and problem that is being studied, the work's importance and usefulness in real life, the main contributions, the method, and how the thesis is organized.
- **Chapter 2: Creating and putting into action a self-regulated learning system based on Moodle.** That document talks about the course's structure and topics, the ideas behind the check-in/checkout system, how students and teachers use it, how it works in Moodle, and how to collect information. This chapter lays the groundwork for research in the fields of technology and education.
- **Chapter 3: A plan for using machine learning to help students do better in school.** Prediction is a kind of computer program that looks at school data. It talks about the machine learning pipeline, how to prepare data, how to pick and train models, how to judge frameworks, and how to find out which features are the most important. This chapter lays the groundwork for the techniques of predictive analytics.
- **Chapter 4: Talks about the results of experiments and how to look at predictive models.** It also shows what happened when the framework was used in the real world on four different course datasets. It has all the results for regression and classification models, as well as resubstitution analysis to see if the results can be used in other situations, rankings of feature importance, comparisons between datasets, and an explanation of what the results mean. This chapter shows that the conclusions are correct.
- **Chapter 5: A Brief Overview and Some Thoughts** gives a summary of the

results, explains what they mean for education as a whole, and offers suggestions for how to use them and do more research if necessary. It talks about what the study means in terms of theory, how it can be used in schools, what its flaws are, and some exciting ways to keep working on it. This chapter talks about what the results from the real world mean for education as a whole.

1.7 Conclusion

The first chapter gave a brief summary of the thesis. They talked about what it is, what it means, and how it works. It's hard to help students because you don't know how well they'll do at first. This study takes a different approach to the problem by combining machine learning and educational technology.

In the next chapters, we will talk about the technical and educational parts of this work. The framework they give you will not only help you see the future, but it will also make you think about what it really means to do well in school.

This thesis is useful for both learning analytics and engineering education because it was well-planned, well-analyzed, and easy to understand. It could be helpful in ways that weren't talked about in class.

Chapter 2

Design and Implementation of a Moodle-Based Self-Regulated Learning System

The Moodle learning management system (LMS) uses a self-learning framework to solve a big problem in engineering education: students often put off studying until just before tests.This procrastination makes learning less effective and raises the chances of developing poor skills, which makes it harder to learn more advanced topics. This issue worsens as students' progress through their education.This chapter talks about a fully asynchronous online remedial course that is meant to help students who are already taking regular engineering classes. The system was originally developed for circuit theory courses in the bachelor program of biomedical engineering at Polytechnic of Turin, but its principles are broadly applicable across engineering disciplines and beyond.

2.1 Overview of the educational system

There are two main goals for the framework. The goal is to help students stop putting things off, practice more often and consistently throughout the course, and develop study habits that will help them in other classes as well. From the teacher's point of view, the framework aims to encourage students to study regularly, help them learn faster through regular testing and feedback—thereby raising overall scores and success rates on final exams—and make it easier to find students who might be having trouble so that they can get help right away. The check-in/check-out system is the most important new idea in this framework. It gives teachers a structured way to evaluate students by setting up regular touchpoints throughout the learning process. This method gives both students and teachers regular feedback and encourages them to keep using the course materials.

2.2 Structure and topics of the course

The Moodle platform is used to run the remedial course. Students can work on it at their own pace, and it covers the same topics as regular class lectures. It is divided into weekly modules. There were 13 modules made for the circuit theory implementation. Each one covered about a week's worth of lecture material and focused on basic electrical engineering ideas. The modular approach has many advantages: clear division of course material into manageable parts; each module adds to what you already know to help you learn new things; you can take tests on a regular basis and get feedback right away; you can focus on the areas where you need help. The system is particularly well-suited for engineering topics that build sequentially on fundamental concepts, require consistent practice for mastery, involve problem-solving with multiple solution approaches, and benefit from visual and interactive demonstrations.

For the circuit theory implementation, modules covered topics including:

- Basic circuit elements and Kirchhoff's laws.
- Resistive circuits and analysis techniques.
- Thévenin and Norton equivalents.
- Capacitors and inductors.
- First and second-order dynamic circuits.
- AC circuit analysis.
- Frequency response.
- Power in AC circuits.
- Three-phase systems.

2.2.1 Structure of participation

It is up to the students whether they want to join the remedial program, but students who had trouble with the material before are especially encouraged to do so. A bonus point system is used to encourage participation:

- Up to 4 points (out of a possible 30 in the Italian grading system) are given based on how well you do.
- Points are given out per module in a tiered system:
  - **I**f you pass the check-in test, you get full credit ($\frac{4}{13}$ points per module).
  - **I**f you fail the check-in but pass the check-out on the first try, you get partial credit ($\frac{2}{3}$ of full points).
  - **I**f you fail the check-in but pass the check-out after trying several times, you get less credit ($\frac{1}{3}$ of full points).

This incentive system rewards both getting ready for the test and fixing mistakes, which encourages students to keep working with the material instead of putting off studying until the test.

2.3 The design of the check-in/check-out system

2.3.1 Educational principles of the system

The check-in/check-out system is grounded in educational principles that have been demonstrated by research to be effective:

1. **Formative assessment:** The check-in tests help students identify areas for improvement by revealing knowledge gaps without penalizing them with harsh grades.
2. **Targeted remediation:** Remediation is focused on areas of demonstrated need, as learning materials are provided based solely on check-in performance.
3. **Immediate feedback:** The system provides students with instant feedback on their test results, enabling them to quickly recognize and correct mistakes and misunderstandings.
4. **Distributed practice:** By encouraging engagement with multiple modules over the course of the semester, the system promotes spaced learning rather than last-minute cramming.
5. **Mastery learning:** The framework ensures that students fully grasp foundational concepts before progressing to more complex material.
6. **Self-regulation:** The system supports the development of students' self-regulated learning skills by offering clear indicators of both progress and areas requiring further effort.

2.3.2 Structure of the parts

There are three main parts to each module that work together to make learning easy:

1. **Check-in test:** A first test of the student's basic knowledge of the subject. The outcomes determine whether a student needs to complete the relevant learning units.
2. **Learning units:** Students who don't do well enough on the check-in exam can access the course materials. Among these units are:
    - Short videos that explain important theoretical ideas, as well as step-by-step instructions.
    - worked examples for how to solve problems.
    - Moodle tests with practice questions that provide immediate feedback.

Moodle's conditional activity features only let students access certain units, which helps them focus on their weak points.

3. **Check-out test:** This is a follow-up test that is meant to see how much a student has learned after they have finished the learning units. It is set up the same way as the check-in test.

This three-part framework makes sure that everyone has the same learning goals, uses the same standards for evaluation, and meets the needs of each student.

2.3.3 The philosophy behind the design of assessment

The assessments in the check-in/check-out system must follow some design rules:

1. **Equivalence:** Since check-in and check-out tests assess the same concepts using different problems of comparable difficulty, learning gains can be quantified directly.
2. **Randomization:** The order and parameters of the questions are changed at random to make it harder for people to share answers while keeping the difficulty level the same.
3. **Time limits:** Each test has a 30 - minute time limit, which makes people work harder and makes the tests feel like real tests.
4. **Immediate feedback:** Students get feedback on their work right away, along with the right answers and explanations for any mistakes they might make.
5. **Multiple attempt structure:** Check-in tests only let you try to make a baseline once, but check-out tests let you try as many times as you want, which encourages persistence and mastery.
6. **Balanced difficulty:** The questions range from simple application of concepts to more difficult problem-solving, but they are still doable.

2.4 The student's perspective

For the student, using the system means doing a set of planned activities that help them learn on their own. The first step is initial engagement, which is when students sign up for the remedial course on Moodle. When they sign up, they can see the whole module structure, but some of the content may not be visible until certain conditions are met. At the beginning of each module, students learn about the topic, the goals of the lesson, and what they need to do to pass the test that goes with it. They also learn when the tests will be given, which is usually around the same time as the regular course material.

The check-in assessment experience begins with a message letting students know that a check-in test is available and telling them how long they have to take it, which is usually 48 hours. Students start the test when they are ready, which starts a 30-minute countdown. The question interface shows a list of random questions that are related to the module topic. These questions often have more than one correct answer and include

math problems, circuit analysis tasks, and exercises to find parameters. Students get immediate grading and feedback on wrong answers after they submit their work. This helps them figure out what they need to work on. A student gets full credit for the module if they score higher than 80. They can then review the material whenever they want. People who don't meet the requirement must finish the learning units before they can take the check-out assessment.

Students who didn't pass the check-in test are expected to do a variety of things during the learning period. These include watching instructional videos that explain concepts that were not understood, going over worked examples that show how to solve problems for questions that were missed, doing ungraded practice activities with instant feedback, keeping track of their progress with Moodle's tools, and using practice problems to see if they are ready for the final test.

Moodle's completion system keeps track of all the required learning activities that have been done, so the check-out assessment experience is set up to make sure they have all been done. Then, within 30 minutes, the students take the check-out test. They can look over their feedback and try to raise their score up to two more times if they need to. This process lets students see how their check-out performance compares to their initial check-in, which gives them a sense of how far they've come.

This system has a number of long-term engagement benefits that will last throughout the semester. Students get regular feedback on how well they understand the material, structured help with areas that need more work, and visual progress tracking through Moodle dashboards. The system encourages students to stay engaged, which helps keep their knowledge fresh between course sections. It also helps them get ready for final exams by giving them regular practice.

2.5 Professor's perspective

The professor has high hopes for the system's ability to monitor student progress, identify issues, and inform pedagogical decisions using hard data.

Module preparation for courses is the responsibility of the professor, who is responsible for developing and organizing all course learning materials and assessments such that they complement one another and the course as a whole. In order to ensure that the test remains valid, they create question banks that are sufficiently diverse and randomized. Teachers can also use conditional release settings to dictate when students have access to course materials and exams depending on their check-in performance. In addition, they monitor due dates to guarantee that module release and availability periods match up with the usual course pace.

You can monitor each and every one of your kids with this technique. By keeping track of which students have completed each course and test, professors can gauge their level of engagement. Using performance dashboards, students can easily see their test scores from the beginning and end of the semester. Teachers can observe how frequently and effectively students use learning materials, such as the number of times they view videos

or take practice exams, with the help of engagement analytics. By comparing students' performance on the first and last assessments, teachers can gauge their level of learning with the help of progress comparison tools. By highlighting children who are struggling or performing worse than normal, the approach aids in the identification of those pupils who may benefit from further support.

Data from the system also reveals novel approaches to enhancing education. Professors can identify challenging ideas and subjects where many students struggle by analyzing trends in assessment outcomes. They can compare the efficacy of various learning resources to determine which ones are most beneficial to students' education. In order to guarantee that exams are equitable, sufficiently challenging, and relevant to the course objectives, question quality analysis is employed. Based on students' progress, the system also allows teachers to rearrange the lessons and modify the rules for moving through them. In the long run, this aids the learning route. Lastly, the information gathered from the remedial course can be utilized to enhance and harmonize the content of normal classes, particularly in regards to clarifying challenging concepts.

2.6 Information about how to set up moodle

2.6.1 Setting up the platform

The implementation makes use of the moodle learning management system and takes advantage of some of its unique features:

1. **Course format:** Weekly format with topics that can be collapsed, which makes it easy to see how time is organized while keeping the visual simplicity.
2. **Activity completion:** : Setting up criteria for completing all resources and activities, which lets content items have prerequisite relationships with each other.
3. **Conditional access:** involves setting up rules that decide when students can access certain activities based on how well they did on previous activities, whether they finished the required activities, when they are available, and requirements for group membership.
4. **Grade calculations:** Custom grade formulas that use the bonus point system based on how well students check in and out.
5. **Groups and groupings:** Putting students into groups based on their regular class sections so that they can get targeted content and reports.

2.6.2 Configuring the quiz module

Using certain technical settings, the check-in and check-out exams make use of moodle's quiz module:

1. **Question banks:** In Moodle, questions are grouped into categories and subcategories, like a tree, instead of being in a flat list. For example, "Resistive Circuits,"

"Operational Amplifiers," and "AC Analysis" are all top-level categories that cover major topics in the course. "Basic," "Intermediate," and "Advanced" are all subcategories that break these down by level of difficulty. This hierarchical structure makes it easy for teachers to find specific content areas when making tests, make sure that all course topics are covered in the right amounts, balance the difficulty levels of each test, and focus on the areas where students usually have trouble.

There are usually at least 5–10 questions in each category that test the same idea or skill. This means that each topic and difficulty level has multiple questions. This number is very important because it lets the system randomly give different questions to different students, making sure that even when they are testing the same idea, each student gets a different problem. It also means that students who take the check-out test more than once will see different questions each time, which makes it less useful for students to share specific answers. For instance, instead of just one question about Thévenin equivalents, the system could have 8–10 different circuit configurations that all test the same idea but with different layouts and parameters.

Also, metadata tagging is used to choose questions, which means adding more than just the category to each question. These tags could include the specific learning goal being addressed (like "Apply Kirchhoff's Current Law"), the estimated time to finish (like "5 minutes" or "10 minutes"), the question format (like "Multiple choice," "Numerical," or "Circuit analysis"), the cognitive level according to Bloom's taxonomy (like "Remember," "Apply," or "Analyze"), and any common misunderstanding being addressed (like "Voltage source behavior").

2. **Random selection parameters:** In order to prevent students from cheating or sharing their answers, the Moodle quiz module employs sophisticated randomization techniques to guarantee that every student has an independent testing experience. Selecting questions at random from a predetermined list of categories is one approach. The system automatically selects questions from the question bank's categorized categories whenever a student begins an assessment during check-in or checkout. On quizzes, teachers don't hand-pick questions for each student. Rather, they programmed the test such that it would randomly select a specific amount of questions from each applicable area. So, they may provide the following: "Choose three questions at random from the 'Basic Circuits' category and two from the 'Thévenin Equivalents' category." Consequently, while the questions given to each student may vary in terms of specificity, they all cover the same ground and range in difficulty. By way of illustration, the "Operational Amplifiers" category could have questions 1, 5, and 8 for Student A, and questions 2, 4, and 9 for Student B.

   To guarantee that every student receives an equitable grade, Moodle employs algorithms. Several factors are considered by these algorithms before selecting questions at random. As a result, you can be certain that every quiz is well-balanced in terms of difficulty, covers all the necessary material, has comparable total projected completion times, and uses a consistent mix of question forms (such as numerical or

multiple choice questions) throughout. The objective is to create exams that are challenging enough to cover the same ground for all students, despite the fact that their questions will be unique. This ensures that no student will receive an exceptionally easy or difficult set of questions.

Alternating the parameter values for each question at random is another sophisticated method. The format and language of the question remain the same with Moodle's "calculated question" type, but the numbers are customized for each student. Assigning values allows for real-time calculation of the correct answer. To illustrate the point, consider a scenario involving Ohm's Law. If the circuit has $12V$ and 6Ω resistance, Student A will need to determine the current, whereas Student B will face the identical task with $9V$ and 3Ω resistance. Sharing answers isn't beneficial because both students are assessed on the same concept but are asked to complete different math problems.

A robust assessment system is created by combining these randomization methods. This system prevents students from cheating by making it more difficult to share answers, ensures that all students are tested at the same level of difficulty, encourages students to demonstrate conceptual understanding rather than rote memorization, allows students to take the test multiple times with different questions each time, and provides ample practice with different problem sets.

3. **Controls for timing:**The moodle quiz module includes several controls for timing to ensure a fair and structured assessment environment. A timer sets a 30-minute time limit for each attempt, and the test is auto-submitted when time runs out. The time remaining is clearly displayed to students throughout the test, allowing them to manage their pace effectively. Additionally, attempt logs keep track of both the start and submission times for each student, providing a record of their assessment activity.

4. **Restrictions on the attempt:** There are specific restrictions on the attempt process in the moodle system. Each student is only allowed one attempt when checking in. For the checkout process, students are permitted a maximum of three attempts. When multiple attempts are allowed, the highest grade achieved across all attempts is retained as the final score.

5. **Security settings:** The moodle system incorporates several security settings to maintain assessment integrity. Questions are shuffled to discourage the sharing of answers between students, and feedback is delayed until after submission to prevent immediate sharing of solutions. Additional measures include limiting the use of JavaScript and detecting pop-up blockers in the browser, further supporting a secure and controlled testing environment.

6. **Configuring feedback:** Configuring feedback in the moodle system allows students to view correct or incorrect responses immediately upon submission. The system can provide detailed explanations of common problems with specific feedback for each question, along with general feedback that explains how to solve

similar problems. Additionally, students have the ability to review their previous attempts before making a retry, supporting ongoing learning and improvement.

2.6.3 Implementing learning materials

Several different kinds of moodle activities are used to implement the course material between tests:

The moodle system supports a variety of learning tools and resources to enhance student engagement and understanding. For video assets, an HTML5 player with playback controls and an embedded video.js player ensure a uniform viewing experience across all devices. Transcripts are available for easy access, content completion is monitored, and videos are broken down into manageable conceptual chunks for better comprehension.

Practice exams are also provided, allowing students to try as many times as they want without scoring, while still tracking completion. Full feedback is given after each attempt, and the adaptive option permits multiple tries for each question. After each attempt, students can view tips for possible solutions, aiding their learning process.

Lesson activities are designed to present content that branches based on student responses, incorporating questions to ensure attention and understanding. Progression through the material is conditional, requiring students to demonstrate comprehension before moving forward, and navigation controls are available to help them review challenging concepts.

Additionally, the platform offers a range of supplementary resources, including summarized theories in PDF format, interactive circuit simulators using H5P, and multiple methods for exploring working examples of solutions. These features collectively support diverse learning styles and reinforce key course concepts.

2.6.4 Configuring activities based on conditions

The conditional activity system in moodle is designed to adapt course content to meet the demands of students. Access limits are set so that if a student's check-in performance falls below a certain threshold, they are prompted to review the relevant learning materials. All learning materials must be completed before a student can proceed to the checkout assessment, and date limits are established to align with the normal progression of the course.

Completion tracking is set up to ensure engagement and mastery. For example, students must view at least 90% of a video for it to be marked as complete, achieve a minimum score of 80 on practice quizzes, and view all content pages within lessons for completion to be tracked. In the case of discussion-based activities, manual completion options are available.

Limitation visualization is implemented by displaying activities that are not currently

accessible as grayed out. Progress indicators show students which tasks have been completed, and visual learning paths are provided in the form of prerequisite maps. A notification system alerts users when new content becomes available.

Activity sequencing is also enforced, requiring students to progress sequentially through essential material, while offering multiple, adaptable routes through supplemental resources. Determination chains ensure that concepts are introduced in the correct order, and content is automatically unlocked once the necessary criteria are met.

2.7 Architecture for collecting data

2.7.1 Moodle logging system

The moodle logging system provides comprehensive tracking of how students interact with various components of the learning environment. System events are recorded, including the times and frequency of logins and logouts, the number of times pages are viewed, navigation patterns, and any form submissions or data entries. Quiz interaction data is also captured, detailing the start and end times for each attempt, the time spent on each question, navigation between questions, changes to answers, submission patterns, and the final answers along with their correctness.

In terms of resource engagement, the system logs events such as playing, pausing, and seeking within videos, file download activities, page scrolling, time-on-page metrics, and click patterns in interactive content. Completion events are tracked as well, with the system recording activity completion timestamps, records of criteria satisfaction, progress percentage updates, and distinctions between manual and automatic completion. This robust logging infrastructure supports both effective course management and detailed analysis of student learning behaviors.

2.7.2 Collecting source data

For every student, the system records a full suite of raw data points:

1. **Information concerning the check-in assessment:**
    - `Time_taken_checkin:` The total elapsed time from when a student started the assessment until they submitted their answers, displayed in a minutes and seconds format.(for instance, "18 mins 32 secs")
    - `Time_taken_checkin(s):` The same measurement expressed in seconds
    - `start_time_norm_checkin:` When the student started relative to the availability window
    - `Grade_checkin:` Score achieved (0–100)
    - `duration_norm_checkin:` A normalized value between 0 and 1 indicating what fraction of the maximum allowed time (30 minutes) the student actually used to complete the assessment.

- `total_quiz_attempts_checkin`: The number of submission attempts
- `total_quiz_time_checkin`: Active time answering questions
- `total_quiz_time_checkin_seconds`: Same measurement in seconds

2. **Learning period data:**
   - `lecture_sessions`: The count of distinct learning sessions where a student accessed the lecture materials, with a new session counted after 30 minutes of inactivity.
   - `complete_sessions`: Count of sessions where lecture was viewed to completion
   - `total_lecture_time`: Time spent watching throughout all sessions
   - `total_lecture_time_seconds`: Same measurement in seconds
3. **Check-out assessment data:** Check-out assessment data includes metrics that are the same as check-in, but with the "checkout" suffix. It's possible to compare performance from the beginning to the end.

2.7.3 Data extraction methods

Several technical methods are used to obtain the raw data from moodle. Database queries are executed directly on the moodle database to obtain extensive data, and stored procedures are used for regularly needed reports. Tools for analyzing log data include moodle log viewers integrated into the platform, with the ability to filter and export, as well as custom report templates for specific analysis needs. External processing involves data export to "CSV" format for external analysis, secure "ETL" (Extract, Transform, Load) pipelines, and integration with statistical analysis tools.

