EXPLAINABLE AI (XAI): IMPROVING AT-RISK STUDENT PREDICTION WITH THEORY-GUIDED DATA SCIENCE, K-MEANS CLASSIFICATION, AND GENETIC PROGRAMMING

____

A Dissertation

Presented to

The Faculty of the Department of Library Science and Technology

Sam Houston State University

____

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Education

____

by

Ngoc Van P. Bui

August 2022

EXPLAINABLE AI (XAI): IMPROVING AT-RISK STUDENT PREDICTION WITH THEORY-GUIDED DATA SCIENCE, K-MEANS CLASSIFICATION, AND GENETIC PROGRAMMING

by

Ngoc Van P. Bui

____

ABSTRACT

Bui, Ngoc Van P., *Explainable AI (XAI): Improving at-risk student prediction with theory-guided data science, K-Means classification, and genetic programming*. Doctor of Education (Instructional Systems Design and Technology), August 2022, Sam Houston State University, Huntsville, Texas.

This research explores the use of eXplainable Artificial Intelligence (XAI) in Educational Data Mining (EDM) to improve the performance and explainability of artificial intelligence (AI) and machine learning (ML) models predicting at-risk students. Explainable predictions provide students and educators with more insight into at-risk indicators and causes, which facilitates instructional intervention guidance.

Historically, low student retention has been prevalent across the globe as nations have implemented a wide range of interventions (e.g., policies, funding, and academic strategies) with only minimal improvements in recent years (Stolk et al., 2007). In the US, recent attrition rates indicate two out of five first-time freshman students will not graduate from the same four-year institution within six years (NCES, 2020). In response, emerging AI research leveraging recent advancements in Deep Learning has demonstrated high predictive accuracy for identifying at-risk students, which is useful for planning instructional interventions.

However, research suggested a general trade-off between performance and explainability of predictive models (Arrieta et al., 2020; Gunning et al., 2019). Those that outperform, such as deep neural networks (DNN), are highly complex and considered black boxes (i.e., systems that are difficult to explain, interpret, and understand). The lack of model transparency/explainability results in shallow predictions with limited feedback prohibiting useful intervention guidance. Furthermore, concerns for trust and ethical use are raised for decision-making applications that involve humans, such as health, safety, and education.

To address low student retention and the lack of interpretable models, this research explored the use of eXplainable Artificial Intelligence (XAI) in Educational Data Mining (EDM) to improve instruction and learning. More specifically, XAI has the potential to enhance the performance and explainability of AI/ML models predicting at-risk students. The scope of this study includes a hybrid research design comprising: (1) a systematic literature review of XAI and EDM applications in education; (2) the development of a theory-guided feature selection (TGFS) conceptual learning model; and (3) an EDM study exploring the efficacy of a TGFS XAI model.

The EDM study implemented K-Means Classification for explorative (unsupervised) and predictive (supervised) analysis in addition to assessing Genetic Programming (GP), a type of XAI model, predictive performance, and explainability against common AI/ML models. Online student activity and performance data were collected from a learning management system (LMS) from a four-year higher education institution. Student data was anonymized and protected to ensure data privacy and security. Data was aggregated at weekly intervals to compute and assess the predictive performance (sensitivity, recall, and f-1 score) over time. Mean differences and effect sizes are reported at the .05 significance level. Reliability and validity are improved by implementing research best practices (J. Cohen, 1988; Field, 2018; He et al., 2016).

KEYWORDS: Explainable artificial intelligence; XAI; Genetic programming; Machine learning; Educational data mining; Learner analytics; At-risk student prediction; Student retention

ACKNOWLEDGEMENTS

I would like to express my utmost gratitude to those who have provided support, patience, care, and friendship during my journey here at SHSU. Without you, this work would not have been possible.

Thank you, Dr. Donggil Song, for guiding me from the start. It was your work, dedication, guidance, support, and mentorship that have helped me succeed in more ways than you would ever know.

Thank you, Dr. Sheng-Lun Cheng, for your mindfulness, dedication to student learning, critical feedback, and shared insights that have helped me continue to learn, grow, and improve.

Thank you, Dr. William Angrove, for your suggestions and encouragement on distance learning. You have inspired me to discover how instructional technology can positively impact learning in a connected society.

Thank you, to my family and friends, for your care, love, and patience throughout my academic pursuits.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

Introduction

The research contributes to the field of Instructional Systems Design and Technology (ISDT) by designing, developing, and evaluating an at-risk student predictive model using Educational Data Mining (EDM) process grounded in instructional theory and evidence-based practices. To prepare for a fast-evolving and technology-driven society, this work leverages emerging computational techniques to support and inform instructional practice. This work explored how improving the explainability of AI models can help identify interaction characteristics of at-risk students as well as improve the prediction performance of existing systems.

Recent advancements in AI such as Deep Neural Networks (DNN) have demonstrated unprecedented performance in pattern recognition, automation, and prediction (Adadi & Berrada, 2018). As a result, there has been widespread research and adoption of AI across diverse fields and applications (Arrieta et al., 2020). In education, AI models have been used in EDM to predict at-risk students and enable timely interventions for impro student performance (Chitti et al., 2020), which are associated with higher student retention (Baldelovar, 2016; McCoy & Byrne, 2017). Research advancing at-risk student predictions can help address the poor state of student retention, which has been slow and stagnant over the past two decades (Beer & Lawson, 2017; NCES, 2022; Tinto, 2006).

Although current AI implementations have demonstrated high accuracies for predicting at-risk students, emerging research suggests a general trade-off between predictive performance and explainability (Gunning & Aha, 2019). AI models that outperform, such as deep neural networks (DNN), are highly complex, inherently unexplainable, and are considered black boxes, which are systems that are too difficult for humans to explain, interpret, and understand (Knight, 2017). The lack of transparency in at-risk student predictions is evident in educational data mining (EDM) research, which prompted the need for more interpretable models (Chitti et al., 2020).

In response to the lack of transparent AI, the field of Explainable AI (XAI) emerged with goals to improve model informativeness, trustworthiness, causality, transferability, confidence, fairness, accessibility, interactivity, and privacy (Arrieta et al., 2020). This research contributes to XAI by exploring selected explainable techniques and explainable models for improving at-risk student predictions. More specifically, this study evaluates the performance and explainability of K-Means Classification, an explainable technique, and Genetic Programming (GP), and explainable model, using a developed Theory-Guided Feature Selection (TGFS) model that incorporates relevant theories such as Activity Theory, Significant Learning Model, and Social Learning Theory.

Statement of the Problem

This research addresses two primary problems: (1) the need to improve the poor state of student retention in higher education; and (2), the need to use Explainable Artificial Intelligence (XAI) to improve the performance and explainability of at-risk student predictions. The ability to improve the accuracy and explainability of at-risk predictions can lead to better feedback for instruction interventions and lower student attrition attributed to poor performance outcomes. Furthermore, model explainability helps improve user trust and adoption in practice. The proceeding paragraphs will give a brief review of the problems as well as the benefits that XAI offers.

First, the problem of student retention is evident in the low and stagnant graduation rates (averaging 56% and 33% for four-year and two-year institutions respectively from 1996 to 2016) of higher education in the US over the past two decades (Beer & Lawson, 2017; NCES, 2022; Tinto, 2006). Compared to the Organization for Economic Co-operation and Development (OECD) (78%) and European countries (79%) (2020), the US national averages are substantially lower. The lack of progress highlights the significant waste of financial, time, and labor investments from students, institutions, and governments and the need for more effective interventions.

In addition, student attrition statistics from Wellman et al. (2012) show that students leave due to both academic and non-academic factors. McCoy and Byrne's (2017) literature review identified contributing factors such as diversity, culture, socio-economic background, academic performance, financial aid, social support, national policies, and standards. The wide array of contributing factors presents the need for diverse research and intervention improvements with considerations for student, institution, and government contexts. The scope of this research focuses on improving student retention from an academic performance perspective by improving at-risk student predictions using state-of-the-art XAI techniques and models.

Second, the need to use XAI can be explained by first understanding the importance and current state of using AI in education. The recent advancements in artificial intelligence (AI), machine learning (ML), availability of large datasets, and distributive computing have enabled unprecedented performance in pattern recognition, automation, and prediction (Adadi & Berrada, 2018; Gunning et al., 2019; Khare et al., 2018; Ramalingam et al., 2018; Samek & Müller, 2019). However, emerging works indicate a trade-off between prediction performance and model interpretability or explainability (i.e., how easy it is for humans to understand a model) (Arrieta et al., 2020; Gunning et al., 2019). High-performing models are typically too complex and hard to interpret or explain, which raises concerns about trust, ethical use, safety, and technology adoption (Adadi & Berrada, 2018; Arrieta et al., 2020; Crowe & LaPierre, 2018; Samek et al., 2021; Samek & Müller, 2019; Sun & Medaglia, 2019; Zhai et al., 2021). Unfortunately, these challenges are evident in existing early warning systems as their predictions lack meaningful insight or informative feedback for intervention guidance (Cano & Leonard, 2019).

This research addresses these concerns by exploring eXplainable AI (XAI) research to improve its trust, reliability, transparency, performance, adoption, and acceptance. In addition to improving educational outcomes, this study addresses the lack of XAI literature for educational applications (Abdul et al., 2018; Adadi & Berrada, 2018; Arrieta et al., 2020; Gilpin et al., 2019; Gunning et al., 2019; Gunning & Aha, 2019; Samek et al., 2021; Samek & Müller, 2019; Vilone & Longo, 2020). Building upon the limited and emerging works in XAI, this research responds to the prompt for adopting Theory-Guided Data Science (TGDS) to enhance predictive performance and explainable results in higher education (Pillay, 2020; Xing et al., 2015). Implementing TGDS has the potential to improve prediction performance as well as address the lack of theoretical grounding in EDM and Learner Analytics research (Clow, 2013; Conijn et al., 2017). This research implemented an EDM protocol to analyze learner LMS data of a dual-mode (i.e., traditional and online) higher education institution to develop, assess, and analyze explainable techniques and explainable models for predicting at-risk learners.

Purpose of the Study

The purpose of this study is to explore the efficacy of XAI to improve the performance and explainability of at-risk student predictions. More specifically, this study will implement Theory-Guided Data Science (TGDS), K-Means Clustering, and Genetic Programming (GP), all of which have been demonstrated in early works to enhance predictive performance and explainable results in higher education (Pillay, 2020; Xing et al., 2015). This study proposes a novel theory-guided feature selection (TGFS) conceptual learning model, which aims to improve model explainability as well as addresses the lack of theoretical grounding in Educational Data Mining (EDM) and Learner Analytics research (Clow, 2013; Conijn et al., 2017). EDM protocols from the literature are leveraged to provide a robust and reliable data collection, research, and analysis process within an online Learning Management System (LMS) of a dual-mode higher education institution.

Research Questions

The following are the XAI EDM study research questions to be investigated:

- RQ1: What are the associations between activity factors and student final grades?
- RQ2: What are characteristics of at-risk students that can be identified using K-Means Cluster Analysis?
- RQ3: What are characteristics of at-risk students that can be predicted using XAI techniques and models?

- RQ4: How do explainable models compare to complex models when LMS student activities are used as features to predict at-risk students?

Significance of the Study

This study addresses concerns, challenges, and gaps in the current literature by contributing innovative research in the field of XAI research with a specific focus on improving educational outcomes. First, this study contributes to the critical need to improve student retention in higher education to address the historically low and stagnant progress (NCES, 2020; Tinto, 2006). Although emerging AI technologies such as Deep Neural Networks are highly effective at predicting at-risk students, they are considered black-box systems, which lack explainability, are difficult to understand, and may raise concerns for trust, fairness, safety, ethics, regulatory compliance, and technology acceptance (Arrieta et al., 2020). Second, this study contributes to the gap in XAI research in the educational domain, is which evident by the scarce number of works cited in recent explainable systematic literature reviews (Adadi & Berrada, 2018; Anjomshoae et al., 2019; Arrieta et al., 2020; Vilone & Longo, 2020). And third, this study focuses on online learning environments, which enable EDM for XAI implementation while promoting technological sustainability and educational equity thanks to the widespread accessibility of distance learning. This research has broader implications as XAI research contributes to fields beyond education, such as those in the medical, government, and commercial industries.

Implications to the Field of ISDT

What are the implications of XAI and EDM research to online learning and more specifically to the field of Instructional Systems Design and Technology (ISDT)? this question can be answered by first framing the definition of the field. According to Reiser and Ely (1997), the Association of Education Communication and Technology (AECT) defined Instructional Technology as “the theory and practice of design, development, utilization, management, and evaluation of processes and resources for learning” (p. 69). This research aligns well with this definition in two ways to advance research and practice in the field.

First, this research implemented a theory-guided feature selection model (TGFS) that integrates learning theories such as Activity Theory, Bruner’s Social Learning Theory, and Fink’s Significant Learning Theory. By leveraging theory, at-risk predictions and the explainability of those predictions can be improved (Xing et al., 2015). Furthermore, theory-guided models address shortcomings in data-centric approaches, such as EDM (Karpatne et al., 2017). TGDS also improves the chance of implementing influential factors with causal relationships that are grounded in sound instructional principles. When incorporated into future at-risk student detection and prediction systems, TGFS can provide targeted explanations that map back to theory to provide intervention guidance.

Second, this research evaluates the predictive performance and explainability of theory-grounded XAI models. Evaluative research provides preliminary evidence that can alleviate the cost of implementing and testing in practice. In addition, when implemented as an early warning system, XAI models can provide more formative feedback to both instructors and students. For example, early warning at-risk indications can provide teachers and faculty with valuable information about students that may most likely fail, and therefore, need additional special attention and intervention. It allows educators to focus their resources to improve the overall success of most students.

In summary, this research aligns with the field of ISDT by integrating theory with the design, development, and evaluation of XAI models to improve learning and instruction within online learning environments. Leveraging EDM research and processes ensures an effective, efficient, and comprehensive approach to training and predicting learning outcomes based on student LMS interaction data.

Limitations

This section identifies the limitations of the study, which are “factors that may affect the study but which are out of the control of the researcher” (Ostler, n.d., p. 6). Limitations in this study are acknowledged for transparency and mitigated, when possible, to enhance validity, reliability, and robustness.

First, samples collected from the target population for the XAI EDM study were limited to student activity and performance parameters and did not include demographic or prior performance parameters. As such, sampling demographic could not be directly computed. To address this limitation, the central limit theorem was leveraged by obtaining large random samples (greater than 30) of the population, which allows for the assumption that the sampling distribution is normal with the mean equal to the population mean (Field, 2018). This in effect allows us to approximate the sampling statistics using inferential statistics (Kwak & Kim, 2017). Therefore, the population demographic, which was retrieved from NCES (2021), is reported instead of the sampling statistics.

Second, there were limitations in computing hardware such as time, memory, and processing constraints due to the large size of the LMS activity logs retrieved. To address this, data was extracted from the LMS database in smaller weekly segments and bounded to LMS activity logged during the spring semester of 2021. The data was also imported into a local LMS database that allowed for the processing and analysis of data that eliminated the restriction of working memory.

Third, unbalanced data sets were observed during the clusters obtained during k-Means clustering, which have negative implications on model accuracy and post-hoc statistical analysis (Banerjee et al., 2018; Field, 2018). Consequently, Banerjee et al. (2018) found that “sampling methods can have a major influence on reducing the gap between sensitivity and specificity of a model” (p. 362). For asymmetric class distribution, the specificity and sensitivity of the minority class could be adversely impacted and result in a high number of false negatives and false positives (Banerjee et al., 2018). To address this, stratified k-fold cross-validation was selected during supervised learning, which prevents overfitting and creates “an unbiased estimate of the population proportion” (Berrar, 2018, p. 4).

To address post-hoc analysis impacts, random under-sampling will be used to obtain balanced data set for mean difference analysis. For large samples (greater than 30), the sampling distribution will be normally distributed according to the central limit theorem regardless of the underlying population distribution characteristic (Field, 2018). This allows for the use of parametric tests for statistical analysis.

Delimitations

This section identifies the delimitations of the study, which are “factors that may affect the study over which the researcher has control… [to] …set boundaries or limits” for feasibility and scope (Ostler, n.d., p. 6).

First, as part of the EDM data tuning and normalizing process, a substantial number of samples were removed from the population due to outliers, errors, missing values, or inconsistent data. This may introduce sampling exclusion bias as the data set will not be inclusive of all students.

Second, the study is limited to 15-week LMS data collected from a Doctoral University of High Research Activity by The Carnegie Classification of Institutions of Higher Education with a large online student population (25 percent of students study exclusively online). This delimitation was necessary to keep the study feasible and within scope while targeting online students, which is of particular interest for this study as it facilitates EDM for predictive modeling. Limiting participants to a single institution will also mitigate concerns of volunteer bias that is common in soliciting a wider internet population (L. Cohen et al., 2018, p. 372).

Third, this study excluded performance-related metrics from the feature selection due to inconsistent implementation of student assessment metrics (e.g., grade scores from assignments and quizzes) across courses. As a result, features selected for at-risk student prediction were isolated to student learning management system (LMS) logged activity frequencies (e.g., access counts to pages, tools, and functions within the LMS environment). Thus, this study is primarily focused on predicting at-risk students based on activity-only related features.

Assumptions

As mentioned previously, random sampling of sufficient size will be obtained to leverage the central limit theorem to assume that the demographic makeup of the randomly selected samples (i.e., sample mean) is representative of the population demographic (i.e., fits a normal distribution of the population sample means), which was retrieved from NCES (2021). In addition, stratified k-fold cross-validation will be used for training/test split sampling for classification. Random undersampling will be used for post-hoc statistical analysis, such as association comparative analysis or hypothesis testing.

For K-Means cluster analysis, outliers are removed to alleviate skewed effects and feature frequency values normalized due to inconsistent variables scales with large standardized deviations as recommended by Cohen et al. (2018). K-Means clustering assumes that variables are continuous with spherical clustered shapes (Everitt et al., 2011). Typically used as a post-hoc analysis technique to discover patterns, clustering can be useful to the group and categorize data for subsequent analysis (L. Cohen et al., 2018). For this study, the clusters and their characteristics define the at-risk groups' ground truths labels, which are used in the subsequent classification and comparative analysis.

To ensure the validity and reliability of analytic findings, an analysis of the violation of assumptions will be performed for each respective statistical and/or analytic technique. In quantitative research, assumptions to consider include statistical power, normality, linearity, sample size, variance, homogeneity, skew, kurtosis, and outlier effects (L. Cohen et al., 2018). Depending on the results of the assumption checks, appropriate statistical tests will be selected and used. For statistical significance testing, this study will adopt Cohen et al.’s (L. Cohen et al., 2018) recommendation for “statistical power at 0.80, alpha at 0.05, and beta at 0.20” (p. 752). Statistical significance is a measure of the probability (p-value) of a calculated effect due to chance. Statistical significance is defined at the $p<0.05$ level (the probability of finding an effect is 5% due to chance).

Chapter Summary

This chapter described the XAI EDM study, its guiding research questions, research design, conceptual framework, reliability and validity considerations, and implementation methods and procedures. The study implements an EDM approach that leverages XAI techniques (e.g., TGDS and K-Means Cluster Analysis) and XAI models (e.g., Genetic Programming) to improve at-risk student prediction and explanations. The scope of the study includes an exploratory analysis to characterize learning groups and a classification analysis to assess the prediction performance and explainability of a theory-guided model. Finally, an emerging Genetic Programming model will be evaluated on performance and explainability against other common leading AI/ML models.

The remaining chapters will present the relevant sections of the research, which are organized as follows: Chapter 2 provides a literature review describing the problems and background of XAI and EDM/Learner Analytics research; Chapter 3 gives provides relevant theories and conceptual models; Chapter 4 presents the XAI EDM Study, which includes the methods and discussion of results; Chapter 5 addresses reliability and limitations of our study; and Chapter 6 concludes with directions for future research.

CHAPTER II

Literature Review

Introduction

This chapter presents a literature review of the problem space revolving around student retention and the need for improving current at-risk student prediction systems using Explainable Artificial Intelligence (XAI) models in Educational Data Mining (EDM) research. The findings reported include (1) the demand, motivations, progress, and challenges of XAI; and (2) the common study types, target outcomes, and feature categories implemented in EDM research. The results of this literature review provided guidance and direction for this research.

Methods

This study adopted Xiao and Watson's (2019) systematic literature review process to enhance reliability, rigor, quality, and repeatability. A *systematic literature review* follows a rigorous process of identifying, screening, inclusion/exclusion, data extraction, and analysis of relevant past and current research. Initial sources were first identified by conducting relevant keyword searches (i.e., using search strings "*XAI AND 'explainable artificial intelligence' AND predict AND 'at-risk' AND student AND performance AND outcome AND learning*" or "*'educational data mining' AND 'Learner Analytics'*") in the Google Scholar and EBSCO databases. Selected works were then screened to remove duplicates based on the title name. Abstracts are reviewed to filter items based on the inclusion criteria of being XAI and/or EDM relevant studies. A full-text review was then performed to extract, codify, and categorized common themes, key findings, and limitations of XAI and EDM research.

In total, 16 sources (Abe, 2019; Al Breiki et al., 2019; Al-Omar, 2018; R. S. J. D. Baker et al., 2011; Berens et al., 2019; Bienkowski et al., 2012; Chamizo-Gonzalez et al., 2015, p.; Conijn et al., 2017; Imran et al., 2019; Ndou et al., 2020; Pillay, 2020; Raju et al., 2020; Song et al., 2019; Xing et al., 2015; Yang et al., 2021; Zheng, 2020) were selected, reviewed, codified, and summarized. The objective was not exhaustive, but precursory to analyze, coded, categorized, and triangulate common research aims, predictor variables, features selected, analysis techniques, and study limitations, all of which helped guide and support the research.

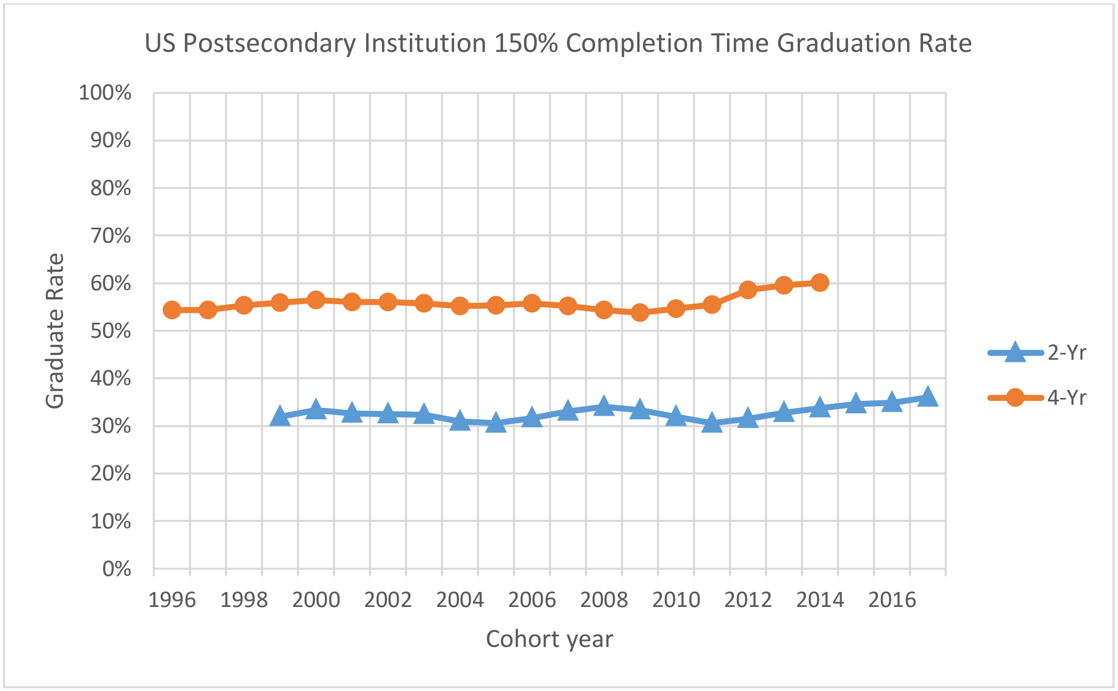
The Problem of Student Retention and the need for XAI

The Need to Address the Low and Stagnant Student Retention

Improving student retention has been an ongoing challenge, both in the US and the world according to the National Center for Educational Statistics (2022) and Beer and Lawson (2017). This is evident in the historically low student retention rate and stagnant progress (Tinto, 2006), which can be observed from 1996 to 2010 with only marginal improvements thereafter (see Figure 1). On average, only 56% graduate within six years from a four-year institution, and, worse, 33% graduate within three years from a two-year institution. When compared to other nations, the US graduation rate (60%) in 2018 was substantially lower than the Organization for Economic Co-operation and Development (OECD) (78%) and European countries (79%) (OECD, 2020). In addition to increasing student burdens, attrition costs are significant and widespread for higher education institutions across the globe (2017).

Figure 1

US Postsecondary Institution 150% Completion Time Graduation Rate



Note. Data from the National Center for Education Statistics (NCES) Trend Generator (NCES, 2022). In the public domain.

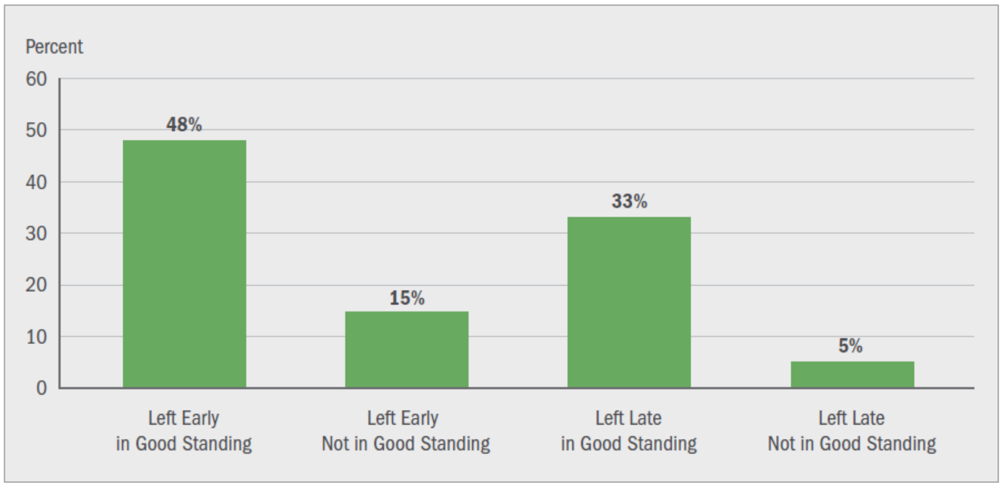
These statistics highlight the critical need to improve student retention in the US not only for national and institutional progress but more importantly, student success. Addressing student attrition is important as it reflects (Long et al., 2006): the waste of institutional cost and government-funded expenditures in producing an outcome; the student financial and emotional burden; a critical measure of university quality and educational support services; educational equity; and waste of potential talent.

The Need for Explainable AI (XAI) in Education

Implementing effective learning interventions requires a grasp of the underlying causes of student attrition, the effectiveness of implementations in practice, and the current state of research. Prior research has identified a wide range of root causes stemming from the socio-economic background, financial aid support, family background, race, culture, work status, personal choices, class size, and academic preparation/performance (Beer & Lawson, 2017; Stolk et al., 2007; Wills et al., 2018). To address this, many nations, states, and local governments have implemented various interventions such as student support policies, financial aid, and academic support programs with mixed results (Stolk et al., 2007). Stolk et al. (2007) found that determining consistent factors and interventions was difficult due to the varying nature of countries, institutions, cultures, and demographic backgrounds. Nevertheless, a common and robust measure of student attrition commonly referenced by the authors was academic performance. Concerning academic performance, Wellman et al. (2012) characterized two groups of student attrition: (1) those in good standing that leave early (48%) or late (33%); and those not in good standing that leave early (15%) or late (5%) (see Figure 2). It is the latter group (i.e., 20% of students not in good standing or at-risk) that is the target of this research.

Figure 2

Attrition Timing and Academic Standing of Students



Note. Chart showing the percentage of student attrition for time and academic standing. Reprinted from Measuring (and Managing) the Invisible Costs of Postsecondary Attrition (p. 6), by J.

Wellman, N. Johnson, and P. Steele, 2012, American Institute for Research (*https://files.eric.ed.gov/fulltext/ED536120.pdf*).

A common strategy in addressing student retention involves the use of early warning systems that collects and monitors student activity/performance to analyze and provide real-time indications of at-risk students. Researchers have explored how educational data mining (EDM) and learning analytics can be used to design, model, and predict student outcomes for guiding instructional interventions (Alshammari et al., 2013; Du et al., 2020; Şahin & Yurdugül, 2020). An area of promise in predicting at-risk students includes the use of artificial intelligence (AI) and machine learning (ML). The advancements in deep learning, availability of large datasets, and distributed parallel computing have enabled AI/ML to achieve unprecedented performance and accuracy in pattern recognition, automation, and prediction (Adadi & Berrada, 2018; Gunning et al., 2019; Khare et al., 2018; Ramalingam et al., 2018; Samek & Müller, 2019). This is evident and popularized by recent AI breakthroughs.

For example, headline news in 2016 of Google's AlphaGo beating the world's leading professional, Lee Se Dol, in the game of Go highlighted a pivotal case where machines exceeded human cognitive performance (Moyer, 2016; Silver et al., 2017). Kwon (2020) later performed a probability distribution analysis of AlphaGo's game reading and decision-making ability against that of professional go players and found that it had statistically "surpassed human abilities" (p. 1). This is just one of many examples in which AI has demonstrated task superiority over humans, among others in "complex visual tasks" (Samek & Müller, 2019, p. 1) such as in medical imaging pattern recognition (Subbiah et al., 2020). As a disruptive technology, AI has the potential to advance knowledge work automation tasks, which may supplement or even replace large market sectors involving human cognitive abilities (Manyika et al., 2013).

However, the high performance of AI models typically comes at the cost of explainability, which is the interpretability or understandability of a system or model (Arrieta et al., 2020; Gunning et al., 2019). Due to their highly complex nature, most AI models lack explainability. This raises concerns for trust, ethical use, and safety that can inhibit technology adoption in applications involving humans, such as transportation, healthcare, legal, military, finance, and engineering (Adadi & Berrada, 2018; Crowe & LaPierre, 2018; Samek et al., 2021; Samek & Müller, 2019; Sun & Medaglia, 2019; Zhai et al., 2021). For decision-making applications in education, the lack of explainability can negatively impact trust, fairness, educational equity, and technology acceptance. Non-explainable models result in shallow feedback that limits meaningful insight needed for effective learning intervention guidance.

This research addresses the limited transparency of complex black box systems by exploring the effectiveness of eXplainable artificial intelligence (XAI) for improving the performance and explainability of at-risk student predictions. As an emerging technology, XAI can be integrated into an early warning system (EWS) to provide at-risk student predictions with real-time feedback and intervention guidance for improving student learning and retention (Veerasamy, 2020). XAI encompasses explainable models and/or explainable techniques to help humans better understand, interpret, and trust machine learning (Gunning & Aha, 2019) as well as addressing model causality, transferability, fairness, and confidence (Arrieta et al., 2020).

Although XAI has been widely researched, there is limited literature addressing educational applications (Abdul et al., 2018; Adadi & Berrada, 2018; Arrieta et al., 2020; Gilpin et al., 2019; Gunning et al., 2019; Gunning & Aha, 2019; Samek et al., 2021; Samek & Müller, 2019; Vilone & Longo, 2020). When used to support teaching and learning, XAI can help teachers and students better understand the reasons and logic behind at-risk alerts to provide the meaningful insight needed for effective decision-making, intervention planning, or self-regulated learning.

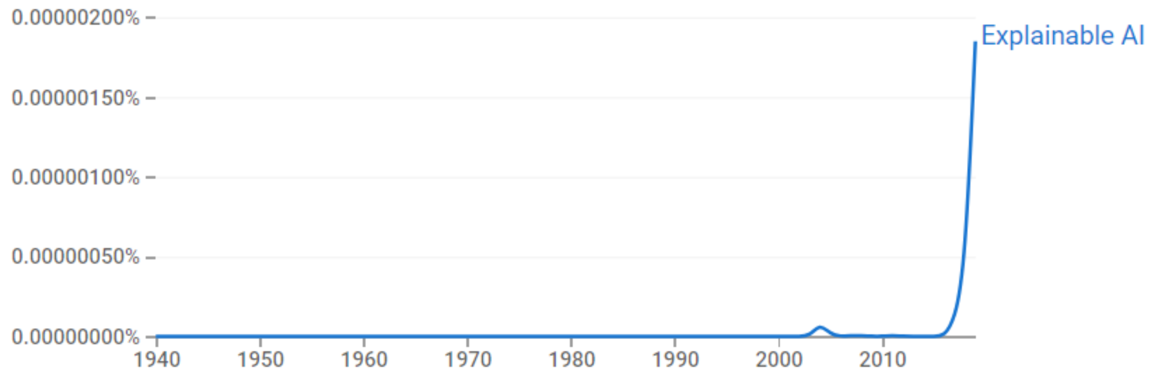
Why XAI?

The Emerging Demand for Explainability

The rapid development and spread of AI have created the need to control where decision-making applications involve humans or human safety. The potential risks of negative consequences have motivated academic and government entities to increase awareness and research in developing more explainable systems, which later became known as the field of Explainable AI, or XAI (Duval, 2019). The increased interest is evident in the sharp rise in relevant publications (Figure 3) and online Google search trends (Figure 4). Arrieta et al. (2020) also found a similar trend in increasing explainability publications (XAI and Explainable Artificial Intelligence) since 2017 (Figure 5). This trend is also evident in the open-research community based on similar XAI keyword search terms from Knoth and Zdrahal's (2012) CORE open publication database (CORE, n.d.), as illustrated in Figure 6. As the field of AI and ML continues to grow and evolve, so will the field of XAI to address its challenges and barriers to trustworthiness, safety, ease of use, and adoption.

Figure 3

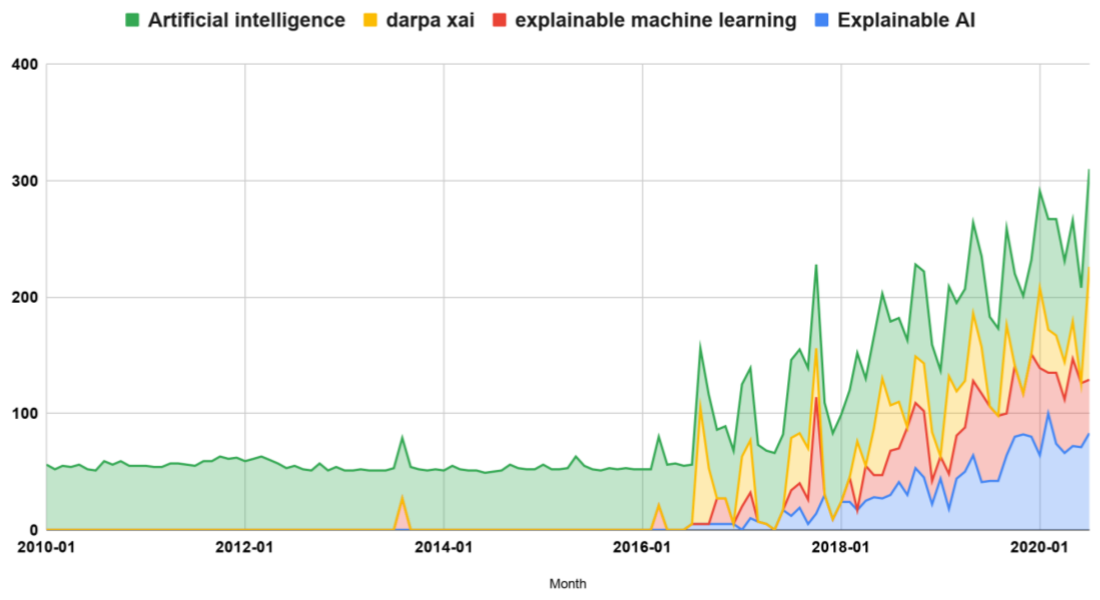
Chart Showing Publication Trends for XAI



Note. Chart retrieved from (Google Trends, n.d.) using the search string "Explainable AI" filtered for years ranging from 1940 to 2019. In the public domain.

Figure 4

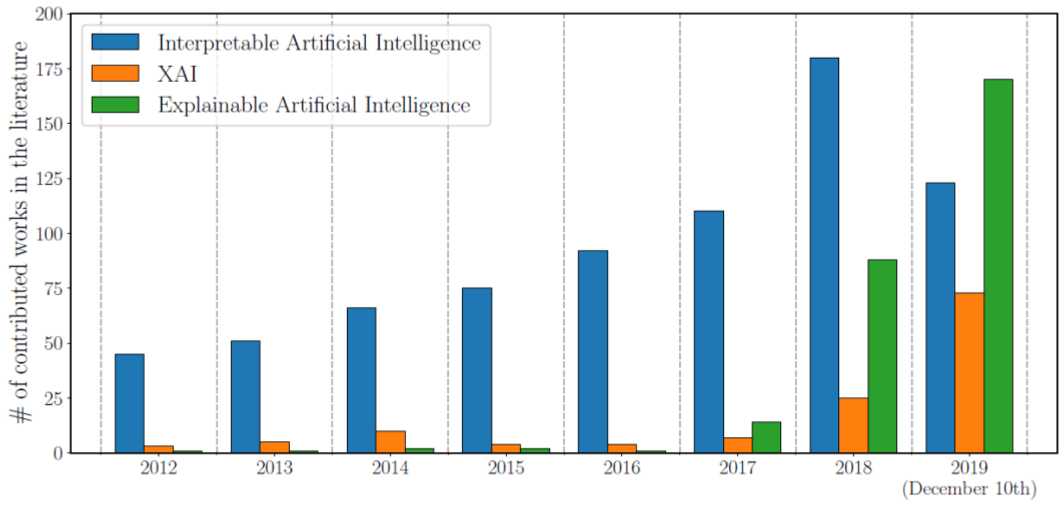
Chart Showing Google Search Trends for AI, ML, and XAI



Note. Chart adapted with data retrieved from (Google Trends, n.d.). In the public domain.

Figure 5

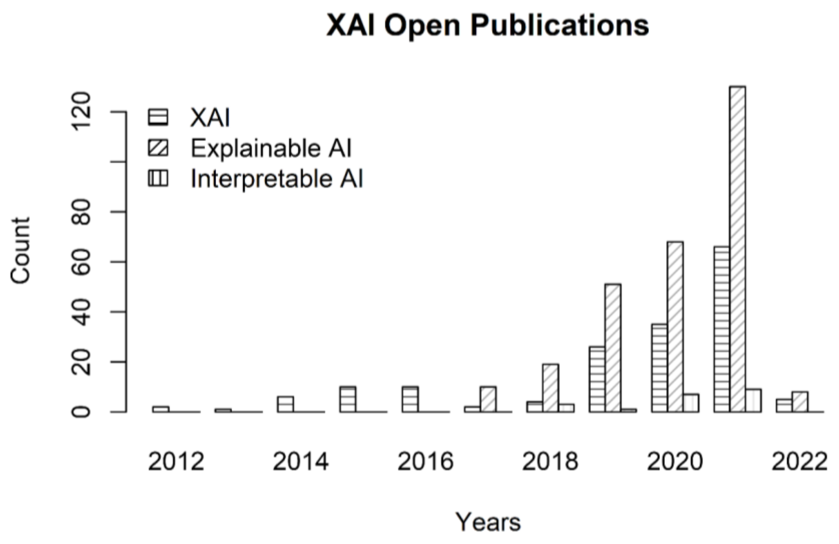
Chart of Explainability Related Publications



Note. The chart shows the rising trend in private research databases. Reprinted from (Arrieta et al., 2020, p. 3). From "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities, and challenges toward responsible AI," by Arrieta, A. B. et al., 2020, *Information Fusion*, *58*, 82–115 (https://doi.org/10.1016/j.inffus.2019.12.012). Copyright 2020 by Elsevier B.V.

Figure 6

Chart of XAI Open Publications



Note. This chart shows the number of XAI open publications based on XAI keyword search terms queried from the CORE database. Adapted from CORE API, by Petr Knoth and Zdenek Zdrahal, 2021 (https://core.ac.uk/services/api). In the public domain.

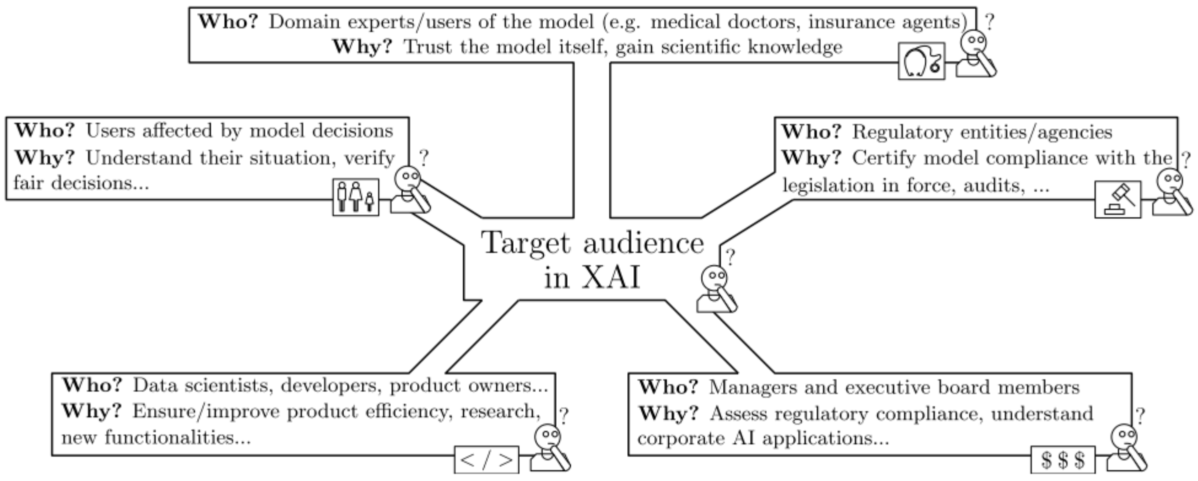
Although the interest in XAI has increased, the field is still in its infancy (evident by the low percentage of XAI publications compared to that of AI and ML in Figure 4 and Figure 5) and lacks a common definition (Arrieta et al., 2020). For this study, the following definition of XAI is adopted to take into account human understanding and target audience (both of which are critical and relevant within the educational context): "Given an audience, an explainable Artificial Intelligence… produces details or reasons to make its functioning clear or easy to understand" (Arrieta et al., 2020, p. 6).

Motivations for XAI

The motivation for this research stems from the increasing demand for human and AI interaction in multiple applications and domains, which requires XAI to address concerns for transparency, fairness, ethics, trust, and regulatory compliance. Arrieta et. al. (2020) also provide rationales for XAI based on the target audience (see Figure 7). Target audiences include domain experts, regulatory entities, managers, executive members, scientists, developers, and affected users. The benefits of XAI include gaining trust, scientific knowledge, regulatory compliance, operational efficiency, research, and increased explainability. However, there is a lack of emphasis on educators and learners, whose primary interest is to improve educational outcomes and student retention. XAI systems can help improve the learning and instruction of students and educators by providing more detailed and informative indicators of at-risk performance. An easy-to-understand, interpretable, and explainable system ultimately results in higher confidence and trust from educational users (Gunning & Aha, 2019; Shin, 2021).

Figure 7

Chart Presenting Rationales for XAI Based on Target Audience


Who? Domain experts/users of the model (e.g. medical doctors, insurance agents)
Why? Trust the model itself, gain scientific knowledge
Who? Users affected by model decisions
Why? Understand their situation, verify fair decisions...
Who? Regulatory entities/agencies
Why? Certify model compliance with the legislation in force, audits, ...
Target audience in XAI
Who? Data scientists, developers, product owners...
Why? Ensure/improve product efficiency, research, new functionalities...
Who? Managers and executive board members
Why? Assess regulatory compliance, understand corporate AI applications...
</ >
$$$

Note. From "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities, and challenges toward responsible AI," by Arrieta, A. B. et al., 2020, *Information Fusion*, *58*, 82–115 (https://doi.org/10.1016/j.inffus.2019.12.012). Copyright 2020 by Elsevier B.V.A preliminary search of the current literature also identified the following benefits of XAI:

- Supports effective decision-making:
  - In Educational Science, the need for explainability is required to improve and support effective decision-making, such as in e-Learning environments where human and AI system interaction is prevalent (Alonso & Casalino, 2019).
- Aligns with major organizational interest:
  - In 2016, the US Defense Advanced Research Project Agency (DARPA) launched the XAI program to fund research for "new or modified machine learning techniques that produce explainable models that, when combined with effective explanation techniques, enable end-users to understand, appropriately trust, and effectively manage the emerging generation of

Artificial Intelligence (AI) systems" (*Broad Agency Announcement: Explainable Artificial Intelligence (XAI) DARPA-BAA-16-53*, 2016, p. 5).

- Supports learning and instruction:
    - XAI systems not only have the potential to assist educators in designing courses but also provide students with explanations about their learning activities, which is expected to improve student perceptions and learning outcomes (Alonso & Casalino, 2019).
- Increases trust and safety:
    - “Entrusting important decisions to a system that cannot explain itself presents obvious dangers” (Adadi & Berrada, 2018, p. 52138). The lack of transparency in AI systems presents concerns of trust, especially in critical decision-making applications, such as medical diagnosis, criminal judgment, and auto-pilot vehicle systems (Adadi & Berrada, 2018; Crowe et al., 2017; Sun & Medaglia, 2019).
- Promotes technology acceptance:
    - In education, the lack of trust in black box AI raises concerns for fairness and educational equity, which become barriers to technology acceptance. XAI can provide solutions and explainability techniques to increase trust and promote technology acceptance of AI (Arrieta et al., 2020).
- Addresses regulatory compliance, fairness, and trust:
    - “AI needs to provide justifications to comply with legislation, for instance, the ‘‘right to explanation’’, which is a regulation included in the General

Data Protection Regulation (GDPR) that comes into effect across the EU on May 25, 2018” (Adadi & Berrada, 2018, p. 52143).

- “XAI systems are expected to be beneficial to society through fairness, transparency and explainability, regarding not only technical but also ethical and legal issues.” (Alonso & Casalino, 2019, pp. 2–3).
- “For commercial benefits, for ethics concerns or regulatory considerations, XAI is essential if users are to understand, appropriately trust, and effectively manage AI results” (Adadi & Berrada, 2018, p. 52142).

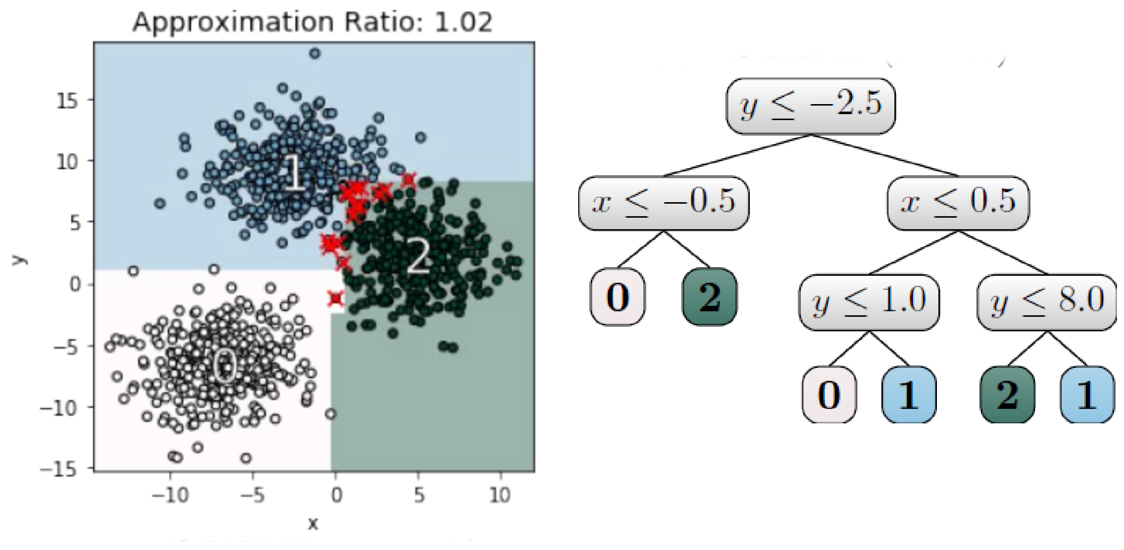
Although XAI can offer many benefits, this research focuses on improving educational outcomes. More specifically, this research will investigate the efficacy of XAI techniques and XAI models for enhancing and augmenting instruction by providing early at-risk student prediction and formative explainable feedback.

An Explainable Example

In the current literature, there are a limited number of studies exploring the use of unsupervised learning for XAI research as most focus on the interpretation of supervised techniques (Frost et al., 2020). The limitation of only using supervised techniques is that they only provide accurate predictions of at-risk students, Of the limited unsupervised implementations, Frost et al. (2020) investigate an expanding k-Means Clustering (ExKMC) methodology implementing binary threshold trees for feature characterization (see Figure 8). When kept small, binary threshold tress (i.e., a type of decision tree) are by nature easy to understand, highly interpretable (Arrieta et al., 2020; Awaji, 2018), and flexible as it supports both categorical and numerical data types without sacrificing accuracy (Song, 2021a). From Figure 8, we can see that k-Means Clustering provides adequate pattern recognition of similar groups while the binary decision tree provides an explainable logical structure facilitating meaningful insight based on input parameter values.

Figure 8

Binary Decision Tree



Note. Figure showing a binary decision tree (bottom) representation of a k-Means Cluster (top) classification. Adapted from "ExKMC: Expanding Explainable k-Means Clustering," by N. Frost, M. Moshkovitz, and C. Rashtchian, 2020, arXiv.org, p. 2 (https://arxiv.org/abs/2006.02399v2).

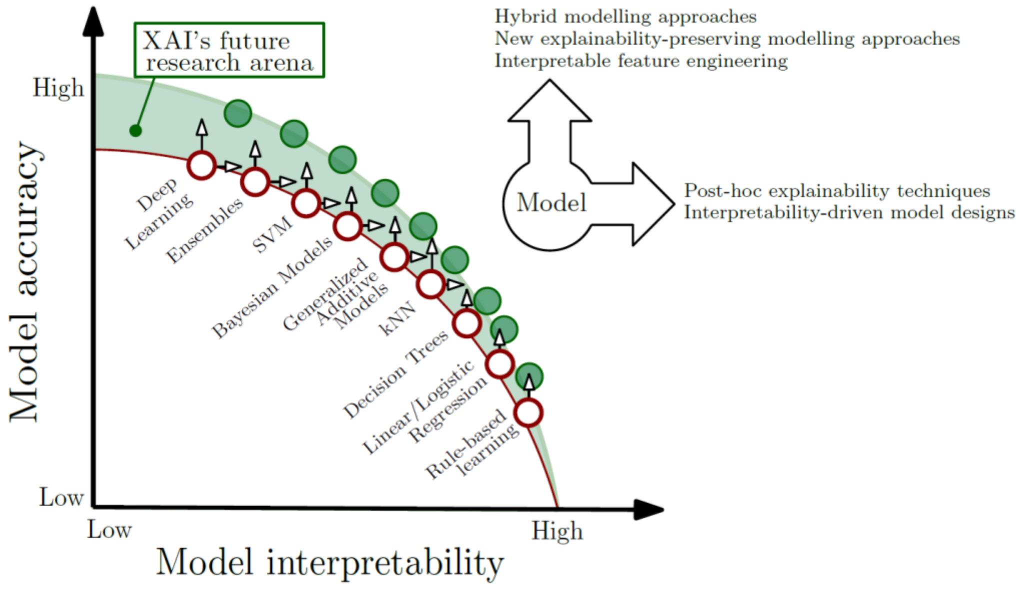
The Forefront and Challenges of XAI

The works of Arrieta et al. (2020) provide a comprehensive literature review of recent XAI research spanning 400 studies, which categorizes these models into three types: (1) transparent models (e.g. Linear/Logistic Regression, Decision Trees, K-Nearest Neighbors, Rule-based Learning, General Additive Models, and Bayesian Models); (2) shallow models (e.g. Tree Ensembles, Random Forests, Multiple Classifier Systems, Support Vector Machines); and (3) Deep Learning models (e.g. Multi-layer Neural Networks, Convolutional Neural Networks, Recurrent Neural Networks, and Hybrid Transparent Models). Transparent models have varying levels of understandability by design while shallow and Deep Learning models are considered complex and difficult to interpret black box systems.

The goal of XAI research is to further develop and increase the explainability of existing models while preserving or improving predictive performance and accuracy. In Figure 9, Arrieta et al. (2020) depict the horizon of XAI research within the green band, which represents studies implementing various methods to increase the explainability of both ML and Deep Learning models. Figure 9 also illustrates Gunning et al.'s (2019) finding that there is a general tradeoff between accuracy and explainability based on the model implemented. Explainability techniques used include implementing hybrid models and post-hoc techniques to increase the accuracy of more explainable systems while improving the interpretability of accurate models (Arrieta et al., 2020). Although XAI research spans various industries and domains, this research specifically focuses on implementing XAI for educational at-risk predictive systems while leveraging instructional theory to enhance explainability and improve educational outcomes.

Figure 9

Chart Illustrating the Tradeoff between Model Accuracy and Model Interpretability


XAI's future research arena
High
Low
Model accuracy
Low
High
Model interpretability
Deep Learning
Ensembles
SVM
Bayesian Models
Generalized Additive Models
kNN
Decision Trees
Linear/Logistic Regression
Rule-based learning
Hybrid modelling approaches
New explainability-preserving modelling approaches
Interpretable feature engineering
Model
Post-hoc explainability techniques
Interpretability-driven model designs

Note. From "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," by Arrieta, A. B. et al., 2020, *Information Fusion*, *58*, 82–115 (https://doi.org/10.1016/j.inffus.2019.12.012). Copyright 2020 by Elsevier B.V.

Educational Data Mining and Learner Analytics

Common Study Types

Most of the literature reviewed implemented EDM and/or Learning Analytics to collect student LMS data, analyze and classify patterns, and build predictive models to provide early detection of at-risk students and intervention guidance in hopes to improve educational outcomes and student retention. From Table 1, study types found include predictive (13), analysis (3), correlation (2), comparative (1), and frequency analysis (1). We also included literature reviews (3) to provide additional insight that may have been missed from the limited number of studies selected. Most studies (13) focused on predicting student learning outcomes, such as final grade or pass/fail result, which is consistent with (Conijn et al., 2017). Note that some works had multiple studies, which we counted as separate experimental implementations.

Table 1

Target Study Types

Study Types	**Sources**	**Studies**
Predictive	(Abe, 2019; Al Breiki et al., 2019; R. S. J. D. Baker et al., 2011; Berens et al., 2019; Conijn et al., 2017; Imran et al., 2019; Ndou et al., 2020; Xing et al., 2015; Zheng, 2020)	13
Literature Review	(Bienkowski et al., 2012; Pillay, 2020; Raju et al., 2020)	3
Analysis	(Al-Omar, 2018; Song et al., 2019; Yang et al., 2021)	3
Correlation	(Abe, 2019; Chamizo-Gonzalez et al., 2015)	2
Comparative	(Al Breiki et al., 2019)	1
Frequency Analysis	(Al Breiki et al., 2019)	1
Grand Total		**23**

Common Target Outcomes

Table 2 gives a frequency summary of the target study outcomes of the experimental studies reviewed (excluding three literature reviews) categorized by category predicted or analyzed, which includes student achievement (credits earned), behavior (personality traits), perceived usability (sentiment), performance (activity, final exam, final grade, and pass/fail result), preparations for learning (PFL test score), and retention (dropout). Note that student performance had the highest research interest with a primary focus on predicting final grades.

Table 2

Target Study Outcomes

Outcome Categories	Outcome Variables	Studies
Achievement	Credits Earned (Abe, 2019)	1
Behavior	Student Personality Traits (Zheng, 2020)	1
Perceived Usability	Sentiment (Al-Omar, 2018)	1
Performance	Activity (Xing et al., 2015)	1
	Final Exam (Yang et al., 2021)	1
	Final Grade: 4 studies from (Al Breiki et al., 2019); and (Chamizo-Gonzalez et al., 2015; Conijn et al., 2017; Ndou et al., 2020; Song et al., 2019)	8
	Pass/Fail (Abe, 2019; Imran et al., 2019)	2
Preparations for Future Learning (PFL)	PFL Test Score (R. S. J. D. Baker et al., 2011)	1
Retention	Dropout (Berens et al., 2019)	4
Grand Total		**20**

Table 3 gives the study distribution of the predicted target outcome variables (excluding three literature reviews) sorted by studies implemented. Predicted categories found include performance (activity, final exam, final grade, and pass/fail result), student achievement (credits earned), behavior (personality traits), perceived usability (sentiment), and preparations for learning (PFL test score), and retention (dropout). The performance category had the highest research implementations with a primary focus on predicting final grades. This study will investigate factors impacting final grade, as it is the most common target outcome to predict.

Table 3

Target Predictor Variables

Predicted Categories	Variables	Studies Implemented
Performance	Final Grade	(Al Breiki et al., 2019; Chamizo-Gonzalez et al., 2015; Conijn et al., 2017; Ndou et al., 2020; Song et al., 2019)
	Pass/Fail	(Abe, 2019; Imran et al., 2019)
	Activity	(Xing et al., 2015)
	Final Exam	(Yang et al., 2021)
Retention	Dropout	(Abe, 2019; Berens et al., 2019)
Achievement	Credits Earned	(Abe, 2019)
Behavior	Student Personality Traits	(Zheng, 2020)
Perceived Usability	Sentiment	(Al-Omar, 2018)
PFL	PFL Test Score	(R. S. J. D. Baker et al., 2011)

Common Feature Categories

Table 4 summarizes the feature selection categories sorted by studies implemented. The categories of features (i.e., predictor variables) include student activity, context, performance, learner characteristics, time, achievement, learner perception, concurrent courses, course content, participation, and SafeAssign Score. Among these, student activity, context, and performance were the most prominent predictors used. For this study, student LMS activity will be selected as potential features for prediction.

Table 4

Feature Categories

Features Categories	Studies Implemented
Student activity	(R. S. J. D. Baker et al., 2011; Bienkowski et al., 2012; Chamizo-Gonzalez et al., 2015; Conijn et al., 2017; Song et al., 2019; Xing et al., 2015; Yang et al., 2021; Zheng, 2020)
Context	(Abe, 2019; Berens et al., 2019; Bienkowski et al., 2012; Imran et al., 2019; Ndou et al., 2020; Zheng, 2020)
Performance	(Abe, 2019; Al Breiki et al., 2019; Berens et al., 2019; Conijn et al., 2017; Imran et al., 2019)
Learner Characteristics	(Bienkowski et al., 2012; Ndou et al., 2020; Zheng, 2020)
Time	(Bienkowski et al., 2012; Ndou et al., 2020; Song et al., 2019)
Achievement	(Abe, 2019; Berens et al., 2019)
Learner Perception	(Al-Omar, 2018; Bienkowski et al., 2012)
Concurrent Courses	(Berens et al., 2019)
Course Content	(Bienkowski et al., 2012)
Participation	(Abe, 2019)
SafeAssign Score	(Zheng, 2020)

Summary of Literature Findings

This section provides a summary of literature review findings related to EDM research using logged student activity as features for predicting students' outcomes. Table 5 highlights the selected literature and its results.