2.7.4 Data privacy considerations

To protect student privacy while enabling meaningful analysis, several measures are implemented. Anonymization is applied to exported data by hashing (SHA-256) the first name and last name of students. Access controls ensure that only authorized instructional workers have access to the raw data. Aggregation is used in reports for broader audiences, presenting data in an aggregated, non-identifiable form.

2.8 Important definitions of metrics

Some metrics needed clear operational definitions to make sure that measurements and interpretations were consistent:

2.8.1 Metrics for evaluation

1. **Test duration:** The time it takes from the start of the quiz to the end, which can be as long as 30 minutes. This number tells you how well you manage your time and maybe how hard the questions are. It is written down in both raw seconds and as a percentage of the most time allowed.
2. **Start time normalization:** Changed to a scale from 0 to 1
   - 0 means that you can start as soon as the test is ready.
   - 1 means starting as late as possible while still having all the time you need.
3. **Quiz attempt:** when a student sends in answers to be graded all at once. You can usually only take the check-in test once, but you can take the check-out test more than once. If a student submits answers, gets feedback, and tries again, they have tried more than once.
4. **Grade:** A score (0–100) calculated based on correct responses. This is the main way to measure how well students are doing and put them into groups.

2.8.2 Engagement metrics

1. **Lecture session:** lecture session is a time when you can watch lecture videos without any breaks that last more than 30 minutes. When a student stops working for more than 30 minutes, a new session begins when they start working again. This number shows you how often people are interested.
2. **Complete session:** A lecture session that the student viewed at least 90% of the available lecture content. This indicated to me that the student had engaged with the content and was engaged in a thorough review of the content as opposed to selectively viewing content.
3. **Total lecture time:** The total amount of time spent going over all the lecture materials in all the sessions. This number tells you how much money is being spent on all learning activities.

2.8.3 Putting students in groups

To make it easier to analyze, students are put into two groups:

1. **Below 80 group:** The below 80 group is made up of students who got less than 80 on their first check-in test. This group wants to find things that make things better, and "Grade_checkout" is the thing they are most interested in.
2. **All students group (Check-in):** The all students group (Check-in) is the full dataset that has all the students, no matter how well they did at first. This dataset gives a complete picture of all the students, from the lowest to the highest achievers. It is different from the "Below 80" or "Below 70" groups, which only look at students who are having trouble.

This classification enables more targeted analysis and potentially diverse intervention strategies based on initial performance levels.

2.9 Machine learning approaches for educational data

This section provides a high-level overview of machine learning approaches applicable to the educational data collected through the check-in/checkout system. Detailed implementation will be addressed in Chapter 3.

2.9.1 Using regression to predict performance

There are many regression methods that can be used to predict continuous performance outcomes (grades):

1. **Linear regression models:** are useful for figuring out how student behavior affects outcomes, especially when it's important to be able to understand the model and control its complexity. **Ridge Regression,** which uses L2 regularization, is great for check-in and checkout data where features might be related, like the amount of time spent on tests and the number of tries. Including more than one temporal and engagement metric helps keep predictions stable while still being easy to understand and preventing overfitting. **Lasso Regression,** which uses L1 regularization, is great for choosing features because it finds the behavioral indicators—like start time or session length—that best predict performance. It automatically sets less important features to zero, which makes models more parsimonious. This is useful when there are more features than students. **ElasticNet** uses both L1 and L2 regularization, which makes it a good choice for selecting features and shrinking coefficients. This method is especially helpful when it's important to deal with correlated features and pick the right variables, which is often the case when groups of related behaviors affect outcomes together. **ElasticNet** is better at choosing features than **Lasso** alone, especially when behavioral metrics are very similar to each other. These **linear regression** methods work best when there is a roughly linear relationship between student behaviors and outcomes and when the coefficients need to be easy to understand.

2. **Ensemble methods based on trees: Random Forest** is an ensemble approach that creates a collection of decision trees, each of which selects attributes to utilize in a random fashion among the other trees. With this strategy, non-linear correlations between engagement patterns and performance are effectively captured, which is a significant accomplishment. Continuous time measures and discrete counts of sessions are two examples of the mixed sorts of features that are frequently found in educational data. It also works well with these types of features. **Random Forest** is an effective method for dealing with outliers, which are frequently seen in educational settings where certain students may study in unconventional ways. In addition to this, it provides trustworthy measurements of the value of features, which can assist you in locating crucial behavioral signs. On the other hand, **Gradient Boosting** is a method that trains trees in a sequential manner, with each

new tree correcting the errors that were made by the tree that came before it. This approach is excellent for identifying minute trends in the manner in which the timing of engagement influences the results of performance. It is particularly effective at replicating the intricate relationship that exists between starting knowledge (as judged by check-in) and end performance (as measured by checkout). The **Gradient Boosting** technique can also be used to discover interaction effects, which are situations in which particular combinations of behaviors, such as beginning early and participating in numerous sessions, have an effect on outcomes simultaneously. Due to the fact that it typically produces superior outcomes compared to single models, it is an excellent option for the analysis of educational data. In order to discover non-linear relationships and interactions between features, it is not necessary to provide **Random Forest** or **Gradient Boosting** with instructions on how to proceed.

3. **Support vector regression: (SVR)** is a good way to describe datasets that are about average size and have complex, non-linear patterns. In order to do this, kernel functions are used to move data into higher dimensions. There may be a complicated link between how people interact with information over time and how well they learn. **SVR** is a good way to deal with this. Not only that, but it doesn't change much depending on how many variables are in the feature space. This makes it easy to keep track of many behavioral measures. This method can correctly record performance patterns even if the link between what was done in the study and what happened in the end is not linear or has threshold effects instead of clear linear trends. **SVR** is helpful when the connection between what students do and how well they do is difficult and hard to model with simpler tools. This is because it makes the connection easier to model.

2.9.2 Classification approaches

For categorical outcome prediction (such as pass/fail or performance categories):

1. **Logistic regression:** With a probabilistic view, binary or multinomial classification models give coefficients that are simple to understand and show how features change the chance of a result. It's easy to see how these models show how different actions can change your chances of passing a check-out test. Also, they can make risk scores that help teachers step in early when students show signs of having trouble staying interested. They make things easy to understand while telling you what actions are most likely to cause certain results. Because of this, they are very useful for finding out if students will meet the needs for conditional pathway assignment.

2. **Tree-based classifiers: Random Forest** and **Gradient Boosting** algorithms are used to make the system more accurate while still being easy to understand. Because educational choices are made in a hierarchical way, these methods allow for different intervention paths based on how behaviors are observed together. They can also spot important boundaries in behavior patterns that separate students who are doing well from those who might be having trouble. Ensemble methods, like

Random Forest and **Gradient Boosting**, give better accuracy while still letting you get meaningful choice boundaries that you can act on.

3. **Support vector machines: (SVM)** classifiers are helpful for finding the best lines that separate different levels of performance. These classifiers are also flexible because you can choose the kernel for boundaries that aren't straight. Not only do they help you focus on the boundary cases where help would be most useful, but they also help you figure out the right way to separate the different student performance groups. Support vector machines are very useful when it's hard to tell the difference between students who pass and those who fail because they can handle complicated and non-linear relationships between engagement patterns and categorical outcomes.

2.9.3 Clustering for finding patterns

K-means clustering is useful for two reasons: it helps you find similar patterns of activity and it shows you how each student acts differently. It was easy to put students into groups based on how they did on the check-in and checkout tests. We can find common ways of learning, like "early starters," "last-minute crammers," and "consistent engagers." Teachers can use this method to get data-driven profiles of each student that can help them spot patterns in how they act and plan more focused lessons without having to make groups ahead of time.

Dendrograms are often used to show how behavior patterns are connected when **Hierarchical Clustering** is used to make nested groups. This method finds the behavior patterns that are most closely related and counts how many there are. To learn more about how patterns of interaction change or connect with one another, look at clusters with different levels of detail. You can use **Hierarchical Clustering** to find connections between different ways of learning and managing your time.

DBSCAN and other types of **Density-Based Clustering** can find different kinds of groups. This is very useful for finding patterns in how people spend their time. It works well with time-based data and can find groups that other distance-based algorithms can't. It can also find strange patterns of interaction that don't fit into normal groups, just like study habits that change a lot. This method takes into account that engagement groups don't have a spherical shape in time. One of the goals of the check-in/checkout system is to look for patterns in how students spread out their schoolwork over time. This is very helpful for that.

2.9.4 Feature selection and importance

It's important to do feature selection and importance analysis to find out which behavioral signs have the most impact on how well students do in the check-in/checkout system. Each way to look at school data has its own pros and cons.

You are not tied to any one model when you use filter methods. Instead, they use statistical measures to pick traits before training the model. The linear relationship

between each feature, like `start_time_norm`, and the goal variable (grade) is measured by **correlation analysis**. This quickly finds behavioral metrics, like lecture viewing time or assessment start time, that have direct linear relationships with performance. Simple relationships can be quickly found with this method, which is easy to understand and use. The **Chi-square test** checks to see if the goal variable and categorical features don't depend on each other. Because of this, they can be used to see patterns in behavior, such as how often people turn in quizzes or do tasks that they are supposed to. An **ANOVA** test compares the averages of several groups to see if there are any factors that separate the performance groups. Statistically, this means we can be sure that behavioral measures can tell the difference between groups. It also tells you how much one measure can tell you about another. It can show both straight-line and curved connections, which makes it very useful for figuring out complicated patterns in the way kids behave.

When training models on different sets of features, **Wrapper Methods** test groups of features and pick the ones that work best. The Recursive Feature Elimination (RFE) method starts with all the features and keeps getting rid of the least important ones until it finds the smallest set of behavioral signs that can be used to guess how well a student will do. You start with no features in forward picking and add the best ones one at a time. First, you choose all the features. Then, you get rid of the ones that aren't useful. These methods focus on the most important behavioral markers, make models easier to understand, and take into account how features interact with each other.

Embedded Methods choose features as part of the process of training the model. This is the best of both the filter and wrapper ways put together. One type of regularization called L1 regularization (Lasso) makes the loss function worse by making the values of less important features equal to zero. It also automatically picks out the most important factors from a large set of behavioral signs. Features are ranked by how often and how early they are used to split in models like Random Forest and Gradient Boosting. These measures are based on trees. This gives good rankings and shows how different parts of student involvement are connected and affect each other in non-linear ways.

These ways of choosing and ranking features help find the parts of a student's behavior that best show how well they will do in the check-in/check-out system. This makes it easy to come up with tests and decide where to put interventions. Focusing on the most important things can help teachers better guide how their students learn and raise the quality of their education.

2.9.5 Metrics for evaluation and validation

To assess model performance appropriately:

1. **Regression metrics:**
    - **Root mean squared error (RMSE)**:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.

- **Mean absolute error (MAE)**:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.

- **R-squared (R^2)**:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where SS_{res} is the sum of squares of residuals, SS_{tot} is the total sum of squares, y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the mean of the actual values.

- **Relative Mean Squared Error (RelMSE)**: The **Relative Mean Squared Error (RelMSE)** is a normalized measure of the predictive accuracy of a model. It is calculated by dividing the Mean Squared Error (MSE) by the variance of the actual observed values. This normalization makes RelMSE a scale-independent metric, meaning it can be used to compare models across datasets with different scales of target variables. A lower RelMSE value indicates better predictive performance.

$$RelMSE = \frac{MSE}{\text{Var}(y)} = \frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where y_i represents the actual observed value for the i-th instance, $\hat{y}_i$ represents the predicted value for the i-th instance, $\bar{y}$ represents the mean of the actual observed values, n is the total number of observations, MSE is the Mean Squared Error, and $\text{Var}(y)$ is the variance of the actual observed values.

2. **Classification metrics:**

- **Accuracy:**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision:**

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall:**

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-score:**

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Area Under ROC Curve (AUC):** Measures the area under the Receiver Operating Characteristic curve, representing the trade-off between true positive rate and false positive rate.
- **Confusion matrix analysis:** A summary of prediction results on a classification problem, showing TP, TN, FP, FN counts.

3. **Validation strategies:**
   - K-fold cross-validation
   - Stratified sampling
   - Temporal validation for educational time series

2.10 Summary

The moodle-based check-in/check-out system is a whole new way to learn on your own that keeps students from putting things off. The system encourages regular participation, quick feedback, and data-driven decisions about how to teach by using a structured sequence of tests and conditional learning materials.

The system uses moodle's advanced features, such as conditional activities, progress tracking, and full logging, to create a learning space that meets the needs of each student. This system gathers a lot of data on how students act, how interested they are, and how well they learn.

The first results show that the people who took part did much better than the people who didn't, which had a big positive effect on how well the students did. This means that the method works well to help students study better and do better in school in general.

This system gives us a lot of useful data to use for machine learning analysis. This can help us figure out what the best way to learn is and make educational changes that are more tailored to each student. In the next chapter, we'll talk more about the machine learning tools that were made to look at this educational data and help teachers and students learn.

2.11 Related work

2.11.1 Learning management systems in education

Learning management systems (LMS) have changed the way educational content is delivered and the way student progress is tracked in higher education. Moodle is an open-source LMS that has become very popular in engineering education because it is flexible

and can be expanded. Costa et al. (2012) [5] assessed moodle's efficacy in higher education environments, emphasizing its capacity to facilitate diverse pedagogical methodologies via its modular architecture. Their work showed how moodle's plugin system lets schools change the platform to fit their specific educational needs without having to do a lot of customization.

Martín-Blas and Serrano-Fernández (2009) [17] documented how moodle could enhance physics education through interactive resources and automated assessment, specifically for engineering disciplines. They said that when traditional teaching was combined with structured online activities like those in the check-in/checkout system, students were much more interested and did better.

Gamage et al. (2019) [8] investigated the conditional access features that are important to the check-in/checkout system. They showed how adaptive content release based on how well a student did could help them learn better in math classes. Their method, like the one in this chapter, used test scores to get to the next set of materials, building on what they already knew before moving on to more difficult ideas.

2.11.2 Addressing procrastination in self-regulated learning

Procrastination is a major obstacle to academic success, especially in online and self-paced learning settings. Steel and Klingsieck (2016) [23] performed an extensive analysis of academic procrastination, estimating that 70-95% of undergraduate students procrastinate to some extent, with around 50% indicating that it adversely affects their academic performance. Their research establishes the theoretical framework for creating systems that specifically mitigate procrastination tendencies.

The check-in/checkout system's strategy for addressing procrastination corresponds with the findings of Michinov et al. (2011) [18], who discovered that the establishment of consistent deadlines during a course diminished procrastination and enhanced performance in online learning contexts. Their research revealed a substantial negative correlation between procrastination and academic performance, which could be alleviated through the implementation of structured intermediate deadlines.

Zimmerman (2015) [28] suggested a cyclical model for the growth of self-regulated learning that includes phases of forethought, performance, and self-reflection. The check-in/checkout system puts this model into action by giving each phase specific checkpoints. The check-in assessment starts forethought, the learning materials guide the performance phase, and the check-out assessment helps with self-reflection. Wong et al. (2019) [25] discovered that scaffolded methods of self-regulation were notably effective in online settings lacking conventional classroom cues.

2.11.3 Systems for formative assessment and feedback

The check-in/checkout method is based on the basic ideas of formative assessment. Black and Wiliam's (2009) [2] seminal work established that effective formative assessment entails eliciting evidence of student comprehension, delivering feedback that propels learners

forward, and engaging students as proprietors of their learning — all fundamental components of the system delineated in this chapter.

Gikandi et al. (2011) [10] examined the application of computerized formative assessment through a systematic review of online formative assessment in higher education. Their findings underscored the significance of prompt feedback, multiple attempt frameworks, and low-stakes assessment opportunities — elements explicitly integrated into the check-in/checkout design.

Hattie and Timperley's (2007) [11] important model of feedback divides feedback into four types: task, process, self-regulation, and self-level. The first three types are the best for learning. The feedback mechanisms in the check-in/checkout system concentrate on task and process feedback, offering precise guidance on misunderstandings and procedural mistakes instead of broad evaluative comments, which is in line with their suggestions for creating effective feedback.

2.11.4 Approaches to adaptive learning

Adaptive learning systems that modify content according to student performance have garnered considerable interest in educational technology research. Essa (2016) [7] explained how adaptive systems that use data could make learning more personal by looking at how well people know something and how interested they are in it. The conditional release features of the check-in/checkout system exemplify a type of adaptivity, although they are less automated compared to certain AI-driven methodologies.

Oxman and Wong (2014) [19] identified four generations of adaptive learning systems, with the check-in/checkout method most closely aligning with their "second generation" model, which employs rule-based adaptivity contingent on assessment performance. They observed that these systems offer considerable advantages over non-adaptive methods while remaining technically viable without intricate machine learning frameworks.

Van Lehn's (2011) [24] meta-analysis of tutoring systems revealed that systems offering targeted feedback and adaptive content sequencing produced results akin to individualized human tutoring. The check-in/checkout system is not a fully automated intelligent tutor, but its conditional structure shows how learning experiences can be changed based on what the learner already knows.

2.11.5 Learning analytics and educational data mining

The data collection architecture delineated in Section 2.7 facilitates educational data mining and learning analytics methodologies akin to those articulated by Romero and Ventura (2020) [22] in their extensive examination of the discipline. They emphasized the potential of utilizing data from learning management systems to analyze student behavior patterns and forecast academic results, which is a primary objective of the check-in/checkout system's analytics framework.

The system's ability to track how students behave over time is like what Cerezo et al. (2016) [4] found: that interactions with LMS content based on time were good indicators

of how well students did in school. Their examination of moodle logs indicated that both the quantity and quality of engagement, as well as the timing, were essential determinants of student success.

Lastly, the way students are grouped based on their first performance (above or below 80) is like the work of Koedinger et al. (2015) [14] on knowledge component modeling, which stresses the need to find and fill in specific knowledge gaps instead of treating all students as if they have the same needs. Their research showed that adaptive systems that change based on each person's level of knowledge work much better than systems that work for everyone.

2.11.6 Student behavior analytics

Our approach to feature engineering expands on research by Hung and Zhang (2008) [12], who discovered behavioral patterns within online learning spaces like frequency of login and length of interaction. Their research lacked the temporal normalization we use, though, which adjusts for windows of assessment availability. The efficiency features crafted in our analysis draw on work by You (2016) [27], which found time management to be an important element in online learning success. Likewise, Jo et al. (2014) [13] discovered that procrastination habits (captured in our start_time_norm features) had a significant effect on academic performance in online classes.

Our engagement features are in line with findings by Macfadyen and Dawson (2010) [16] that LMS interaction metrics are good predictors of learner success. Our approach adds granularity through features like avg_lecture_session_length, timing_strategy_change, addressing issues raised by Gašević et al. (2015) [9] regarding the context-sensitivity of learning data.

The division of our dataset into below-80 and above-80 sets represents an improvement over generic methods. This approach is in line with suggestions by Lee et al. (2016) [15], who advocated for tailored interventions using baseline performance metrics.

2.11.7 Machine learning for performance prediction

Our approach builds on a series of landmark studies on predicting academic performance. Romero et al. (2013) [21] compared different data mining methods for student performance prediction and concluded that tree-based models outperform other methods, which informed the comparison between Random Forest and Gradient Boosting models. The method of hyperparameter optimization that we employed is as recommended by Bergstra and Bengio (2012) [1], whose presentation illustrated the efficiency of grid search for model tuning. Our selection of evaluation measures RMSE, MAE, and R^2 is as recommended by conventional practices in educational data mining, based on Peña-Ayala (2014) [20].

Our tree-based model feature importance analysis follows work by Xing et al. (2015) [26], who used analogous techniques to identify most likely engagement indicators to predict online learning success. Our method, however, differs in that it divides students by early performance, enabling more specific insight.

Brooks et al. (2011) [3] pioneered the use of temporal features in prediction models in the context of student performance but largely concentrated on frequency of logins. We take it a step further by integrating standardized temporal metrics with consideration of individual variation in assessment strategies.

Chapter 3

Machine Learning Framework for Educational Performance Prediction

3.1 Introduction

This chapter talks about the machine learning method that was used to look at the course data that the Moodle sign-in and sign-out system from chapter 2 collected. It begins by talking about the educational framework and some basic ideas about how machines can learn. Then it talks about the pipeline that was used to figure out how well students would do in class and which parts of the lesson were most helpful.

There were two main goals for the machine learning strategy: To guess how well students will do on their final exams (check-in grades) based on how well they did on their first tests (less than 80/70 on check-in). To learn what makes a good first performance (a check-in score of 80/70 or higher).

We want to help both students who are having trouble and students who are doing well. For the first group, we want to give them specific help, and for the second group, we want to recognize and encourage their effective learning strategies through targeted modeling.

3.2 A brief overview of the machine learning pipeline

This study created a systematic machine learning pipeline to deal with the moodle check-in/check-out system's unique educational setting and data features. Each part was carefully made to meet these needs.

3.2.1 Building the pipeline

The following steps make up the overall design of the machine learning pipeline:

1. **Data preparation:** Getting the raw data from moodle and cleaning it up.
2. **Feature engineering:** the act of creating new features that can find important trends in education.
3. **Data preprocessing:** preparing the data for training the model, which means handling outliers and scaling features.
4. **Model training:** Teaching Multiple Regression models with Hyperparameter Optimization.
5. **Model evaluation:** Checking How Well the model Works by Using Test Data.
6. **Feature importance analysis:** Identifying the factors that most significantly influence student performance.