Table 5

Study Findings

Outcome Variable	**Features**	**Findings**	**Reference**
Activity	Logged interaction frequency data based on activity theory: subject (initiated activity, received activity), community (chat activity), division of labor (content activity), object (module activity, group activity).	Investigated a Theory-guided GP model, which outperformed other models in prediction accuracy and interpretability. Predicted student performance (represented by several activities completed) with higher accuracy: fitness (80.2%), sensitivity (80.3%), and specificity (80.3%). Outperformed other models (except for Naïve Bayes) in predicting at-risk (low number of activities completed) students: (fitness=89.5%; sensitivity=85%; and specificity=94.4%). As a white box model, GP is preferred over Naive Bayes as it offers explainable results useful for intervention guidance.	(Xing et al., 2015)
Credits Earned	Attendance Rate	Demonstrated a strong correlation between attendance and academic achievement (units acquired) for regular, postponed, and dropout groups.	(Abe, 2019)

(continued)

Outcome Variable	**Features**	**Findings**	**Reference**
Dropout	Age, Gender, Location, Nationality, Immigration Background, Health Insurance, Educational Background, Prior Courses Completed, Number of Concurrent Courses Enrolled, Employment Status, Relevant Exam Score, Other Exam Score, GPA, Failed Exams per Semester, Exams Not Participated per Semester, Number of No-Show Exams per Semester	Three types of tests were analyzed: based on dropout rate, average dropout rate, and performance data only. AdaBoost predictive accuracy is strong (67 - 95% from enrollment to the fourth semester) and improves with increasing semesters. Demographic prediction is low at enrollment (67% and 50% for state and private universities respectively). Accuracy is high in the fourth semester (up to 83% for private). Demographic data is only useful during enrollment and the first semester as it does not substantially improve accuracy after performance data is available. Performance data is better than demographic data at predicting dropouts.	(Berens et al., 2019)
	Average Attendance Rate	Poor classification performance at the beginning of the first semester. There was a good predictive performance in the 6th week (all methods had precision and recall > 72%) and at the end of the semester (neural network method had about 85% accuracy) for poor and good performers, but poor predictive power (precision and recall < 65%) for medium performers. Predictive performance was higher at the end of the first and second semesters.	(Abe, 2019)
	Entrance Exam Type, Entrance Exam Scores, Unit Acquisition Rate, Average Attendance Rate, Semester GPA	Prediction accuracy increases over time using the Naives Bayes method. At the end of the first year, prediction accuracy ranged from .797 to .814 for all methods.	(Abe, 2019)

(continued)

Outcome Variable	**Features**	**Findings**	**Reference**
	Unit Acquisition Rate, Attendance Rate	For students with low acquisition (rate < 37) and low attendance (< 79%), 96% (24 of 25 students) drop out of school or postpone graduation.	(Abe, 2019)
Final Exam	Student behavioral: content interaction in the form of backtrack reading rate (BRR), reading time (RT), adding annotations (AN), and deleting annotations (D-AN).	Classified three groups: Comprehensive learning group (CLG), Reflective learning group (RLG), and Selective learning group (SLG). Median learning outcome scores for the three groups were 84, 83, and 76, respectively. CLG had the highest RT, AN, and D-AN. RLG had the highest BRR and next highest RT, AN, and D-AN. CLG had the lowest all (BRR, RT, AN, and D-AN). High interaction (CLG) can be compensated with less interaction but more reflective (BRR) behaviors. Low interaction in all categories will likely lead to lower performance.	(Yang et al., 2021)
Final Grade	Biographical (grade 12 scores, year started, age, and others) and enrollment observations (socioeconomic, psycho-social, pre- and intra-collect scores, and individual attributes).	All six analysis techniques resulted in high predictive accuracy ranging from 83% up to 95% for 1st, 2nd, and final year outcomes. Random forest was most accurate.	(Ndou et al., 2020)
	Influential Course GPA	The smaller subset can offer accurate prediction (96.4%), but slightly slower than using the full course seethe highest test accuracy was obtained with Random Forest.	(Al Breiki et al., 2019)

(continued)

Outcome Variable	Features	Findings	Reference
	LMS log data (clicks, online sessions, total time online, number of course page views, irregularity of study time/interval, the largest period of inactivity, time until first activity, average time per session, number of resources viewed, number of links viewed, number of content page views, number of discussion post views, the total number of discussion posts, number of quizzes started, number of attempts per quiz, number of quizzes passed, number of quiz views, number assignments submitted, assign. (submission) views, number of wiki edits, number of wiki views, average assessment grade	Although there were statistically significant correlations for most of the predictor variables, the authors found that there was low predictive power and low portability across courses. A larger percentage of the explained variance is attributed to the student, rather than the course level. Overall, the authors found mixed effects for predictors in both size and direction when predicting final exam scores. LMS log data alone is limited. For early intervention, other variables are needed to improve predictive power. Recommended variables include learning dispositions, personality characteristics, self-disclosure data about dispositions, entry test results, learning styles, motivation, and engagement; all of which have been previously shown with a significant correlation with the final grade. The authors also stress the need for theoretical grounding. The portability of LMS predictors still needs more research.	(Conijn et al., 2017)
	Course, Model Selection Frequency	Used various classification algorithms to identify courses influencing final GPA. Frequency analysis was performed to identify the most influential courses. Findings revealed some courses were more influential than others.	(Al Breiki et al., 2019)
		Naive Bayes and SVM-based SMO algorithms were the most accurate (84.83%) classification methods. Higher accuracy was also achieved for processed data (replaced missing values) versus unaltered data.	(Al Breiki et al., 2019)

(continued)

Outcome Variable	Features	Findings	Reference
		SMOReg (SVM) performed the best concerning correlation coefficient (.9698) between predicted and actual GPA scores.	(Al Breiki et al., 2019)
	Student active and passive participation from LMS data mining. Active activities include assignment upload, forum add a post, forum update post, and forum view discussion. Passive activities include content (assignment, blog, course, forum, resource) view interactions	11 variables were found significantly correlated to positive learning outcomes for at least one of the four courses analyzed. Found that learner activity can influence learning where the type of activity may depend on the course type. The final regression model included active assignment upload, active forum posts, and passive resource view as influential factors (p < .001) contributing to positive learning outcomes. The model explains 84% of the variance in learning outcomes.	(Chamizo-Gonzalez et al., 2015)
	System Access, Session Time, Discussion Length, Discussion Quality, Conversation Length, Conversation Quality	The final grade is correlated with system access, time spent, discussion length, discussion quality, and conversation quality. No correlation was found for conversation length. System access was correlated with time spent and discussion length. There is a strong correlation between discussion quality and conversation quality. Two factors were identified comprising learner participant/interaction in online courses. Factor 1 was labeled Interaction quality based on high loadings from discussion quality and conversation quality. Factor 2 was labeled LMS-oriented Interaction based on high loadings from system access, time spent, and discussion length.	(Song et al., 2019)
Pass/Fail	33 Features including Student Grades, Demographics, Social Related Features, and School-Related Features. Of the 33, 12 were selected using a filter method using an information gain-based selection algorithm	J48 achieved the highest accuracy (95.78%) for predicting student performance.	(Imran et al., 2019)

(continued)

Outcome Variable	**Features**	**Findings**	**Reference**
	GPA, Units, Attendance Rate, Related Subject Grades, Cumulative Units	Predictive performance for various subjects can be effective (accuracy of 68 - 95%). Accuracy can be improved using related subject grades. However, accuracy performance varies depending on the subject and features used; causality is not yet clear and requires further study.	(Abe, 2019)
PFL Test Score	Student LMS Interaction Behavior Each feature is the proportion of time a specific student behavior occurs in the LMS log file. Features include: "help avoidance"; "long pauses after error messages"; "long pauses after reading on-demand hint messages"; "long pauses after reading an on-demand hint message and getting the current action right"; "off-task behavior"; "long pauses on skills assessed as known"; "gaming the system"; "contextual slip/carelessness"; and "the presence of spikes during learning the moment-by-moment learning model".	Cross-validation correlation was better than Bayesian Knowledge Tracing at predicting student performance on preparations for future learning (PFL) tests. It can predict with decent accuracy and achieves most of its predictive power in around 20% of the Cognitive Tutor activities for College Genetics, which highlights the potential for early warning at-risk learner detection. Features related to helping such as help avoidance or long pauses after on-demand hints, under certain conditions, can help predict student performance.	(R. S. J. D. Baker et al., 2011)
Sentiment	Student Usability Sentiment, Instructor Usability Sentiment	Three clusters were identified for each program type (distance students, external students, e-Learning students, and instructors). Concluded that Blackboard is reliable and well-designed but violates basic usability guidelines. Distance education students suffer the most, followed by external students. Overall, the usability of LMS is low for all user types.	(Al-Omar, 2018)

(continued)

Outcome Variable	Features	Findings	Reference
Student Personality Traits	Demographic features (age, gender, nationality, and marital status from survey data. Learning behaviors (late submission, number of attempts, and SafeAssign score) mined from Blackboard LMS. Personality traits were obtained from Big5 Framework and Ten-Item Personality Inventory (TIPI) questionnaire.	When treated as categorical variables, personality traits are best classified by neural networks. As numerical variables, support vector regression is best. Classification accuracy was low (roughly 20-45%) for all personality traits (openness, conscientiousness, extraversion, agreeableness, and neuroticism).	(Zheng, 2020)

The following provides a summary of the literature findings:

- Most EDM approaches target predictive models with student performance outcomes (grades, pass/fail, dropout) as labels. Common features include activity/behavioral data from LMS/EDM and/or performance data (grades, pass/fail status, and course types). Some include demographics or pre-assessments as well.
- Interaction is typically correlated with higher learning outcomes, but not always. Predictive accuracy increases over time. Prediction is most accurate with performance outcomes. However, activity data or demographic/pre-assessment data may be useful for early at-risk prediction when performance data is not yet available. When performance data is available, activity data adds marginal improvements in prediction accuracy.
- Incorporate, if possible, demographic, pre-course performance data, activity/behavioral data, and performance data as potential features. Data collection will be limited based on what is available within the LMS.

- To provide early at-risk prediction, predict at weekly intervals rather than at the end of the course. Predictive models should include past data to enhance prediction accuracy.
- Explainable models such as decision trees or Genetic Programming are preferred to be able to offer insightful intervention guidance thanks to improved transparency.
- There is a lack of XAI studies in education as well as a lack of EDM studies incorporating theory.

These findings provide valuable support, insight, triangulation, and guidance for this study, such as feature selection, model selection, and analytic methods.

Theoretical Supports

Introduction

This section gives a review of the theoretical supports leveraged by the XAI EDM study, such as Theory-Guided Data Science (TGDS) and its use for improving model performance and explainability; Activity Theory and its use to model human-computer interaction; Fink's Significant Learning Theory to model interactions between human, knowledge, and learning; and Social Learning Theory to consider the importance of self-efficacy in learning as well as incorporate elements of human to human interaction. Using these theoretical constructions, a novel Theory-Guided Feature Selection (TGFS) Conceptual Learning Model is proposed and described.

Theory-Guided Data Science

This study adopts Karpatne et al.'s (2017) theory-guided framework, which couples theory with predictive data science. TGDS (Karpatne et al., 2017) provides two primary benefits: (a) enhances explainability by providing scientific consistency from theory-based models; and (b) promotes scientific discovery by providing novel predictive insight from data science. Leveraging both technology and theory ensures an informed approach to embracing innovation that is sound in learning principles and instructional methods.

TGDS aims to address the limitations of a data-science-only or theory-only approach. This provides new insights into the flexibility and innovative nature of data science while improving the reliability and explainability of predictive models correlated to theory (Karpatne et al., 2017). As such, the predictive model, feature selection, and post-study explainability analysis will be supported by Activity Theory, Fink's Significant Learning model, and Bandura's Social Learning Theory.

This study extends Xing et al.'s (Xing et al., 2015) work by investigating the effectiveness of using LMS activity data for predicting at-risk students. To supplement this, we incorporated Fink's (2003) Holistic View of Active Learning, which comprises: (1) student access to information and ideas (i.e., content); (2) student experiences (e.g., doing, observing, actual, simulated, or authentic settings, etc.); and (3) leaner reflection activities (e.g., essays, discussions, etc.). Using both models, we can obtain an initial comprehensive list of educational data mining parameters for feature selection.

Activity Theory

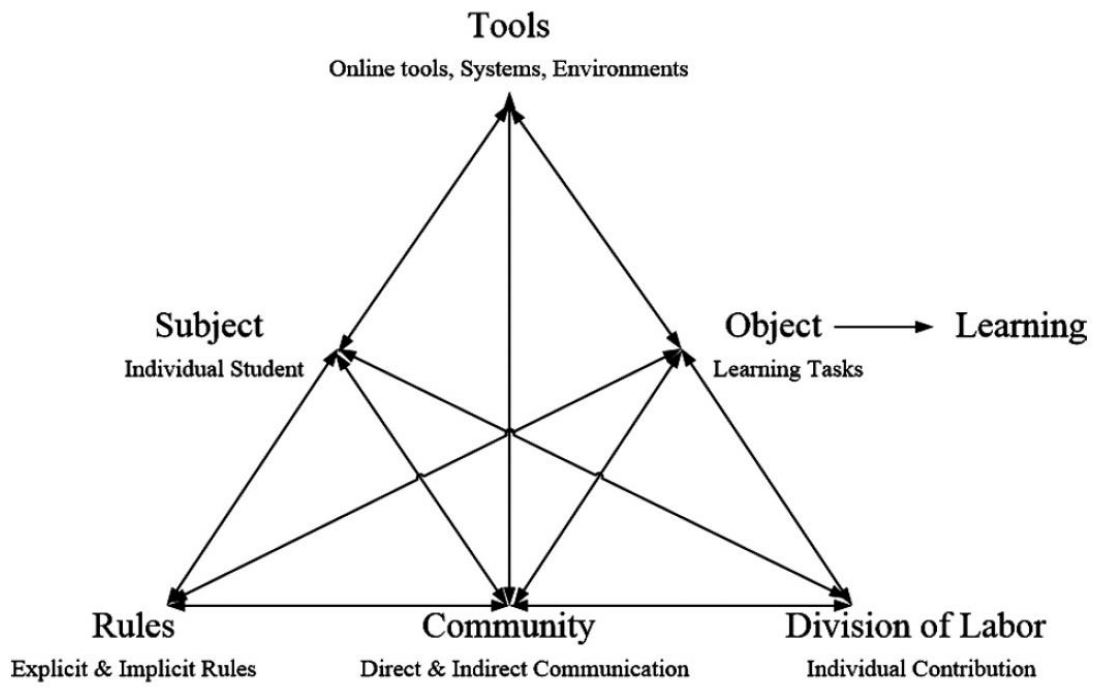
This study adopts Xing et al.'s (Xing et al., 2015) interpretation and implementation of Engeström activity theory, which posits learning as a contribution of interacting elements within a complex system. The interactions are described as tools, subjects, rules, community, and division of labor as illustrated in Figure 10. Observe that

Xing et al.'s implementation is an adaptation, in which the object represents the primary interaction of tools, rules, community, and division of labor. Note that subject represents the student and is a secondary factor that interacts with other parameters, but not the object (i.e., learning tasks) directly. Under this model, learning is assumed to be directly supported by the learning tasks implemented. Using activity theory as a guide, we propose the following interpretations concerning feature selection:

- The *subject* represents the student. Subject information such as student behaviors, demographics, skills, and knowledge can also serve as potential features.
- The *tools* are supporting applications and functions within the LMS that subjects directly interact with for learning support, such as helpful resources, plugins, and applications.
- The *rules* are constraints placed by the system, such as the LMS functions, instructors, or students themselves. For example, rubrics are constraints placed by instructors for controlling the quality of subject tasks.
- The *community* represents the contribution of social learning and can be measured by student interactions with communicating elements of the LMS (e.g., email, discussion forums, or social learning apps).
- The *division of labor* represents the individual contribution of the subject to overall learning improvement and can be measured by their interaction with the course content, such as learning modules.

Figure 10

Activity Theory



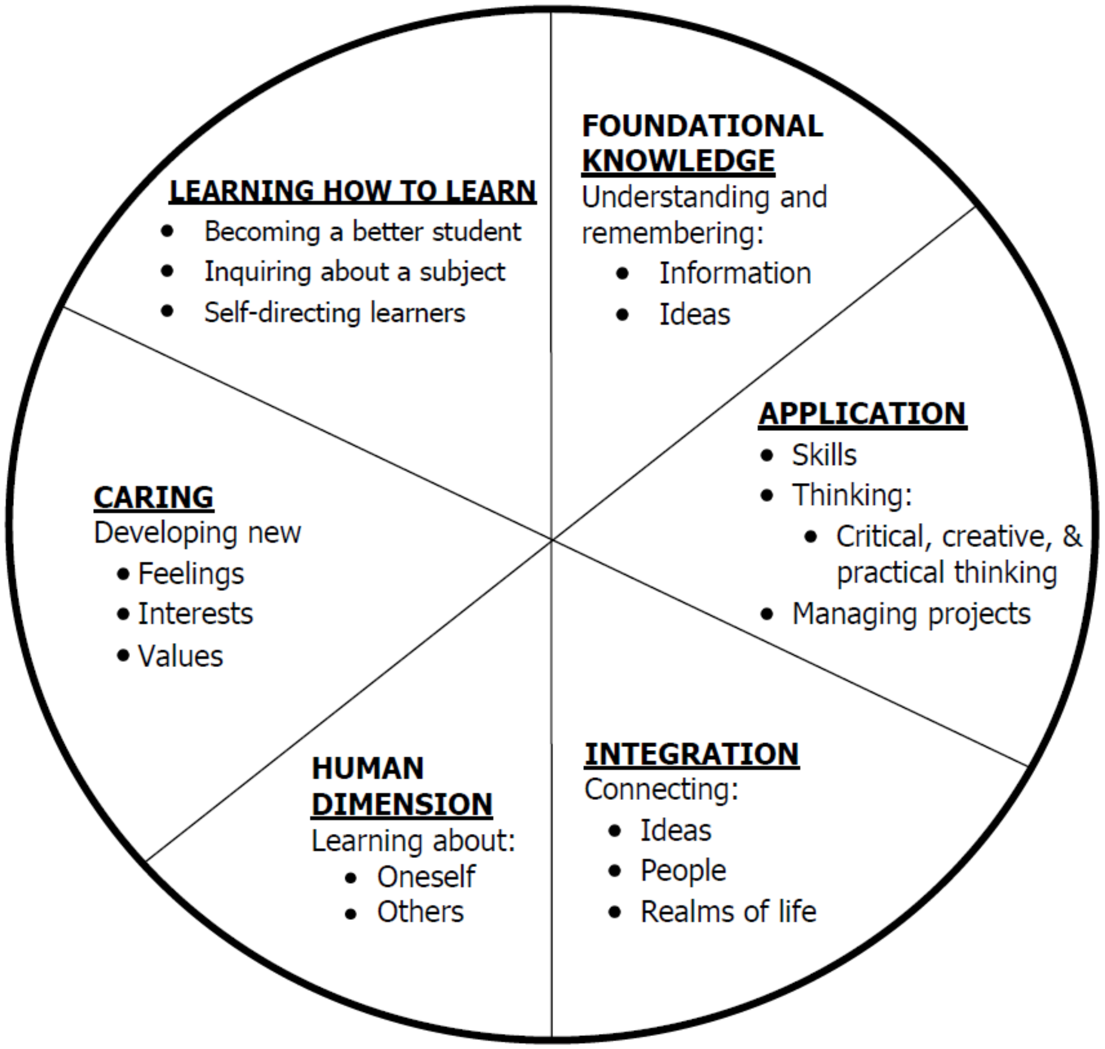
Note. Image reprinted from (Xing et al., 2015).

Fink's Significant Learning Theory

Although activity theory presents a holistic perspective of learning based on student interaction, there is a lack of emphasis on other learner-centric factors such as self-regulation, motivation, autonomy, and self-efficacy. To add focus on learner needs, we analyzed Fink's (Fink, 2003) Significant Learning theory as shown in Figure 11 for identifying possible features for student at-risk prediction.

Figure 11

Fink's Taxonomy of Significant Learning



Note. Image reprinted from (Fink, 2003).

According to Fink, learning is a complex interactive process between the six domains of foundational knowledge, application, integration, human dimension, caring, and learning how to learn. Concerning LMS activities, the following factors are applicable and considered for feature selection:

- The *human dimension* can be analogous to activity theory's community factor as both leverages the social interaction dimension. LMS activities applicable to this dimension include email, chat, forum discussions, and social media communication. The human dimension is also closely related to *integration*, which requires communication to connect ideas, people, and realms of life.

- *Foundational Knowledge* represents the information and ideas that are independently gained (e.g., interaction with content and resources) or indirectly gained (e.g., social, and interactive communication).
- *Learning How to Learn* measures a learner's self-regulated learning capability or autonomy, drive, and volition. These can be measured by their independent and self-driven actions to acquire knowledge and skills, such as accessing LMS resources and tools.

Unfortunately, logging limitations of LMS activity prevent us from using intrinsic learning dimensions such as *caring*, which requires measures such as student feedback. *Application*, on the other hand, can be measured indirectly by performance outcomes, such as assignment grades, quizzes, projects, or exams. Direct measures will require in-depth content analysis to assess the level of application quality. Depending on the nature of the performance activities, specific application metrics can be retrieved. However, the scope of this study does not include using performance measures as we discovered a large variance in performance outcome implementation that could not be generalized across large data sets.

Social Learning Theory

Bandura's (1977b) Social Learning Theory and Vygotsky's (1985) Zone of Proximal Development can help provide a comprehensive framework for understanding the social aspects of human learning.

In Bandura's (1977b) social learning theory, "vicarious, symbolic, and self-regulatory processes [are prominent roles] in psychological functioning" (p. vii). Social learning theory posits a "continuous reciprocal interaction between cognitive, behavioral, and environmental determinants" in which human development is impacted by both their independent capabilities as well as external influences (p. vii). Social learning theory posits that the role of self-efficacy is to not only influence behaviors and environments but also be influenced by them (Bandura, 1977a, 1977b; Schunk & Pajares, 2009). "Assuming requisite skills and positive values and outcome expectations, self-efficacy is a key determinant of individuals' motivation, learning, self-regulation, and achievement" (Schunk & Pajares, 2009, p. 37).

In Vygotsky's (1985) Zone of Proximal Development (ZPD) theory, learning is supported by external influences, which provide knowledge and experiences beyond what any single learner can achieve alone. In ZPD, learning lags development, which corresponds to a state of mastery where the learner can demonstrate knowledge and skills independent of external support (e.g., the ability to solve a complex math problem without the help of a peer, coach, or teacher) (Vygotsky, 1985). Beyond development, there will be slightly harder problems that the learner will be unable to complete alone but likely successful with external support. This slightly higher difficulty level is ZPD and can serve as a predictor of future learning that can be facilitated by social support. Thus, learning can be promoted by social learning aspects of human interaction.

For continuous monitoring and evaluation of social learning, formative assessments can be implemented in conjunction with collaborative social activities, such as using rubrics with discussion boards, e-portfolios, reflective journaling, and wikis

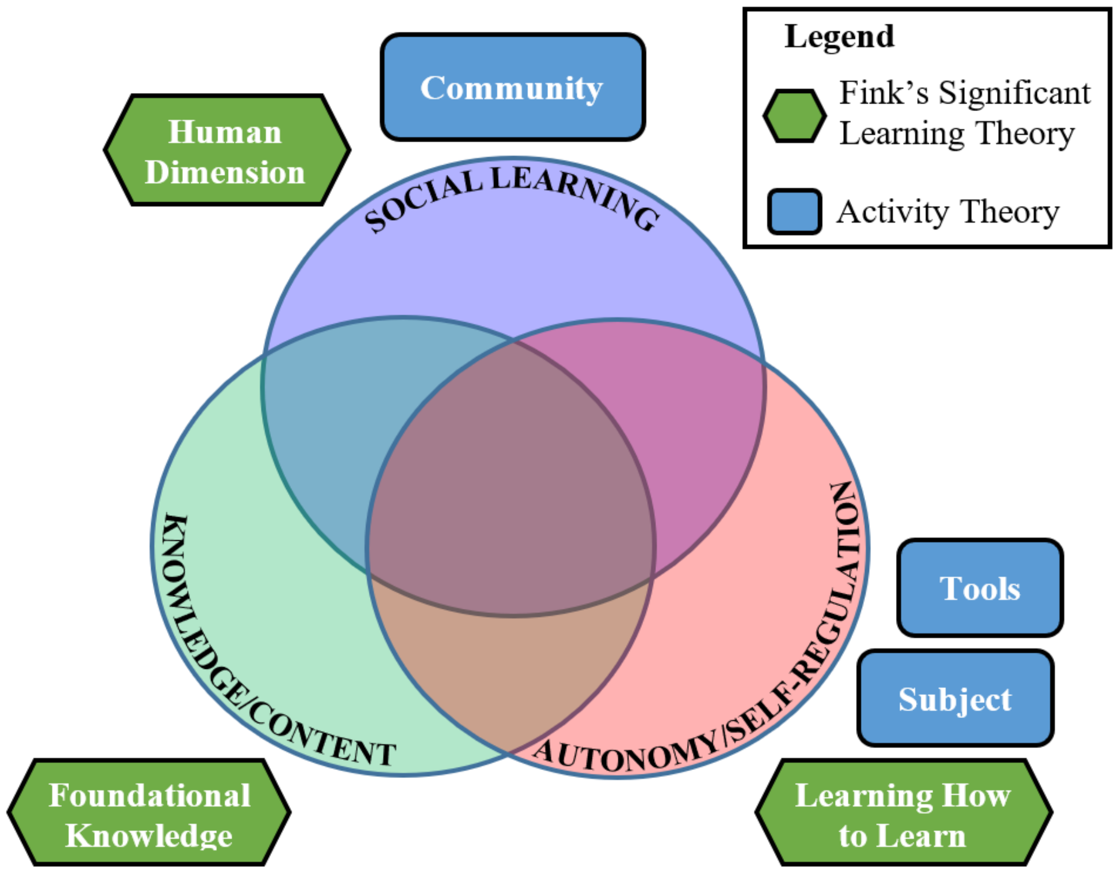
(Perera-Diltz & Moe, 2014). Therefore, a student's interaction with assessment-relevant material within the LMS can promote the learning process.

Theory-Guided Feature Selection (TGFS) Model

Synthesizing relevant elements of the theoretical constructs described previously (i.e., Activity, Fink's Significant Learning, and Social Learning Theory), a novel Theory-Guided Feature Selection (TGFS) conceptual learning model that promotes student-centric social learning is developed and shown in Figure 12. Note that this model is neither comprehensive nor complete, as it was designed as a guide to identify and select relevant theory-guided features for the XAI EDM study. Theoretical elements such as intrinsic value or self-efficacy were excluded as those features are not readily available from LMS data collection. Future studies could incorporate these measures by administering additional data collection instruments such as interviews or surveys.

Figure 12

Theory Guided Feature Selection (TGFS) Model



The TGFS Conceptual Learning Model can help identify and categorize viable features when inspecting and analyzing potential logged data of institutional learning management systems. The Venn diagram identifies the three primary theoretical learning constructs: social learning, knowledge/content, autonomy, and self-regulation. Features that fall in overlapping areas can boost learning by leveraging the benefits of multiple scaffolds. The supporting adjacent components represent Significant Learning and Activity Theory supports that provide further justification and support for the respective theoretical learning construct. When used as a guide, data analysts can quickly identify variables of interest, theoretical support, and learning scaffolding potential. As a post-analysis tool, the TGFS model can help provide explanations to model predictions as well as guide learning interventions.

Conceptual Framework

Conceptually, this study frames around Karpatne et al.'s (2017) theory-guided data science (TGDS), which couples theory with predictive data science. TGDS (Karpatne et al., 2017) provides two primary benefits: (a) enhances explainability by providing scientific consistency from theory-based models; and (b) promotes scientific discovery by providing novel predictive insight from data science. Leveraging both technology and theory ensures an informed approach to embracing innovation that is sound in learning principles and instructional methods.

The conceptual design of this study extends Xing et al.'s (Xing et al., 2015) genetic programming (GP) prediction model guided by Activity Theory, which describes a learner's social and technology-mediated interactions with the learning environment. To supplement this, we incorporated Fink's (2003) Holistic View of Active Learning, which comprises: (1) learner access to information and ideas (i.e., content); (2) learner experiences (e.g., doing, observing, actual, simulated, or authentic settings, etc.); and (3) leaner reflection activities (e.g., essays, discussions, etc.). Finally, social learning and self-regulation theory is adopted from Bandura's Social Learning Theory (Bandura, 1977b) and Vygotsky's (1985) Zone of Proximal Development.

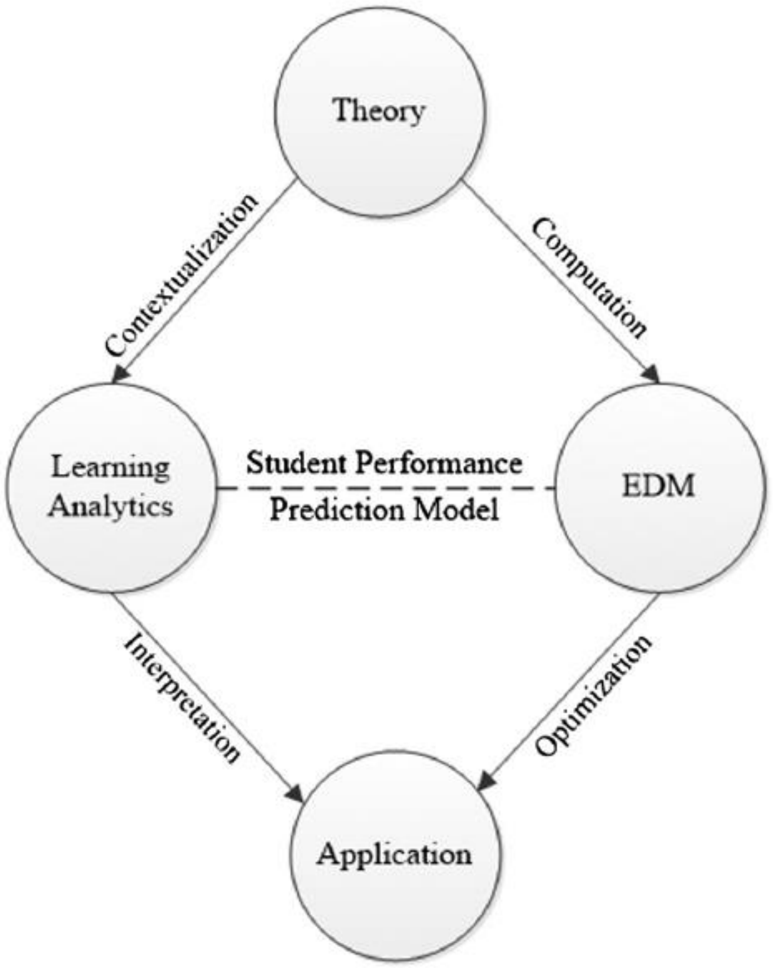
Synthesizing these theoretical frameworks, a novel Theory-Guided Feature Selection (TGFS) Conceptual Learning Model is developed and proposed (see Figure 12). Using this model, an initial comprehensive list of educational data mining parameters for data collection can be identified, categorized, and used for guiding instructional interventions.

Conceptual Model

The conceptual model for this study leverages Xing et al.'s (2015) theoretical framework for the student prediction model (see Figure 13) as it incorporates the various elements of our study, such as theory, learning analytics, educational data mining (EDM), and application (i.e. the XAI performance, explainability, and statistical evaluation).

Figure 13

Theoretical Framework for Student Prediction Model



Note. Chart reprinted from Xing et al. (2015).

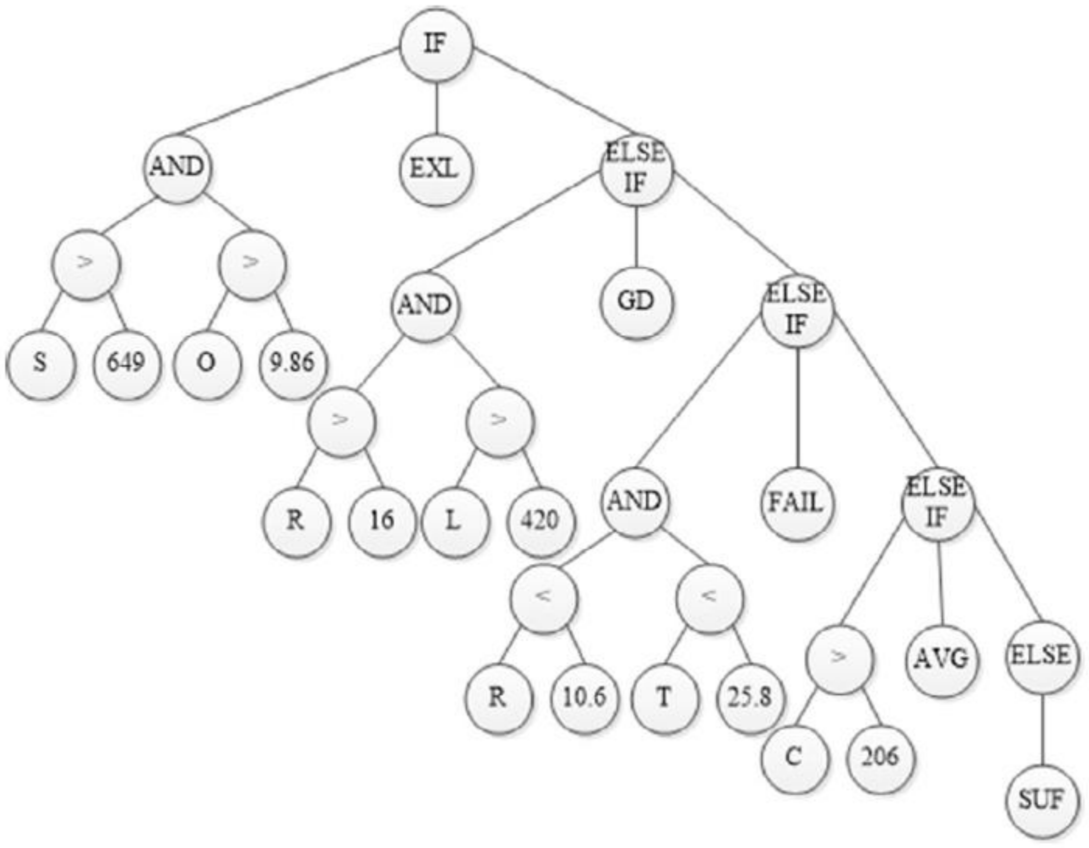
Unlike prior research, which has focused on individual components for target studies, Xing et al.'s (2015) framework provide a system view identifying "links between theory and computation, and optimization and interpretation… to develop an easily interpreted and applied prediction model for real-life learning environments" (p. 179). This view shows how learning analytics is supported by theory to provide relevant context for explainability and interpretations while EDM is supported by computation theory (e.g., AI and ML) to improve the accuracy and prediction of outcomes. When combined, a high-performing explainable model can be developed for learning applications, such as predicting at-risk students and/or providing explainable learning interventions.

Genetic Programming

Genetic Programming (GP) is an evolutionary technique that builds models based on parent/offspring selection, mutation, and evolution. Some are subsets of Decision Trees but have the additional ability to evolve to optimize performance, accuracy, and efficiency. This is achieved by iteratively training a predefined decision tree until the desired accuracy is achieved. The visual design of the trees, conditional expressions used, and numerical weights are transparent to the user and therefore provide explainability for analyzing why a particular path was undertaken (see Figure 9).

Figure 14

Example of a GP Decision Tree



Note. Chart reprinted from Xing et al. (2015).

Following a genetic tree is intuitive and easy to interpret, which makes it useful for not only predicting (e.g., at-risk indication of drop-out) but also providing explanations (e.g., formative feedback for at-risk indicators). GP models have been shown accurate and effective in many studies and serve as an ideal implementation for predicting at-risk students (Kalles & Pierrakeas, 2006; Xing et al., 2015).

Emerging works in genetic programming also include stack-based classifiers that have demonstrated comparable performance to state-of-the-art models in a variety of applications (La Cava et al., 2017, 2019). Consistent with this, Perkis (1994) notes that stack-based genetic programming is “a promising technique… worthy of further consideration” (p. 152) as it provides “efficiency and simplicity of implementation” that are “comparable or superior to current methods” for certain problem sets.

This study integrates and advances the emerging works of Xing et al. (2015) and La Cava et al.’s (Cava, 2017; 2019) Ellyn-GP model by exploring theory-guided feature selection, k-Means clustering, and stack-based genetic programming multi-classifier to understand and predict at-risk student groups. Tree-based GP models from the Ski-Kit Learn, such as GP Learn binary classifier, are also explored.

Section Summary

This section introduced the theoretical underpinnings of the study, such as the use of Theory-Guided Data Science to improve model validity and reliability through the incorporation of sound instructional theories and principles. Applicable theories identified and described include Activity Theory, Fink’s Significant Learning Theory, and Social Learning Theory. Synthesizing these theoretical constructs, a novel Theory-Guided Feature Selection (TGFS) Conceptual Learning Model was developed and proposed. The conceptual model will be used in the EDM study to analyze and guide the feature selection process. The TGFS model also provides a framework for model explanation and intervention planning. Finally, Xing et al.’s (2015) conceptual model was referenced and described to capture the overall research approach that leverages theory, learning analytics, and EDM toward at-risk student prediction applications.

Chapter Summary

This chapter provided a relevant literature review concerning the target research problem of student retention and the need for XAI at-risk student predictive systems. Early works were identified to provide support for the emerging demand for explainability, motivations for XAI, as well as prompt future research. Extensive studies were systematically synthesized to identify the current state of EDM research, as well as common features and outcome variables studied. A summary of literature findings revealed promising works leveraging theory-guided data science (TGDS) and genetic programming. Finally, relevant theoretical frameworks were presented, which provided support for a theory-guided feature selection model. A brief description of genetic programming and its benefits was summarized.

CHAPTER III

Methods

Introduction

The objective of this research is to explore the performance and efficacy of theory-grounded explainable artificial intelligence (XAI) to predict and explain at-risk students in an online learning management system (LMS). To achieve this, quantitative research methodologies in educational data mining (EDM) and learning analytics (LA) were adopted, both of which offer data-centric techniques to improve and understand online learning in higher education.

Educational data mining (EDM) is employed to collect student performance, logged interaction activity, and learning outcomes from a higher institution LMS system. Data will be anonymized and protected to ensure data privacy and security. Interaction frequency was computed at weekly intervals to explore the model's predictive performance over time using common measures of AI/ML accuracy (f1-score, recall, and precision). Efficacy will be determined by analyzing the predictive performance and interpretability of the GP-TGDS model against other known AI/ML models (e.g., student retention and learning outcomes). Mean differences and effect sizes are reported at the .05 significance level.

In the following sections, detailed methods and procedures are described to identify the process of feature selection, data collection, processing, analysis, and performance evaluation.

Research Questions

We investigated the following research questions (RQs):

- RQ1: What are the associations between activity factors and student final grades?
- RQ2: What are the characteristics of at-risk students that can be identified?
- RQ3: What are characteristics of at-risk students that can be predicted using XAI techniques and models?
- RQ4: How do explainable models compare to complex models when LMS student activities are used as features to predict at-risk students?

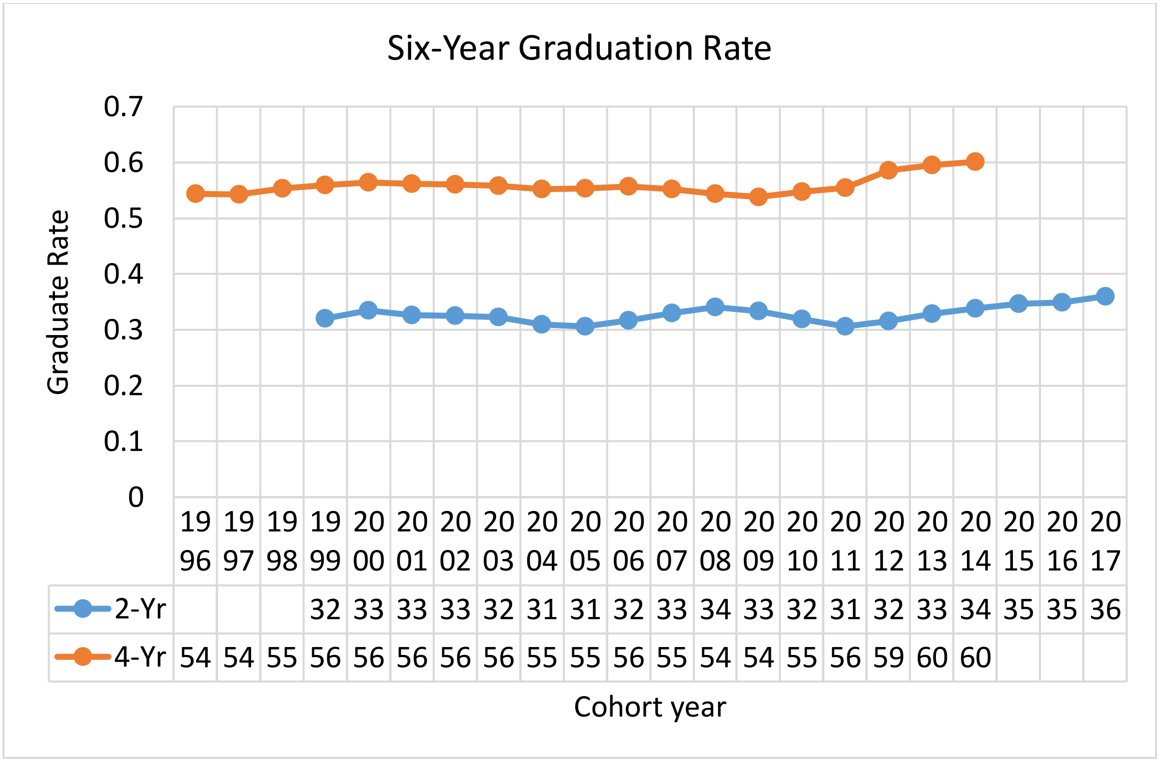
Population, Setting, and Demographics

The target population selected for this study is a large four-year higher education institution that offers distance learning and has average student retention near or below the US national average six-year graduation rate to capture a representative population for studying at-risk students. Offering distance learning is required as these study targets institutions delivering online learning using a learning management system to enable the collection of existing student activity logged data for educational data mining.

Figure 15 gives the six-year graduation rate for the US national average and the selected institution for comparison (NCES, 2021). As shown, the selected institution has historically underperformed concerning the national average for cohorts 2004 to 2014. In addition, 14% enrolled in distance-only classes, 44% enrolled in some distance classes, and 42% are not enrolled in any distance classes (NCES, 2021). Of all graduates, 55% are enrolled in distance-only classes, 13% are enrolled in some distance classes, and 32% are not enrolled in any distance education. Note that graduates comprise only 14.7% of all students. Thus, the institution meets the inclusion retention and distance learning inclusion criteria.

Figure 15

150% Graduation Rate for All Full-time/First-Time Students (Fall 2015)



	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017
2-Yr				32	33	33	33	32	31	31	32	33	34	33	32	31	32	33	34	35	35	36
4-Yr	54	54	55	56	56	56	56	56	55	55	56	55	54	54	55	56	59	60	60			

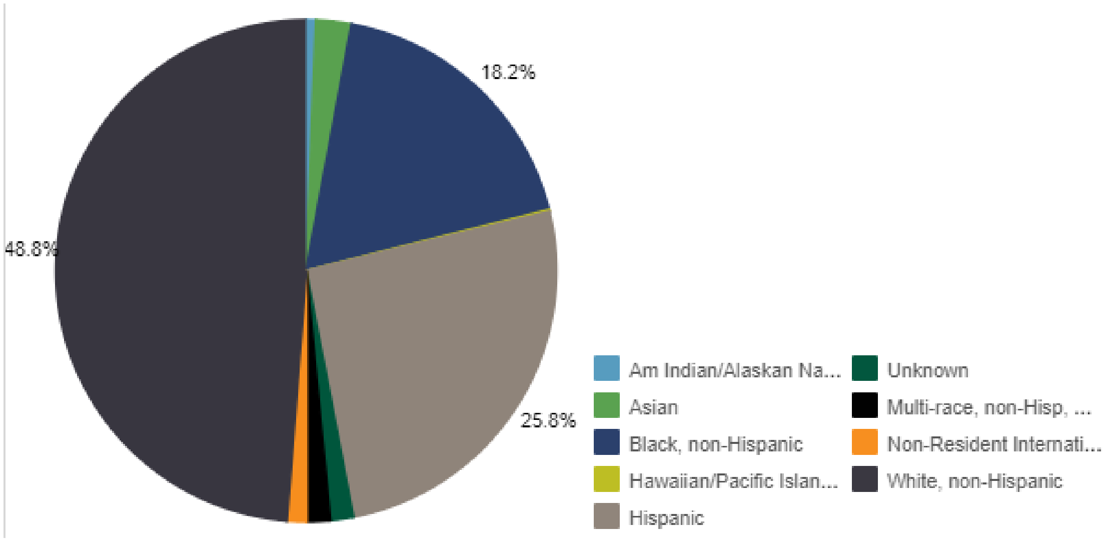
Note. Data adapted from NCES and Texas University System websites (NCES, 2021; The Texas State University System, 2022). In the public domain.

Raw student activity logs and performance data were collected from a Blackboard Learning Management System (LMS) database spanning 15 weeks in the Spring semester of 2021. According to the Texas University System (TSUS) enrollment database (The Texas State University System, 2022), the 2021 Spring semester had a total enrollment size of 19,753 students who pursued undergraduate (85.3%), master's (12.3%), doctoral research (1.7%), or post-baccalaureate (0.7%) degrees. Two-thirds (64.1%) were females and most (69.7%) were enrolled in full-time status. The race/ethnicity was primarily

White (48.7%), Hispanic/Latino (25.8%), and Black or African American (18.1%) (see Figure 16 for additional details).

Figure 16

Student Proportions of Race/Ethnicity (Spring 2021)



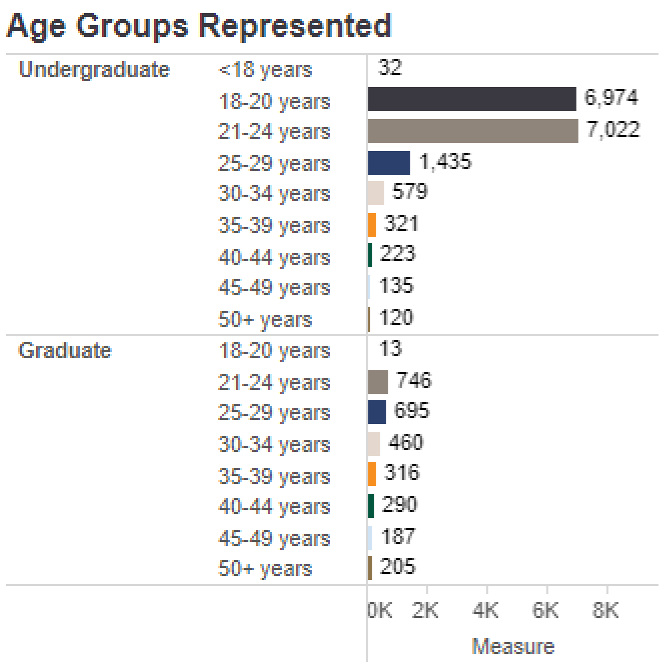
- Most graduates (74%) work part-time.

Note. Image reprinted from The Texas University System (2022). In the public domain.

Most students were in-state (98.7%) and between the ages of 18 and 24 (78.1%; see Figure 17). Using EDM processes, the data was analyzed, cleaned, and tune for follow-on statistical, exploratory, classification, prediction, and explainability analysis to investigate the study's research questions. It is important to note that according to the institution's leadership, the school used the Blackboard LMS to track both traditional and online grades. Since the majority of students were enrolled in traditional classroom environments, there was a large number of students with a lack of LMS interaction activity that was excluded to prevent skewing of results.

Figure 17

Student Proportions of Race/Ethnicity (Spring 2021)



EDM Process

This section provides a brief background on EDM and describes the process adopted to collect, process, and analyze data for this study. According to Aliaga and Gunderson (2002), quantitative research is "explaining phenomena by collecting numerical data that are analyzed using mathematically based methods (in particular statistics)" (as cited in Muijs, 2004, p. 1). Both EDM and LA fall into this definition as they "share a common interest in data-intensive approaches to education research, and share the goal of enhancing educational practice" (R. Baker & Inventado, 2014, p. 62). In contrast, Siemens and Baker (2012) identify some differences between the respective field definitions:

- The International Educational Data Mining Society defines EDM as "developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learn in" (pp. 252).
- The Society for Learning Analytics Research defines LA as "the measurement, collection, analysis, and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs" (pp. 252-253).

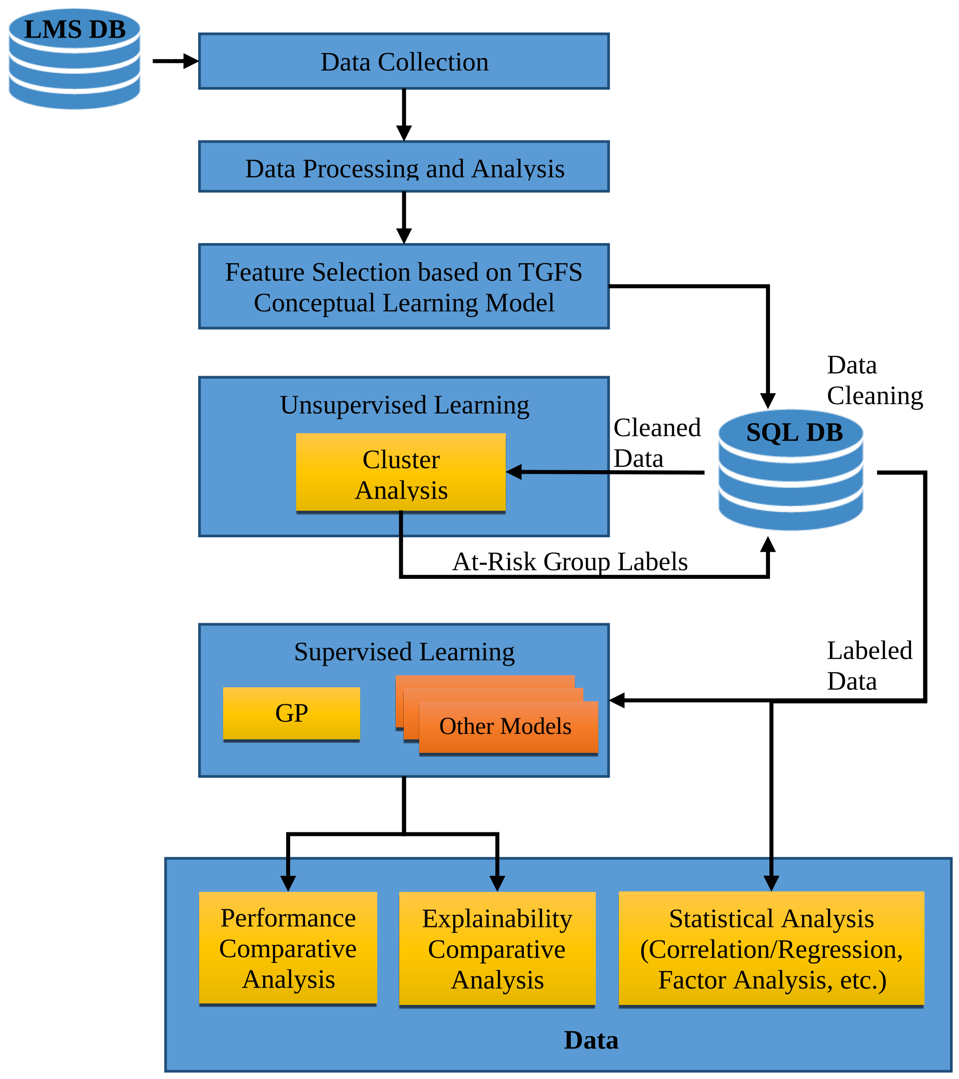
Furthermore, EDM emphasizes automated discovery techniques to inform learners and educators (such as an at-risk indication system), whereas LA takes a step further to provide human judgment and analysis of that data to "empower instructors and learners" (e.g., providing informative feedback and suggestions for intervention guidance) (Siemens & Baker, 2012, p. 253).

Thus, while EDM provides a methodology and process for collecting, reducing, and developing predictive models, LA adds a layer of interpretation and explainability by including human interpretations (i.e., using visualizations or theory) of automated results. This provides a comprehensive and holistic framework for developing XAI models that not only predicts at-risk students but also enhance the explainability of at-risk indicators by providing informative feedback.

Figure 18 illustrates the planned EDM/LA research methodology adapted from Yang et al. (2021) and Imran et al. (2019). *To improve the reliability of the study, this process includes a post-hoc statistical analysis for determining the statistical significance and explaining the variance of influential factors.*

Figure 18

Planned EDM/LA Research Methods Process


LMS DB
Data Collection
Data Processing and Analysis
Feature Selection based on TGFS Conceptual Learning Model
Data Cleaning
SQL DB
Cleaned Data
Unsupervised Learning
Cluster Analysis
At-Risk Group Labels
Supervised Learning
GP
Other Models
Labeled Data
Performance Comparative Analysis
Explainability Comparative Analysis
Statistical Analysis (Correlation/Regression, Factor Analysis, etc.)
Data

Feature Selection Process

The selection of student activity features was guided by a combination of literature review, data inspection/analysis, and theoretical support, which are discussed in the proceeding subsections.

Literature Support

Preliminary literature research results identified student activity as the most used feature for predicting at-risk students (see Table 4). The following summarizes the finding concerning predicting student outcomes:

- Using correlation and logistic regression, Chamizo-Gonzalez et al. (2015) found a positive correlation between student activity (e.g., assignment uploads, discussion forum posts, discussion forum views, assignment views, blog views, course views, and resource views) and learning outcomes.
- Using Cross-Validated Correlation and Bayesian Knowledge Tracing, Baker et al. (2011) were able to predict student performance, with the former method demonstrating superior accuracy.
- Bienkowski et al.'s (2012) literature review found that LMS logged data such as student responses coupled with prior skills, knowledge, performance, course content, time, and demographics demonstrated prediction accuracy up to 81% for those who failed. They concluded that, in general, EDM can help build models to explore factors influencing student learning (Bienkowski et al., 2012).
- In a correlation study analyzing the relationship between learner interaction with an LMS and virtual agent, Song et al. (2019) found high loadings from system access, time spent, and discussion length.
- Using various correlation and regression methods, Conijn et al. (2017) found statistically significant correlations for the majority of predictor

variables (e.g., various student activity, interaction time, and performance metrics). However, there was low predictive power and low portability across courses.

- Finally, using EDM, activity theory-guided feature selection, Genetic Programming, and student activity features, Xing et al. (2015) demonstrated superior at-risk prediction performance when compared to other common models, except for Naives Bayes. They concluded that the additional explainability of Genetic Programming made it a preferable choice.

Most findings demonstrated positive results and support for using student LMS activity as potential features for at-risk prediction.

Data Inspection/Analysis Support

After performing a detailed analysis of the Blackboard LMS database schema and underlying logged data, student activity metrics were identified as the most abundant and consistent across all courses. The logged activities were also available during the early weeks of instruction, which is an advantageous time to detect and respond to early signs of at-risk behaviors. Performance metrics such as quizzes, assignments, or pre-course grades were excluded due to the lack of sample size and inconsistent implementations across courses and weekly implementation. The data inspection/analysis activities were an iterative process that spans the entire study from initial data collection and literature review through statistical, clustering, and predictive analysis where additional cleaning, tuning, and outliers were addressed.

Theoretical Support

From the literature review (Table 6), a major limitation observed was that most studies lacked theoretical support. Only Xing et al.'s (2015) Genetic Programming leveraging activity as a basis for feature selection demonstrated the most promise with superior results when compared to other models. Their work provided inspiration and guidance for extending EDM theory-guided data science to improve prediction performance and explainability. Continuing their work, this study develops and explores a theory-guided feature selection model incorporating learning theories such as Activity Theory, Fink's Significant Learning Theory, and Social Learning Theory. Refer to section *Theory-Guided Feature Selection (TGFS) Model*, Figure 12, and Figure 22 for a description of the model, visual representation, and initial feature selection results respectively.

Table 6

Studies with Theory-Grounded Models

Study Type	Literature Reference	Theory-Grounded
Analysis	(Al-Omar, 2018; Song et al., 2019; Yang et al., 2021)	No
Comparative	(Al Breiki et al., 2019)	No
Correlation	(Abe, 2019; Chamizo-Gonzalez et al., 2015)	No
Frequency Analysis	(Al Breiki et al., 2019)	No
Predictive	(Abe, 2019; Al Breiki et al., 2019; R. S. J. D. Baker et al., 2011; Berens et al., 2019; Conijn et al., 2017; Imran et al., 2019; Ndou et al., 2020; Zheng, 2020)	No
	(Xing et al., 2015)	Yes

Analysis Methods

This section identifies and describes the analysis methods performed, such as statistical analysis (descriptive and inferential), power analysis, feature analysis, exploratory cluster analysis, classification analysis, prediction performance analysis, and explainability analysis.

Statistical Analysis

Descriptive statistics were performed to summarize, describe, and identify associations within the collected data using measures of central tendency, variation, range, confidence interval, distribution shape, and association/correlation (L. Cohen et al., 2018). Descriptive statistics also help check for violations of normality, which may drive the type of hypothesis tests used (e.g., parametric or non-parametric) to ensure reliability and validity of results (L. Cohen et al., 2018). Descriptive statistics were used in analyzing both the population and sampled datasets used for inferential statistics.

Inferential statistics were performed to identify statistically significant mean differences within the data parameters (independent and dependent variables). The selection of test type will depend on a variable of factors, such as (McCrum-Gardner, 2008): measurement scale (nominal, ordinal, interval); independent vs. paired (pre/post) groups; parametric vs. non-parametric assumptions (normal vs. non-normal); and independent or paired groups. Since this research explored multiple independent and dependent variables, comparisons between and within groups were conducted using the MANOVA with IBM SPSS.

Power Analysis

To effectively detect significant differences, McCrum-Gardner (2008) recommends conducting a power analysis to identify the minimum sample size required. Field (2018) identifies power as the probability of finding an effect assuming one exists, which is mathematically represented as ($1-\beta$), where β is the probability of not finding an effect when one exists (i.e. Type II statistical error). Cohen (1988) recommends using a power of 0.80 ($\beta = 0.2$) in conjunction with $\alpha = 0.05$ as a guide for sufficient power. Using G*Power (2021) as recommended by Field (2018), the applicable sample size for applicable tests with inputs set to two-tail, $\alpha = .05$, $1-\beta = .80$, N1 = N2, and Cohen's d = 0.5 (medium effect) are as follows:

Independent t-tests: N1 = N2 = 64

Wilcoxon-Mann-Whitney test: N1 = N2 = 67

Paired Sample t-test: N = 34

Wilcoxon Signed Rank test: N = 35

Correlation (Point biserial model): N = 64

Observe that the Wilcoxon-Mann-Whitney test requires the largest sample size and thus is the determining factor. To detect a medium effect (d = 0.5), the minimum sample size required for each group is 67 (134 total) in case a normal distribution cannot be assumed. If we wanted to be able to detect a small effect (d = 0.3), then we would need an individual group sample size of 184, or a total study sample size of 368. Thus, a sample size of 368 places the minimum limit that enables sufficient power to detect a small effect.

Feature Analysis (RQ1)

Feature analysis is an iterative process of selecting features to reduce noise and redundancy while improving model performance and explainability (Layton, 2017). In this study, feature analysis is performed at various stages, such as during database feature identification, data collection and pre-processing, and clustering/classification analysis. The initial potential features are identified in the previous “Initial Database Features Identification.” However, final feature determination will be dependent on further analysis, which may be excluded when addressing high correlation and collinearity during K-Means Clustering analysis. Python packages and libraries such as Pandas (*Pandas,* 2010/2021), SciPy (*SciPy.Org — SciPy.Org*, n.d.), SciKit-Learn (*Scikit-Learn: Machine Learning in Python — Scikit-Learn 0.24.2 Documentation*, n.d.), Seaborn (Michael, 2021), and matplotlib (*Matplotlib: Python Plotting — Matplotlib 3.4.3 Documentation*, n.d.) was used for computational analysis and data visualizations.

To address RQ1, a correlation analysis was performed to identify significant features ($p < .05$) correlated to final grades. Feature pairs with high correlations (above 0.6) were reduced by excluding the feature with the lowest number of samples to mitigate problems associated with high collinearity between independent variables (Abdulhafedh, 2021; Field, 2018; Kondo et al., 2017), which introduces redundant factors that could reduce model prediction accuracy (Layton, 2017). Features with insignificant correlations ($p > .05$) to final grade and a higher percentage of missing values are excluded to reduce noise from irrelevant factors.

Exploratory Cluster Analysis (Process Overview)

Before classification (i.e., supervised learning and prediction) can be performed, the dataset must be grouped and labeled into classes that can be predicted based on feature patterns. This process is known as clustering, which is an unsupervised learning technique (e.g., K-Means Clustering, Hierarchal Clustering, Spectral Clustering, etc.) used to split datasets into subsets based on similarities (Song, 2021b, 2021c). For this study, K-Means Clustering will be used as the clustering method.

K-Means Clustering is a specific type of unsupervised machine learning analysis method used to explore patterns and groups based on similarities or differences in the underlying dataset parameters (Layton, 2017; Song, 2021d). The technique is especially useful when either no class exists for prediction (Song, 2021c) or when the data set is too large and impractical for manual classification assignment (Song, 2021d); both of which apply to this study. Other advantages include “simple mathematical ideas, fast convergence, and easy implementation” (Yuan & Yang, 2019, p. 226). However, Yuan and Yang (2019) note that K-Means algorithms become a challenge with dealing with analyzing “massive data sets” (p. 226).

In brief, the K-Means Clustering algorithm employs an iterative process of finding representative “centroids” or means of cluster samples given a pre-determined number of groups (i.e., K) (Layton, 2017, p. 231). The centroids will change slightly after each iteration of the assignment (i.e., selecting a data point closest to the previously calculated centroid) and update (i.e., computing the next centroid based on the new data point) (Song, 2021c, p. 5) and stop either after a pre-determined number of iterations or when the centroid updates converge and stabilize (Layton, 2017).