This pipeline is for two groups of students: all students (check-in) groups, and those who go below 80 and 70.

3.2.2 Data flow organization

The pipeline arranges the data transformations in a modular way so that they can be easily copied and changed to fit new situations.

1. **Input layer:** Part 2.7 tells you how to get raw data from moodle logs, which is the first part of the input layer.
2. **Transformation layer:** The transformation layer oversees preprocessing and feature engineering.
3. **Modeling layer:** This is where you teach the model and make small changes to its hyperparameters.
4. **Output layer:** Performance metrics and feature importance rankings

This tiered method lets you optimize each part on its own, and it also makes it easy to try out different methods at different points.

3.3 Feature engineering implementation

3.3.1 Making new features

We used the raw features from chapter 2 to create several types of derived features that help us better understand how students behave and learn:

1. **Efficiency metrics:**
    - **time_efficiency_checkin:** The ratio of start time to duration, showing how quickly students worked after starting.
    - **time_efficiency_checkout:** Similar ratio for checkout assessment.

- **avg_time_per_quiz_checkin:** Average time spent per quiz attempt
  `total_quiz_time_checkin_seconds / total_quiz_attempts_checkin`
- **avg_time_per_quiz_checkout:** Average time spent per quiz attempt in checkout
  `total_quiz_time_checkout_seconds / total_quiz_attempts_checkout`

2. **Study pattern features:**
   - **avg_lecture_session_length:** Average duration of each lecture session
     `total_lecture_time_seconds / lecture_sessions`
   - **timing_strategy_change:** Difference between normalized start times of checkout and checkin, indicating strategy adjustment.
3. **Performance improvement features:**
   - **grade_improvement_potential:** Maximum possible improvement from checkin score
     `100 - Grade_checkin`
   - **actual_grade_improvement:** Actual score improvement
     `Grade_checkout - Grade_checkin`
   - **improvement_efficiency:** Ratio of actual improvement to possible improvement
     `actual_grade_improvement / max(0.1, grade_improvement_potential)`

These derived features convert raw behavioral data into metrics that more clearly reflect the educational implications of student interaction with the system.

3.3.2 Logic for computing features

The implementation of the efficiency metrics accounts for edge cases such as zero durations or attempts by applying minimum thresholds to avoid division by zero:

- `time_efficiency = start_time_norm / max(0.1, duration_norm)`
- `avg_time_per_quiz = total_quiz_time_seconds / max(1, total_quiz_attempts)`

The average length of a lecture session is calculated as the total number of seconds spent watching lectures divided by the maximum of 1 and the number of lecture sessions:

- `avg_lecture_session_length = total_lecture_time_seconds / max(1, lecture_sessions)`

Performance improvement features are only calculated for the below-80/70 group, as they are designed to reflect learning progress:

- `grade_improvement_potential = 100 - Grade_checkin`
- `actual_grade_improvement = Grade_checkout - Grade_checkin`

- `improvement_efficiency = actual_grade_improvement / max(0.1, grade_improvement_potential)`

3.3.3 Strategy for choosing features

A methodical approach for selecting features was established:

1. **Correlation analysis:** We got rid of any features that weren't needed and kept the ones that were very similar to each other.

2. **Domain knowledge filter:** We looked at the features to see how useful they would be for learning based on learning theory and previous research.

3. **Initial model-based selection:** We first tried out the features in simple models to see how well they worked at making predictions. We didn't include them if they didn't help much.

4. **Iterative refinement:** The feature set got better over time as the model worked better and was easier to understand.

This process resulted in a targeted collection of features that achieve a balance among predictive capability, interpretability, and educational significance.

3.4 Data preprocessing methods

3.4.1 Outlier treatment implementation

As was already said, eliminating of the "outliers" was a big part of getting the data ready. It was easy to get rid of the odd ones with the winsorization method: Values greater than the 5th and 95th percentiles were found for each number-based trait. These very high and very low numbers were replaced by values at the 5th and 95th percentiles. This method was applied to each feature separately to keep the overall distribution and lessen the impact of numbers that are too high or too low. Other methods we looked at were IQR filtering and Z-score based detection. Winsorization was the best at keeping data points and making strange numbers less important. This problem with finding outliers is shown by three different approaches in figures 3.1 and 3.2 for both groups.

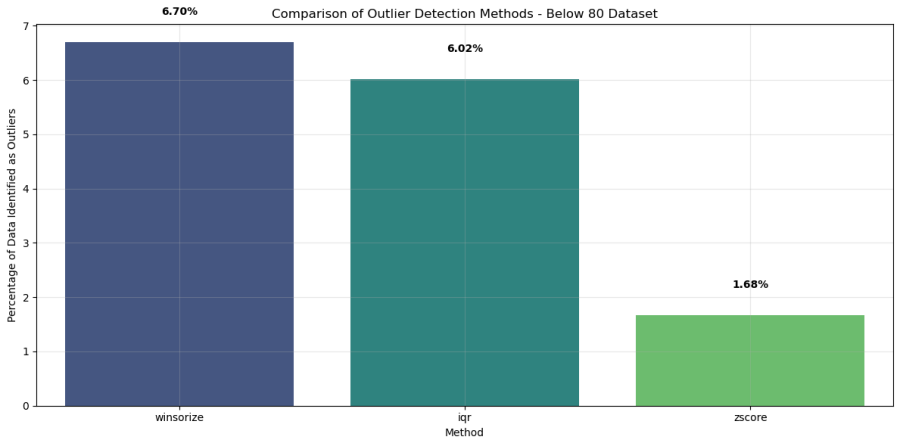

Figure 3.1. Comparison of outlier detection methods for below 80

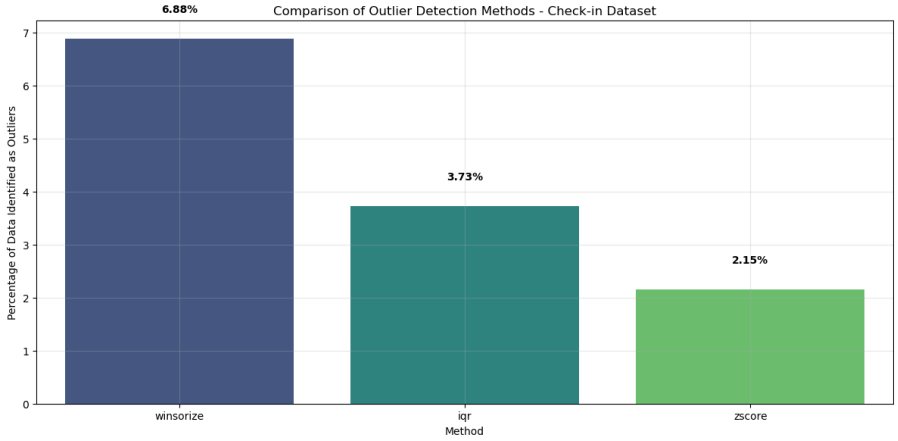

Figure 3.2. Comparison of outlier detection methods for all students(check-in)

3.4.2 Implementation of feature scaling

The analysis in this section led to the use of the "Yeo-Johnson" method with the power transformer for feature scaling: The transformer was applied to all numerical features. The Yeo-Johnson method was chosen because it can handle both positive and negative values without needing to be changed first. This method worked well to normalize the features, no matter what shape they were originally in. This scaling method worked especially well for the temporal features, like normalized start times and durations, because people tend to put things off, which made their distributions skewed. We finally chose power transformer ('Yeo-Johnson'). As shown in figures 3.3 and 3.4, the metric values for

the power transform method are lower than the other two methods for both groups, The Below 80 Dataset, which is made up of students who are having a hard time and have initial scores below 80%, was transformed using the Yeo-Johnson method to fix problems with distribution.

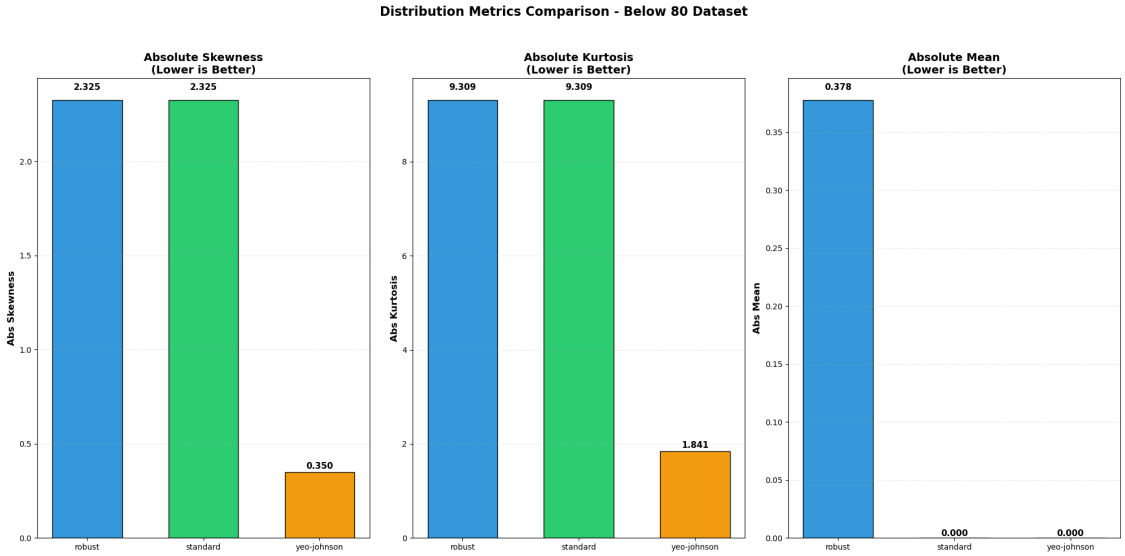

Figure 3.3. Distribution metrics comparison for below 80

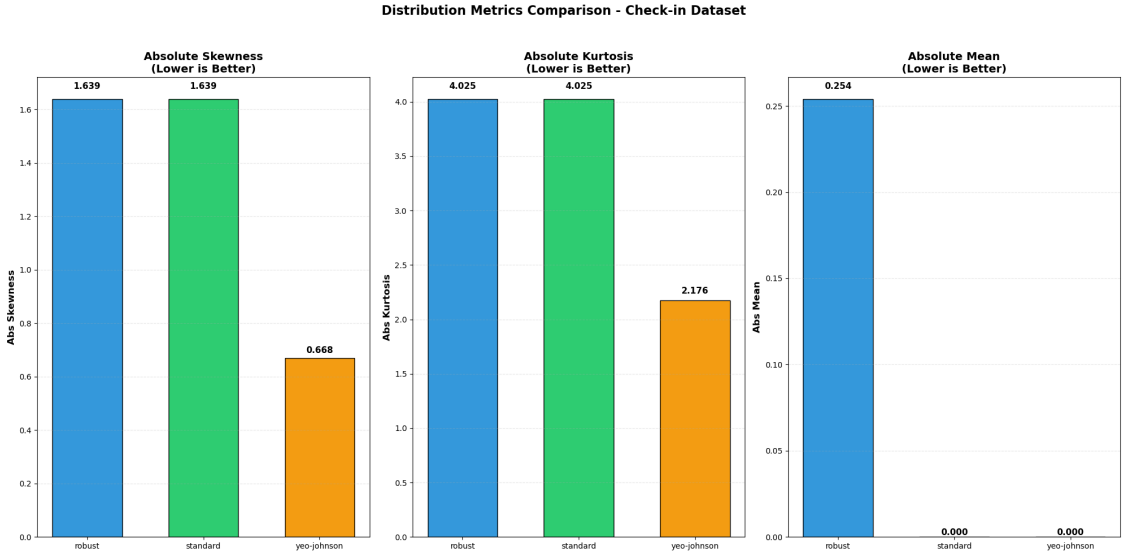

Figure 3.4. Distribution metrics comparison for all students(check-in)

As shown in figures 3.5, With `Grade_checkin`, you can see each student's first test results. This graph shows that students who are having a hard time get scores that are closer to the top of their range, but still below 80%. The skewness value is -0.5569 and the kurtosis value is -0.9547. Performance at start-up can change a lot, as shown by the wide range from -2.0809 to 1.1511. This is because the median (0.2980) is greater than the mean (-0.0), which backs up the left skew. Some students are doing much worse than this, but most of the students who are having trouble are not too far below the level.

It also shows how long it took students to finish their first test in `Time_taken_checkin(s)`. The distribution has a strong left-skewed shape (skewness = -1.1095) and a peak shape that is close to normal (kurtosis = -0.4525). There is a median number of 0.6665, which means that most students who are having trouble use almost all of their free time. Students who are having trouble may feel rushed during tests because they are having trouble understanding the information or solving problems quickly. This could be because they are clustered at the maximum time limit.

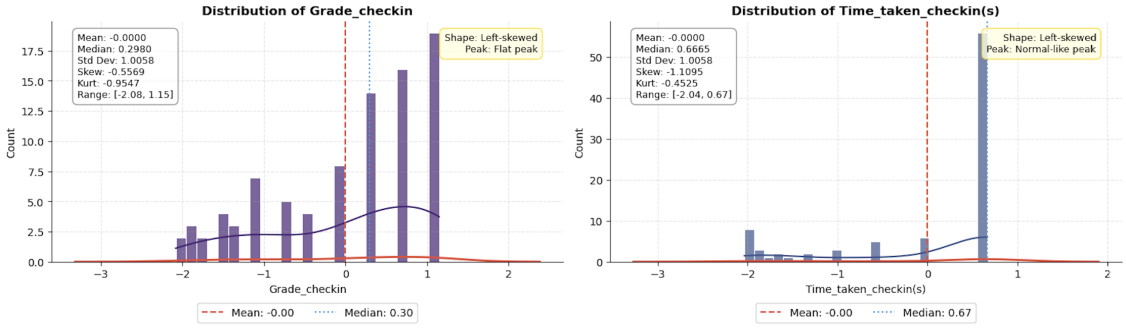

Figure 3.5. Scaled features using yeo-johnson for below 80

As shown in figures 3.6, The feature `start_time_norm_checkin` shows when the first test was given by students in relation to the due date. The distribution of this feature is a little off-center (skewness = -0.2845), and its peak is very flat (kurtosis = -1.3030). The large range (-1.8544 to 1.5502) shows that students who are having trouble are starting out in a variety of ways, with no clear trend. This shows that starting time by itself isn't a trait of students who don't do well in school—some start early, others put things off, and their performance problems are caused by other things.

The feature `duration_norm_checkin` also displays the amount of available time that was used for the first evaluation. The strong left skew (skewness = -1.0930) matches the trend of `Time_taken_checkin`, with the middle value (0.6675) being the same as the highest value (0.6675). This shows that students who are having trouble usually make the most of their time, suggesting that the problem is not a lack of effort but a lack of efficiency. The fairly flat distribution (kurtosis = -0.5103) shows that this is a pattern that most students who are having trouble with school follow.

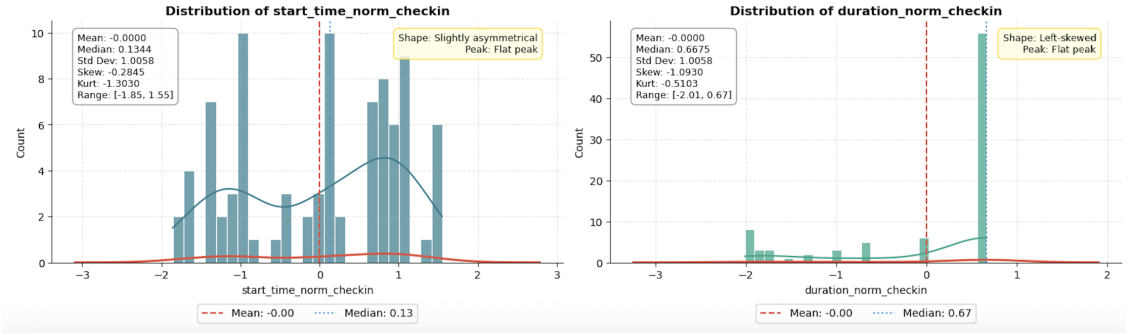

Figure 3.6. Scaled features using yeo-johnson for below 80

As shown in figures 3.7, `total_quiz_attempts_checkin` shows normal peak characteristics (kurtosis = 0.0000) and a perfectly symmetrical shape (skewness = 0.0000), indicating that students' struggling quiz scores follow a normal distribution. It is clear that this behavior remains relatively constant because all four of these measures are identical. Students who are struggling with the content may be taking the required amount of quizzes without taking the time to adequately prepare for them, which could explain the striking homogeneity.

While the overall distribution of `total_quiz_time_checkin_seconds` is highly symmetrical (skewness = 0.0674), the kurtosis value is 8.0225, and the peaks are extremely acute. This leptokurtic pattern indicates that the majority of struggling students devote approximately the same amount of time to quizzes, with a small number of extreme cases. A high kurtosis indicates that some time investments are highly clustered. This might be due to the fact that standardized quiz formats ensure that all students, regardless of their performance, spend the same amount of time on the test.

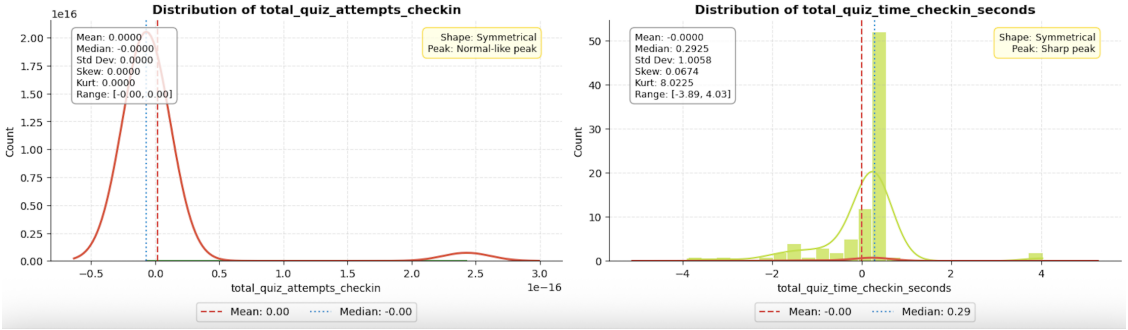

Figure 3.7. Scaled features using yeo-johnson for below 80

As shown in figures 3.8, The number of class sessions that were seen is shown by the

feature `lecture_sessions`. Its distribution is almost even (skewness = 0.0886) and its peak is flat (kurtosis = -1.0726). The range of -1.2352 to 2.1885 shows that there is a lot of variation in the number of classes that students who are having trouble go to. There isn't a clear link between how many classes a student goes to and how well they do, as shown by the flat distribution. This goes against the idea that all students who are having trouble go to fewer sessions.

The `complete_sessions` feature, on the other hand, has the highest kurtosis (8.9419) and a modest left skew (skewness = -0.4679). This sharp peak with heavy tails shows that most students who are having trouble finish about the same number of lessons. However, there are a few that stand out. The fact that the median, 25th, and 75th percentiles are all -0.3451 points points to a dominant pattern with outliers on both ends. This means that most students who are having trouble finish their lessons in the same way.

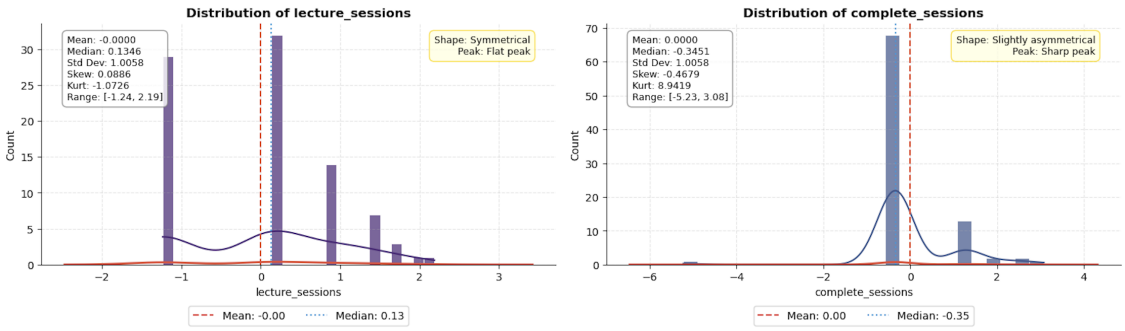

Figure 3.8. Scaled features using yeo-johnson for below 80

As shown in figures 3.9, With a skewness of 0.3039% and a kurtosis of 3.4456, the feature `total_lecture_time_seconds` has a small right skew. This means that the total lecture time is mostly around certain values, with a few students who spend a lot more time. The median is less than the mean (-0.1502 vs. -0.0), which means that most students who are having trouble spend less time in class than the norm. However, some students spend a lot more time, which is shown by the right shift. This shows that the group that is having trouble can participate in classes in different ways.

`Total_quiz_attempts_checkout`, on the other hand, is the same as its check-in cousin. It has a normal peak (kurtosis = 0.0000) and perfect symmetry (skewness = 0.0000). The final assessment phase quiz tries were all done in the same way, as shown by the fact that all statistical values were the same. This says that there are set course requirements rather than different ways that each student who is having trouble tries to do things.