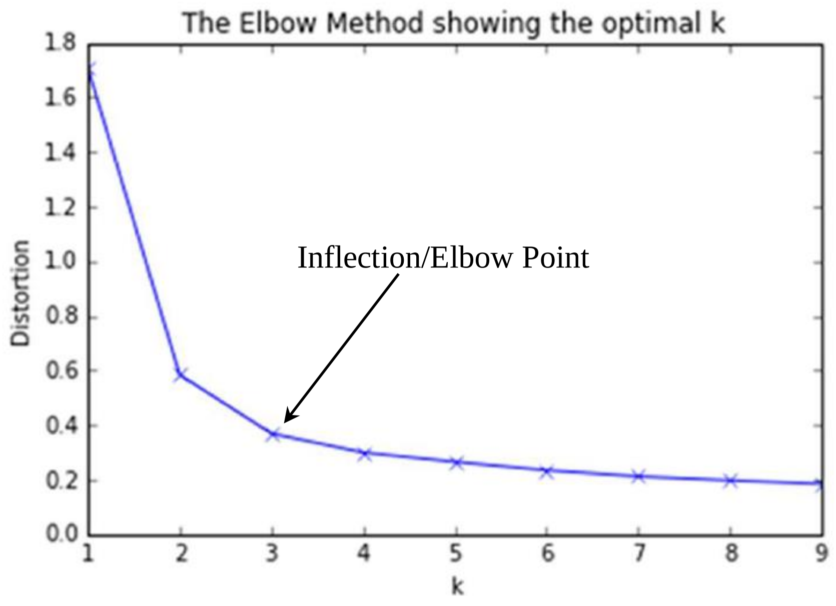
One of the key goals of K-Means cluster analysis is finding the optimal number of clusters (i.e., K) representative of the patterns that exist assuming we have no additional insight besides the dataset given. Although any number can be used as a pre-determined input to the K-Means Cluster Analysis, Yuan and Yang (2019) note that “in practice, the K value is generally difficult to define” (p. 228), especially to find meaningful groups. In a comparative study, Yuan and Yang (2019) investigated the performance of four alternate methods, namely, the Elbow Method, Gap Statistic, Silhouette Coefficient, and Canopy. Their results are summarized as follows:

- The Elbow Method involves graphing a sum of squared error (SSE) metric versus K and then finding the point of inflection (Yuan & Yang, 2019, p. 233).
- Gap Statistic is a complex algorithm iterating values of K to find a maximum “Gap” value based on Monte Carlo sampling and “reference measurements”(Yuan & Yang, 2019, p. 229).
- The Silhouette Coefficient algorithm involves calculating and determining a maximum “S(i)” metric representing the “cohesion” (i.e., similarity) of clusters for different K values (Yuan & Yang, 2019, pp. 230–231).
- The Canopy algorithm involves divided data sets, overlapping subsets, distance comparisons (given a pre-determined range), and iterative aggregation and deletion (Yuan & Yang, 2019).

In summary, Yuan and Yang (2019) found that all four methods were feasible when using small data, with the Elbow Method and Canopy algorithms computationally more efficient than the Gap Statistic and Silhouette Coefficient methods. For simplicity and ease of implementation, this study uses a Python Kneed Package to automate the process of finding K and the Elbow Method for visual verification. Figure 19 gives an example scenario from Yuan and Yang (2019) where the “elbow” point was identified at K = 3.

Figure 19

Exemplar Inertia Graph for Implementing the Elbow Method



Note. Image adapted from “Research on K-Value selection method of K-Means Clustering algorithm,” by C. Yuan and H. Yang, 2019, *J, 2*(2), p. 229 (https://www.mdpi.com/2571-8800/2/2/16).

Exploratory Cluster Analysis (RQ2)

After post-processing and the final selection of features, K-Means clustering analysis was performed to explore and identify clusters for prediction. The optimal number of clusters, K, was determined using the Python kneed package (Arvai, 2020; Satopaa et al., 2011), which detects the “knee” point based on where the “curve becomes more ‘flat’” (p. 3). This was visually verified using the elbow method to confirm correctness and consistency (Yuan & Yang, 2019). The cluster activity frequency was then plotted with the outcome variable (mean final grades) for pattern analysis. Meaningful labels and descriptions of each cluster are given to identify and characterize at-risk activity behavior patterns to investigate and answer RQ2.

As an unsupervised learning method, both activity features and final grade outcomes were used as input variables, which has positive and negative implications. As a benefit, using final grades as an input feature helps to discriminate clusters concerning at-risk outcomes. However, in practice, final grades are not available and cannot be used as a feature for prediction. Although at-risk groups can easily be identified and labeled during unsupervised learning, prediction of those labels will be limited without the underlying assumption of final grades as an input predictor variable. This was later confirmed by evaluating the performance of variable models against the unsupervised K-Means model.

Classification Analysis (RQ3)

To address the predictive limitations of the unsupervised k-Means cluster model, a new k-Means cluster analysis was conducted to develop a model based on activity as input features (i.e., final grades are not included as a feature). This results in natural groups for student activity behaviors. The mean final grades are then computed and graphed against each cluster's activity features for post-hoc analysis, classification (i.e., creating outcome labels to be predicted), and characterization of at-risk student groups. Mean final grades define the level at-risk while the features (i.e., interaction frequency) identify associated group behaviors. The final labels are then used as ground truths (i.e., accepted as true for grouping outcomes). This process is known as a "cluster-then-label" process, which has been shown effective and accurate in labeling big data, especially when it is cost-prohibitive and resource-intensive (Beil & Theissler, 2020; Peikari et al., 2018). This approach investigates the predictability of at-risk characteristics to address RQ3.

The process of classification and prediction involves the training and testing of sampled datasets (i.e., training and testing re-sampled datasets) obtained from a population sample (i.e., the dataset collected for this study). The sampling method used to select the training and testing sets is of critical importance as it could impact model performance (Banerjee et al., 2018; Berrar, 2018). For example, a common problem when training a model is that of “overfitting”, which is when a model is “perfectly adapted to the data set at hand but then unable to generalize well to new, unseen data” (Berrar, 2018, p. 1). To address this, various sampling methods have been devised. The following list gives a summary of common methods highlighted by Berrar (2018).

Table 7

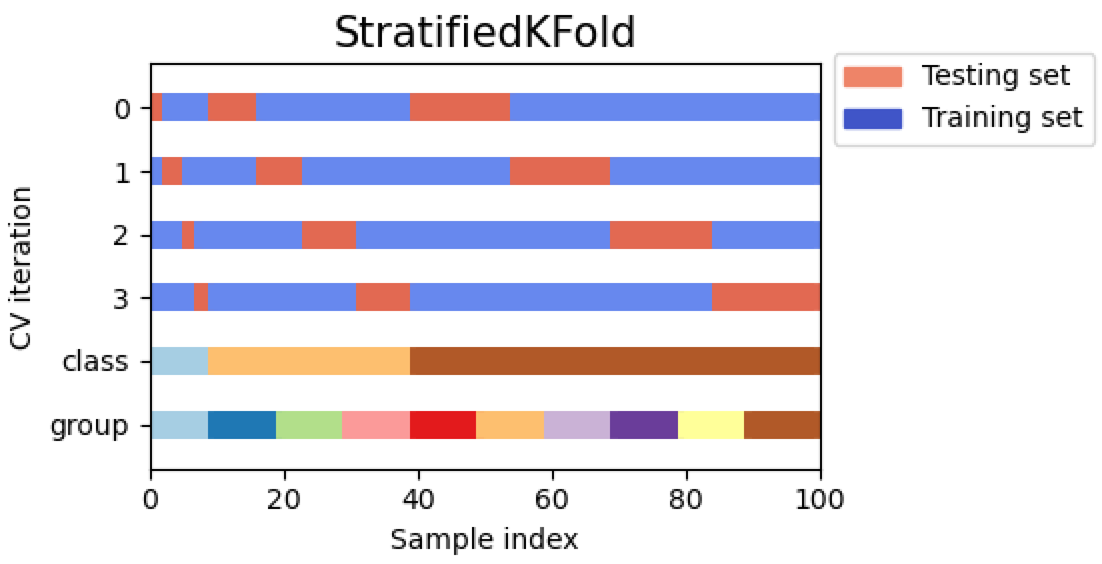
Training Split Methods

Training Split Method	Description
Single hold-out random sampling	Random sampling is performed on the data set until a percentage of the training set is reached (e.g., 10-30% for testing and the remaining for training).
K-Fold random subsampling	Single hold-out random is performed K times, where each training + testing group sample size equals the dataset size divided by K. Performance is average over K sets. Could result in overlapping sets.
K-fold cross-validation	Similar to K-Fold random subsampling but ensures there is no overlap in each training/test set. This alleviates overfitting and overtraining from training/testing the same samples multiple times.
Stratified K-fold cross-validation	K-fold cross-validation with stratified random is performed to ensure class proportions of each individual set are reflects that of the population sample. This provides an "unbiased estimate of the population proportion" (2018, p. 4).
Leave-one-out cross-validation (LOOCV)	This is a special case of K-fold cross-validation where each training set is excluded in the validation set (i.e., the validation set contains k-n sets at each iteration). Although this provides an "unbiased estimate of true prediction error," there is a tradeoff of higher variance and computational load.
Jackknife	Like LOOCV but focused on estimating the bias or variance of a statistic rather than the generalization ability of a predictive model.

Based on Table 7, stratified 10-fold cross-validation (see Figure 20 for an example illustration) was selected for training set splitting to help alleviate overfitting and disproportionate sampling due to unbalanced data sets (Berrar, 2018; *Visualizing Cross-Validation Behavior in Scikit-Learn — Scikit-Learn 0.24.2 Documentation*, n.d.). Furthermore, 10-fold cross-validation was found to have a small bias and accuracy, as long as the data sets were sufficiently large (Berrar, 2018). Finally, due to its widespread use, stratified k-fold cross-validation is natively supported by the SciKit-Learn API, which makes it readily accessible for this study.

Figure 20

Stratified K-Fold Cross-Validation Training Split



Note. Image reprinted from (*Visualizing Cross-Validation Behavior in Scikit-Learn — Scikit-Learn 0.24.2 Documentation*, n.d.)

Performance measures (i.e., precision, recall, and f1-score) are averaged across K sets, reported, and analyzed to identify the prediction performance, which is described in the following section.

Prediction Performance Analysis (RQ4)

Using the classification model and training split methodology described in the previous section, a performance evaluation was conducted to compare the prediction performance of various AI/ML models with varying levels of explainability and complexity. Measures of precision, recall, and f1-score were computed at weekly intervals of cumulative data to analyze prediction performance over time, which allows for assessing early warning potential.

According to Arrieta et al. findings (Arrieta et al., 2020, p. 30), there is a "common trade-off between model interpretability and performance". This work contributes to the XAI goals of improving the performance of explainable models and explainability of complex models by integrated theory-guided implementations of AI/ML. As will be demonstrated, when models are designed with theory in mind, explainable model performance can improve without further modification. This study will demonstrate that explainable models can be comparable, if not better in specific cases, than complex black-box models.

Depending on the nature of the data and field of study, different predictive performance measures are used. For balanced data sets, accuracy, as given in equation (1), might be sufficient (Sokolova et al., 2006). However, when data sets are unbalanced, overall accuracy is biased and is a poor measure of model performance. For this study, unbalanced datasets are expected as there will be disproportions in the number of at-risk students compared to other groups. As such, traditional accuracy may be a poor measure and will not be used.

$$accuracy = (TP+TN)/(TP+FP+TN+FN) \qquad \textbf{(1)}$$

Where,

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

To address unbalanced data sets, alternative measures such as the confusion matrix (Table 8), precision, recall, and f1-score will be used to assess model performance. Precision, recall, and f1-score are defined by equations (2), (3), and (4) respectively (*Scikit-Learn: Machine Learning in Python — Scikit-Learn 0.24.2 Documentation*, n.d.) for the confusion matrix parameters (i.e., TP, TN, FP, and FN).

Table 8

Confusion Matrix

		Predicted	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

$$precision = TP/(TP+FP) \tag{2}$$

$$recall = TP/(TP+FN) \tag{3}$$

$$f1\text{-}score = 2 \cdot (precision \cdot recall)/(precision+recall) \tag{4}$$

Intuitively, the three metrics are interpreted as follows (*Scikit-Learn: Machine Learning in Python — Scikit-Learn 0.24.2 Documentation*, n.d.):

- Precision measures the “ability of the classifier not to label as positive a sample that is negative.”
- Recall measures the “ability of the classifier to find all the positive samples.”
- f1-score is the weighted average of precision and recall.

Both precision and recall are important for at-risk student detection. When precision is low, a classifier may identify students who are not at-risk as at-risk and result in unnecessary interventions for those who do not need it. On the other hand, when the recall is low, the classifier will not be able to detect at-risk students, which is the primary objective of educational data mining. Thus, recall is a more critical measure for predicting at-risk students.

Explainability Analysis (RQ4)

Explainability analysis was performed to explore the explainability of implemented XAI models, such as the decision tree, stack-based genetic programming multi-classifier, and tree-based genetic programming binary classifier. Visual models are explored, compared, and contrasted.

Chapter Summary

This chapter introduced the XAI EDM Study and associated target research questions as well as the theoretical and conceptual framework. The underlying demographics of the dataset were also discussed to characterize the generalizability of the study. A detailed description of the methods and procedures was provided to identify the EDM process and analysis methods. In the next section, the XAI EDM Study Results will be reported and discussed.

CHAPTER IV

Results

Introduction

This chapter provides the results of the XAI EDM study, which comprises the results of the EDM process, theory-guided feature selection, feature analysis, exploratory clustering analysis, classification analysis, prediction performance comparative analysis, and explainability analysis.

EDM Process Results

The following list summarizes the data collection and processing steps performed to prepare the LMS log data for analysis.

1. Data Collection
   - 1.1. Export LMS data to CSV files
   - 1.2. Analyze/develop python processing scripts
   - 1.3. Process/import data into working MySQL database
2. Extract initial features and outcome variables into new tables
   - 2.1. Activity data is aggregated as frequency counts
   - 2.2. Datasets are grouped weekly
   - 2.3. Compute the normalized final grade scores (score/total possible)
3. SQL Inclusion/Exclusion Pre-processing
   - 3.1. Excluded records with missing final grades
   - 3.2. Exclude records with final grades > 1.0
4. Export features and outcome variable table to CSV file
   - 4.1. The first column is the course ID

4.2. The second column is the student ID

4.3. Third through (Nth – 1) column are features

4.4. The nth column is final_grade (target outcome variable)

4.5. Two types of datasets are extracted:

4.5.1. Weekly aggregate data of activity frequency (15 files for weeks 1 to 15).

4.5.2. Cumulative aggregate data of activity frequency (15 files for weeks 1 to 15).

5. Python Pre-processing

5.1. Import CSV file into Python Jupiter Notebook

5.2. Exclude statistically insignificant features (p > .01)

5.3. Exclude highly correlated features (r > .7)

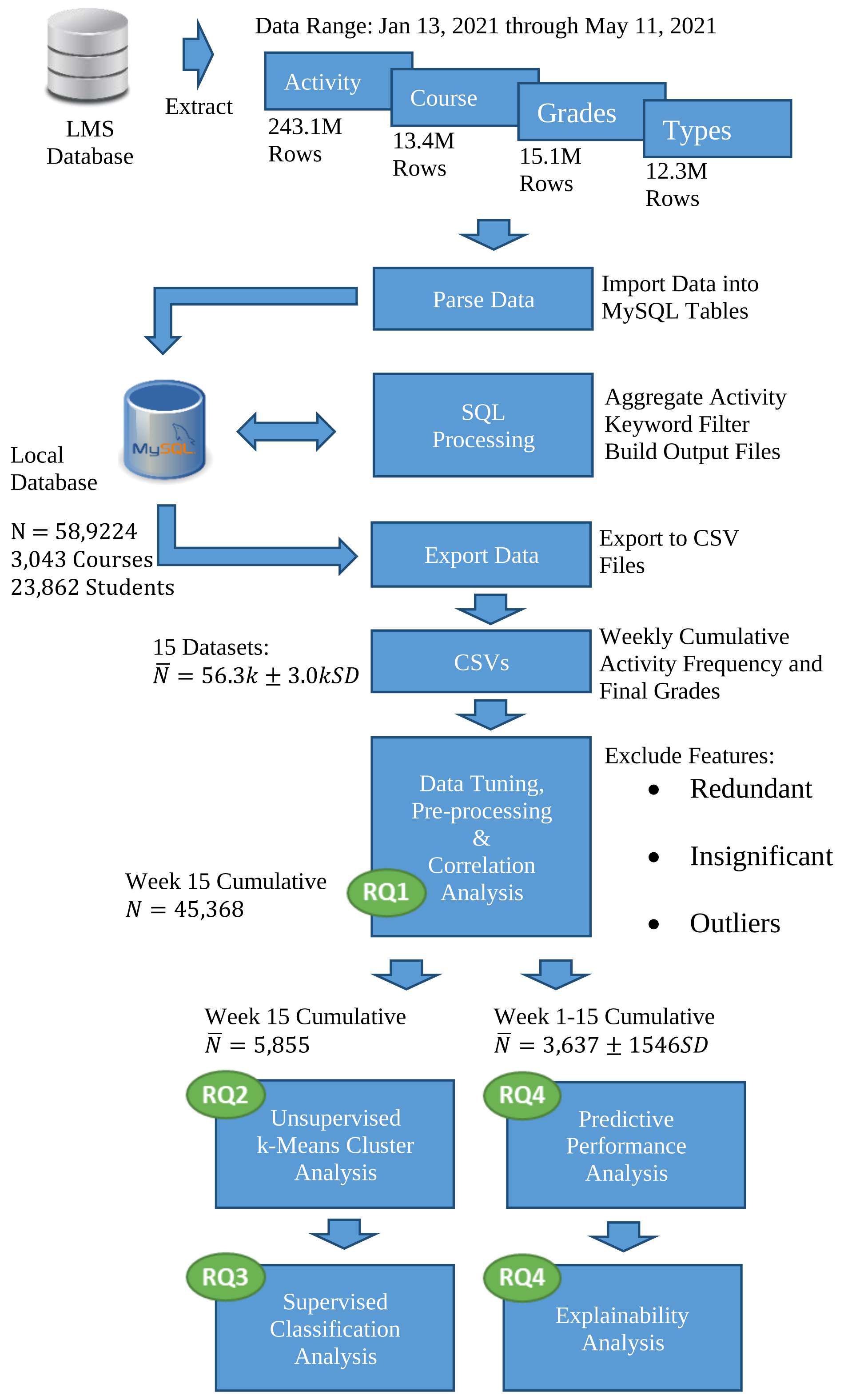
5.4. Exclude outliers beyond 2 standard deviations

5.5. Transform/Scale Data to normalize activity frequency between 0 and 1 (MinMaxScaler)

The Python pre-processing step includes feature exclusions to reduce collinearity effects, which could negatively impact model performance and accuracy (Abdulhafedh, 2021; Field, 2018; Kondo et al., 2017; Layton, 2017). Figure 21 illustrates the actual data collection and processing steps performed with associated sample sizes identified at each stage. The dataset comprised comma-separated value (CSV) files representing selected data tables extracted from the LMS database. Personal identifiable information was excluded to ensure the privacy and confidentiality of user data. Data was also securely stored and processed on a password-protected computer implementing multi-factor authentication and volume encryption.

Figure 21

Data Collection and Processing Results Flow Chart



Descriptive Statistics

Table 9

Descriptive Statistics of Raw Data Before Feature Selection

	N	Minimum	Maximum	Mean	Std. Deviation
disc_cnt	45295	1.0	349327.0	514.521	3506.8233
mod_cnt	24538	1.0	64860.0	227.692	649.6914
email_cnt	13055	1.0	15100.0	27.105	221.1084
mobile_cnt	16130	1.0	35425.0	338.426	690.1081
res_cnt	21327	1.0	748155.0	2702.885	17587.5283
grade_cnt	42492	1.0	11295.0	89.271	204.8992
quizexam_cnt	25573	1.0	167918.0	383.805	2311.3193
rubric_cnt	5754	1.0	7880.0	64.218	355.1623
announce_cnt	28967	1.0	5705.0	90.929	203.5616
tools_cnt	3204	1.0	6581.0	124.147	493.2645
final_grade	45365	.00	1.00	.8378	.20456

Feature Selection Results

A preliminary feature selection analysis was performed by reviewing the LMS database schema to identify relevant tables to extract. With a specific interest in student activity and performance outcomes, the activity and grades table were selected for extraction. The course and grade type tables were also included as they provided cross-reference keys (i.e., user and course primary keys) that linked the various tables together.

Following the import of data into a local MySQL database, feature selection analysis was performed to identify relevant and available student-logged activity data. This involved relevant keyword searches on student activity types supported by the literature research and theoretical background. This highlights an important point that activity features are limited by the existing data. As a result, the following 10 keywords were identified:

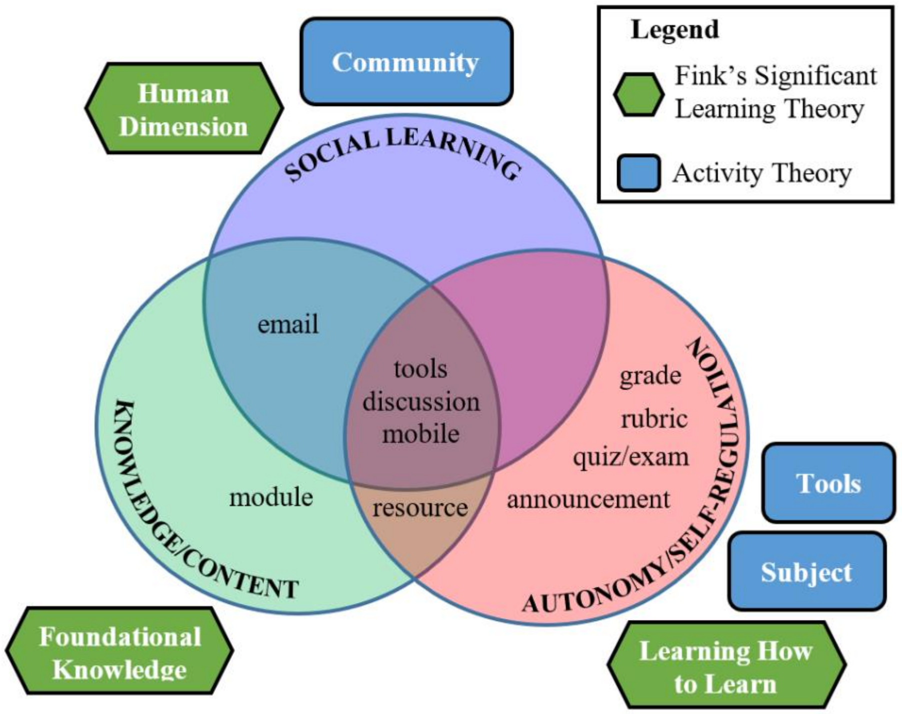
1. “discussion”
2. “module”
3. “email”
4. “mobile”
5. “resource”
6. “grade”
7. “quiz” OR “exam”
8. “rubric”
9. “announcement”
10. “tools”

These keywords represent potential and relevant features selected based on LMS logged data that occurred the most often and are selected to ensure sufficient data collection and sample size. The features are then compared against the theoretical framework identified in Figure 22. Features that can be categorized into the areas of *Social Learning*, *Knowledge/Content,* and *Autonomy/Self-regulation* are therefore supported by the underlying theoretical supports and can serve as preliminary influential learning factors. The Venn model also identifies how each activity is related to one or more learning dimensions that may provide scaffolding effects. In theory, a greater number of overlapping dimensions for any activity would result in greater learning potential. However, there may be a point of diminishing returns due to task or cognitive overload.

We can observe that some features can impact different aspects of the learning process. Tools, discussion, and mobile are similar in that they provide collaborative functions enabling the learning of content while requiring some level of student autonomy and self-regulation. Email is typically a learner-learner or learner-instructor interaction that facilitates clarification of understanding. The resource component represents supplementary material that aids the learning process but is not typically required. Students who access resources are more self-driven and are eager to learn how to learn. Student activity access to grades, quizzes/exams, and rubric pages are related to self-assessment, which can promote learning through self-regulation in response to gaps. Finally, autonomous, and self-regulated learners are more likely to check announcements to ensure learning tasks are completed on time or instructor notifications are addressed. The learning model shown provides a relevant framework for effective initial feature selection.

Figure 22

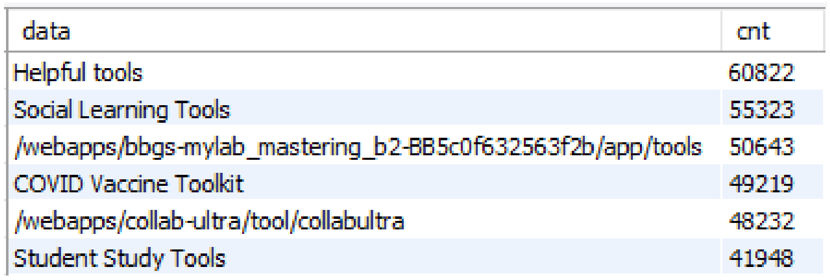
Conceptual Model of Learning Support Dimensions



Using structured query language (SQL), the frequency count of relevant keywords was obtained from the activity log table. Figure 23 gives an example of keyword search results identifying the top sorted frequency count of relevant log text containing the keyword “tool”. This process was repeated for relevant keywords identified by either visually inspecting the data or purposefully selecting from theory.

Figure 23

Screenshot of SQL Keyword Search Frequency Results



data	cnt
Helpful tools	60822
Social Learning Tools	55323
/webapps/bbgs-mylab_mastering_b2-BB5c0f632563f2b/app/tools	50643
COVID Vaccine Toolkit	49219
/webapps/collab-ultra/tool/collabultra	48232
Student Study Tools	41948

Analysis Results

Feature Analysis (RQ1)

This section presents the results of the feature analysis, which uses correlation to and RQ1 (i.e., determine the associations between activity factors and student final grades). Statistical significance is determined at the .05 level and effect size is interpreted using Cohen et al.'s (2018) recommendation for weak (±0.1), modest (±0.3), moderate (±0.5), strong (±0.8), and very strong (≥ 0.8) values of Pearson's *r* respectively. For initial correlation analysis, the 15th-week cumulative data set (N=45365) was used to account for all samples collected. Figure 24 and Figure 25 give the Pearson's correlation coefficient (*r*) and p-value matrix, respectively.

Figure 24

Correlation (Pearson's R) Matrix

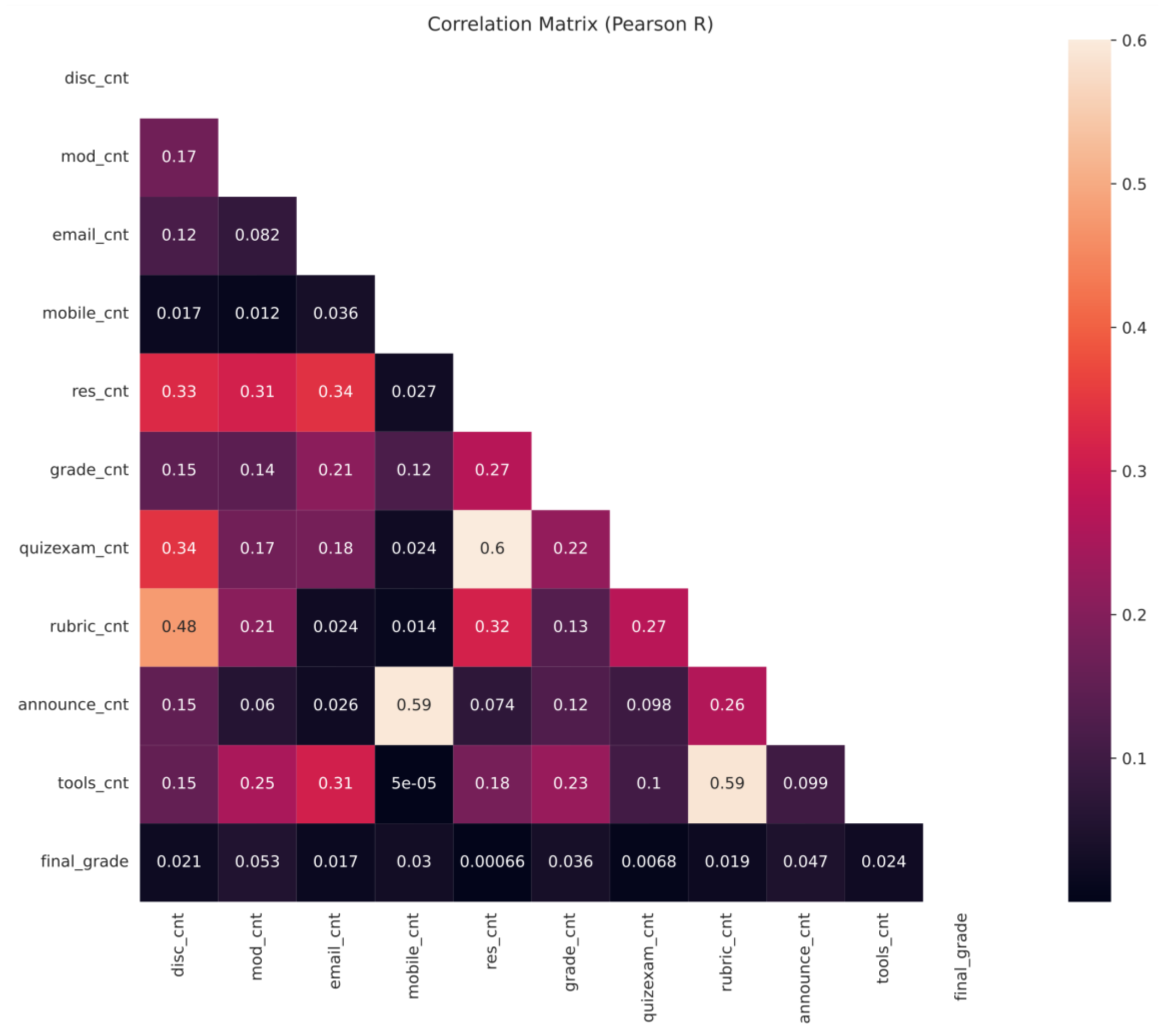
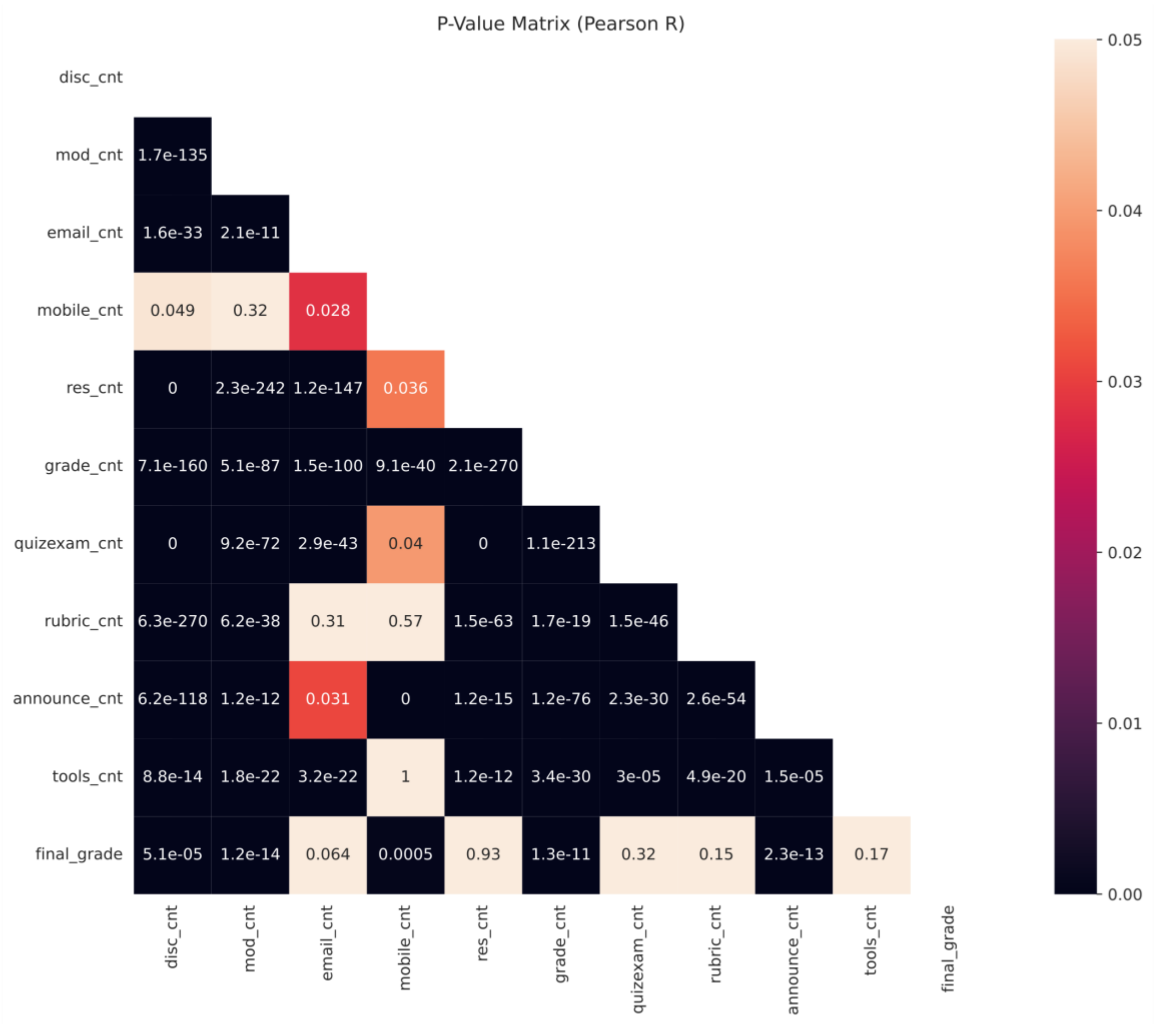


Figure 25

P-Value Matrix (Pearson's R)



Overall, the Pearson's correlation results show that final grades have a statistically significant and weak positive correlation ($r < .1$, p < .001) to student interaction frequency with discussions forums (disc_cnt), module content (mod_cnt), mobile access (mobile_cnt), grade pages (grade_cnt), and announcement notifications (announce_cnt). All other activities (email, resource, quiz/exam, rubric, and tools access frequency) were not statistically significant (p > .05).