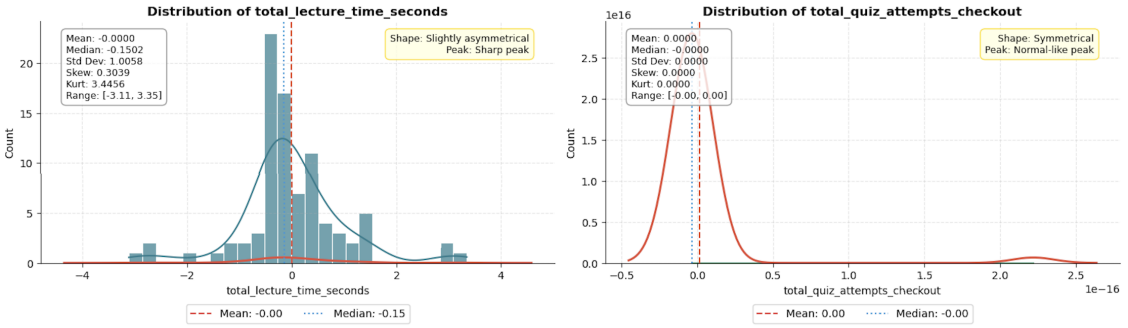

Figure 3.9. Scaled features using yeo-johnson for below 80

As shown in figures 3.10, The feature `total_quiz_time_checkout_seconds` has a distribution that is slightly skewed to the right (skewness = 0.1596) and has high kurtosis (2.7470). The leptokurtic pattern shows that most students who are having trouble spend about the same amount of time on their final quizzes, but some outliers spend a lot more time. The large interquartile range (-0.7906 to 0.8195) shows that there is a lot of variation, even though the distribution is peaked. This shows that different people used different strategies for spending time on their final exams.

On the other hand, the feature `Time_taken_checkout(s)` shows how long it took to finish the final test. It has a small left skew (skewness = -0.2769) and a flat distribution (kurtosis = -1.2173). The big difference between -2.2155 and 1.1855 and the broad interquartile range (-1.0441 to 1.0827) show that there are many different ways to manage time during final assessments. Students who are having trouble show more varied patterns of how they use their time at checkout than they did in the first assessment. This suggests that they have learned how to manage their time better over the course.

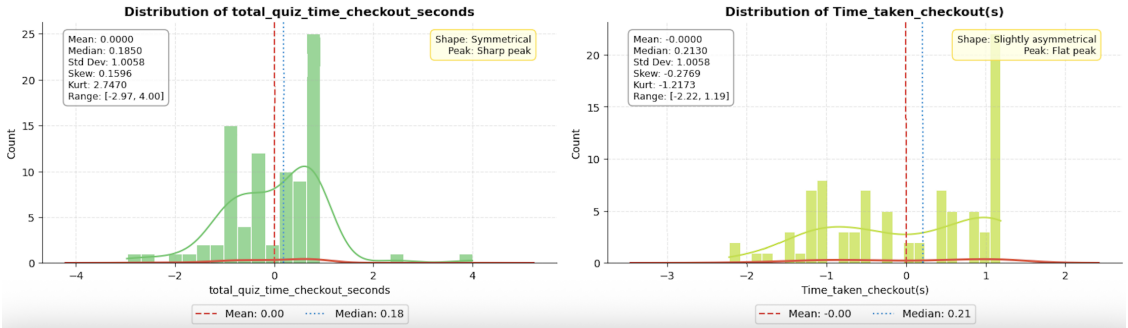

Figure 3.10. Scaled features using yeo-johnson for below 80

As shown in figures 3.11, The feature `start_time_norm_checkout` shows when students

started their final exam in relation to the due date. A flat distribution (kurtosis = -1.1703) and a slight left skew (skewness = -0.3373) show that students had different starting behaviors. The negative median (-0.0422) shows that a little more than half of the students who are having trouble start their final exam earlier than usual. Compared to the first test, the distribution is still pretty flat, which shows that students who are having trouble with timing still use a variety of tactics.

The `duration_norm_checkout` function, on the other hand, shows how much of the available time was used for final exams. There isn't much skew in this feature (skewness = -0.2077), and the distribution is flat (kurtosis = -1.2891). For the first test, most students used all of their time, but for the final test, students used a wider range of time limits. The large range, from -2.1514 to 1.2134, and the wide interquartile range, from -1.0440 to 1.0967, show that students who are having trouble managing their time come up with different ways to do it by the end of the course.

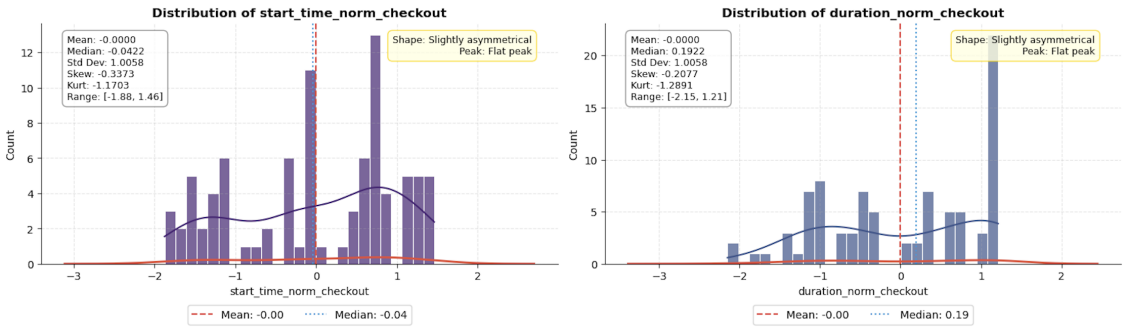

Figure 3.11. Scaled features using yeo-johnson for below 80

As shown in figures 3.12, The feature `time_efficiency_checkin` is a derived metric that shows the relationship between start time and duration. It has a nearly symmetrical distribution (skewness = -0.1194) and a moderately sharp peak (kurtosis = 1.2951). The positive median (0.1082) suggests that just over half of the struggling students show above-average efficiency on their first tests. The leptokurtic distribution shows that most students use the same strategies to be efficient, but there are some very different students at both ends of the scale.

The feature `time_efficiency_checkout` also has a very small skew (skewness = -0.1004) and a moderately sharp peak (kurtosis = 1.0937). This leptokurtic pattern suggests that the same efficiency methods are used during final exams. The median of almost zero (0.0306) shows that the efficiency strategies are balanced, with about the same number of students being more or less efficient. This suggests that there are different but similar ways to measure the efficiency of final assessments.

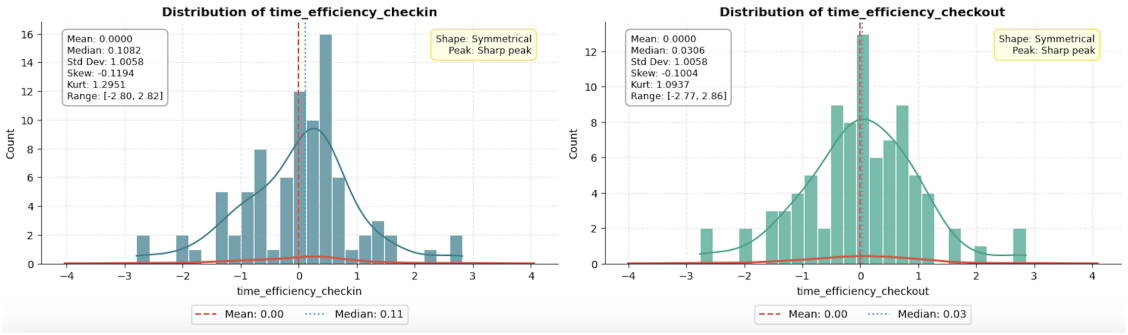

Figure 3.12. Scaled features using yeo-johnson for below 80

As shown in figures 3.13, The feature `avg_time_per_quiz_checkin` has a strong left skew (skewness = -1.0978) and a peak that is not very high (kurtosis = -0.4856). The median (0.6797) is close to the maximum (0.8271), which means that most of the time, struggling students spend a lot of time on each question during their first quizzes. This means that these students need more time to think about and answer questions. This could be because they don't understand the material or aren't sure of their answers.

The feature `avg_time_per_quiz_checkout`, on the other hand, shows how long each quiz question takes on average in final assessments. There is a little bit of a left skew to this feature (skewness = -0.2577) and a flat distribution (kurtosis = -1.1949). The wide range from -2.2103 to 1.1907 and the broad interquartile range from -0.9791 to 1.1311 show that people manage their time at the question level in different ways. The timing of the final quiz is more varied than that of the first quiz, which suggests that students who are having trouble come up with their own ways to manage their time on questions.

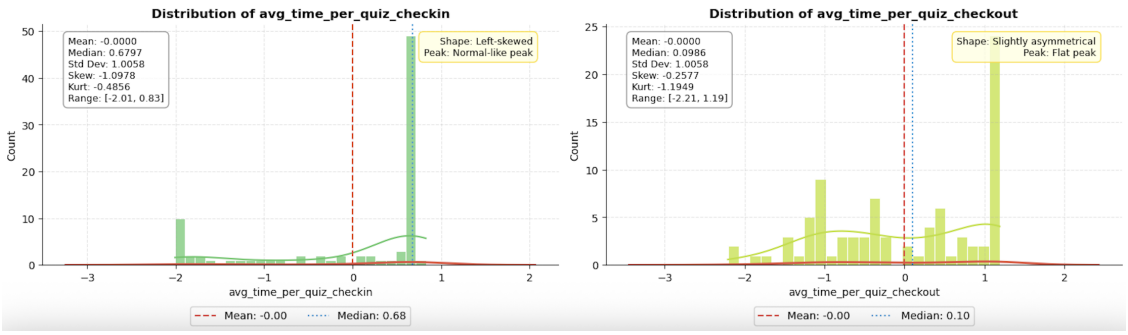

Figure 3.13. Scaled features using yeo-johnson for below 80

As shown in figures 3.14, The `avg_lecture_session_length` feature has a high kurtosis (3.2978) and a small right skew (0.2343). The leptokurtic distribution shows that most

of the sessions for students who are having trouble are about the same length, but some sessions for these students are much longer. When you compare the median (-0.0660) to the mean (-0.0), you can see that it is skewed to the right. This means that most students who are having trouble have sessions that are shorter than usual. However, some students may have sessions that are much longer to make up for their performance problems.

`Timing_strategy_change`, on the other hand, displays how the start time changed between the first and last tests. This point has a strong skew to the right (skewness = 0.9090) and a flat peak (kurtosis = -0.7385). As shown by the negative median (-0.3724), most students who are having trouble start their final exam earlier than they did their first exam, which was due on the due date. However, the right skew shows a group that goes toward later starts. This suggests that timing strategy can be changed in different ways.

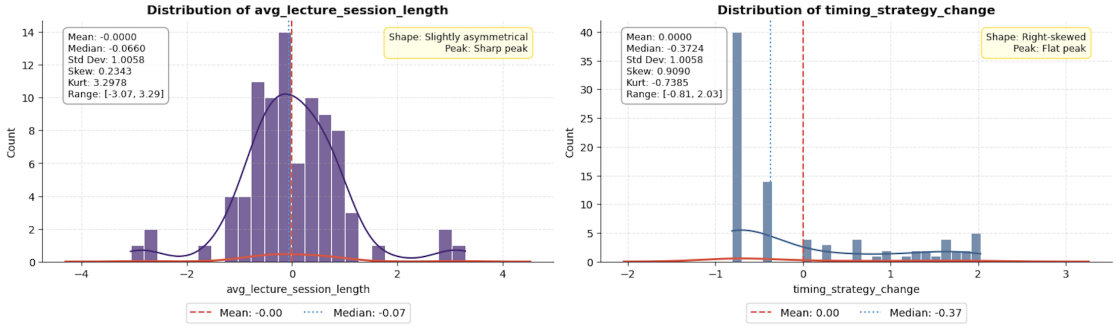

Figure 3.14. Scaled features using yeo-johnson for below 80

Figure 3.15 shows that, The feature `grade_improvement_potential`, which is found by subtracting the initial grade from 100, shows how much a student could have improved. The distribution is flat (kurtosis = -1.2093) and not very skewed (skewness = 0.1156), which means that students who are having trouble can get better in different ways. The negative median (-0.1066) shows that more than half of them have below-average improvement potential, which is shown by scores below 80. The platykurtic distribution shows that there are no clear groups at any level.

The feature `actual_grade_improvement` shows the real difference in grades between the first and last tests. The peak of this feature isn't too sharp (kurtosis = 0.6628), and the shape is very even (skewness = 0.0561). Because the median is positive (0.0721), it means that more than half of the students who are having trouble are doing better than average. The leptokurtic pattern shows that most students' scores go up by about the same amount, but some students' scores go down a lot or up a lot.

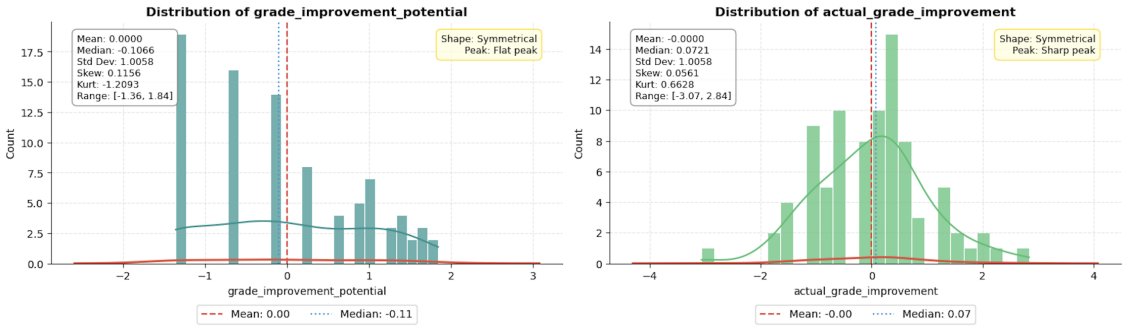

Figure 3.15. Scaled features using yeo-johnson for below 80

Figure 3.16 shows that, The resulting statistic of `improvement_efficiency` compares actual enhancement to potential. It demonstrates a slight left skew (skewness = -0.2138) and a platykurtic distribution (kurtosis = -1.2362). The positive median (0.0454) indicates that just over half of underperforming pupils convert their potential for development into actual enhancements more efficiently than average. The platykurtic distribution exhibits diverse efficiency patterns without apparent clustering, suggesting that students possess differing abilities to leverage opportunities for enhancement.

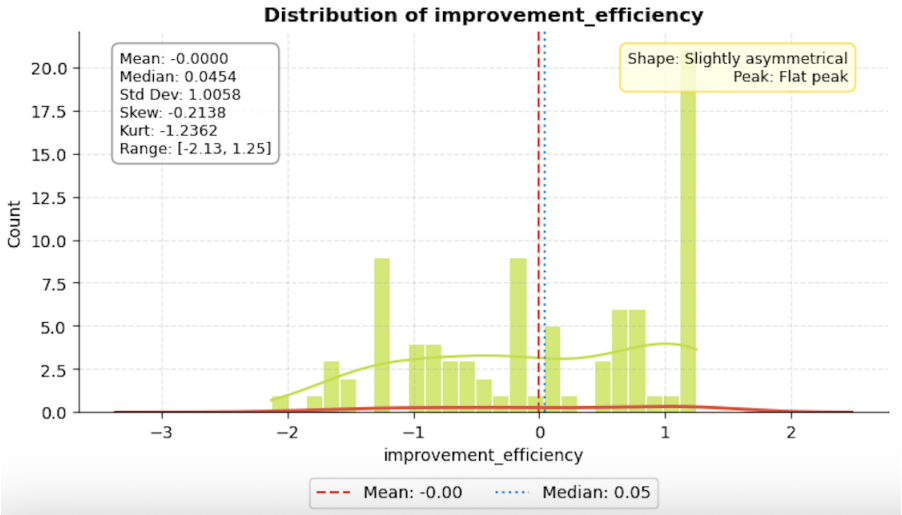

Figure 3.16. Scaled features using yeo-johnson for below 80

Figure 3.17 shows the `Time_taken_checkin(s)`, which is the total amount of time students spent on their first check-in assessment. The distribution has a small left-skew (skewness = -0.4649) and a very flat peak (kurtosis = -1.1282), which shows that students handle their assessment time in very different ways. The median (0.3653) is higher

than the mean (0.0), and the values range from -1.9204 to 2.6627. This shows that some students finish tests very quickly, while others tend to use more of the time they have. This platykurtic pattern shows that there isn't one "right" way to time assessments for all students. Instead of speed being a factor in success, it seems that students adjust their pace based on how they think, with both fast and slow approaches showing up across the performance spectrum.

Figure 3.17 shows `start_time_norm_checkin`, which shows when students start their assessment in relation to the availability window. This feature shows that students of all skill levels use very different methods to plan their work. The distribution is slightly left-skewed (skewness = -0.2267) and the peak is very flat (kurtosis = -1.2778). The wide range from -1.8471 to 1.5664 goes against the idea that starting earlier leads to better performance. The data, on the other hand, shows that students of all performance levels spread their start times out over the available window. There are successful students among both those who start early and those who start closer to the deadline. This flat distribution suggests that being able to change the timing may be more important than sticking to one "optimal" start time pattern.

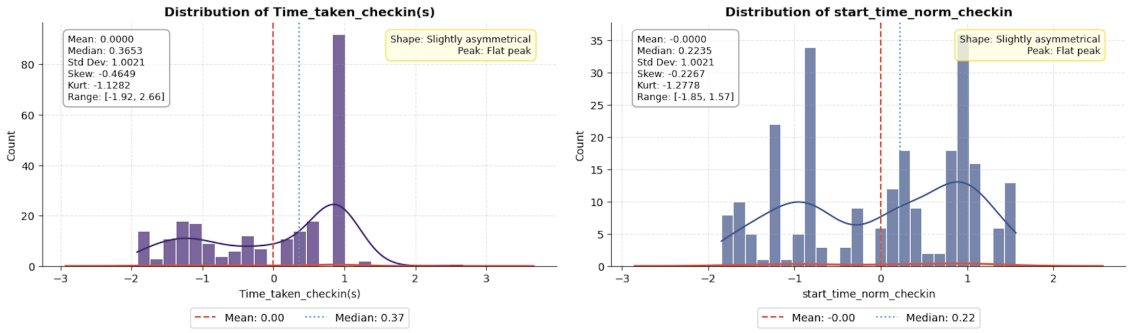

Figure 3.17. Scaled features using yeo-johnson for all students(check-in)

The `duration_norm_checkin`, which represents the proportion of time that was available for the check-in assessment, is depicted in Figure 3.18. The skewness of this feature is -0.2642, which indicates that it is slightly skewed to the left, and the kurtosis value is -1.1597, which indicates that it has a flat peak. The fact that the students' scores ranged from -1.9529 to 1.5863 demonstrates that they employed a wide variety of approaches when it came to managing their time. Some students are able to complete their assignments in a short amount of time, while others require practically all of their available time. The platykurtic distribution demonstrates that there is no discernible pattern in the data. Due to the fact that students vary their speed to meet their own knowledge of the content and how they want to work, rather than adhering to a single strategy that is considered to be the most effective, this variability shows that effective time management is highly individualistic.

The distribution pattern of `total_quiz_attempts_checkin`, which is depicted in Figure

3.18, is the most extreme of all the patterns in the dataset. This feature has a very high peak (kurtosis = 7.8537) and a significant right skew (skewness = 3.1286); the skewness value is 3.1286. It demonstrates that the majority of students attempt the same number of quizzes, but a tiny percentage of students attempt a higher number. The fact that the 25th, 50th, and 75th percentiles all have the same value (-0.2939) makes this pattern even more obvious than it already was. It would appear from this that the majority of the factors that affect the number of times people take tests are not their choice but rather the requirements of the course or the limits imposed by the structure. There is a possibility that some pupils are learning by repetition, or that they require many attempts in order to accomplish the goal of fully comprehending the content.

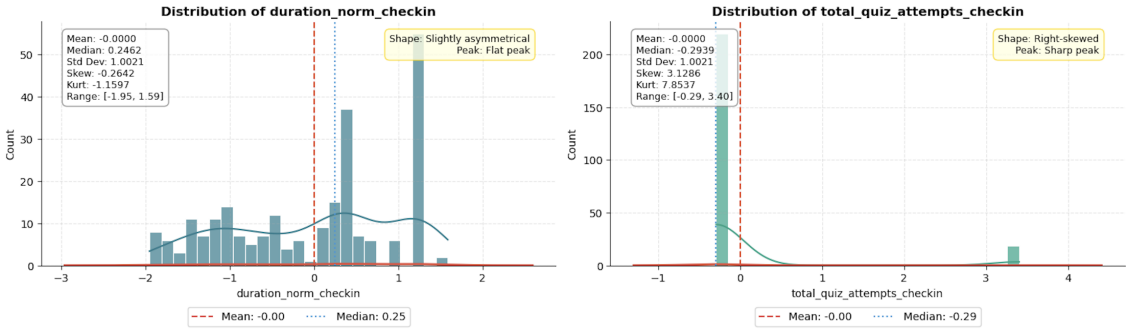

Figure 3.18. Scaled features using yeo-johnson for all students(check-in)

Figure 3.19 shows `total_quiz_time_checkin_seconds`, which counts the total time spent on quiz activities. This feature shows a distribution that is almost symmetrical (skewness = 0.0886) and has a peak that is not too sharp (kurtosis = 2.1029). The leptokurtic pattern shows that most students spend a "typical" amount of time on quiz activities, with fewer students spending very little or very long amounts of time on them. The range from -2.9341 to 3.0835 shows that most students spend about the same amount of time on their work, but there are some students who spend a lot more or a lot less time than others. This means that there may be an ideal amount of time for quiz activities for all students, but this amount of time may vary from person to person based on how they learn and how well they understand the material.