The lack of significant correlations may be attributed to the fact that the correlation analysis was based on the initial population dataset (45,365 samples covering 2,908 courses and 22,953 students), which included both online and traditional in-classroom courses. As mentioned previously, only a small number of undergraduate and graduate students are enrolled online, which averages around 25% of total enrolled students according to the institution's leadership. As such, the majority of students enrolled in courses that required little to no student activity. This explanation is supported by Chamizo-Gonzalez et al.'s (2015) findings that correlation varies not only by activity type but also by course type. When all courses were included in the correlation analysis, correlations were significantly lower than the maximum of individual correlations between final grade and student activity frequency. In their study of 4,989 students, Chimizo-Gonzalez found the following correlations with respect to final grade: resources views (r = .13, p < .001; r ranged between 0 to .4), content page views (r = .17, p < .001; r ranged between .01 and .4), discussion post views (r = .04, p < .01; r ranged between .01 and .24), and quiz page views (r = .14, p < .001; r ranged .02 and .39). In short, general predictive power is low, but may be high concerning sub-groups such as course types.

To address these problems, additional sample exclusions were performed to remove insignificant features, pairs with strong correlations, and outliers. The features email, resources, quiz/exam, rubric, and tools (email_cnt, res_cnt, quiz_cnt, exam_cnt, rubric_cnt, and tools_cnt respectively) were excluded as they have statistically insignificant (p > .05) correlations to final grade due to a lack of associative and predictive power. Features with strong correlations (Pearson's r > 0.6) were also checked and removed to mitigate collinearity concerns (L. Cohen et al., 2018). Reducing collinearity by feature exclusion can help improve statistical significance (Shrestha, 2020) as well as model effectiveness. For the notable correlated pair, the feature with a higher number of missing values was excluded to mitigate data loss.

Finally, the dataset was analyzed to identify and remove outliers beyond 1.5 the Interquartile Range. Only a single iteration was performed to alleviate data loss. After removing outliers, a total sample size of 5,855 remained. Afterward, a Minmax scaler was performed to normalize all activity features to the ratio range between 0 and 1, which allows for a relative comparison between activity frequency and final grade. Figure 26 and Figure 27 give the given sample distribution histograms for the final grade and features, respectively. Observe that the final grade distribution has a non-normal, left-skew, and leptokurtic shape whereas the feature distributions have a non-normal, right-skew, and leptokurtic shape. This data sample served as the dataset for the proceeding exploratory clustering analysis.

Figure 26

Final Grade Distribution Histogram

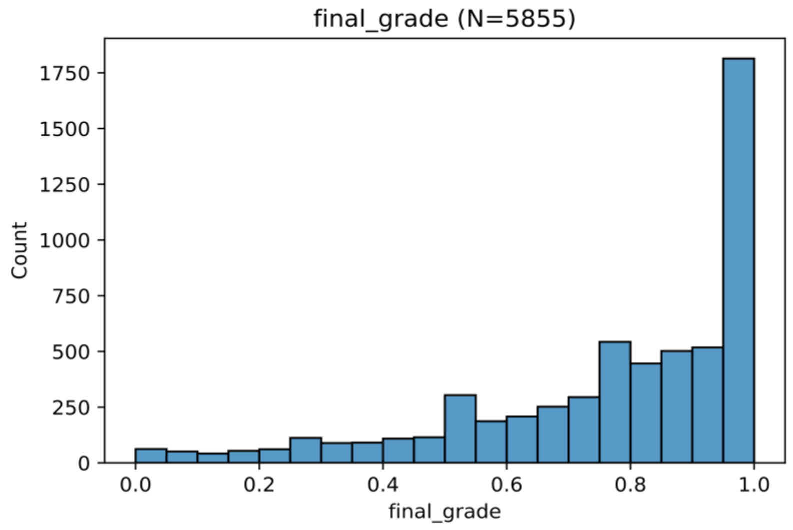
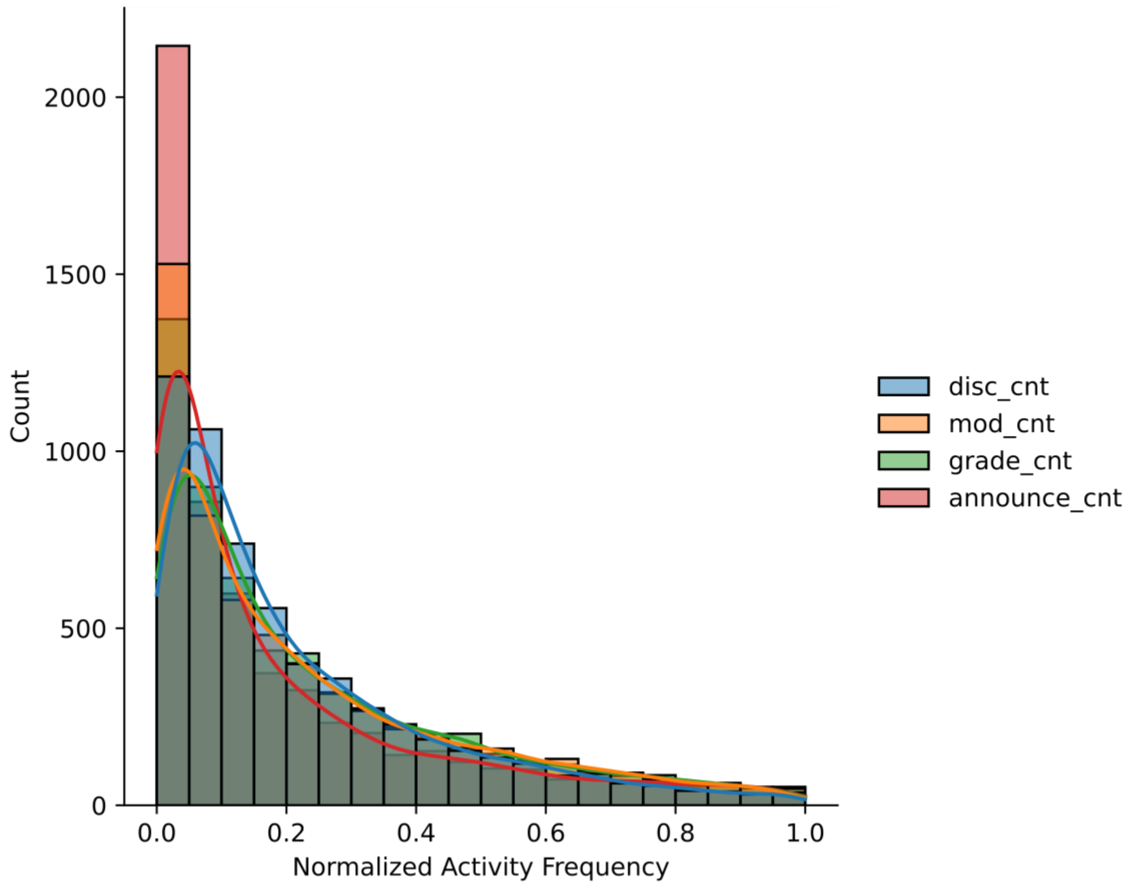


Figure 27

Feature Distribution Histogram



The results of the exploratory cluster analysis, which is described in the next section, revealed a confounding cluster characterized by a high mean final grade and low activity frequency across all features. This group can be explained as students who did not interact with the LMS for most of their learning, such as those who prefer to use external resources, or students who belong to traditional in-classroom courses. They may use the LMS to obtain course materials, but not actively participate in discussion posts or module content access as these were not implemented for traditional courses. Given the fact that this group represented the largest cluster size (34.4%), it is more likely that this group contained traditional learners as discussion posts and module access are required components of online courses requiring frequent participation. Samples from traditional courses not only underestimate student activity in online courses but also skew the correlation results due to significant group size. As such, this group was excluded from the data set and a correlation analysis was recomputed to obtain a more reliable assessment.

Figure 28 and Figure 29 give the correlation results after excluding the confounding group, which corrects for traditional learning effects. Observe that the correlation results have improved as discussion, module, and grade access frequencies had modest positive and statistically significant correlations (*r* = .22, .21, and .17 respectively with p < .001) to the final grade. In addition, announcement page access frequency had a weak positive and statistically significant correlation (*r .096, p < .001)* with the final grade. The updated correlations after sampling exclusion demonstrated a significant improvement compared to previous results (*r* = .02, .05, .03, and .05 for discussion, module, grade, and announcement activity respectively).

The results indicate that discussion, module, and grade access frequency have a modest positive association with learning outcomes while announcement access frequency has a weak association. These results are comparable to Chimizo-Gonzalez's findings for specific courses (2015). This highlights the importance of detecting and removing bias from traditional students, which negatively impacted features associated with content (disc_cnt, mod_cnt), social interactions (disc_cnt), and motivation/self-regulatory activities (grade_cnt and announce_cnt).

Figure 28

Correlation Matrix (Pearson's R) After Sample Exclusion

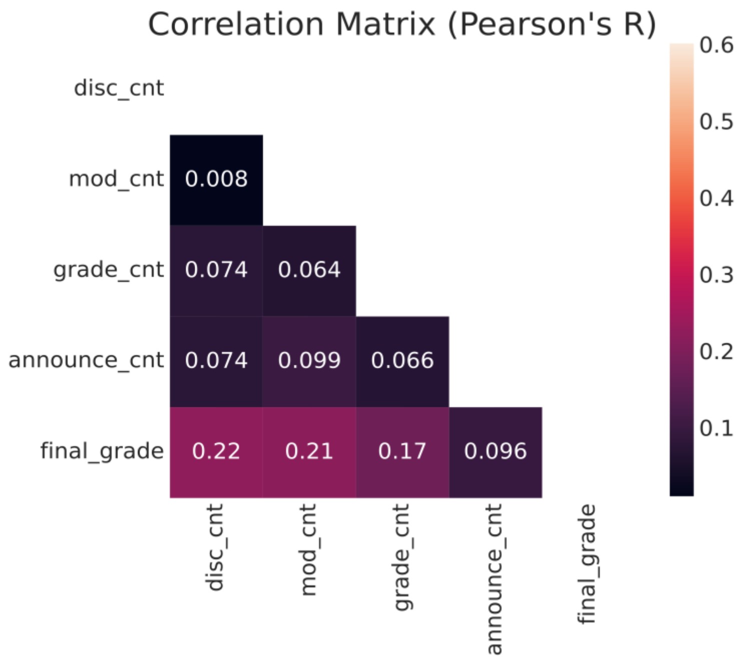
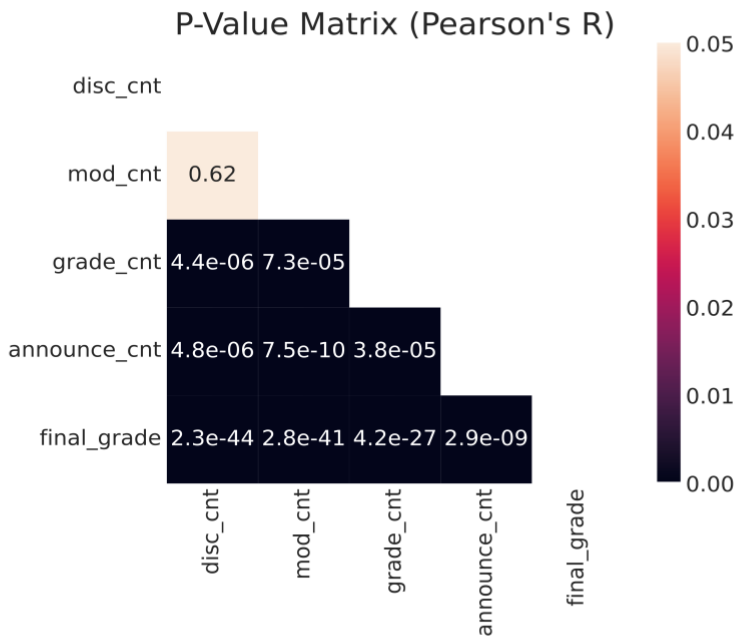


Figure 29

P-Value Matrix (Pearson's R) After Sample Exclusion



Exploratory Cluster Analysis (RQ2)

To answer RQ2 (i.e., What are characteristics of at-risk students that can be identified using educational data mining?), a k-Means cluster exploratory analysis was conducted on the 15th-week cumulative dataset. For exploratory analysis, *the final grade was included as a feature*, which helped improve clustering results. However, using final grade as a feature also has predictive power trade-offs, which will be discussed later. As mentioned previously, outliers and missing values were excluded. The remaining dataset was then normalized between 0 and 1 using the SciKit-Learn MinMaxScaler function to alleviate unbalanced distributions among feature frequencies. Normalizing also enables relative comparisons between feature frequency as a ratio of maximum LMS interaction.

K-Means clustering is comprised of: (1) finding the optimal cluster size K (see Figure 30); and (2) obtaining/analyzing k-Means clustering results (Figure 31 and Figure 32) to identify characteristics of low (at-risk) and high (not-at-risk) performing students.

Figure 30

K-Means SSE (Exploratory)

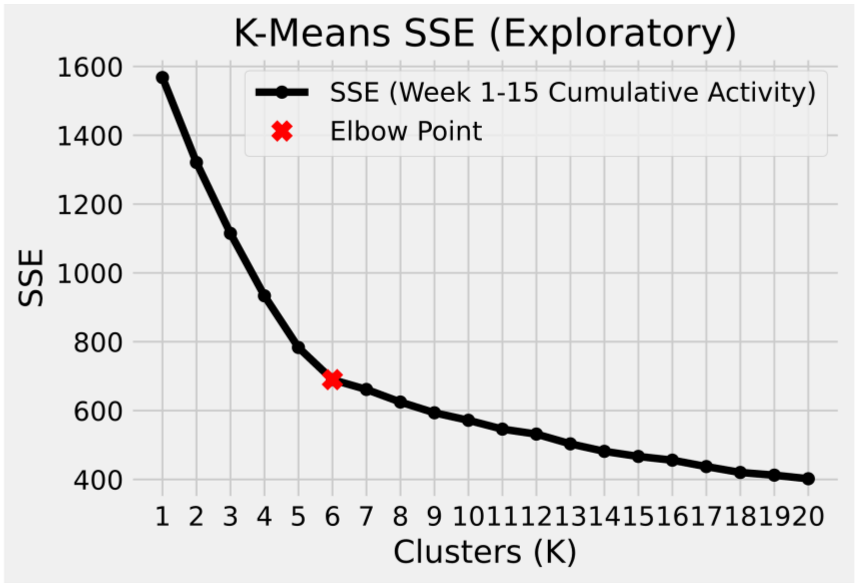
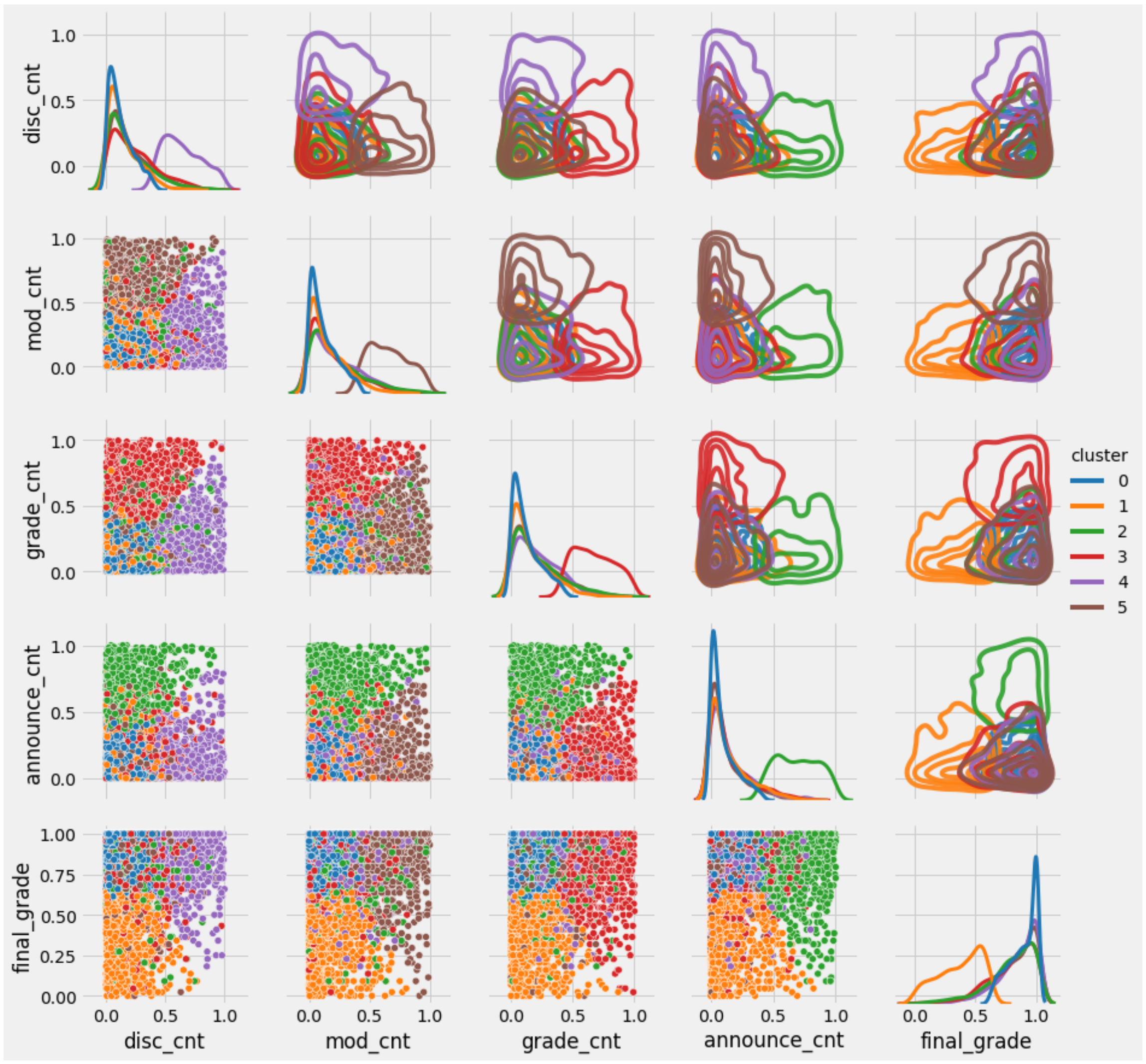


Figure 31

K-Means Cluster Matrix (Exploratory)



In Figure 31, the diagonal charts represent the histograms for the associated clusters concerning the associated feature of interest. Observe that there is significant skewing with multiple peaks of different heights, all of which signify unbalanced distributions. Concerning final grades, there is distinct clustering for all features. Figure 32 gives k-Means results as normalized mean activity (left axis) and final grade (right axis) concerning each cluster (bottom axis). Analyzing this chart, student at-risk characteristics were labeled and described based on activity frequency and its relationship to mean final grade scores (see Table 10).

Figure 32

K-Means Cluster Summary (Exploratory)

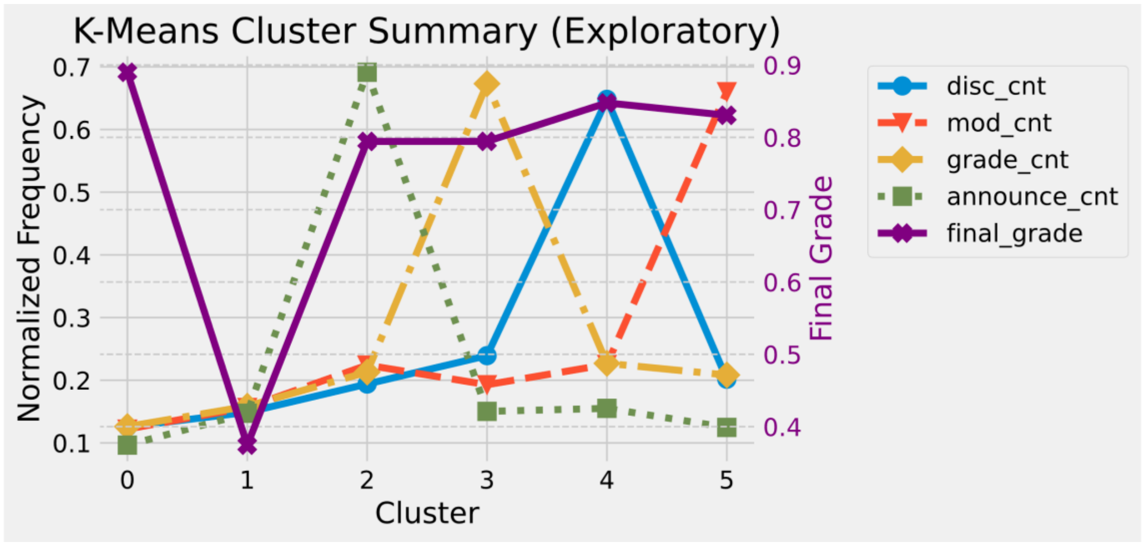


Table 10

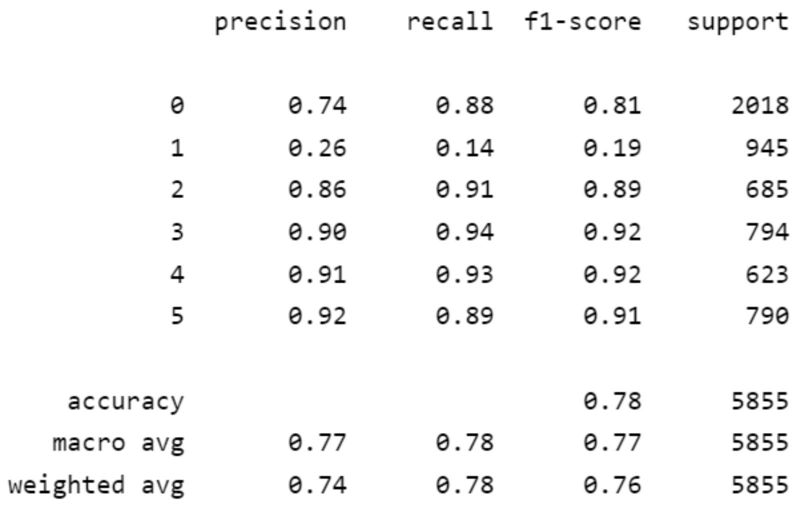
Exploratory Group Labeling and Characterization

K	Label	At-Risk	LMS Activity Characteristic	Mean Final Grade
0	Inactive High Performer (IHP)	No	Lowest activity for all features.	.88 ± .11SD N = 2018 (34.4%)
1	Inactive Low Performer (ILP)	Yes	Very low activity for all features.	.37 ± .18SD N = 945 (16.1%)
2	Announcement-Active Good Performer (AAGP)	No	High announcement activity; all others very low.	.79 ± .19SD N = 685 (11.7%)
3	Grade-Active Good Performer (GAGP)	No	High-grade access activity; all others very low.	.79 ± .19SD N = 794 (13.5%)
4	Discussion-Active Good Performer (DAGP)	No	High discussion activity; all others very low.	.84 ± .16SD N = 623 (10.6%)
5	Module-Active Good Performer (MAGP)	No	High content activity; all others very low.	.83 ± .18SD N = 790 (13.5%)

Table 10 identifies one at-risk group (Inactive Low Performer; cluster 1) characterized by having very low relative activity among all features. Note that the Inactive High Performer group also had similar characteristics, as it had the lowest activity among all features with slightly lower activity than the at-risk Inactive Lower Performer group. This confounding result is problematic as there are no obvious discriminating factors between the two groups, which was found to limit predictability. Figure 33 gives the results of conducting a Naives Bayes classification using 10-fold cross-validation. Note that the final grade was removed as a feature during model training as in practice, the final grade is not available until the end of the course. This has some negative implications. Observe that, overall, there are high accuracies for clusters 2 – 3, but lower accuracies for clusters 0 and 1, which are the high performers and at-risk groups. This is expected as clusters 0 and 1 are very similar concerning activity frequencies (both had the lowest) with no obvious differentiating factors distinguishing them apart.

Figure 33

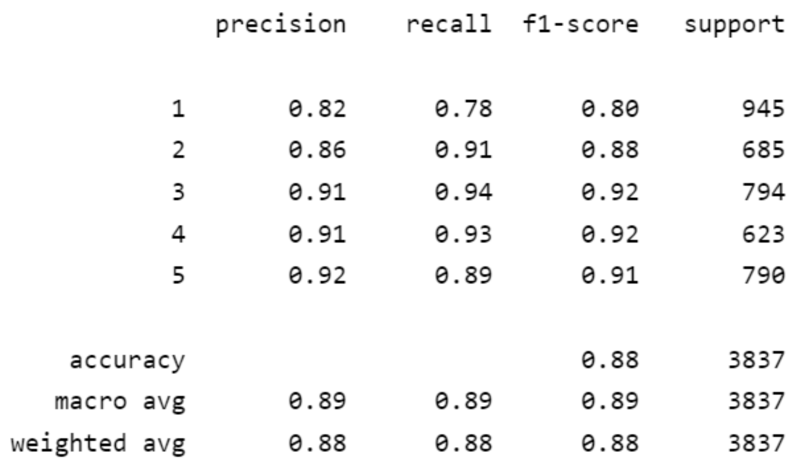
Naives Bayes Preliminary Classification Results


```
              precision    recall  f1-score   support

           0       0.74      0.88      0.81      2018
           1       0.26      0.14      0.19       945
           2       0.86      0.91      0.89       685
           3       0.90      0.94      0.92       794
           4       0.91      0.93      0.92       623
           5       0.92      0.89      0.91       790

    accuracy                           0.78      5855
   macro avg       0.77      0.78      0.77      5855
weighted avg       0.74      0.78      0.76      5855
```

The lack of discriminating factors could be attributed to external factors or features not measured. For example, the Inactive High Performer student could be the type of learner who prefers to study or communicate outside of the LMS environment, which results in lower LMS activity. Another explanation is that this group represented students who were enrolled in traditional courses that were delivered in a classroom environment and did not require online participation. The latter explanation is more likely as the confounding group (cluster 0) had almost three times (N = 2018 or 34.4%) the group size compared to other clusters (N = 623 – 945 or 10.6% - 16.1%). This is consistent with the institution enrollment statistics where the majority of students were enrolled in traditional in-classroom courses. Without a discriminating factor, the predictive power of these classifications will be limited once the final grade is removed from the feature set during supervised classification. To address this, clusters with low activity (mean values less than 0.2) and high final grades (values greater than 0.8) were excluded to remove traditional learners from the dataset. As a result, Figure 34 shows that the Naïve Bayes prediction performance for the at-risk group (cluster 1) has been significantly improved (61% improvement of f-score from 0.19 to 0.8).

Figure 34

Naives Bayes Preliminary Classification Results (Excluding Cluster 0)



```
              precision    recall  f1-score   support

           1       0.82      0.78      0.80       945
           2       0.86      0.91      0.88       685
           3       0.91      0.94      0.92       794
           4       0.91      0.93      0.92       623
           5       0.92      0.89      0.91       790

    accuracy                           0.88      3837
   macro avg       0.89      0.89      0.89      3837
weighted avg       0.88      0.88      0.88      3837
```

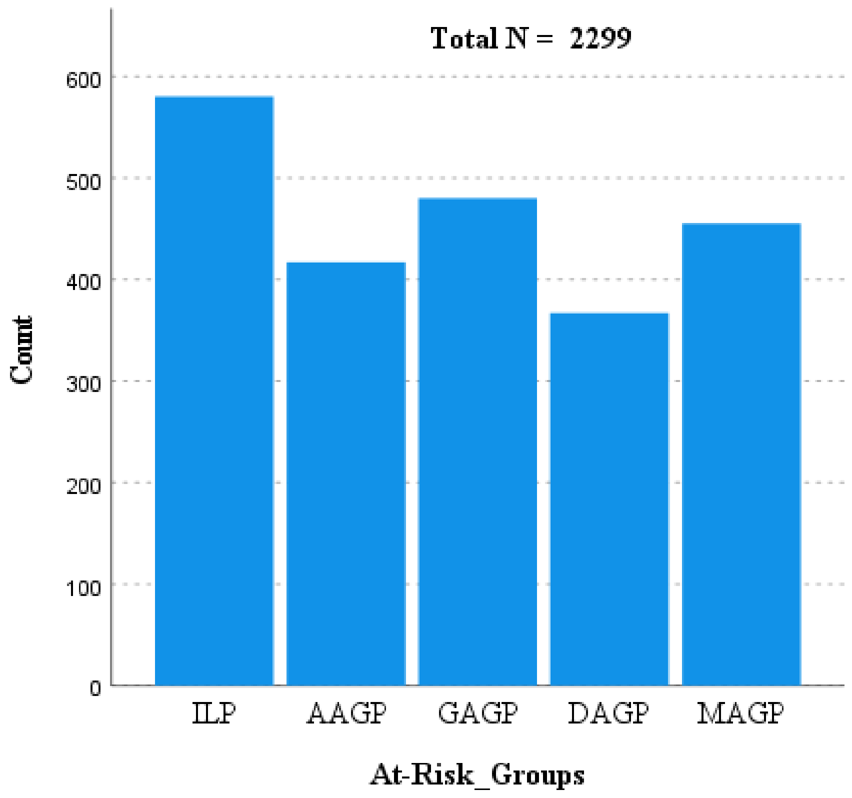
Statistical Analysis of Mean Differences between Independent Groups

This section describes the statistical analysis performed to determine if mean differences between final grade scores differ among the five at-risk groups (ILP, AAGP, GAGP, DAGP, and MAGP) identified in Table 10 and Figure 34. First, a description of the descriptive statistics and assumptions checks will be given followed by the test for mean differences using the Kruskal-Wallis test.

Descriptive Statistics. After excluding the IHP group, the updated population sample had a sample size of N = 3837. A 60% random sampling (N = 2299) was performed on the population for statistical analysis. The associated descriptive statistics are given in Table 11 and group size proportions are depicted in Figure 35, where the groups ILP, AAGP, GAGP, DAGP, and MAGP have a sample size of 580 (25.2%), 417 (18.1%), 480 (20.9%), 367 (15.9%), and 455 (19.8%) respectively. This ensures that the smallest sample size for each cluster is near or greater than 368, which was the group size determined from prior power analysis with sufficient power to detect small effects at the .05 level. Although the groups are not equal, they are comparable and sufficiently large.

Figure 35

Proportions of Random-Sampled Data Set



From Table 11, the overall mean final grade was 0.705 (±0.266SD) with the following normalized mean frequencies: 0.268 (±.238SD) for discussion access; .286 (±.258SD) for module access; .295 (±.261SD) grade page access; and .242 (±.266SD) for announcement page access.

Table 11

Descriptive Statistics

	N	Minimum	Maximum	Mean	Std. Deviation
disc_cnt	2299	0	0.922	0.268	0.238
mod_cnt	2299	0	0.922	0.286	0.258
grade_cnt	2299	0	1	0.295	0.261
announce_cnt	2299	0	1	0.242	0.266
final_grade	2299	0	1	0.705	0.266

Assumption Checks. Normality was assessed by analyzing the skewness and kurtosis of each distribution using equations (5) and (6) to determine significance, where Z = z-score, S = skewness value, K = kurtosis value, and SE = standard error (Field, 2018). Significance is determined when the absolute value of Z is greater than 1.96 (i.e., p < .05). The calculated Z-scores for skew and kurtosis are given in Table 12, which shows there is a statistically significant presence of both skew and kurtosis (Zs > 1.96 and Zk > 1.96). Therefore, normality is violated.

$$Z_{skewness} = \frac{S-0}{SE_{skewness}} \tag{5}$$

$$Z_{kurtosis} = \frac{K-0}{SE_{kurtosis}} \tag{6}$$

Table 12

Calculated Z-scores for Skew and Kurtosis

	Skewness			Kurtosis		
	Statistic	SE	Zs	Statistic	SE	Zk
disc_cnt	1.027	0.051	20.13725	0.232	0.102	2.27451
mod_cnt	0.829	0.051	16.2549	-0.347	0.102	3.401961
grade_cnt	0.827	0.051	16.21569	-0.376	0.102	3.686275
announce_cnt	1.156	0.051	22.66667	0.228	0.102	2.235294
final_grade	-0.76	0.051	14.90196	-0.381	0.102	3.735294

To check for violations in homogeneity, Levine's test was conducted using SPSS one-way ANOVA test and shown in Table 13. Levine's test results rejected the null hypothesis of equal variances for all independent and dependent variables (p < .05). Therefore, homogeneity is violated.

Table 13

Test of Homogeneity of Variances

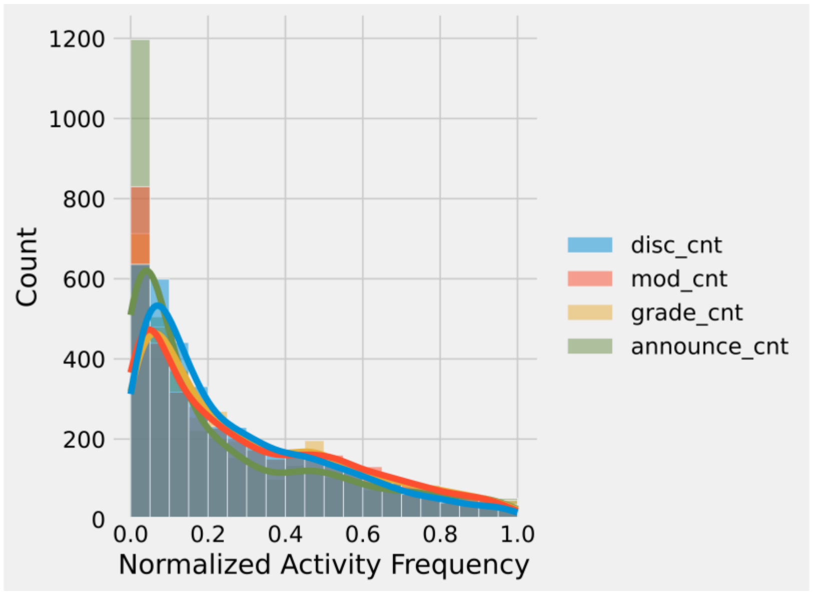
		Levine Statistic	df1	df2	Sig.
disc_cnt	Based on Mean	15.257	4	2294	.000
mod_cnt	Based on Mean	11.840	4	2294	.000
grade_cnt	Based on Mean	6.923	4	2294	.000
announce_cnt	Based on Mean	7.443	4	2294	.000
final_grade	Based on Mean	2.481	4	2294	.042

Since there were violations in both normality and homogeneity, the Kruskal-Wallis test was selected for testing mean differences. The Kruskal-Wallis is a non-parametric test that is commonly used for assessing mean differences between independent samples when assumptions of normality or homogeneity are violated (L. Cohen et al., 2018; Field, 2018). In addition, there are assumptions for randomly generated samples, independent group samples, and similar underlying distributions (L.

Cohen et al., 2018). As previously noted, the data was randomly selected from the population where each group represent independent samples. This meets the first two assumptions. Figure 36 gives the distribution histograms of the activity frequencies, all of which demonstrated a right-skewed distribution, which meets the last assumption requiring similar distribution shapes.

Figure 36

Distribution Comparison



Kruskal-Wallis Mean Difference Test Results. A Kruskal-Wallis test was conducted to evaluate differences in final grades among the five at-risk groups (ILP, AAGP, GAGP, DAGP, and MAGP). The test, which was corrected for tied ranks, was significant *H(4) = 1078.807, p < .001* (see Table 14).

Table 14

Independent-Samples Kruskal-Wallis Test Summary

Total N	2299
Test Statistic	1078.807[a]
Degree Of Freedom	4
Asymptotic Sig.(2-sided test)	.000

Note. a. The test statistic is adjusted for ties.

Follow-up tests were conducted to evaluate pairwise differences among the three groups, controlling for Type I error across tests by using the Bonferroni approach. Effect size is calculated as $r = Z/\sqrt{N}$, where Z is the standard test statistic and N is the total sample size of group comparisons (Field, 2018). The results of these tests are presented in Figure 37 and Table 15. In summary, there was a significant difference in mean final grades among at-risk groups, *H(4) = 1078.807, p < .001*. In summary, the pairwise comparison with adjusted p-values indicated the following:

- ILP group had a significantly lower mean final grade than groups:
  - GAGP, H(4) = -954, p < .001, r = -.72
  - AAGP, H(4) = -976, p < .001, r = -.73
  - MAGP, H(4) = -1102, p < .001, r = -.83
  - DAGP, H(4) = -1120, p < .001, r = -.82
- GAGP group had significantly lower mean final grade than groups:
  - MAGP, H(4) = -148, p = .006, r = -.11
  - DAGP, H(4) = -166, p = .003, r = -.12
- AAGP group had significantly lower mean final grade than groups:
  - MAGP, H(4) = -126, p = .051, r = -.09

  - DAGP, H(4) = -143, p = .025, r = -.11

- There were no significant differences between:
  - GAGP and AAGP, H(4) = 22.398, p = 1.0, r = .02
  - MAGP and DAGP, H(4) = 17.45, p = 1.0, r = .01

Figure 37

At-Risk Group Final Grades Boxplots

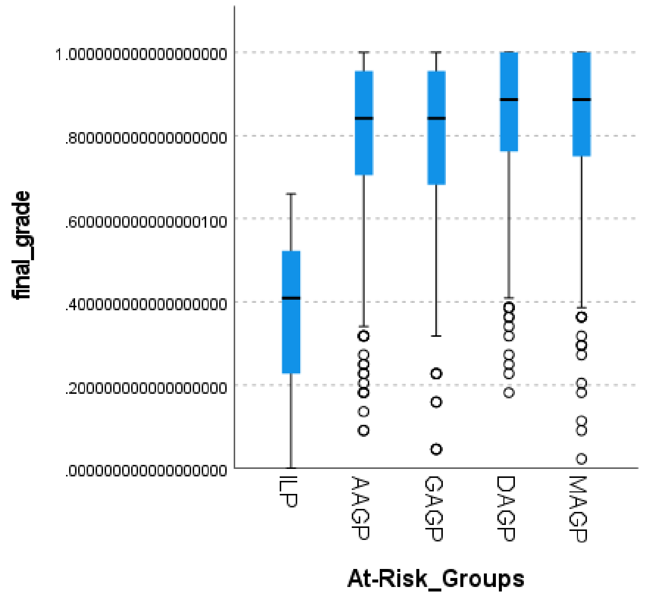


Table 15

Final Grade Pairwise Comparisons of At-Risk Groups

Sample 1-Sample 2	Test Statistic	Std. Error	Std. Test Statistic	Sig.	Adj. Sig.[a]	N	*r*
ILP-GAGP	-953.940	40.854	-23.350	.000	.000	1060	-0.72
ILP-AAGP	-976.338	42.509	-22.968	.000	.000	997	-0.73
ILP-MAGP	-1102.148	41.463	-26.581	.000	.000	1035	-0.83
ILP-DAGP	-1119.599	44.161	-25.352	.000	.000	947	-0.82
GAGP-AAGP	22.398	44.322	.505	.613	1.000	897	0.02
GAGP-MAGP	-148.208	43.321	-3.421	.001	.006	935	-0.11

(continued)

Sample 1-Sample 2	Test Statistic	Std. Error	Std. Test Statistic	Sig.	Adj. Sig.[a]	N	*r*
GAGP-DAGP	-165.658	45.910	-3.608	.000	.003	847	-0.12
AAGP-MAGP	-125.810	44.885	-2.803	.005	.051	872	-0.09
AAGP-DAGP	-143.260	47.388	-3.023	.003	.025	784	-0.11
MAGP-DAGP	17.450	46.453	.376	.707	1.000	822	0.01

Note. Each row tests the null hypothesis that the Sample 1 and Sample 2 distributions are the same. Asymptotic significances (2-sided tests) are displayed. The significance level is .050.

a. Significance values have been adjusted by the Bonferroni correction for multiple tests.

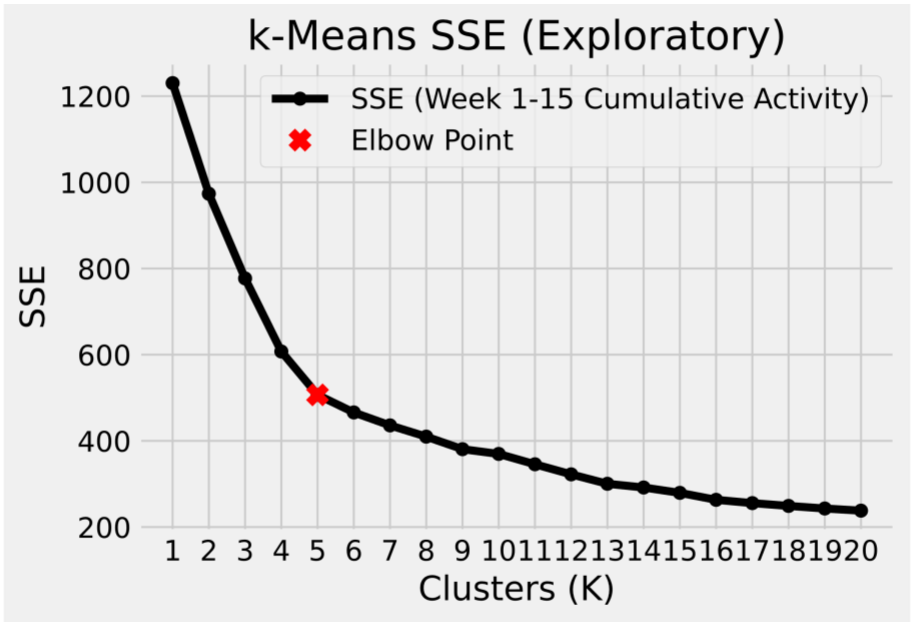
Classification Analysis (RQ3)

In this section, the cluster analysis was re-performed based on activity-only features, which would reveal naturally occurring clusters associated with their mean final grades for interpretation and analysis. This method also is more representative of implementation in practice (i.e., final grades are not available as a feature for early at-risk prediction).

Using only activity as features, K-Means clustering was repeated for the 15th-week cumulative data set. Figure 38 gives the SSE plot with the elbow point identified at K = 5 using the Python Kneed package. We can verify that this point is credible due to the sharp change in slope (elbow) at K=5. Overall, this indicates that there are 5 natural clusters when considering only activity data as features.

Figure 38

K-Means SSE (Classification)



Using K=5, k-Means clustering was conducted to explore naturally occurring patterns. The k-Means classification matrix is shown in Figure 39. Like the exploratory clustering analysis, the diagonal plots show skewed distributions, which indicates the presence of unbalanced data sets. Furthermore, the overall cluster separation is distinct between feature-to-feature interactions. However, this is no obvious separation between the final grade and each of the activity features (last row). In the last column of the contour plot, there is significant overlap. To get a better measure of final grade cluster separation, mean activity and scores were analyzed.

Figure 39

K-Means Cluster Matrix (Classification)

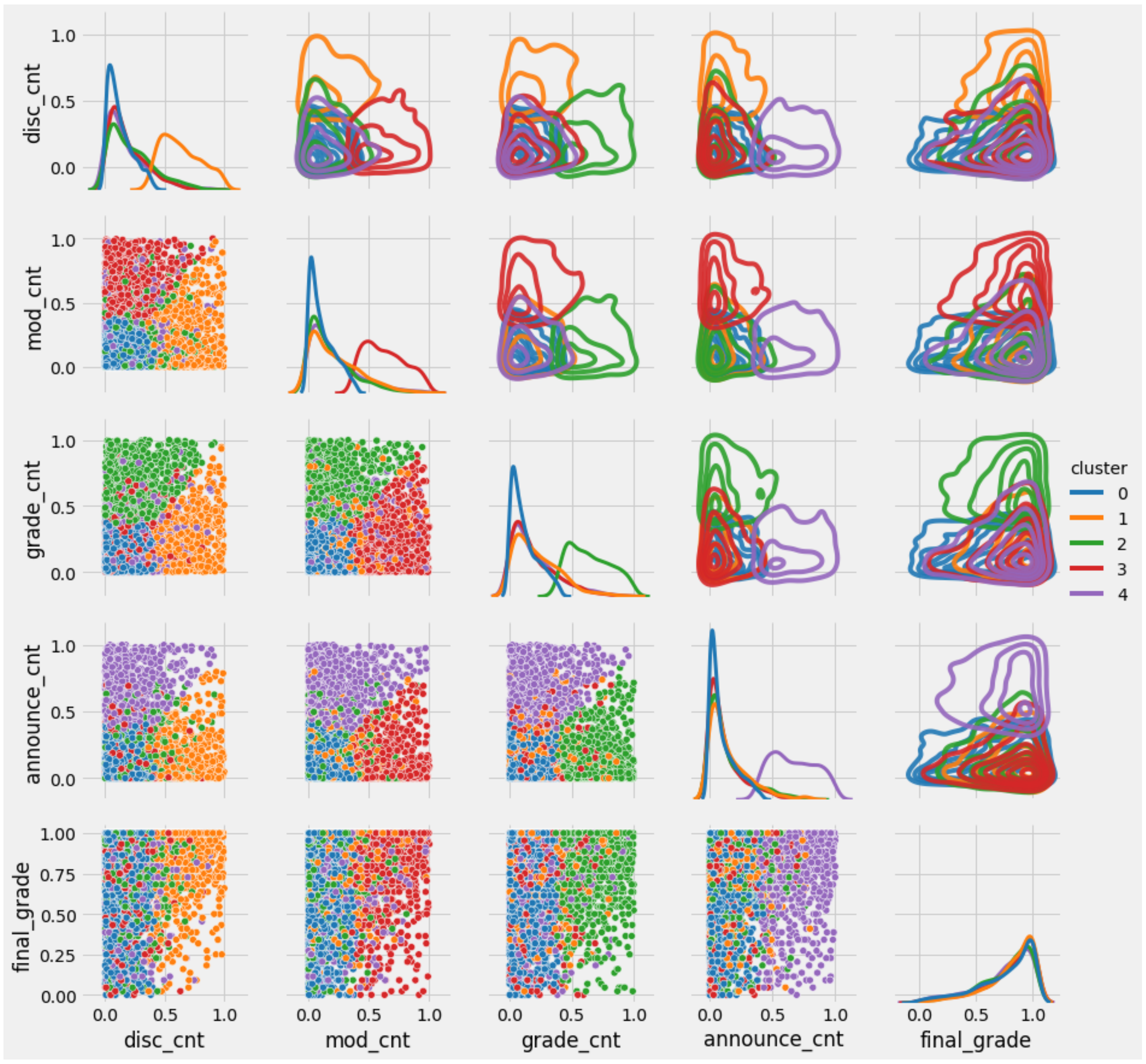
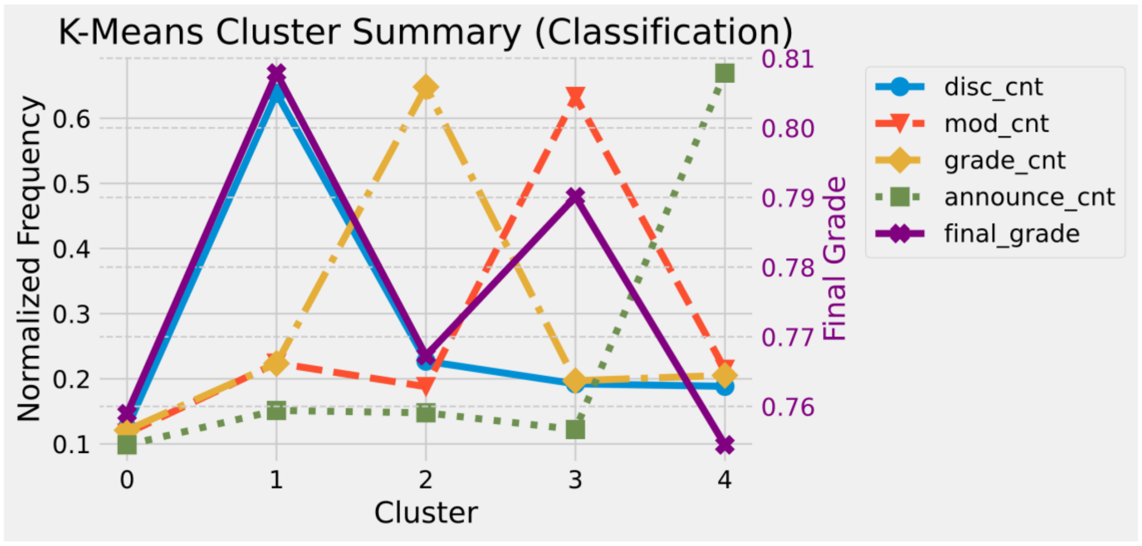


Figure 40 depicts the k-Means cluster summary, which plots the mean normalized activity and final grade concerning each cluster. Like the exploratory analysis, there are discriminating factors for groups 1 through 4 (disc_cnt, grade_cnt, mod_cnt, and announce_cnt respectively). The final grade range is also narrow (76 – 81). Thus, the differences in at-risk and not at-risk students are much more subtle with significant overlap.

Figure 40

K-Means Cluster Summary (Classification)



Analyzing Figure 40, the at-risk groups were labeled and characterized in Table 16. Observe that higher final grades were associated with higher discussion and module activities (clusters 1 and 3) but not higher grade page nor announcement activity frequency (clusters 2 and 4). Furthermore, low activities across all features resulted in lower final grades. Finally, clustering based on activity-only features demonstrated a narrow range of predictable at-risk final grade levels. The characterization found is partially consistent with studies leveraging activity theory (Xing et al., 2015) and social learning theory (Nabavi, 2012) in that certain types of student interactions, such as discussion and module access frequency, with the learning environment can influence performance outcomes.

Table 16

Exploratory Group Labeling and Characterization

K	Label	At-Risk	LMS Activity Characteristic	Mean Final Grade
0	Inactive Low Performer (ILP)	Yes	Low activity across all features; coupled with a lower final grade.	0.76 ± .25SD N = 2591 (44.2%)
1	Discussion-Active High Performer (DAHP)	No	High discussion activity with a low module, grade, and announcement activity; coupled with a higher final grade.	0.81 ± .21SD N = 674 (11.5%)
2	Grade-Active Low Performer (GALP)	Yes	High-grade page access with low discussion, module, and announcement activity; coupled with a lower final grade.	0.77 ± .22SD N = 909 (15.5%)
3	Grade-Active Good Performer (GAGP)	Marginal	Higher module activity with a lower grade, discussion, and announcement activity; coupled with a marginally higher grade.	0.79 ± .23SD N = 904 (15.4%)
4	Announcement-Active Low Performer (AALP)	Yes	Higher announcement activity with the lower module, grade, and announcement activity; coupled with the lowest grade.	0.75 ± 24SD N = 777 (13.3%)

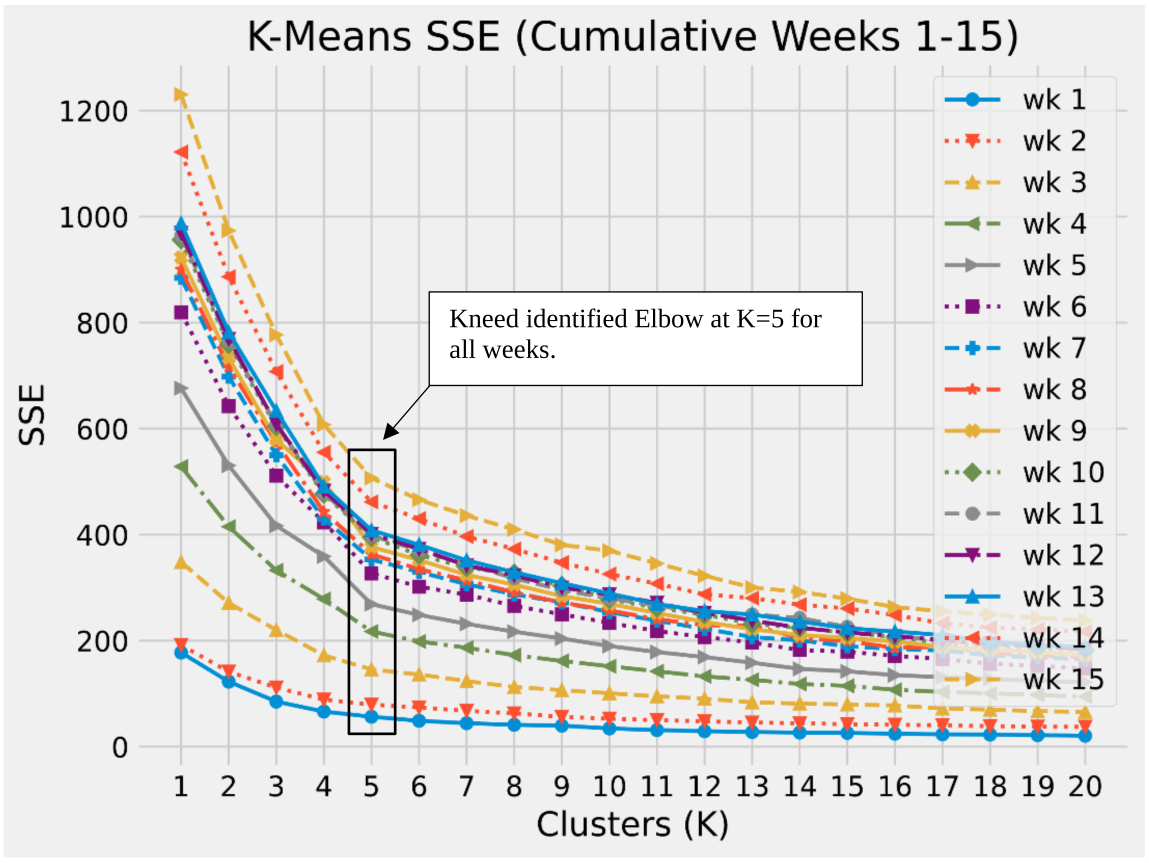
Prediction Performance Analysis (RQ4)

Multi-classifier Performance. This section summarizes the predictive analysis results, which compares the performance of a stack-based genetic programming XAI model to that of other AI/ML models based on LMS student activity as feature inputs for at-risk prediction. Supervised learning was performed by first training each model using stratified 10-fold cross-validation to predict at-risk labeled ground truths obtained from conducting k-Means classification for weekly cumulative datasets.

Using the Python kneed package, cluster size was automatically determined to be K=5 for each week's cumulative dataset, which was also visually verified using the elbow method (Figure 41). Observe how SSE grows over time due to the larger amount of data, which contributed to a higher amount of error aggregated. In addition, the elbow is more pronounced over time as SSE improvements are steeper (angle of the curve before the elbow point), which may be an indicator of better cluster isolation and predictability. This could be attributed to the greater amount of historical data accumulated each week.

Figure 41

K-Means SSE (Cumulative Weeks 1-15)



The predictive performance measures f1-Score (Figure 42), precision (Figure 43), and recall (Figure 44) were then computed for each AI/ML model for each week to show predictive performance trends over time. Observe that prediction performance for all models starts low initially, dips at the 2nd week, and then gradually peaks around the 4th week. It is important to note that the stack-based genetic programming (GP) model was superior for the first 5 weeks, even outperforming the multi-layer perceptron (MLP), a deep learning model. It was also strange that during week 11, GP experienced a drastic dip, but then recovered shortly after. These findings highlight the benefit and power of GP as a potential model for early implementation, as it provides superior at-risk prediction in the early weeks in addition to having more transparency.