Figure 3.19 shows `time_efficiency_checkin`, which is a calculated ratio of start time to duration that shows how well students use their time once they start an assessment. This feature has a distribution that is almost perfectly symmetrical (skewness = -0.0553) and a moderate peak (kurtosis = 0.6224). It shows a tendency toward a central "sweet spot" of efficiency, with the frequency of efficiency gradually decreasing toward both very high and very low efficiency. The range from -2.5009 to 2.6296 includes both very effective methods (where students start late but finish quickly) and less effective ones (where students start early but use most of the time). The moderately leptokurtic distribution suggests that some students are moving toward the best ways to be efficient, but there is still a lot of

variation between students.

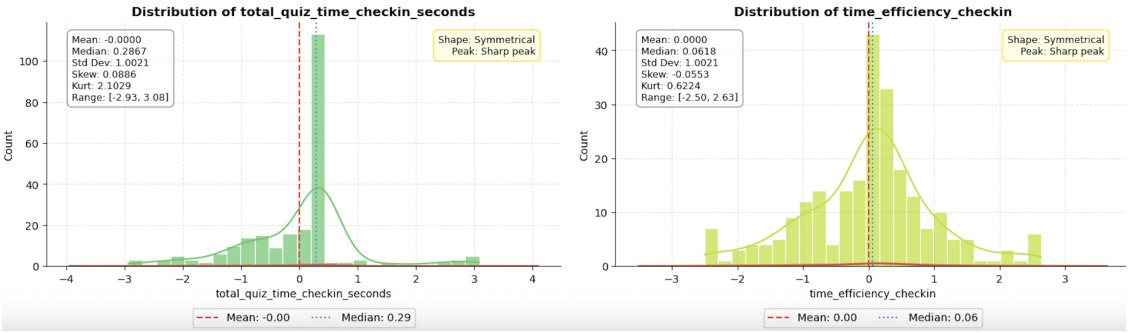

Figure 3.19. Scaled features using yeo-johnson for all students(check-in)

Figure 20 shows the `avg_time_per_quiz_checkin`, which shows how long each quiz question should take. The distribution has a small left skew (skewness = -0.4502) and a very flat peak (kurtosis = -1.0873), which shows that people manage their time differently at the question level. The range from -1.9316 to 2.8381 includes both people who answer quickly and people who need more time to think about each question. The platykurtic distribution suggests that there is no one best way for students to spend their time on each question. Some students do better with quick, intuitive answers, while others do better with careful thought. This means that their response styles may show different levels of proficiency. This suggests that the best time to answer questions is different for everyone and may depend on their cognitive style, how well they know the material, and how hard the questions are, rather than following a set rule.

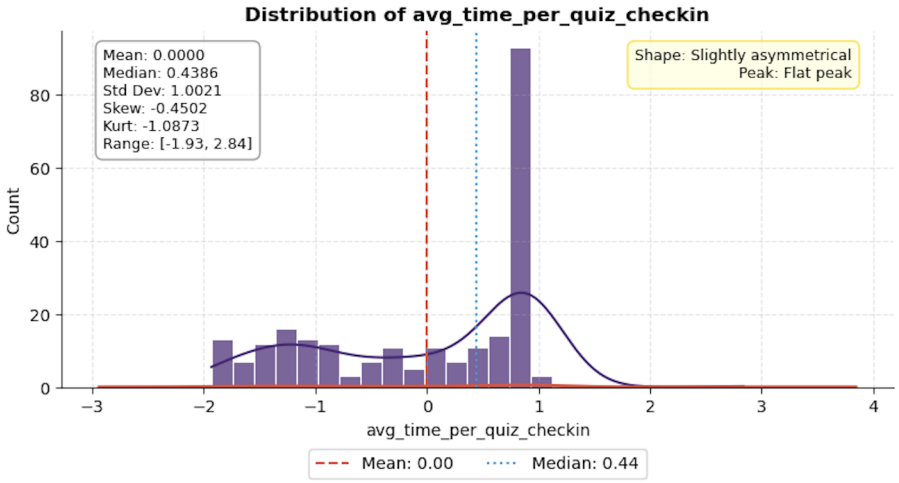

Figure 3.20. Scaled features using yeo-johnson for all students(check-in)

3.4.3 Analysis with high-achieving students: implications for education and comparison

Both significant parallels and noticeable variances in behavioral patterns are revealed by comparing the distribution patterns of the overall student cohort with the high-achieving subset (Above 80). Different timing tactics are evident across all performance levels, as both groups display extremely flat distributions for essential timing parameters including assessment duration, start time, and time consumption. This constant platykurtic pattern calls into question the long-held belief in education that certain timing strategies (such getting a head start or making the most of available time) always lead to higher results. A tiny percentage of students in both groups choose to practice a lot more by taking the quiz numerous times, which indicates that both groups have a tendency to minimize repeated efforts. The distributions of the two groups' quiz attempt counts are both highly right-skewed.

Despite these shared features, there are also some notable distinctions that help to define the traits shared by students who consistently perform at a high level. While the entire cohort displays an even more prominent concentration around usual values (kurtosis 2.1029 vs 1.0947) for quiz time investment (`total_quiz_time_checkin_seconds`), top achievers exhibit better consistency. That top performers keep a more balanced approach with reasonable regularity is supported by the fact that average students closely stick to projected time investments. Another little difference is shown by efficiency patterns. Very high achievers tend to cluster around optimal efficiency levels, while the general population exhibits a little more variability in their start timing and duration balance (kurtosis 0.6224 vs 0.6703). The most noticeable difference is the distribution of quiz attempts; the entire cohort is very homogeneous (kurtosis 7.8537 vs. 3.8783), suggesting that average students follow the rules to a T, while top scorers are more likely to stray from the norm when necessary.

A statistical transformation's efficacy machine learning techniques that presume normalcy are better suited to the check-in dataset after applying the Yeo-Johnson transformation, which greatly enhanced its statistical features. The mean absolute skewness was reduced by 59.21% through the modification, from 1.6386 to 0.6684. While this normalization was considerable, it was not quite as drastic as the one shown in the Below 80 dataset, which reduced skewness by 84.93%. Two of the seven features showed near-normal distributions after transformation; however, the total quiz attempts checkin feature, which had a very peaked and right-skewed initial distribution, nevertheless showed substantial skewness issues. We may have more faith in the educational insights gleaned from the analysis because to this significant improvement in distributional characteristics, which boosts the validity of the machine learning models and makes the feature importance rankings obtained from them more reliable.

Relevance to education and Real-World Use several significant observations with crucial implications for educational practice can be derived from the distribution patterns observed across the entire student cohort. The fact that timing feature distributions are flat across the board is strong evidence that pupils do not all benefit from the same "correct"

method of time management. This goes against the grain of conventional educational wisdom, which tends to recommend certain timing strategies as ideal, and instead implies that good time management is extremely subjective, with successful students using a variety of methods according to their unique cognitive preferences and learning styles. Instead of forcing pupils to follow rigid time management plans, teachers would be better to assist students in developing their own unique solutions.

A contrasting pattern emerges from the extremely peaked distribution of quiz attempts, which may indicate strong structural impacts or course expectations that the majority of students strictly follow. Given the lack of variation, it's reasonable to wonder if the existing frameworks are flexible enough to meet the requirements of all students or if some demographics would benefit from additional leeway to make their own decisions in the classroom. There is a lot of individual variance in how students balance the timing factors, yet efficiency measurements are in the middle ground, with somewhat peaked distributions suggesting some convergence toward optimal efficiency levels. Lastly, the significance of customization is highlighted by the flat distribution for question-level timing diversity. This indicates that students at different performance levels use different strategies to answer individual questions, and there is no clear winner.

All things considered, these results point to the need for more adaptable and individualised evaluation frameworks and time management recommendations in educational systems. Teachers should encourage students to engage in metacognitive reflection and experiment with various techniques in order to discover their own optimal strategies rather than dictating a single best way. Adaptive frameworks that cater to the wide variety of students' cognitive processes and work patterns should replace strict time management prescriptions, according to the study.

3.4.4 Strategy for splitting the train and test sets

The train-test split is implemented with the following parameters: 80% of the data is for training and 20% is for testing. Stratification is based on grade performance bins to ensure the distribution is representative. A random state is set so that it can be repeated. This method makes sure that the model's performance can be tested on a dataset that is like the original data but not used for training. This gives a good idea of how well the model will work on new data.Figure 3.21 shows the classification method, which puts students into certain grade ranges based on how well they do. There are ten different grade ranges in datasets: 0–10, 10–20, 20–30, 30–40, 40–50, 50–60, 60–70, 70–80, 80–90,and 90–100. This detailed classification lets the models tell the difference between small changes in performance levels.

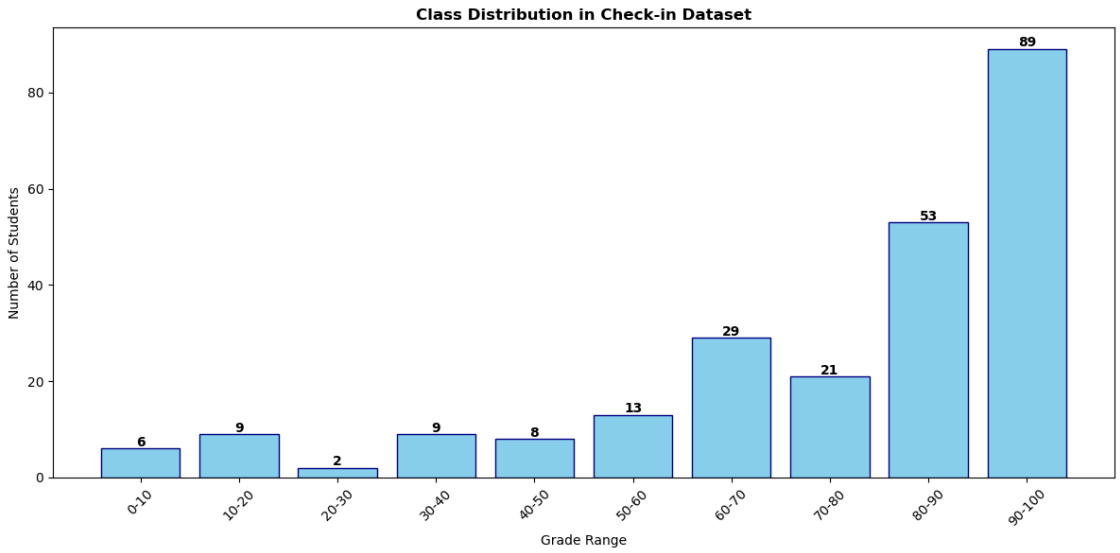

Figure 3.21. Class distribution in check-in

3.5 Model selection and training

3.5.1 Models and their justifications for regression

Based on the machine learning approaches introduced in section 2.9, several regression algorithms were selected for implementation:

1. **Ridge regression:** The method was chosen because it can handle multicollinearity with L2 regularization. It provides coefficients that are comprehensible while controlling model complexity, making it a good fit for educational datasets where features might be linked.

2. **Lasso regression:** L1 regularization makes it better at choosing features by helping to identify the most important predictors while making less important features zero. This results in a sparse model that only focuses on the most significant factors.

3. **Random forest regressor:** Its strength lies in its ability to model complex patterns and nonlinear relationships. Additionally, it is robust to outliers and can handle large datasets. It also provides metrics for feature importance that are based on reducing variance.

4. **Gradient boosting regressor:** It is used for strong prediction performance through sequential model construction. This approach is designed to capture subtle patterns in the data and provides thorough information about the value of features.

5. **Support vector regression:** It was chosen because it works well with datasets of medium size and is not affected by the number of dimensions in the feature space.

Additionally, it is able to use kernel functions to model interactions that aren't linear.

Each student group is given the best possible model to work with thanks to this varied range of algorithms that enable thorough comparison.

3.5.2 Models and their justifications for classification

Based on the machine learning approaches introduced in Section 2.9, several classification algorithms were selected for implementation:

1. **Logistic Regression:** This was the main classification model because it was easy to understand and worked well. This model figured out the chances of a student being in each grade range category and then used the highest chance to give the final grade. It also gave understandable coefficients that showed how much and in what direction each feature affected the outcome.

2. **Random Forest classifier:** This made a group of decision trees, each of which was trained on a random sample of the data with replacement (bootstrap sampling). At each split, only a random sample of features was looked at. The final predictions were made by a majority vote among all the trees. This method had a number of benefits for educational data. It was able to handle outliers in student behavior, find non-linear relationships between study patterns and performance, automatically handle feature interactions (like when the effect of starting time depends on the length of time), and measure the importance of features.

3. **Gradient Boosting classifier:** The Gradient Boosting classifier made an ensemble of trees one after the other, with each new tree trying to fix the mistakes made by the previous ensemble. Most of the time, trees were shallow and focused on certain patterns in the data. Learning happened slowly, with each tree making a small difference. We included this method because it works very well on tabular data, can find small patterns in student behavior, can learn complex decision boundaries very well, and gives us detailed metrics on feature importance.

3.5.3 Implementing hyperparameter tuning for regression

Grid search cross-validation was used to implement a systematic hyperparameter tweaking technique for each model type:

1. **Ridge regression:**

   - Hyperparameters: alpha (regularization strength), solver algorithm.
   - Search space: alpha values [0.01, 0.1, 1.0, 10.0, 100.0], solvers ['auto', 'svd', 'cholesky', 'lsqr', 'sag'].

2. **Lasso regression:**

   - Hyperparameters: alpha (regularization strength), selection method.

- Search space: alpha values [0.001, 0.01, 0.1, 1.0], selection methods ['cyclic', 'random'].

3. **Random forest:**
   - Hyperparameters: n_estimators, max_depth, min_samples_split, min_samples_leaf.
   - Search space: estimators [100, 200], max_depth [None, 10, 20], min_samples_split [2, 5, 10], min_samples_leaf [1, 2, 4].
4. **Gradient boosting:**
   - Hyperparameters: n_estimators, learning_rate, max_depth, subsample.
   - Search space: estimators [100, 200], learning_rate [0.01, 0.1], max_depth [3, 5, 7], subsample [0.8, 1.0].
5. **SVR:**
   - Hyperparameters: kernel, C (regularization), gamma.
   - Search space: kernel ['linear', 'rbf'], C [0.1, 1, 10], gamma ['scale', 'auto'].

5-fold cross-validation was used with the grid search to make sure that all possible hyperparameter combinations were thoroughly tested. The optimization metric was the negative mean squared error.

3.5.4 Implementing hyperparameter tuning for classification

A systematic grid search with 5-fold cross-validation was implemented for each classification model to find the optimal hyperparameter configuration:

1. **Logistic regression:**
   - Hyperparameters: C (regularization strength), penalty, solver.
   - Search space: C [0.1, 1.0, 10.0, 100.0], penalty ['l1', 'l2'], solver ['liblinear', 'saga'].
2. **Random forest:**
   - Hyperparameters: n_estimators, max_depth, min_samples_split, min_samples_leaf.
   - Search space: estimators [100, 200], max_depth [None, 10, 20], min_samples_split [2, 5, 10], min_samples_leaf [1, 2, 4].
3. **Gradient boosting:**
   - Hyperparameters: n_estimators, learning_rate, max_depth, subsample.
   - Search space: estimators [100, 200], learning_rate [0.01, 0.1], max_depth [3, 5, 7], subsample [0.8, 1.0].

3.5.5 Training process implementation

These steps were used to teach the model. We used the pre-set hyperparameter grid to make a grid search object for each type of model. We used 5-fold cross-validation to discover patterns in the training data that were laid out in a grid. Cross-validation's average performance was looked at in order to find the best hyperparameter configuration. With the best hyperparameters, the whole training dataset was used to teach the model again. The reserved test set helped us see how well the last model worked. For students under 80 years old and over 80 years old, we got the best models when we used the method on those groups of data.

Along with the usual cross-validation test, we used the **Resubstitution method** to see how well the model could generalize and to see if it might be too good at fitting the data. In the resubstitution method, you test the model on the same data that was used to train it. Then, you look at how well it did and compare it to the cross-validation results. We can learn a lot about how well the model works in both training and testing situations from this comparison. The R^2 difference is the difference between the training R^2 and the cross-validation R^2. The RMSE ratio is the cross-validation RMSE divided by the training RMSE. We found the difference in F1 score for classification models and the difference in accuracy (training accuracy minus cross-validation accuracy). If training and cross-validation give very different results, it could mean that the model is too good at what it does. It could mean that the model works well when the differences in performance are small. With this two-part evaluation method, we can pick models that not only do well on performance metrics but also do well when it comes to generalization. When used in education, this is important because models will be used with different groups of students in the future.

3.6 Framework for model evaluation

3.6.1 Performance metrics implementation

To fully evaluate model performance, three complementary indicators were used:

1. **Root mean squared error (RMSE):** This metric makes bigger mistakes stand out and gives an error measurement in the same units as the target variable.

2. **Mean absolute error (MAE):** This number shows how far off predictions are from actual values on average, which helps you understand the average prediction error.

3. **Coefficient of determination (R^2):** This scale-invariant metric shows how much of the variance in the dependent variable is explained by the model.

4. **F1 Score (weighted):** The primary metric, balancing precision and recall across all grade ranges.

5. **Accuracy:** The proportion of correctly classified students.

6. **RelMSE:** Relative Mean Squared Error, calculated by treating class midpoints as predictions.

We calculated these metrics for each model on both the cross-validation folds (during hyperparameter optimization) and the reserved test set (for final evaluation).

3.6.2 Standards for choosing a model

The method used to find the best model for each group of students was based on a weighted evaluation of different factors:

1. **Predictive performance:** Test set metrics are given the most weight, with RMSE, MAE, R^2, Accuracy and RelMSE all getting the same amount of weight.
2. **Model robustness:** The ability to perform consistently across different cross-validation folds, with a preference for models that do so.
3. **Model interpretability:** Assessing the clarity of feature significance and the comprehensibility of coefficients.
4. **Parsimony:** A preference for simpler models when the differences in performance are small.

This thorough method makes sure that only the best-performing models are chosen, and it also gives teachers a lot of useful information.

3.7 Approaches to analyzing the importance of features

3.7.1 Extracting tree-based importance

To find out how important each feature was, we used the built-in important attribute to see how much each feature decreased impurity (variance) on average across all trees in tree-based models (Random Forest and Gradient Boosting), The featureimportances attribute was extracted from the trained model. Features were ranked in descending order of importance. To make it easier to understand, the top features and their values were shown in a visual way. This method gives a good idea of how much each feature affects how well the model predicts. Tables 3.1, 3.2, 3.3, and 3.4 illustrate this important feature with (positive) correlation values.

3.7.2 Analyzing the coefficient of a linear model

To find out how important features are for linear models (Ridge and Lasso), we looked at the sizes of the coefficients, We found the absolute values of the standardized coefficients. We put the features in order based on how big their coefficients were. The sign of each coefficient stayed the same so that we could tell if the correlations were positive or negative. People were able to understand better with the help of graphs of coefficients. This method not only shows which way each association goes, but it also shows which features are more likely to predict the outcome.

3.7.3 Permutation importance implementation

The use of permutation importance made it possible to give a feature importance metric that is not tied to any one model: We used the trained model on the test data to find a baseline score. The values of each characteristic in the test set were mixed up at random. We used this new dataset to test the model's performance and found that its accuracy had gone down. The method was done many times for each attribute in order to get a good estimate. The features were ranked based on how much the model's performance dropped on average. This method finds out how important each feature is by looking at how much model performance drops when the feature's data is destroyed through permutation.

3.7.4 Framework for understanding education

We used a structured interpretation method to turn the meaning of statistical features into useful information for education:

1. **Feature categorization:** Important features were put into groups based on their type, such as temporal, engagement, performance, etc.

2. **Pattern identification:** Patterns that were the same across different models and groups of students were found.

3. **Educational context mapping:** We connected statistical relationships to educational theories and past research.

4. **Intervention potential assessment:** The capacity of each essential trait to influence treatments was evaluated.

This method makes sure that the model's insights can be used effectively in the classroom, closing the gap between statistical findings and teaching.

Table 3.1. Feature importance for random forest (below 80 group)

Feature	**Importance**
improvement_efficiency	0.726651
grade_improvement_potential	0.079688
Grade_checkin	0.078777
actual_grade_improvement	0.035801
total_lecture_time_seconds	0.012662
avg_time_per_quiz_checkout	0.012128
avg_lecture_session_length	0.011511
total_quiz_time_checkout_seconds	0.009893
start_time_norm_checkout	0.007482
start_time_norm_checkin	0.005192

Table 3.2. Feature importance for gradient boosting (below 80 group)

Feature	**Importance**
improvement_efficiency	0.733996
grade_improvement_potential	0.137062
Grade_checkin	0.093656
actual_grade_improvement	0.013603
total_lecture_time_seconds	0.007167
avg_lecture_session_length	0.006059
start_time_norm_checkin	0.003146
time_efficiency_checkout	0.001443
lecture_sessions	0.000916
start_time_norm_checkout	0.000900

Table 3.3. Feature importance for random forest (all students group)

Feature	**Importance**
duration_norm_checkin	0.668678
time_efficiency_checkin	0.106954
start_time_norm_checkin	0.089300
total_quiz_time_checkin_seconds	0.070663
avg_time_per_quiz_checkin	0.039418
Time_taken_checkin(s)	0.024843
total_quiz_attempts_checkin	0.000144

Table 3.4. Feature importance for gradient boosting (all students group)

Feature	**Importance**
duration_norm_checkin	0.699977
time_efficiency_checkin	0.088488
start_time_norm_checkin	0.075485
total_quiz_time_checkin_seconds	0.070360
avg_time_per_quiz_checkin	0.034475
Time_taken_checkin(s)	0.030034
total_quiz_attempts_checkin	0.001180

3.8 Pipeline integration and implementation

3.8.1 Software implementation

The machine learning pipeline was implemented using the python scientific computing stack:

1. **Data processing:** Use pandas to change data and make new features.
2. **Machine learning:** Scikit-learn for training, testing, and figuring out which features are most important.
3. **Visualization:** Use matplotlib and seaborn to see and understand the results.
4. **Statistical analysis:** SciPy for tests and changes to statistics.

The implementation uses object-oriented design principles to create modular parts that can be tested and changed.

3.8.2 Making workflows automatic

A structured workflow made it possible to run the pipeline automatically:

1. **Configuration management:** configuration management, which includes things like steps in processing, hyperparameters, and choices of features.
2. **Logging and monitoring:** There is a lot of logging in place to keep track of progress and record warnings.
3. **Intermediate result storage:** Checkpoints set up to save intermediate results and let you run parts of the program again.
4. **Parallel processing:** Steps that require a lot of computing power, like grid search, have been made parallel.

This automated workflow makes it possible to repeat experiments and try out different pipeline setups.

3.8.3 Integration with moodle system

The current implementation works with extracted data, but the pipeline was built with the idea that it could be integrated with the Moodle system:

1. **Data extraction interface:** The data extraction interface must work with the Moodle database structure and have a clear input format.
2. **Real-time processing capability:** The modular design makes it possible to make predictions in real time.
3. **Feedback integration:** The output format is compatible with moodle's intervention systems.
4. **Scalability considerations:** processing optimizations to handle more students.

These design choices make it easier to use the machine learning pipeline as part of the moodle check-in/check-out system in the future.

3.9 Ethical considerations in model development

When we made and used machine learning models, we had to think about a lot of important moral questions. A lot of things were done to cut down on bias and make sure that all kids were treated the same. For this, a balanced dataset with all levels of performance had to be made. Features had to be chosen fairly so that they didn't have any systematic flaws. The model had to be tested on various groups of students from different backgrounds to see how well it works for them. Also, when the interpretations were done, extra care was taken to avoid strong interpretations that would back stereotypes.

Many steps were taken to make sure that the data was not linked to any specific people. There was access control put in place to limit who could see the raw data and trained models, and student "IDs" were changed to secure hash values to hide their identities. Statistics were given as a whole instead of as separate pieces of data. Only behavioral data that was useful for education was collected to keep sensitive data to a minimum.

There was also a lot of information about the features, such as full descriptions of all the functions that were used. This made the modeling process better and simpler to understand. There was a reason for choosing certain models, and it was clear that they weren't sure of their skills. Results that are easy to understand through ways that give clear results were what was most important. These moral problems make sure that the machine learning method teaches effectively while also upholding students' rights and promoting fair teaching.

3.10 Summary

This chapter outlined a thorough machine learning approach for evaluating and forecasting student performance in the moodle check-in/check-out system. The pipeline includes data preparation, feature engineering, preprocessing, model training, evaluation, and feature importance analysis. Each step is tailored to a different group of students. The methodology emphasizes predictive efficacy and interpretability, which can lead to insights that can improve teaching. The models can find things that help students succeed and suggest possible interventions by looking closely at patterns over time, engagement habits, and performance trajectories.

Chapter 4

Experimental Results and Analysis of Predictive Models

4.1 Overview

It is important to test the moodle check-in/checkout system from chapter 2 and the machine learning framework from chapter 3 a lot to make sure they work well. In this chapter, we look back at all of our prediction models that were tested on four different course datasets: "Module1, Module2, Module3, and Module4". Because each dataset shows a different way that the check-in/checkout system is used in a class or lecture, they are a good way to see how well our method works in general.

A number of important research questions are answered by the experimental evaluation:

1. How well do we think we can guess how students who are having trouble will do in the end?
2. Is it possible to accurately put students into performance groups based on what they did at the start of the day?
3. What are the most important behavioral traits that can show if a person will do well in school?
4. What is the difference between regression and classification in terms of how well they can predict?
5. How similar are the patterns that can be used to guess what will happen in different classes and with different groups of students?
6. What does the fact that our models work well with both training and testing data tell us about how useful they are in real life?

Looking at these questions across a number of different datasets helps us not only figure out how well our models work, but also understand how people learn in general, which is

useful for all course implementations.

4.2 Setting up the experiment

4.2.1 Descriptions of the datasets

We evaluated data from four alternative course implementations, each with their own set of features. Module 1 demonstrates a very even distribution of student performance levels, with many students performing well in the higher performance areas. There are two major categories in it: "below 80 students," who scored less than 80 on their first check-in assessment, and "Check-in (All Students)," which includes all students, regardless of how well they performed on their first assessment.

Module 2 features a broader range of performance values, with more students in lower performance levels than Module 1. The subset's structure did not change.

Module 3 employed a below 70 criteria to segregate students who were struggling, allowing us to observe how modifying this threshold affected prediction models.

Module 4, like Module 3, employed a below 70 threshold. This allowed us to compare our results to those of other organizations using the same criterion.

The Moodle check-in/checkout mechanism provides the same core set of functionalities for all datasets. These comprised temporal metrics (when students completed examinations), engagement measures (how long they spent on learning materials), and performance indicators (test scores).

4.2.2 How to evaluate

We used a method for systematic evaluation to make sure we got a good picture of how well the model worked:

1. **Preparing the data:** Outliers were removed, features were scaled using the Yeo-Johnson transformation, and each dataset went through train-test splitting (80% training, 20% testing).
2. **Modeling approaches:** The two ways we used to model were different, but they worked well together:
    - Regression models are used to find out the exact grades that students who are below the cutoff (80 or 70) will get when they leave school.
    - To put all of the students into grade ranges based on how they check in, you can use classification models.
3. **Types of models:** We trained and compared a number of different algorithms:
    - Different kinds of regression are Ridge, Lasso, Random Forest, Gradient Boosting, and Support Vector Regression (SVR).

- There are different ways to classify things, such as Logistic Regression, Random Forest, and Gradient Boosting.

4. **Hyperparameter optimization:** Grid search with 5-fold cross-validation was used to fine-tune each type of model.
5. **Performance metrics:** To judge the models, we used different metrics, such as
   - These are the relative mean squared errors (RelMSE), MAE, R^2, and RMSE for regression.
   - Accuracy, F1 Score (weighted), and RelMSE (to compare with regression) are used for classification.
6. **Feature importance analysis:** The sixth step was to look at which behavioral factors had the most significant impact on predictions across a range of datasets and model types.
7. **Generalization assessment:** We checked how well models work in two different ways to see how general they are:
   - CV stands for "cross-validation." 5-fold cross-validation is a way to guess how well the model will work with new data.
   - Resubstitution, testing models on the same data that was used to train them to see if they are too good at what they were taught.
8. **Overfitting analysis:** We looked at overfitting and used a number of methods to do so:
   - The rate of change (R^2) between training and testing.
   - RMSE is the difference between how well the training went and how well the tests went.
   - The difference in how well something works when it is trained and when it is tested (for classification).

This systematic approach allowed us to thoroughly evaluate model performance and identify consistent patterns across different course implementations.

4.3 Results and analysis of the module1 dataset

4.3.1 Regression models for students with fewer than 80 (Module1)

The below 80 subset of module 1 in table 4.1 demonstrated strong predictive performance across all regression models, with **Lasso Regression** emerging as the most effective.

All models achieved R^2 scores exceeding 0.91, indicating that they could explain more than 91% of the variance in final checkout grades. Notably, the Lasso model reached an R^2 of **0.992**, reflecting outstanding accuracy in predicting final grades for students who initially scored below 80.

Table 4.1. Regression results for module 1 (below 80 group)

Model	**RMSE**	**MAE**	R^2	**RelMSE**
Lasso	1.33	0.94	0.992	0.008
Ridge	1.42	0.96	0.991	0.009
SVR	2.01	1.33	0.981	0.019
GradientBoosting	2.57	1.27	0.969	0.031
RandomForest	4.29	3.21	0.913	0.087

The low RMSE values with the best model achieving an RMSE of **1.33** suggest that the predicted grades were typically within 1.3 points of the actual values. This level of precision is particularly valuable for supporting educational decision-making.

Resubstitution analysis for module1 regression models

To check for possible overfitting, we looked at the results of 5-fold cross-validation and the performance on the training data (resubstitution method) in table 4.2. This comparison shows some important patterns: Linear models don't overfit much, and Lasso and Ridge regression do a great job of generalizing, with very small differences between training and validation performance (R^2 difference of 0.003–0.005). Tree-based models have a moderate amount of overfitting. For example, Gradient Boosting and especially Random Forest show bigger differences between training and validation performance. The Random Forest model fits the training data almost perfectly ($R^2 = 0.998$), but it doesn't fit the validation data as well ($R^2 = 0.913$). The RMSE ratio backs this up. The ratio of cross-validation RMSE to resubstitution RMSE is much higher for tree-based models (3.17 for Gradient Boosting and 6.60 for Random Forest) than for linear models (about 1.27). These results show that all of the models work very well for Module1, but the linear models (Lasso and Ridge) offer the best balance of accuracy and generalization. Even though the tree-based models are flexible, they seem to be overfitting, which could make them less useful in real classrooms where being able to generalize to new students is very important.

Table 4.2. Resubstitution analysis for regression models (module1)

Model	**CV R^2**	**Resubstitution R^2**	**R^2 Difference**	**CV RMSE**	**Resubstitution RMSE**	**RMSE Ratio**
Lasso	0.992	0.995	0.003	1.33	1.05	1.27
Ridge	0.991	0.996	0.005	1.42	1.12	1.27
SVR	0.981	0.989	0.008	2.01	1.57	1.28
GradientBoosting	0.969	0.997	0.028	2.57	0.81	3.17
RandomForest	0.913	0.998	0.085	4.29	0.65	6.60

Analysis of feature importance

While Lasso delivered the best overall performance, tree-based models provided more interpretable rankings of feature importance. The **Gradient Boosting** model identified the following top predictors:

1. **improvement_efficiency (73.4%)**: This was by far the most influential feature. It represents the ratio of actual grade improvement to the maximum possible improvement, highlighting that a student's ability to close their personal performance gap was the strongest indicator of their final success.
2. **grade_improvement_potential (13.7%)**: This feature captures how much a student could potentially improve, calculated as $100 -$ initial grade. It was a significant factor in predicting outcomes.
3. **grade_checkin (9.4%)**: The initial performance of students remained an important predictor, suggesting that even among initially underperforming students, the starting point influenced their eventual outcomes.
4. **Other features:**: Various engagement metrics like lecture time and session length collectively accounted for only about 3.5% of the predictive power.

4.3.2 Classification models for the check-in dataset (Module1)

We applied classification models to the full Check-in dataset, which includes all students, to predict the grade range rather than the exact score.

Table 4.3. Classification results for module1 check-in dataset

Model	Accuracy	F1 Score	RelMSE
LogisticRegression	0.55	0.49	0.45
RandomForest	0.42	0.39	0.86
GradientBoosting	0.38	0.36	0.95

As shown in table 4.3, The **Logistic Regression** model achieved an accuracy of **55%**, correctly predicting the grade range for over half of the students. The highest grade range (90–100) performed the best, with an F1 score of **0.72**.The 70–80 range had a respectable performance with an F1 score of **0.62**.The 60–70 range performed moderately, with an F1 score of **0.56**.The lowest grade ranges had poor classification performance, likely due to class imbalance and fewer data points in those categories.

Although the confusion matrix is not shown, it is likely that most misclassifications occurred between adjacent grade ranges. From an educational perspective, such mistakes are less concerning than errors that span distant grade intervals.

Resubstitution analysis for module1 classification models

Cross-validation and resubstitution were used to test classification models, and the results are shown in table 4.4. In a lot of different ways, the resubstitution analysis shows that the models tend to fill in too much. A difference of 0.09 in accuracy between training and validation shows that logistic regression is a little too good at what it does. This shows that it works well in general. On the other hand, tree-based models are way too good at what they do. On the training data, Random Forest and Gradient Boosting are

both very good at what they do (0.97 and 0.89, respectively), but on the validation data, they are much less good (0.42 and 0.38, respectively). The big difference between them (0.55 and 0.51) shows that the model is way too good at predicting the data. It's clear that there is a pattern because the gaps in F1 scores between training and validation are the same as the gaps in accuracy. This makes the worry about tree-based models fitting too well even stronger. Logistic Regression is the best way to put all of the students in Module 1 into groups, as shown by these results, because it is both accurate and useful in many situations. Many times, tree-based models try to fit too well, which makes it hard to believe that they can correctly guess how new students will do in school.

Table 4.4. Resubstitution analysis for classification models (module1)

Model	CV Accuracy	Resubstitution Accuracy	Accuracy Difference	CV F1	Resubstitution F1	F1 Difference
LogisticRegression	0.55	0.64	0.09	0.49	0.61	0.12
RandomForest	0.42	0.97	0.55	0.39	0.97	0.58
GradientBoosting	0.38	0.89	0.51	0.36	0.88	0.52

Analysis of feature importance

The **Random Forest** classifier identified the following features as the most important predictors:

1. **duration_norm_checkin (23.4%)**: Percentage of available time used during the check-in assessment.
2. **time_efficiency_checkin (21.8%)**: Ratio of start time to total duration, reflecting time management behavior.
3. **start_time_norm_checkin (20.8%)**: The normalized start time of the check-in test within the available window.
4. **avg_time_per_quiz_checkin (17.8%)**: Average time allocated per quiz question.
5. **total_quiz_time_checkin_seconds (16.2%)**: Total time dedicated to quiz activities.

These features primarily reflect students' test-taking strategies rather than their engagement with learning materials. This indicates that test planning and time management are strong predictors of academic performance within this dataset.

4.3.3 A comparison of regression and classification in the check-in dataset (module1)

We trained regression models on the check-in dataset and used the RelMSE score to evaluate how well they predicted the complete student cohort. As shown in table 4.3 the regression method had a minor advantage in RelMSE (0.41 vs. 0.45), implying that utilizing bin midpoints to represent grades was slightly more accurate when predicting precise grades than when predicting grade ranges. The categorization approach provided

numerous real-world benefits, including easier-to-understand grade range projections, the elimination of the ceiling effect in high grades, and more logical performance measures for educational decision makers. The Module 1 dataset has a classification accuracy rate of 55%, with 80% of predictions falling within 10 points of the actual values. This shows that it will be quite useful in real life, despite the very high RelMSE.

Table 4.5. Comparison of regression and classification approaches for module1 Check-in Dataset

Approach	**Best Model**	**Accuracy/R^2**	**RelMSE**
Regression	SVR	$R^2 = 0.59$	0.41
Classification	LogisticRegression	Accuracy = 0.55	0.45

4.3.4 Practical insights from module1

We got a lot of useful information from the Module1 analysis. Students who are having trouble (below 80) need to work on closing the gap between how well they did at first and how well they could do. This is the most important thing for them to do to get better. This means that interventions should help students learn as much as they can. All students should know when to take tests and how to use their time wisely while they are taking them. This is because when they start, how long they work, and how well they do all have a big effect on how well they do. The Below 80 group's high predictive accuracy ($R^2 = 0.992$) means that it might be very helpful to act on these predictions early on. This would help teachers figure out which students are most likely to improve and which ones might need more help. It's not as good with numbers, but it helps make decisions about education by putting students into groups based on how well they do. The resubstitution analysis also shows that linear models (Lasso, Ridge, and Logistic Regression) are the best at finding a good balance between being accurate and being able to use what you've learned in new situations. Tree-based models, on the other hand, can work better with training data, but they tend to overfit a lot, which could make them less useful when working with new students.

4.4 Results and analysis of the module2 dataset

4.4.1 Models for regression for students under 80 (module2)

The below 80 group of module 2 in table 4.6 demonstrated strong predictive capability, though not as well as arg01. **Gradient Boosting** was the best at what it did. The models can explain more than 94% of the variation in final checkout marks for this dataset, as shown by the high R^2 scores (above 0.94 for all models). The Gradient Boosting model achieved an R^2 of 0.982, which means it was very good at predicting what would happen.

The higher RMSE values compared to module 1 (3.57 vs. 1.33 for the best models) suggest that this student cohort was somewhat more difficult to predict precisely. This might reflect greater heterogeneity in learning patterns or a more diverse student population in the module2 course.

Table 4.6. Regression results for module 2 (below 80 group)

Model	**RMSE**	**MAE**	**R^2**	**RelMSE**
GradientBoosting	3.57	2.46	0.982	0.018
Lasso	3.87	2.39	0.979	0.021
Ridge	4.29	2.68	0.974	0.026
SVR	4.57	2.11	0.971	0.029
RandomForest	6.59	4.34	0.940	0.060

Resubstitution analysis for module2 regression models

These patterns are clear in table 4.7, which shows how well cross-validation and resubstitution work. The resubstitution analysis shows that linear models are still very good at making generalizations. When you train and test Lasso and Ridge regression, they work almost the same (the R^2 difference is only 0.005–0.006), and the RMSE ratios close to 1. Tree-based models often fit data too well. Gradient Boosting and, especially, Random Forest, on the other hand, do a lot better in training and validation. The Random Forest model has an R^2 of 0.997 when trained on data, but only 0.940 when tested on data. With this dataset, SVR is better at making generalizations than Module1. In Module2, where the R^2 difference is 0.013, SVR is better at generalizing than Module1. This study found that linear models like Lasso and Ridge are the best at making predictions for regression tasks. Tree-based models may be better at validating, but they also have a lot of overfitting, which could make them less reliable when used with new students.

Table 4.7. Resubstitution analysis for regression models (module2)

Model	**CV R^2**	**Resubstitution R^2**	**R^2 Difference**	**CV RMSE**	**Resubstitution RMSE**	**RMSE Ratio**
GradientBoosting	0.982	0.998	0.016	3.57	1.19	3.00
Lasso	0.979	0.984	0.005	3.87	3.36	1.15
Ridge	0.974	0.980	0.006	4.29	3.76	1.14
SVR	0.971	0.984	0.013	4.57	3.35	1.36
RandomForest	0.940	0.997	0.057	6.59	1.46	4.51

Analysis of feature importance (regression)

The Gradient Boosting model found these important predictors:

1. **improvement_efficiency** (60.7%): Once more, how well students closed the achievement gap was the most important factor.
2. **Grade_checkin** (13.8%): This dataset's initial success had a bigger effect than arg01's.
3. **grade_improvement_potential** (12.4%): There was still room for improvement, which was important.
4. **actual_grade_improvement** (11.3%): The change in grade points was also seen as important.

The fact that improvement related metrics are always important in both datasets shows that how well students improve is the best way to predict how well they will do in the end. However, the fact that the importance is spread out a little differently suggests that the specific dynamics may be different for different groups of students or course implementations.

4.4.2 Classification models for the check-in dataset (module2)

The classification models for the module 2 check-in dataset did not do as well as those for module 1, which suggests that this group of students is more diverse or complicated. As shown in table 4.8, The best model, **Logistic Regression**, was only 24% accurate, which is a lot less than the 55% accuracy that arg01 got. If you look at the classification report, the highest ranges (80–90 and 90–100) did better, with F1 scores of 0.50 and 0.46, respectively. The 60–70 range did okay, with an F1 score of 0.34, but the lowest ranges did awful, with F1 scores of 0.00 for several groups.

Table 4.8. Classification model performance for module2 dataset

Model	**Accuracy**	**F1 Score**	**RelMSE**
LogisticRegression	0.24	0.14	1.19
GradientBoosting	0.14	0.13	1.19
RandomForest	0.14	0.12	1.21

This pattern shows that it was especially hard to predict students who would do poorly in this group. This could be because the students' behavior was more varied or there were fewer examples in these groups.

Resubstitution analysis for module2 classification models

As shown in Table 4.9, the resubstitution analysis shows very troubling patterns when you look at how well classification models work when cross-validation and resubstitution are used. It's possible for logistic regression to be too good at what it does because the difference in accuracy is only 0.05. This shows that it can generalize well, even though it mostly fails. Tree-based models, on the other hand, are way too good at what they do. For example, the Random Forest model is 95% accurate on training data but only 14% accurate on validation data, which is a huge difference of 0.81. Gradient Boosting also has a 0.67 difference in how well training and validation work. As well, the F1 score is very different. In the case of Random Forest, the F1 score dropped from 0.95 during training to only 0.12 during validation. The results show that tree-based models don't really find patterns that can be used in other situations. Instead, they just remember the training data. This means they aren't great for use in the classroom, where the goal is to use models with new groups of students.

Analysis of feature importance (classification)

The tree-based classifiers found key factors that were similar to those in module1:

Table 4.9. Resubstitution analysis for classification models (module2)

Model	CV Accuracy	Resubstitution Accuracy	Accuracy Difference	CV F1	Resubstitution F1	F1 Difference
LogisticRegression	0.24	0.29	0.05	0.14	0.21	0.07
RandomForest	0.14	0.95	0.81	0.12	0.95	0.83
GradientBoosting	0.14	0.81	0.67	0.13	0.80	0.67

1. **time_efficiency_checkin** (26.7%): The ratio of the beginning time to the end time.
2. **Average time per question check-in** (22.4%): The amount of time spent on each question.
3. **duration_norm_checkin** (18.9%): The amount of free time that was used.
4. **Total number of quiz time check-in seconds** (16.5%): How much time was spent on quizzes?
5. **start_time_norm_checkin** (15.5%): When the kids started their test.

These timing and efficiency measures were the best predictors again, which is similar to what we saw in module 1 even though the total performance was lower.

4.4.3 Comparison of regression and classification in the check-in dataset (module2)

When the methods for module 2 were compared as shown in table 4.10, the benefit for regression was more clear: The regression method had a much lower RelMSE (0.74 vs. 1.19), which showed that predicting exact grades worked better than predicting grade ranges for this harder dataset.

Both approaches did worse than module 1, which shows that this group of students had more complicated or variable behavior patterns. The classification RelMSE of 1.19 shows that the model's mistakes were bigger than the variation in the grades, which suggests that it wasn't very good at predicting what would happen.

In module 2, only 27.6% of classifications were within 5 grade points of the real value, and 46.6% were within 10 points. This is a much smaller percentage than in module 1.

Table 4.10. Comparison of regression and classification approaches for module 2 check-in dataset

Approach	**Best Model**	**Accuracy/R^2**	**RelMSE**
Regression	Ridge	$R^2 = 0.26$	0.74
Classification	LogisticRegression	Accuracy = 0.24	1.19

4.4.4 Practical insights from module2

We learned a lot from module2. Even if they are in a group with a lot of different people, this is still the most important thing for students who are having trouble (below 80).

This shows how important it is to keep an eye on how well students fill in the blanks in their work. The whole group did much worse at classifying things (only 24.1% correct vs. 55%), which suggests that different groups of students or course settings may need different ways to make predictions. The regression method worked much better than the classification method for this dataset when it came to RelMSE. In other words, if a group has a lot of different types of people, the benefits of grouping them together might not be worth the less accurate numbers. In general, Module1 and Module2 didn't work as well as each other, but the patterns of which features were most important were the same. They can't tell you exactly when someone will be successful, but the basic signs of success don't change. The resubstitution analysis makes it clear that tree-based models are not specific enough. Using what you've learned with different groups of students can help you learn. That's why it's important to use simpler models like linear and logistic regression.

4.5 Results and analysis of the module 3 dataset

4.5.1 Models for regression for students under 70 (module3)

The analysis of the module 3 dataset looked at students who scored less than 70 points. This gave us information about patterns among students who didn't do well. Once again,as shown on table 4.11, Gradient Boosting was the best performer.The regression models trained on the below 70 subset of module3 did very well. The best model, Gradient Boosting, had an amazing R^2 score of 0.994, which meant that it could explain 99.4% of the differences in final checkout grades. This is even better than the best results from module1 and module 2, which means that for this group, it was very easy to predict how well students who started out with low scores would do.

Table 4.11. Regression results for module3 (below 70 group)

Model	**RMSE**	**MAE**	**R^2**	**RelMSE**
GradientBoosting	2.32	1.63	0.994	0.006
Lasso	3.47	2.49	0.986	0.014
Ridge	3.48	2.42	0.986	0.014
RandomForest	4.36	3.32	0.979	0.021
SVR	4.46	2.50	0.978	0.022

The RMSE value of 2.32 is low, which means that the predictions are very accurate, but not quite as accurate as the best model for module1 (RMSE = 1.33).

Resubstitution analysis for module3 regression models

Table 4.12 shows that the resubstitution analysis for module3 shows that all of the models do a good job of generalizing for this dataset when we compare the performance of cross-validation and resubstitution. Compared to how well they did on module2, the tree-based models don't overfit that much. For instance, the R^2 difference for Gradient Boosting

was only 0.005. Linear models are still better at generalizing because Lasso and Ridge have the smallest differences between training and validation performance (R^2 differences of 0.003–0.004). Random Forest is still the most overfitting, with an R^2 difference of 0.019 and an RMSE ratio of 3.28. This means that it is more overfitting than other models, but not as much as it was in earlier datasets. Because all of the models did well at generalizing on this dataset, it seems that the patterns that predict how well students below the 70-point threshold will do may be more stable and easier to find than those for students who do better or for groups of students with a wider range of abilities.

Table 4.12. Resubstitution analysis for regression models (module3)

Model	CV R^2	Resubstitution R^2	R^2 Difference	CV RMSE	Resubstitution RMSE	RMSE Ratio
GradientBoosting	0.994	0.999	0.005	2.32	0.94	2.47
Lasso	0.986	0.990	0.004	3.47	2.98	1.16
Ridge	0.986	0.989	0.003	3.48	3.11	1.12
RandomForest	0.979	0.998	0.019	4.36	1.33	3.28
SVR	0.978	0.986	0.008	4.46	3.49	1.28

Analysis of feature importance (regression)

The Gradient Boosting model found a similar pattern of important predictors to what was found in earlier datasets:

1. **improvement_efficiency** (77.1%): This is even more important than in previous datasets.
2. **Grade_checkin** (13.7%): as important as module2.
3. **grade_improvement_potential** (5.1%): Not as important as it was in earlier datasets.
4. **actual_grade_improvement** (3.0%): Also not as important as in module 2.

The fact that `improvement_efficiency` (77.1%) is so important in this dataset shows that for students who start out doing poorly (below 70), their ability to effectively improve is by far the best predictor of how well they will do in the end. The fact that other factors were less important suggests that improvement efficiency was the main factor that determined outcomes in this group.

4.5.2 Classification models for the check-in dataset (module 3)

The results of the classification for module3 were better than those for module2. As shown in table 4.13, The best model, **Logistic Regression**, got 52.8% of the answers right, which is about the same as the 55% from Module1 and a lot better than the 24.1% from module2. The F1 score of 0.46 shows that there is a fair amount of balance between precision and recall across the grade ranges. The detailed classification report showed that the highest range (58.3–100.0) did very well with an F1 score of 0.58, the lowest range (0.0–16.7) did well with an F1 score of 0.57, and the middle-high range (33.3–58.3)

did okay with an F1 score of 0.60. But the lower-middle range (16.7–33.3) did not do well, with an F1 score of 0.00.

Table 4.13. Classification model performance for module3 Dataset

Model	Accuracy	F1 Score	RelMSE
LogisticRegression	0.53	0.46	0.92
GradientBoosting	0.45	0.45	0.80
RandomForest	0.45	0.44	0.71

This pattern suggests that the model was good at finding both high and very low performers, but not so good at finding people in the middle-low range.

Resubstitution analysis for module3 classification models

Cross-validation and resubstitution were put side by side and the results are shown in table 4.14. The clear patterns can be seen in module 3's resubstitution analysis for classification models. It's clear that Logistic Regression can generalize well because it has an accuracy difference of 0.07. It did well on previous datasets too. Again, though, tree-based models are way too good at what they do: Random Forest gets 96% accuracy on training data but only 45% accuracy on validation data, which is a huge difference of 0.51. Gradient Boosting shows a difference of 0.42, which isn't as big as it could be but is still bad. It's clear that this pattern is true because the difference in F1 scores is the same as the difference in accuracy. This makes it even more clear that tree-based models have trouble with fitting too well. We've seen this over and over again in all of the datasets: tree-based classification models always do a bad job of generalizing, but Logistic Regression does a good job of it. It's even more important to use simpler, more general models first when using them in the classroom because of this.

Table 4.14. Resubstitution analysis for classification models (module3)

Model	CV Accuracy	Resubstitution Accuracy	Accuracy Difference	CV F1	Resubstitution F1	F1 Difference
LogisticRegression	0.53	0.60	0.07	0.46	0.57	0.11
RandomForest	0.45	0.96	0.51	0.44	0.96	0.52
GradientBoosting	0.45	0.87	0.42	0.45	0.86	0.41

Analysis of feature importance (classification)

The tree-based classifiers found key factors:

1. **avg_time_per_quiz_checkin** (23.9%): Average time spent per question.
2. **duration_norm_checkin** (23.7%): The proportion of available time used.
3. **total_quiz_time_checkin_seconds** (19.9%): Total time spent on quizzes.
4. **time_efficiency_checkin** (18.1%): The ratio of start time to duration.
5. **start_time_norm_checkin** (14.3%): When students began their assessment.

These timing and efficiency metrics align with those identified in previous datasets, reinforcing their consistent importance across different student cohorts.

4.5.3 Comparison of regression and classification in the check-in dataset (module 3)

The regression method had a lower RelMSE (0.56 vs. 0.71), which means that for this dataset, predicting exact grades was more accurate than predicting grade ranges. The classification approach's error analysis showed that it worked moderately well: the average error was 21.30 grade points, the median error was 12.50 grade points, only 22.6% of predictions were within 5 points of the actual grade, and only 34.0% of predictions were within 10 points. These results show that the classification accuracy was pretty good (52.8%), but the accuracy of those predictions in terms of grade points was not very good.

Table 4.15. Comparison of regression and classification approaches for module3 Check-in Dataset

Approach	**Best Model**	**Accuracy/R^2**	**RelMSE**
Regression	RandomForest	$R^2 = 0.44$	0.56
Classification	LogisticRegression	Accuracy = 0.53	0.71

4.5.4 Practical insights from module3

Module 3's analysis taught us a lot. For students who got less than 70 points, the most important thing (77.1% importance) was how well they improved. This means that the main goal of programs should be to help students who are doing really badly get better quickly. The cutoff was different (70 or less vs. 80 or less), but the classification was pretty much the same as it was in Module 1. The threshold might not matter as much as the group itself if you want to guess how well a group of students will do. All datasets need to have time and efficiency metrics. This means that both how quickly and how long students spend on tests are good ways to tell how well they will do. When it came to RelMSE, regression did better than classification for this set of data. The 52.8% accuracy rate for classifying isn't great, but it's good enough for educational interventions. These regression models are also better at making predictions than the ones in module 2. This could mean that it will be easier and more stable to find patterns that show which students will get less than 70 points.

4.6 Results and analysis for the module4 dataset

4.6.1 Regression models for students with less than 70 (module 4)

As shown in table 4.16, The best model **Ridge** got a R^2 score of 0.986, which means it explained 98.6% of the differences in final checkout grades. This is a little less than

the 99.4% that Gradient Boosting got in module3, but it still shows that it has amazing predictive power.

Table 4.16. Regression results for module4 (below 70 group)

Model	**RMSE**	**MAE**	R^2	**RelMSE**
Ridge	3.62	2.49	0.986	0.014
Lasso	4.01	3.23	0.982	0.018
SVR	4.13	2.91	0.981	0.019
GradientBoosting	6.43	4.33	0.955	0.045
RandomForest	9.45	7.33	0.902	0.098

It's interesting that Gradient Boosting did better in module3 than in module4, where Ridge regression turned out to be the best model. This means that different algorithms might be able to see different things about how students act, and the best algorithm might be different for each group.

Resubstitution analysis for module4 regression models

Table 4.17 compares the cross-validation and resubstitution performance. The resubstitution analysis for module 4 demonstrates that linear models are still very good at generalizing. Ridge and Lasso regression perform similarly with only a 0.005 R^2 difference between training and validation. Tree-based models are increasingly overfit. For example, Gradient Boosting and, in particular, Random Forest have larger performance gaps between training and validation. The Random Forest model has a significant R^2 difference of 0.096 and an RMSE ratio of 6.95. Ridge regression achieves the optimal balance between high cross-validation R^2 (0.986) and low overfitting (R^2 difference of 0.005). For this dataset, this strikes the best balance between accuracy and generalization. These findings support what we saw in previous datasets: linear models are always better at generalization, whereas tree-based models tend to overfit excessively, despite having higher training accuracy.

Table 4.17. Resubstitution analysis for regression models (module4)

Model	**CV R^2**	**Resubstitution R^2**	**R^2 Difference**	**CV RMSE**	**Resubstitution RMSE**	**RMSE Ratio**
Ridge	0.986	0.991	0.005	3.62	2.91	1.24
Lasso	0.982	0.987	0.005	4.01	3.45	1.16
SVR	0.981	0.992	0.011	4.13	2.73	1.51
GradientBoosting	0.955	0.997	0.042	6.43	1.66	3.87
RandomForest	0.902	0.998	0.096	9.45	1.36	6.95

Analysis of feature importance (regression)

We looked at the feature importance rankings of the Random Forest model because Ridge regression was the best and didn't give us feature importance scores.

1. **improvement_efficiency** (70.6%): This is again the most important predictor, just like in module 3.

2. **actual_grade_improvement** (6.2%): This is more important than in module 3.
3. **Grade_checkin** (4.6%): Not as important as in module 3.
4. **grade_improvement_potential** (4.0%): Like module 3.
5. **time_efficiency_checkout** (2.1%): This is more important than it was in module 3.

Improvement_efficiency was still the most important predictor, but its importance (70.6%) was a little lower than in module 3 (77.1%). The model put more weight on other factors, such as the time and efficiency of checkout, which suggests that this group had slightly different ways of measuring performance.

4.6.2 Classification models for the check-in dataset (module4)

The classification results for module4 were similar to those for module3, providing validation of the patterns observed.

Table 4.18. Classification model performance for module4 Dataset

Model	Accuracy	F1 Score	RelMSE
LogisticRegression	0.51	0.51	0.63
GradientBoosting	0.47	0.47	1.30
RandomForest	0.45	0.43	0.96

As shown in table 4.18, The **Logistic Regression model** was 51.1% accurate, very similar to the 52.8% obtained with module3. which means it put just over half of the students in the right grade ranges. The classification report showed that the highest range (70.3–100.0) did very well, with an F1 score of 0.76. The lowest range (0.0–31.2) did okay, with an F1 score of 0.57. The model did well in some middle ranges and poorly in others. For example, the 50.0–70.3 range got an F1 score of 0.50, while the 31.2–50.0 range got an F1 score of 0.00.

Resubstitution analysis for module 4 classification models

Table 4.19 shows how well cross-validation and resubstitution work. This table shows that the resubstitution analysis for Module 4 supports the patterns that were seen in earlier datasets. Logistic Regression is a method that works; you can expect bulletproof results day across a range of datasets, with an accuracy variance of 0.06. On the other hand, Random Forest has a big problem with overfitting, as shown by an accuracy difference of 0.52, which is a big problem. The model, on the other hand, is much less accurate when applied to validation data (0.45), even though it is almost perfect when applied to training data (0.97). The Gradient Boosting algorithm shows a lot of overfitting, as shown by the fact that its accuracy is off by 0.37. However, it is still too well-fitted to the data, even though it does a better job of generalizing than Random Forest. Even though tree-based models may not be as accurate during training, the fact that the patterns are

the same in all four datasets strongly suggests that Logistic Regression is the best way to generalize classification tasks in education.

Table 4.19. Resubstitution analysis for classification models (module4)

Model	CV Accuracy	Resubstitution Accuracy	Accuracy Difference	CV F1	Resubstitution F1	F1 Difference
LogisticRegression	0.51	0.57	0.06	0.51	0.58	0.07
RandomForest	0.45	0.97	0.52	0.43	0.97	0.54
GradientBoosting	0.47	0.84	0.37	0.47	0.84	0.37

Analysis of feature importance (classification)

These important predictors were found by the tree-based classifiers:

1. **duration_norm_checkin** (28.1%): The amount of time that was used.
2. **time_efficiency_checkin** (19.6%): The relationship between start time and duration.
3. **total_quiz_time_checkin_seconds** (19.2%): Total time spent taking quizzes.
4. **avg_time_per_quiz_checkin** (18.3%): Average amount of time spent on each question.
5. **start_time_norm_checkin** (14.9%): When students started their test.

These timing and efficiency metrics were in line with those found in the other datasets, which shows that they are always important for all groups of students.There are only slight differences in how important these features are compared to those found in module 3. The fact that this pattern holds true across two different datasets with the same threshold is strong evidence that these features are universal predictors of how well students will do.

4.6.3 Comparison of regression and classification in the check-in dataset (module 4)

When we looked at the different methods for module 4, classification came out on top:

As shown in table 4.20, The classification method had a lower RelMSE (0.63 vs. 0.67), which means that for this dataset, predicting grade ranges was more accurate than predicting exact grades.

For module 4, 23.4% of the classifications were within 5 grade points of the real value, and 40.4% were within 10 points. The average mistake was 17.20 grade points, These results are slightly better than those for module3, suggesting that the classification approach was particularly well-suited to this dataset.

4.6.4 Practical insights from module 4

The module4 analysis taught us a lot of useful things. Even for students who do really badly, early behaviors can be used to very accurately predict how they will do in the end (0.986 for the best model). This is shown by the high R^2 values for the Below 70 group.

Table 4.20. Comparison of regression and classification approaches for module4 check-in dataset

Approach	**Best Model**	**Accuracy/R^2**	**RelMSE**
Regression	SVR	$R^2 = 0.33$	0.67
Classification	LogisticRegression	Accuracy = 0.51	0.63

In modules 3 and 4, `improvement_efficiency` was always the most important factor. This shows that for students who scored less than 70, how fast they improve is by far the best indicator of how well they will do in the end. We are sure that this method can be used with other groups that have the same threshold because module3 and Module4 both did a good job of classifying things (51% to 53%). In terms of RelMSE, classification did a little better than regression for this set of data. This means that the best way to teach might depend on the students in the class. It has been shown over and over that linear models like Ridge, Lasso, and Logistic Regression are the best at finding the best balance between accuracy and generalization across a wide range of groups and thresholds.

4.7 A comparison of datasets

4.7.1 Comparing the performance of regression

When you look at regression models across the four datasets, As shown in table 4.21 you can see some interesting patterns: There are a few important things to note. There is always a lot of predictive power, since the best models across all four datasets had R^2 values above 0.98. This means that for students below the threshold (whether 80 or 70), we can predict their final performance with great accuracy. The best algorithms changed from dataset to dataset, which means that the best way to do things depends on the specific traits of the group of students. There was no consistent difference in predictive performance between the Below 80 datasets (module1, module2) and the below 70 datasets (module3, module4), which means that the choice of threshold may not be as important as other factors. The precision also changed, with RMSE values ranging from 1.33 to 3.62. This shows that the predictions were not always as precise across datasets. module1 was the most accurate, and module4 was the least accurate.

Table 4.21. Comparison of best regression model performance across datasets

Dataset	**Best Model**	**R^2**	**RMSE**	**RelMSE**
Module1 (Below 80)	Lasso	0.992	1.33	0.008
Module2 (Below 80)	GradientBoosting	0.982	3.57	0.018
Module3 (Below 70)	GradientBoosting	0.994	2.32	0.006
Module4 (Below 70)	Ridge	0.986	3.62	0.014

4.7.2 Comparison of classification performance

As shown in table 4.22, The classification models all show a similar pattern of performance that changes: Key points include that **Logistic Regression** was the best classification model across all datasets. The predictive power was different, with accuracy ranging from a low of 24% (module2) to a high of 55% (module1). This shows that performance categories were very different across different groups. module3 and module4 also showed similar patterns, with both datasets using the below 70 threshold showing similar classification performance (about 51–53% accuracy). This suggests that the same threshold works well across different groups. In module2, there was an outlier performance, with a much lower accuracy rate of only 24%. This suggests that this group had unique traits that made it especially hard to classify.

Table 4.22. Comparison of best classification model performance across datasets

Dataset	**Best Model**	**Accuracy**	**F1 Score**	**RelMSE**
Module1 (Check-in)	LogisticRegression	0.55	0.49	0.45
Module2 (Check-in)	LogisticRegression	0.24	0.14	1.19
Module3 (Check-in)	LogisticRegression	0.53	0.46	0.92
Module4 (Check-in)	LogisticRegression	0.51	0.51	0.63

4.7.3 Resubstitution analysis comparison

Table 4.23 shows that the patterns in model generalization are the same when resubstitution is done on different datasets. This comparison shows that linear models almost never overfit and that Lasso, Ridge, and Logistic Regression are all very good at generalizing across all datasets, with only small differences between how well they do on training and validation sets. Tree-based models, on the other hand, always have a lot of overfitting. For instance, Random Forest and Gradient Boosting have much bigger gaps between training and validation performance. For regression, the R^2 differences range from 0.019 to 0.096, and for classification, the accuracy differences range from 0.37 to 0.81. Classification models tend to overfit more than regression models. The best classification models, like Logistic Regression, have bigger performance gaps (0.05–0.09) than the best regression models (0.003–0.005). It's interesting that the patterns of generalization performance are very similar across all four datasets. This means that the models used, not any one dataset, are what make the differences.

4.7.4 Patterns of feature importance

Across datasets, the feature importance analyses showed a number of trends:

1. **For students who are (below 80/70):** `Improvement_efficiency` was always the most important factor, accounting for 60–77% of the differences across all datasets. `Grade_checkin` and `grade_improvement_potential` were always secondary predictors. Metrics that measure engagement, like lecture time and session length, didn't have much of an effect.

Table 4.23. Generalization performance comparison across datasets

Dataset	Model Type	Best CV Model	R^2/Accuracy Difference	RMSE Ratio
Module1	Regression	Lasso	0.003	1.27
Module1	Classification	LogisticRegression	0.09	-
Module2	Regression	Lasso	0.005	1.15
Module2	Classification	LogisticRegression	0.05	-
Module3	Regression	Ridge	0.003	1.12
Module3	Classification	LogisticRegression	0.07	-
Module4	Regression	Ridge	0.005	1.24
Module4	Classification	LogisticRegression	0.06	-

2. **For all student (check-in):** Assessment timing metrics were always the best at predicting outcomes. `duration_norm_checkin`, `time_efficiency_checkin`, and `start_time_norm_checkin` were the most important features. The importance of these features was not the same across all datasets.

These patterns show that models' ability to predict student performance varies from group to group, but some features always give good signals about how well students are doing.

4.8 Practical Implications

The test results have a number of real-world effects on how schools work and how learning management systems are built:

4.8.1 For students who started below the threshold (80/70)

The study shows that `improvement_efficiency` is the most important thing for students who started out below the threshold (80/70). This means that interventions should focus on helping students get the most out of what they learn compared to what they could learn by using focused scaffolding and personalized feedback. There is also a big difference in how well the below 80 and below 70 models work. This means that the strategies for helping people need to be changed to fit different levels of performance. For instance, students who are getting less than 70 may need more help or help that is set up in a different way. For the group that did the worst, how they behaved during the checkout process was a strong indicator of how well they would do in the end. This means that showing students how to do well on tests could help this group do better as a whole.

4.8.2 As a check-in dataset for all students

Time management and assessment strategies are very important for students to do well in school because when they start, how much time they give themselves, and how quickly they work all have a big effect on their grades. It's good for all of them to learn how to manage their time and use timing strategies for tests. Regression models may sometimes

give slightly more accurate numbers, but the classification method is much better for making decisions in education because it is easier to understand and can put students into groups based on how well they are doing. Also, the fact that different datasets show different predictive trends shows that different groups of students may need different ways to predict and help them. This shows that one model probably won't work best for all kinds of students.

4.8.3 Things to keep in mind and limitations

There are a few things to keep in mind when trying to figure out what these data mean:

1. **The size of the datasets:** The datasets taught us a lot about behavior, but since they were so small, some of the results may not work in other situations. If these conclusions were backed up by more people, they would be stronger.

2. **Decisions about feature engineering:** The features that were made were based on what we know about the field and what we know about educational theory. But if you use a different way to engineer features, you might get different results.

3. **Causality vs. correlation:** Just because there were relationships that could be used to make predictions doesn't mean they show causality. Some students who do well in school may naturally act in ways that help them do well, rather than these behaviors leading to good performance.

4. **Specificity to the situation:** The big differences in how well the datasets performed show that prediction patterns may only work in some situations and not in all schools.

5. **Grade range boundaries:** The method for classifying used grade range boundaries that were somewhat random, which could have changed the performance that was recorded. If you choose a different limit, you might get different results.

4.8.4 A summary and conclusions

This project showed that it is possible and helpful to use machine learning to guess how well students will do in the moodle check-in/check-out system. The most important things we learned are:

1. **We can accurately predict how students will do in the end if they score below 80** ($R^2 > 0.98$ in both arg01 and arg02). This gives us a good starting point for helping them right away.

2. **Timing techniques are always important:** Across all datasets and model types, how students take tests, especially how quickly and well they do, was found to be a key predictor.

3. **Different patterns for different thresholds:** The way students predicted below the 70-point threshold was different from the way they predicted below the 80-point

threshold. This means that students who aren't doing well might need different kinds of help.

4. **Trade-offs between classification and regression:** Regression models sometimes gave more accurate numbers, but classification models were easier to understand and better for teaching.

5. **Differences by cohort:** The big differences in success across datasets show how important it is to make sure that predictive methods are tailored to specific groups of students.

These results are a step toward building educational support systems that are faster and better suited to each student. Teachers can help students do better in online and blended learning by finding early signs of possible performance problems and figuring out which factors are most likely to predict results.

Chapter 5

Conclusion

5.1 Introduction

This thesis gave a thorough look into how to use the moodle check-in/checkout system to predict how well college students will do. In chapter 4, a lot of experiments were done on four different academic course datasets, which were called Module1, module2, module3, and module4. The results showed both consistent patterns and important differences in how well the predictions worked. This last chapter takes all of these results and talks about what they mean for education in a broader sense. It also gives suggestions for implementation and further research.

The main things that this work adds are:

1. Work on and approval of a machine learning framework for predicting student performance early on.
2. Identifying key behavioral indicators that consistently show academic success across a range of courses.
3. A study that compares the regression and classification methods for predicting performance.
4. Resubstitution analysis is used to test how well a model can generalize.
5. Predictive analytics can help with making useful suggestions for educational interventions.

5.2 Summary of the most important findings

5.2.1 Predictive power for students who are having trouble

One of the most important things this study found was that it was very good at predicting how students who are having trouble at first (scoring below 80 or 70 on their check-in assessment) will do in the future. Our models always got R^2 values above 0.98 for all four

datasets, which means they explained more than 98% of the variation in final checkout grades.

This amazing level of accuracy in making predictions has huge effects on how schools work. It shows that for students who are having trouble at first, their final performance can be predicted very accurately by how they behaved at the beginning. This makes targeted early intervention very possible, as teachers can be very sure of which students who are having trouble are likely to get better on their own and which ones might need extra help.

5.2.2 Why improvement efficiency is most important

In all datasets, `improvement_efficiency`—how well students closed the gap between their initial performance and their potential—was the best predictor of how well they did in the end for students who were having trouble. This factor consistently explained 60–77% of the variation in final grades, which was a lot more than any other predictor, such as initial performance, engagement metrics, or time management factors.

This finding suggests that we need to change the way we think about academic success for students who are having trouble at first. Success doesn't seem to depend on where a student starts or how good they are; instead, it seems to depend on how well they use learning opportunities to get better. This puts the whole emphasis on learning rather than how smart someone is to begin with.

5.2.3 Strategies for timing assessments as universal predictors

Assessment timing metrics consistently emerged as the best predictors for the whole group of students across all datasets:

- The amount of time that was actually used (`duration_norm_checkin`)
- The ratio of the start time to the length of time (`time_efficiency_checkin`)
- What time students started their test (`start_time_norm_checkin`)
- The average amount of time spent on each question (`avg_time_per_quiz_checkin`)

The total amount of time spent on quizzes is shown in seconds.

The fact that these timing and efficiency metrics are important across all datasets suggests that how students prepare for tests—how well they manage their time, how efficiently they work, and how well they plan their strategies—is a universal predictor of how well they will do in school. These patterns of behavior aren't limited to a certain course or subject. They point to basic metacognitive skills that help students do well in all kinds of academic settings.

5.2.4 Things to think about when generalizing models and overfitting

The resubstitution analysis that compared cross-validation and training performance showed important things about how models can be used in real life. Linear models

consistently showed great generalization across all datasets, with only small differences in performance between training and validation (R^2 differences of 0.003–0.006 for regression, accuracy differences of 0.05–0.09 for classification). Concerningly high levels of overfitting were seen in tree-based models, with big differences in performance between training and validation (R^2 differences up to 0.096 for regression, accuracy differences up to 0.81 for classification). Classification models generally showed more overfitting than regression models. This means that guessing exact grades might be easier to use in real life than putting students into performance ranges. These results show how important it is to put model generalization ahead of raw predictive accuracy in educational settings. Complex models like Random Forest may be more accurate on training data, but they are less reliable when making decisions about education in the real world, where new groups of students are always being tested.

5.2.5 Differences in context between datasets

Even though consistent patterns were found, there was a lot of variation in how well the predictions worked across the four datasets. Classification accuracy ranged from 24% (Module2) to 55% (Module1), showing that different groups of students have very different levels of ability to predict performance categories. Different algorithms did better on different datasets; for example, Lasso did best in module1, Gradient Boosting did best in module2 and module3, and Ridge did best in module4. The difference in performance between regression and classification methods also changed between datasets. In three datasets, regression did better than classification, but in module4, classification did better. This difference in context shows that even though some basic patterns are the same for all students, the best way to predict their performance may depend on the unique traits of that group of students. This highlights the importance of having flexible methods that can be used in various classroom settings.

5.3 Implications for theory

5.3.1 Rethinking what makes students successful in university

The fact that `improvement_efficiency` is the most important factor in all datasets goes against traditional ideas of academic success that focus on starting with skills or knowledge. Instead, our results are in line with growth mindset theories (Dweck, 2006) [6] that stress how important it is to learn how to improve over having a fixed set of skills. For students who are having a hard time at first, the data strongly suggests that how well they improve is much more important than where they start.

This has huge implications for educational theory. It means that instead of just measuring or categorizing starting abilities, educational systems should focus more on creating effective ways for people to get better. The high accuracy of `improvement_efficiency` in predicting outcomes backs up teaching methods that focus on progress rather than perfect performance.

5.3.2 The Metacognitive basis of good performance

The fact that assessment timing strategies are important in all datasets shows that metacognition—that is, thinking about how you think and learn—is a key factor in academic success. Many content-specific factors don't seem to be as good at predicting how well students will do on tests than how they prepare for them—when they start, how they divide their time, and how quickly they work.

This fits with ideas about education that stress the importance of "learning how to learn" as a skill that can be used in many different areas. Our results suggest that metacognitive skills like managing time, making plans, and controlling oneself may be able to predict academic success in all kinds of school settings.

5.3.3 The problem of generalization in predicting education

The big overfitting seen in tree-based models shows a big problem in predicting what will happen in schools: it's hard to find models that work reliably with new students while also capturing complex, nonlinear relationships. Flexible, nonlinear models can be very accurate on training data, but they tend to overfit, which makes them less reliable for making decisions in education.

This suggests that simpler, easier-to-understand models might work better in educational settings where applying them to new groups of students is important. It has been shown that linear models are better at generalization across all datasets. This supports the principle of parsimony in educational prediction, which says that simpler models may be more useful in the long run than more complex ones, even if they are less accurate on training data.

5.4 Implications for the practice of teaching

5.4.1 Strategies for early intervention with students who are having trouble

The high accuracy in predicting which students will have trouble allows for targeted early intervention strategies:

1. **Efficiency-focused coaching**: Come up with interventions that are designed to help students learn more effectively, or how they can turn learning opportunities into performance gains.

2. **Personalized improvement targets**: Help students see their full potential and keep track of their progress by setting personalized improvement targets based on how they think they will do in the future.

3. **Optimizing the allocation of resources**: Give limited support resources (like tutoring, extra classes, etc.) to students who are likely to have trouble improving their efficiency.

4. **Motivational feedback**: Give students regular feedback on how they can improve, not just on how well they did, to help them understand and improve the way they learn.

Because these interventions are very good at predicting which students will be having trouble, they can be used with confidence to help the students who are most likely to benefit from it.

5.4.2 All students should learn how to use assessment strategies

The fact that assessment timing metrics are conditions that Teens Should centrals that assessment methods should be improved. Students should learn how to manage their time well during tests, such as how long to spend on different types of questions, in workshops on time management. Students can use strategic planning instructions to help them figure out when to start tests and how to pace themselves during the time they have. Training in efficiency should include structured practice on how to finish tests quickly and feedback on how time is spent. Adding metacognitive prompts, or questions on tests that make students think about how they use their time and how they can improve, can also help them get better. These changes would help all students, not just the ones who are having trouble at first. This is because things like when tests are given consistently predict how well someone will do at all levels.

5.4.3 Choosing the best model for educational use

Our resubstitution analysis gives useful advice on how to use predictive models in school settings:

1. **Prioritize linear models**: Linear models (Lasso, Ridge, and Logistic Regression) should be used in educational settings where they need to be able to generalize to new groups of students, even if they may not be as accurate when training.

2. **Regular validation cycles**: Make sure that predictive models are regularly tested with new groups of students to make sure they keep their ability to generalize over time.

3. **Context-specific model selection**: Choose a model based on how each course or group of students works, keeping in mind that the best way to do things may be different in different situations.

4. **Ensemble approaches**: Think about methods that combine the predictions of several models while lowering the risk of overfitting.

These useful suggestions can assist schools in setting up more dependable and useful predictive analytics systems.

5.5 Suggestions for more research

This study brings up a lot of interesting points that should be looked into further to help us figure out how to better guess how well students will do in school. Making the dataset more varied is the first thing that needs to be done. We need to find out more about all of our students, schools, and academic fields to be sure that what we found is true. It will help us figure out the best ways to guess what will happen in all sorts of schools. It would be helpful to keep track of how students did in different classes and semesters over time. This would help us understand how predictive patterns change over time and whether students' early performance paths stay the same throughout their academic career. You should also think about how people in different groups might predict things in different ways. To make sure the predictive models are fair for all students and don't favor certain groups, we will do this.

Figuring out how the predictive relationships were found and how they work is another important area of study. Some things about a student's behavior were strongly linked to how well they did in school, as shown by our models. There needs to be controlled intervention research, though, to see if changing things, like how things are done or when tests are given, really does help people do better. We could find out more about why some students improve faster than others by doing qualitative research on the ways that students think and act that help them do better. This will add to the number we already have. What about the student or the situation makes the link between behavioral predictors and how well they do in school stronger or weaker? More research into the things that mediate and moderate could help answer this question.

You can make better predictions and keep up with how well they work in general with the help of more advanced modeling methods. Visualizing time in a class would let you see how behavior changes over time. This might help you pick the best time to help. Through transfer learning, it might be easier for students from different classes and groups to share what they know. They wouldn't have to make new models for each class, which would save them time. We could get more accurate results with deep learning methods that can be understood. These methods would still be easy enough for school use. Other methods that protect privacy could let models learn from educational datasets that are spread out across many schools without putting the privacy of the students at risk. It would be easier to see a lot of schools at once.

Implementation science research is needed to both guess what will happen and make learning happen. More research is needed to find the best way to combine predictive analytics with current educational interventions. Once we do this, we can use what we've learned to make plans that will help students do better. If researchers want to make implementation work, they could look into things like how faculty members feel, what stops them from accepting new ideas, and what helps them grow professionally. Find out what predictions and the things that go with them make students feel and what they do. This is important because if the students don't agree with the activities, they won't work as well. Lastly, making clear moral rules about how to use predictive analytics in schools would make sure that these strong tools are used in ways that protect students

and help them. We can make systems that help students learn better by going down these different research paths. Each student will get better service from these systems because they will work better and respond faster.

5.6 Limitations of the study

There are a few things that should be kept in mind when trying to figure out what these research results mean:

1. **Dataset size**: The datasets used were pretty small, but they had a lot of information about behavior. This means that some of the results may not be applicable to other situations.
2. **Context specificity**: The four courses that were looked at might not show all the different types of higher education settings, and the patterns that can be used to predict things might be different in other places.
3. **Correlation vs. causation**: The predictive relationships found don't always mean "cause and effect." For example, behavioral factors may be signs of academic success rather than causes of it.
4. **Feature engineering choices**: The features that were found were based on educational theory and knowledge of the domain. Other feature engineering methods could, however, lead to different conclusions.
5. **Time limitations**: The study only looked at predictions within single courses, not over longer academic paths.
6. **Limited demographic analysis**: The study didn't look into how patterns of prediction might be different for different groups of people.

These limitations give us important background information for understanding the results and point the way forward for future research.

5.7 Conclusion

This study shows that machine learning has a huge amount of potential to predict how well college students will do, especially those who are having a hard time at first. We found that improvement efficiency was the most important predictor and that assessment timing strategies were the most important universal indicators. This gave us both theoretical insights and useful advice for educational practice.

The results show how important it is to focus on the learning process rather than initial performance. This means that educational interventions should focus on helping students improve quickly instead of just finding those who are naturally good or bad at learning. Metacognitive skills like time management and strategic planning are always important. This shows that teaching students "how to learn" is just as important as teaching them specific subject content.

The most important thing that our resubstitution analysis has shown is how important model generalization is in educational settings. Complex models may be very accurate on training data, but they are less reliable for making real-world decisions about education because they tend to overfit. Linear models may be better for supporting educational practice in the long run because they perform well at generalization across all datasets.

Using the check-in/checkout system along with the right predictive models can help teachers build a strong system that encourages students to self-regulate, allows for timely intervention, and ultimately improves learning outcomes in higher education. This way of doing things is a big step toward making school systems more flexible, individualized, and effective so they can help all students do better in university.

Bibliography

[1] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[2] Paul Black and Dylan Wiliam. Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability*, 21(1):5–31, 2009.

[3] Christopher Brooks, Gabe Erickson, John Greer, and Carl Gutwin. Early prediction of student dropout and performance in web-based courses using log data. pages 148–153, 2011.

[4] Rebeca Cerezo, Miguel Sánchez-Santillán, M Puerto Paule-Ruíz, and José C Núñez. Students' lms interaction patterns and their relationship with achievement: A case study in higher education. *Computers & Education*, 96:42–54, 2016.

[5] Carolina Costa, Helena Alvelos, and Leonor Teixeira. Moodle: Morphologic analysis of the learning management system of a higher education institution. *RISTI-Revista Ibérica de Sistemas e Tecnologias de Informação*, (9):1–15, 2012.

[6] Carol S. Dweck. Mindset: The new psychology of success. 2006.

[7] Alfred Essa. A possible future for next generation adaptive learning systems. *Smart Learning Environments*, 3(1):1–24, 2016.

[8] Kelum AA Gamage, Jianglin Xiao, Masood Naeem, Abbes Akram, and Muhammad Ali Imran. Improving assessment outcomes through the application of innovative digital technologies. *International Journal of STEM Education*, 6(1):1–15, 2019.

[9] Dragan Gašević, Shane Dawson, and George Siemens. Let's not forget: Learning analytics are about learning. *TechTrends*, 59(1):64–71, 2015.

[10] JW Gikandi, D Morrow, and NE Davis. Online formative assessment in higher education: A review of the literature. *Computers & Education*, 57(4):2333–2351, 2011.

[11] John Hattie and Helen Timperley. The power of feedback. *Review of Educational Research*, 77(1):81–112, 2007.

[12] Jui-Long Hung and Ke Zhang. Analysis of learners' performance and learning patterns in a web-based learning environment. *Journal of Educational Computing Research*, 38(2):207–223, 2008.

[13] Il-Hyun Jo, Yoon Kim, and Yeonjeong Kang. The effect of procrastination on self-regulated learning in academic writing. *International Conference on Educational Data Mining*, pages 265–266, 2014.

[14] Kenneth R Koedinger, Sidney D'Mello, Elizabeth A McLaughlin, Zachary A Pardos, and Carolyn P Rosé. Data-driven learner modeling to understand and improve online learning. *Current Directions in Psychological Science*, 24(3):198–205, 2015.

[15] David Y Lee, Hyoseon Ryu, Youngju Kim, Sang-Gook Lee, and Soobeom Shin. Predicting student retention in massive open online courses using hidden markov models. *Journal of Computer Assisted Learning*, 32(5):368–389, 2016.

[16] Leah P Macfadyen and Shane Dawson. Mining lms data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*, 54(2):588–599, 2010.

[17] Teresa Martín-Blas and Ana Serrano-Fernández. The role of new technologies in the learning process: Moodle as a teaching tool in physics. *Computers & Education*, 52(1):35–44, 2009.

[18] Nicolas Michinov, Sophie Brunot, Olivier Le Bohec, Jacques Juhel, and Marine Delaval. Procrastination, participation, and performance in online learning environments. *Computers & Education*, 56(1):243–252, 2011.

[19] Steven Oxman and William Wong. Adaptive learning systems: A tool for personalized education. *WestEd. Retrieved from https://www. wested. org/resources/adaptive-learning-systems-a-tool-for-personalized-education*, 2014.

[20] Alejandro Peña-Ayala. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications*, 41(4):1432–1462, 2014.

[21] Cristóbal Romero and Sebastián Ventura. Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(1):12–27, 2013.

[22] Cristobal Romero and Sebastian Ventura. Educational data mining and learning analytics: An updated survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(3):e1355, 2020.

[23] Piers Steel and Katrin B Klingsieck. Academic procrastination: Psychological antecedents revisited. *Australian Psychologist*, 51(1):36–46, 2016.

[24] Kurt VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[25] Joanne Wong, Martine Baars, Dan Davis, Tim Van Der Zee, Geert-Jan Houben, and Fred Paas. Enhancing self-regulated learning by using personalized prompt questions: A review of research. *Educational Psychology Review*, 31:229–264, 2019.

[26] Wanli Xing, Rui Guo, Eva Petakovic, and Sean Goggins. Participation-based student final performance prediction model through interpretable genetic programming: Integrating learning analytics, educational data mining and theory. *Computers in Human Behavior*, 47:168–181, 2015.

[27] Ji Won You. Identifying significant indicators using lms data to predict course achievement in online learning. *The Internet and Higher Education*, 29:23–30, 2016.

[28] Barry J Zimmerman. Self-regulated learning: Theories, measures, and outcomes. *International Encyclopedia of the Social & Behavioral Sciences*, 21:541–546, 2015.