Figure 42

Predictive Performance Over Time (F1-Score)

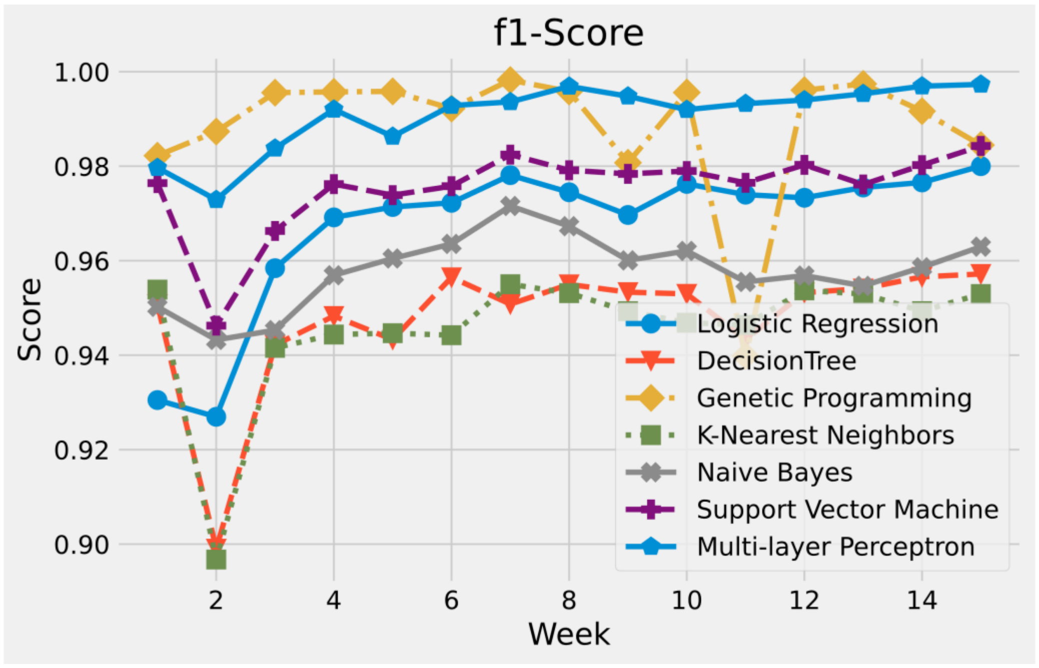


Figure 43

Predictive Performance Over Time (Precision)

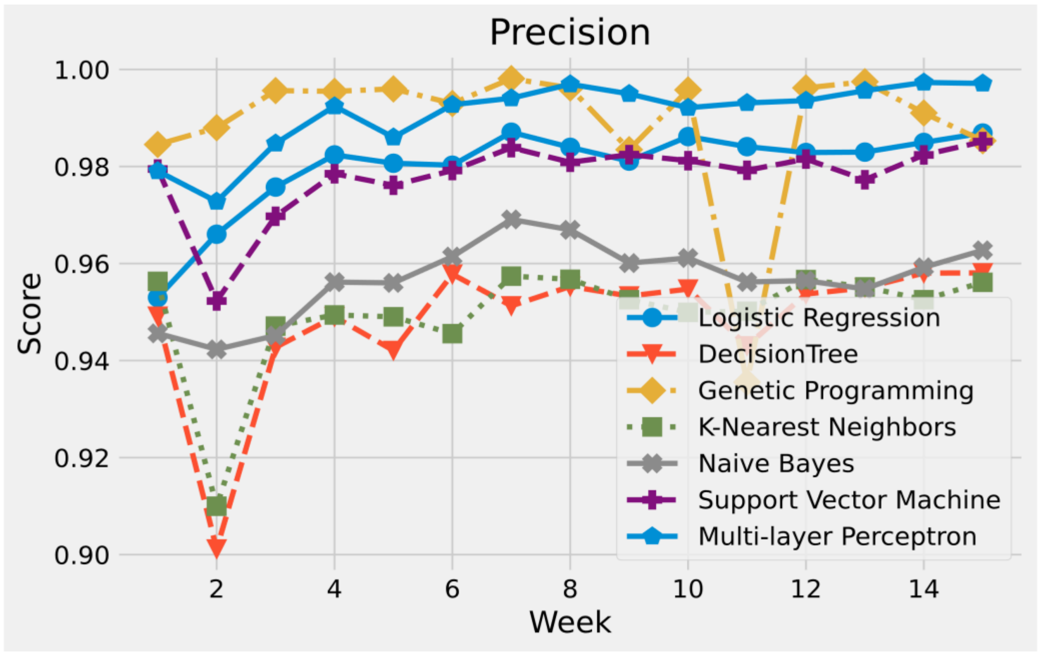


Figure 44

Predictive Performance Over Time (Recall)

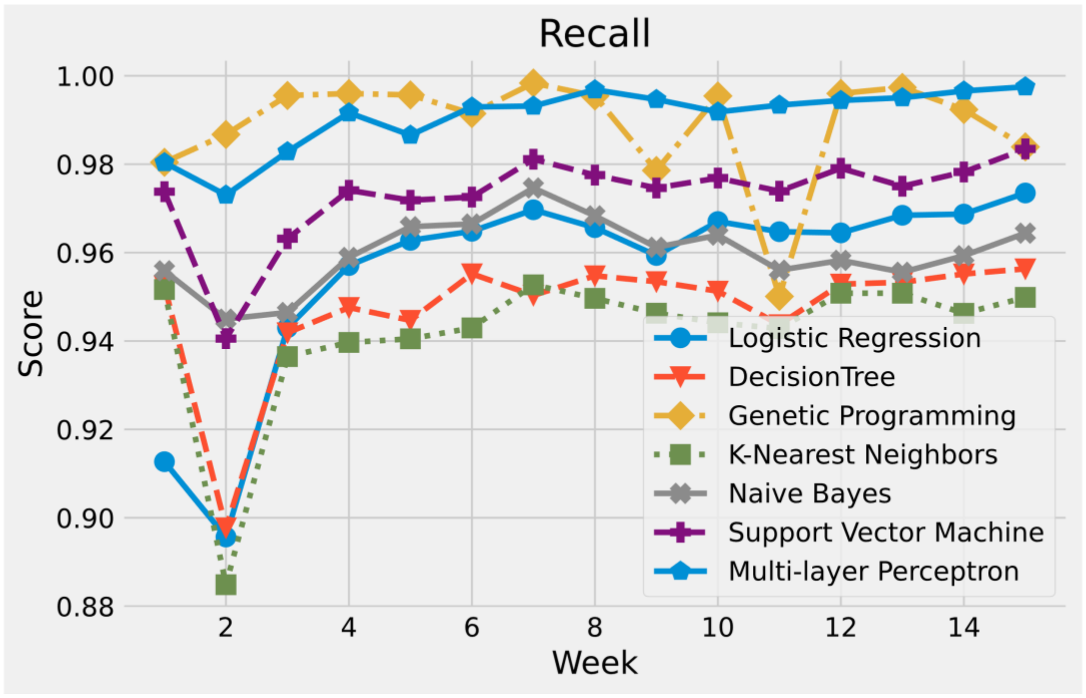
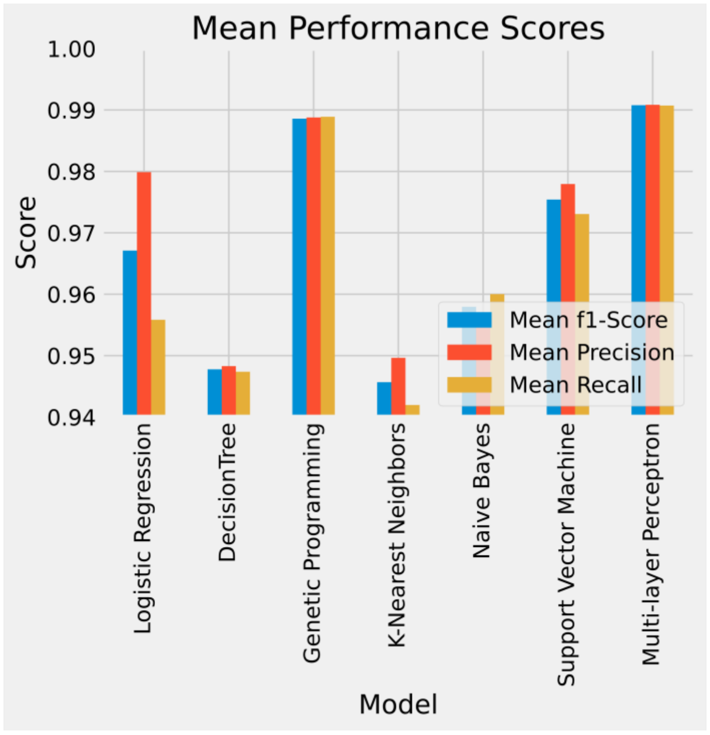


Figure 45 graphs the overall average scores (f1, precision, and recall) across all models for comparison. Observe that MLP is the highest performer, which is expected as it implements deep learning. But it is only marginally better than GP. Considering this, GP may be preferred as it offers additional transparency and explainability as a simpler and easier-to-interpret model. The next best performers include support vector machine (SVM) and logistic regression (LG). SVM appears more consistent among the three measures whereas LG demonstrated the highest precision but lower recall scores. Since recall is more important in education, as it measures the ability to detect a true positive (e.g., true at-risk students), SVM would be a preferred model. Following this reasoning, the next best performers are Naïve Bayes (NB), followed by DT, and finally K-Nearest Neighbors (KNN).

Figure 45

Mean Predictive Performance Scores



Binary Classifier Prediction Performance. This section summarizes the predictive analysis results, which compares the performance of a tree-based genetic programming XAI model to that of other AI/ML models based on LMS student activity as feature inputs for at-risk prediction. More specifically, the GPLearn binary classifier from the Scikit-Learn Python framework was used to explore the efficacy of tree-based GP methods. Unlike the previous section, this analysis focused on the binary classification that re-classified prior groups into at-risk or not-at-risk groups based on whether the group means of their final grades were above or below the overall weekly mean final grades of all groups.

Concerning predictive performance over time, GPLearn demonstrated very high accuracies in the first week (f1-score of 0.94) but gradually declined and dipped in the 10th through 12th week. The performance then improved after the 12th week (see Figure 46). This performance trend was very different from the stack-based method. Nevertheless, an 82-94% f1-score can be practical and effective in practice.

Figure 46

GPLearn Binary Classifier Performance Results Over Time

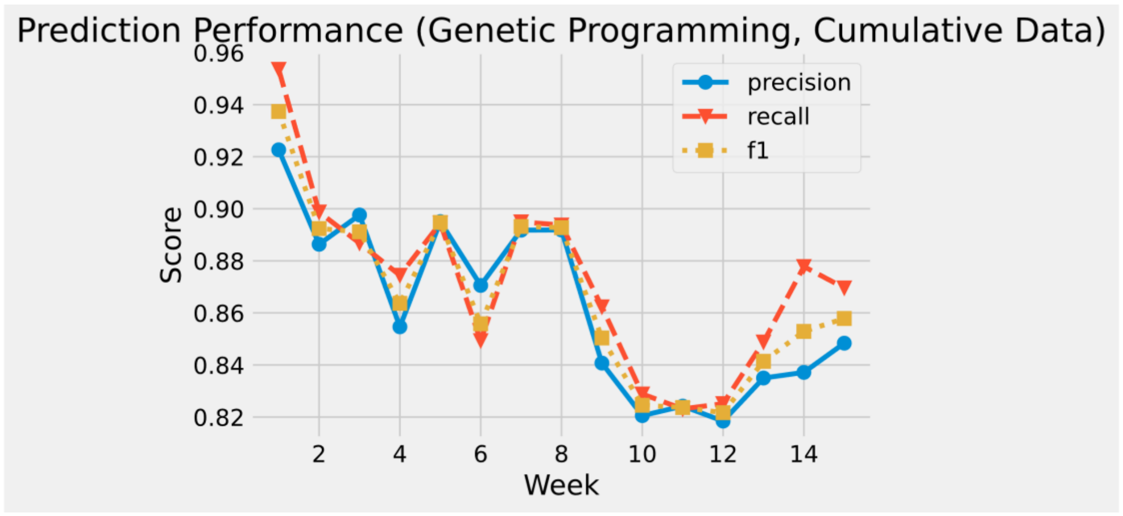
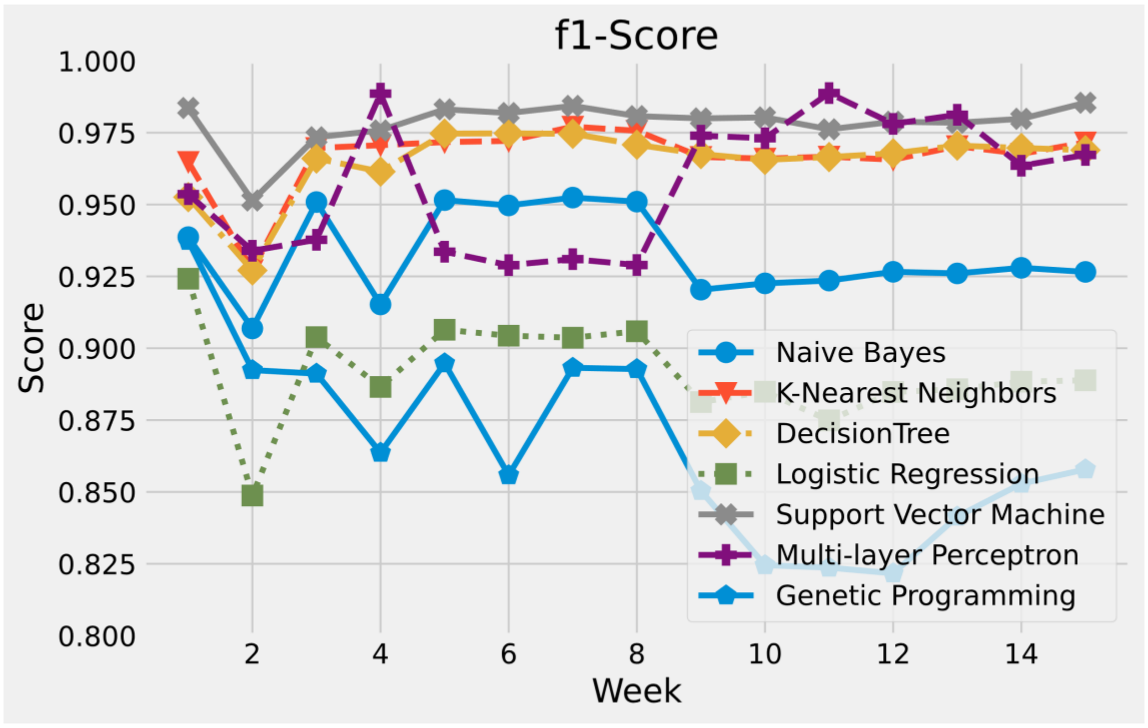


Figure 47 gives the predictive performance of GPLearn and other common AI/ML models for comparison. GPLearn underperformed in comparison to other models except for logistic regression in the first two weeks. Interestingly, the support vector machine, decision tree, and k-Nearest Neighbor were the superior performers. Although MLP had the highest peak scores, there were areas where it dipped in performance (1st-3rd and 5th-8th weeks). Observe that there is a slight dip for all models in the second week, which was a pattern observed from the previous multiclassification analysis results. Concerning overall average performance across all weeks, SVM was the top performer followed by DT, KNN, and MLP. Next was NB followed by LR. GPLearn was the lowest performer (slightly over 86% f1-score).

Figure 47

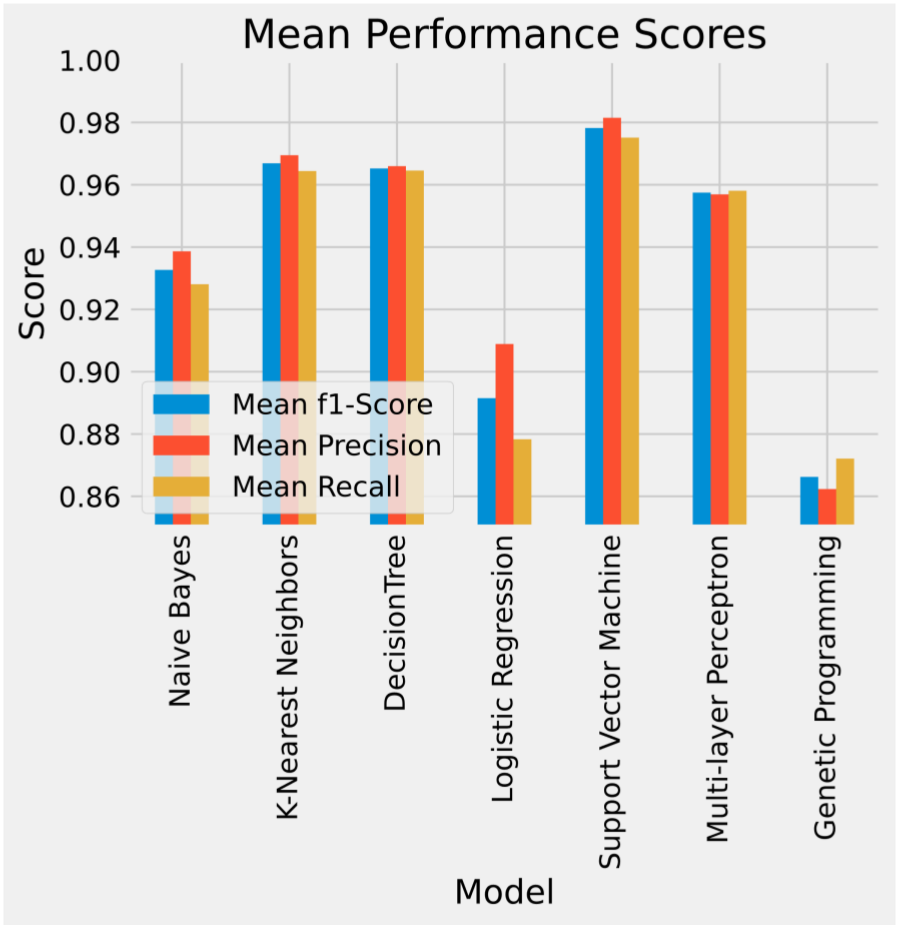
Binary Classifier f1-Scores



Overall, GPLearn underperformed when compared to other models (see Figure 48). Interestingly, the support vector machine, decision tree, and k-Nearest Neighbor were the superior performers. Although MLP had the highest peak scores, there were areas where it dipped in performance (1st-3rd and 5th-8th weeks). These results show that genetic programming may have limited performance as a binary classifier. However, the explainability that GPLearn offers may be worth the performance trade-off. The explainability of GPLearn is further discussed in the next section.

Figure 48

Overall Binary Classifier Performance Results



Explainability Analysis (RQ4)

This section presents an explainability comparative analysis of three XAI models, namely, the stack-based genetic programming classifier, tree-based genetic programming (GP) binary classifier, and decision tree (DT) classifier. These models are considered explainable as they contain visual or rule-based representations of the underlying logic to describe the reasoning behind their predictions. Unlike deep learning methods, which contain a high number of connected and interacting components (known as neurons), explainable models are much simpler and easier to interpret.

Explainability of Stack-Based GP. This section explores the explainability of the Ellyn-GP M4GP algorithm, a stack-based lexicase genetic programming multi-classifier written by La Cava et al. (2017; 2019). The results in the previous section showed that the M4GP model demonstrated exceptional predictive performance comparable to other deep learning leading models such as multi-layer perceptron. In La Cava et al.'s (2019) work, M4GP demonstrated competitive performances against other state-of-the-art models across a broad array of problem sets. But the question remains how explainable are stack-based models? The answer is that they are not directly interpretable due to the complex transformations involved.

Table 17 presents the top 10 models obtained from training M4GP on class labels obtained from the prior K-Means Clustering Classification analysis performed as part of this study. Observe that each of the model solutions appears simple, with only a few first-order terms (7 items for the smallest and 20 for the largest) in the stack (represented by elements within the brackets). Surprisingly, such a model produced accuracies on par with the multi-layer perception with f1-scores above 98%. But understanding the model is not as straightforward since the results are presented in an unfamiliar dimensional space.

Table 17

Stack-Based GP (M4GP) Solutions

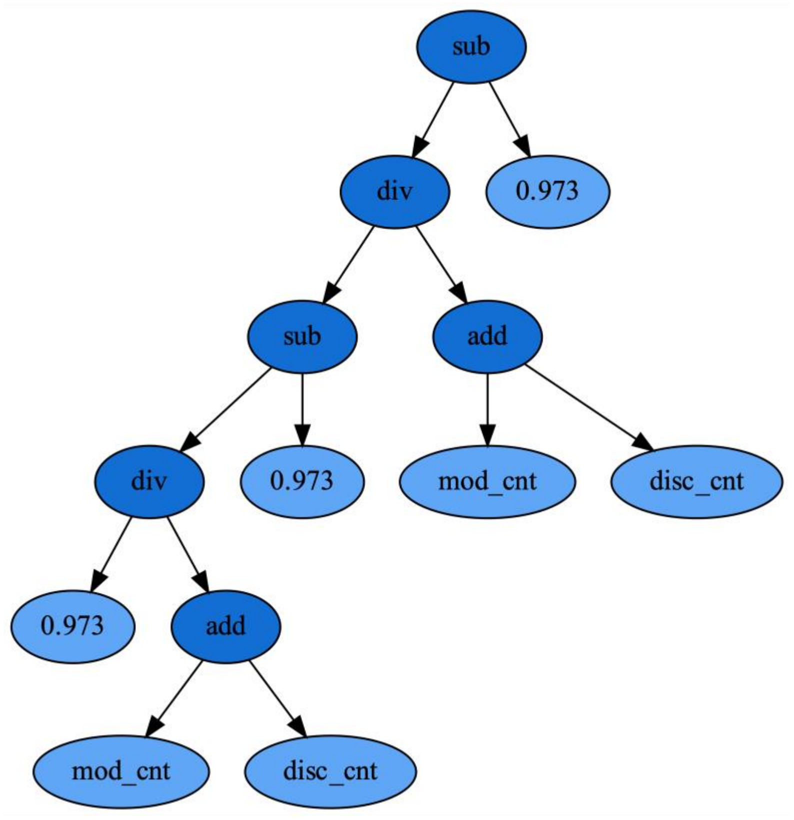
Solutions	**f1-score**
[x_1, 0.509, x_2, x_3, x_0, 0.003, 0.129]	0.9898
[x_1, -0.042, 0.186, 0.014, x_2, -0.176, x_0, x_3]	1.
[x_1, x_2, 0.505, 0.001, 0.044, -0.076, -0.205, 0.128, x_0, x_3, 1.009, 0.0, -0.0]	0.9942
[-0.0, -0.537, x_3, 0.241, 0.001, x_0, x_1, -0.079, -0.132, x_2, -0.007]	1.
[x_2, x_1, 0.021, -0.009, 0.245, 0.006, x_0, x_3, 0.141, 0.003, -0.002]	0.9941
[-0.173, 0.047, -0.203, 0.48, x_1, 0.158, 0.131, x_0, 0.167, x_3, x_2, 0.037]	0.9932
[x_3, -0.005, 0.154, -0.003, -0.026, x_1, -0.11, 0.002, x_2, x_0]	1.
[0.053, 0.0, -0.027, x_3, 0.204, x_2, 0.023, -0.002, 0.108, 0.324, 0.008, 0.013, -0.869, 0.025, 0.025, 0.392, 0.184, x_1, x_0]	1.
[0.421, 0.504, -0.0, x_3, 0.657, x_2, -0.159, 0.558, 0.057, -0.416, -0.969, 0.232, -0.297, -0.012, -0.038, -0.008, -0.115, x_1, x_0]	1.
[0.952, 0.16, -0.001, x_3, -0.015, x_2, -0.025, -0.042, 0.164, -0.341, -0.038, x_0, -0.241, -0.026, 0.007, 0.633, 0.037, x_1, -0.21]	0.9872

According to La Cava et al. (2019), the M4GP algorithm “optimizes models by first performing a transformation of the feature space into a new space of potentially different dimensionality, and then performing classification using a distance function in the transformed space” (p. 260). The model implements a distance-based nearest centroid classifier similar to k-Means clustering where samples are classified based on the minimum distance to cluster centroids. So how can the solutions be interpreted to provide meaningful insight? Unlike K-Means Clustering, which performs clustering on the feature space, M4GP clustering is performed on new n-dimensional space with different p-clustered labels. Unfortunately, there is little to interpret as the space is unfamiliar and only serves the purpose of improving classification accuracy. However, one can observe that the solutions contain variables from the feature space (e.g., *x_n* terms). As such, it can be inferred that the features themselves contribute to the classification if they appear in the solution set.

Explainability of a Tree-Based Binary Genetic Programming Classifier. Another alternative to stack-based genetic programming models is binary tree-based models, which can provide simpler and more interpretable visual representations. For example, Stephen’s (2016) GPLearn model is a binary tree-based classifier, which can leverage Scikit-Learn’s graphviz package to output a syntax tree visual model. To explore this model, the classification analysis was repeated, and at-risk groups were categorized as at-risk, or not-at-risk students based on whether their mean values were above or below the weekly group means of final grades. After training and predicting using the 10-fold cross-validation method, GPLearn demonstrated a precision, recall, and f1-score of r.85, .86, and .85 respectively. Figure 49 gives the associated tree model output.

Figure 49

Tree-Based Genetic Programming Binary Classifier



Observed that the model is easier to interpret than the stack-based M4GP as the representations are directly associated with the feature space. According to Ferreira et al. (2020), the presence or absence of features in this model can be interpreted as an indicator of importance. As such, major influential features can be immediately identified as discussion (disc_cnt) and module (mod_cnt) activity. This is consistent with the previous correlation analysis where discussion and module activity were identified as having the most significant associations to final grades. Furthermore, the model can be interpreted as a function of discussion and module activity and rewritten as equation (7).

As an equation, the binary classification can be explained based on an analysis of module and discussion ranges and their impact on the output results.

$$f(disc_{cnt}, mod_{cnt}) = \frac{\left(\frac{0.973}{(mod_{cnt} + disc_{cnt})} - 0.973\right)}{(mod_{cnt} + disc_{cnt})} - 0.973 \qquad \textbf{(7)}$$

Upon initial inspection, equation (7) may appear arbitrary. However, this is expected as the objective of genetic programming is to develop solutions using randomly selected programs, which may be novel and in this case are equations built from operators (+, -, /) and operands (a rational number between 0 and 1). The resulting solution is just one of many in the solution space, which may contain an infinite number of possibilities. But how exactly does this equation translate to the at-risk state? According to the GPLearn documentation, the output of the equation is transformed into probabilities of each class using a sigmoid function (see equation (8) and Figure 50)

The sigmoid function shows that negative or positive values of the output results can be used to classify binary predictions based on their probability (e.g., $p < .5$ can be classified as one group while $p \geq .5$ can be classified as another) (Stephens, 2016). Therefore, equation (7) can be analyzed to identify what values of discussion and module activity result in an overall positive or negative value. Based on this, the following observations can be made:

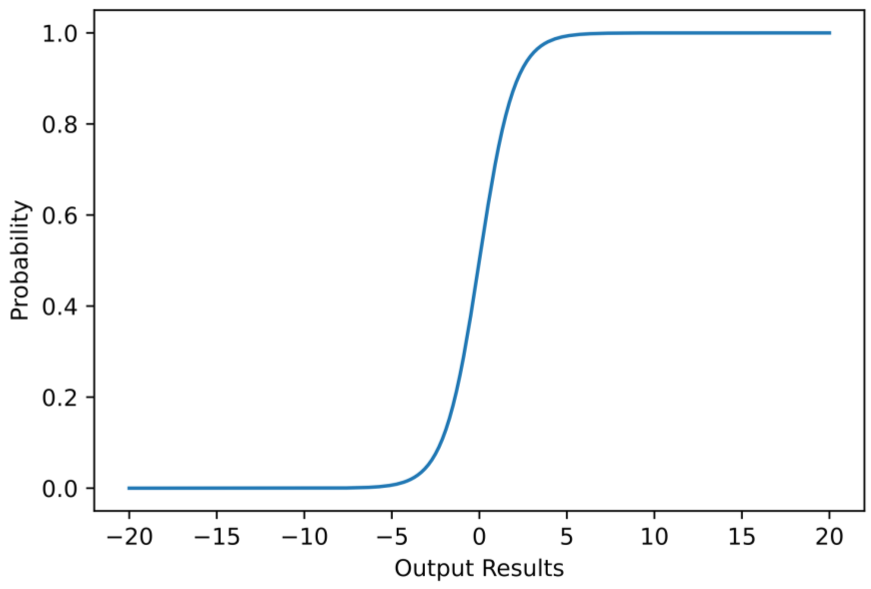
- The terms “mod_cnt + disc_cnt” in the solution equation can range from 0 to 2 as each variable is a normalized frequency ranging from 0 to 1. When both are close to zero, the equation results in a positive value and can be classified as at-risk (i.e., students with both low discussion and low module activity). This is consistent with prior K-Means Clustering Classification analysis where clusters 0, 2, and 4 were lower performing groups that also had low values of discussion and module activity (see Figure 40).

- When one variable is zero and the other is one, the results will still end up as a negative number (the numerator will be close to zero and thus reduce the left operand of the minus sign) and thus classify the student as not-at-risk (i.e., when either discussion or module activity is high). This is consistent with clusters 1 and 3 from the K-Means Clustering classification results (Figure 40) in that discussion and module activity are influential factors alone for higher performing groups.
- When both equal one, the resulting value is a negative number, which can be classified as not-at-risk (i.e., students with both high discussion and high module activity). This is logical as either factor would contribute to higher performance. This is an additional characteristic that was not found in prior K-means Clustering Classification analysis.
- Thus, the GPLearn binary classifier model provides mathematical explainability that is consistent with prior K-Means Clustering Classification as well as more insight into other high-performing groups.

$$Probability = Sigmoid(f) = 1/(1 + e^{-f}) \qquad \textbf{(8)}$$

Figure 50

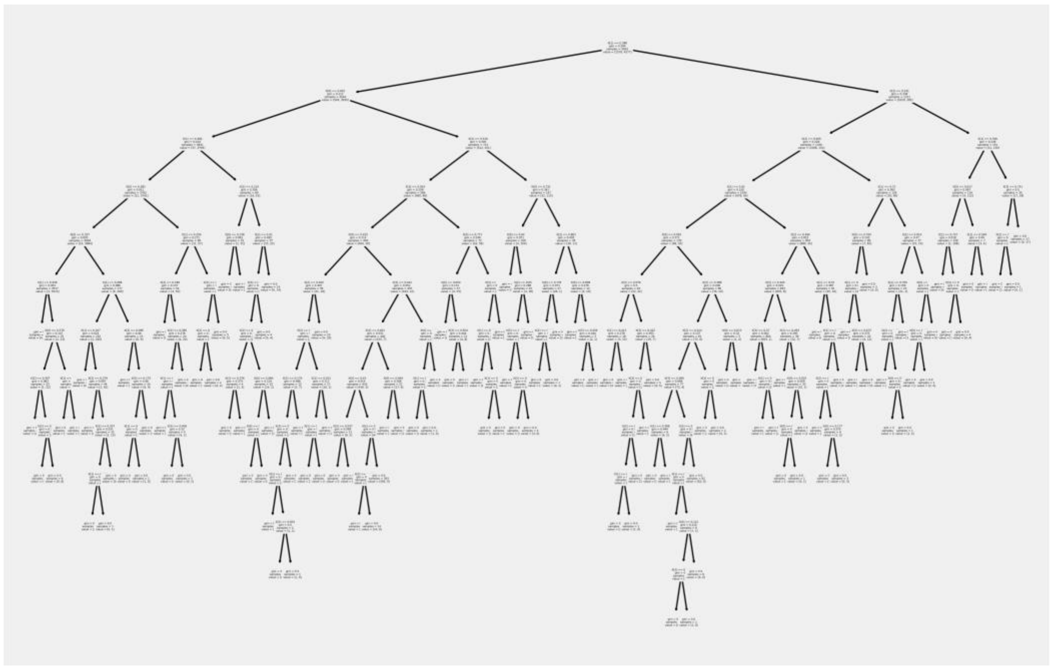
Sigmoid Function



Explainability of a Decision Tree (Binary Classifier). To provide a fair comparison, the same classification process was performed using the binary-class decision tree model, which is shown in Figure 51. For prediction performance, the decision tree outperformed the GPLearn binary classifier with precision, recall, and f1-score of .97 (12% gain in performance). However, the higher accuracy came with the trade-off of explainability as the resulting model has too many components, which renders it too complex for practical interpretations.

Figure 51

Binary-Class Decision Tree



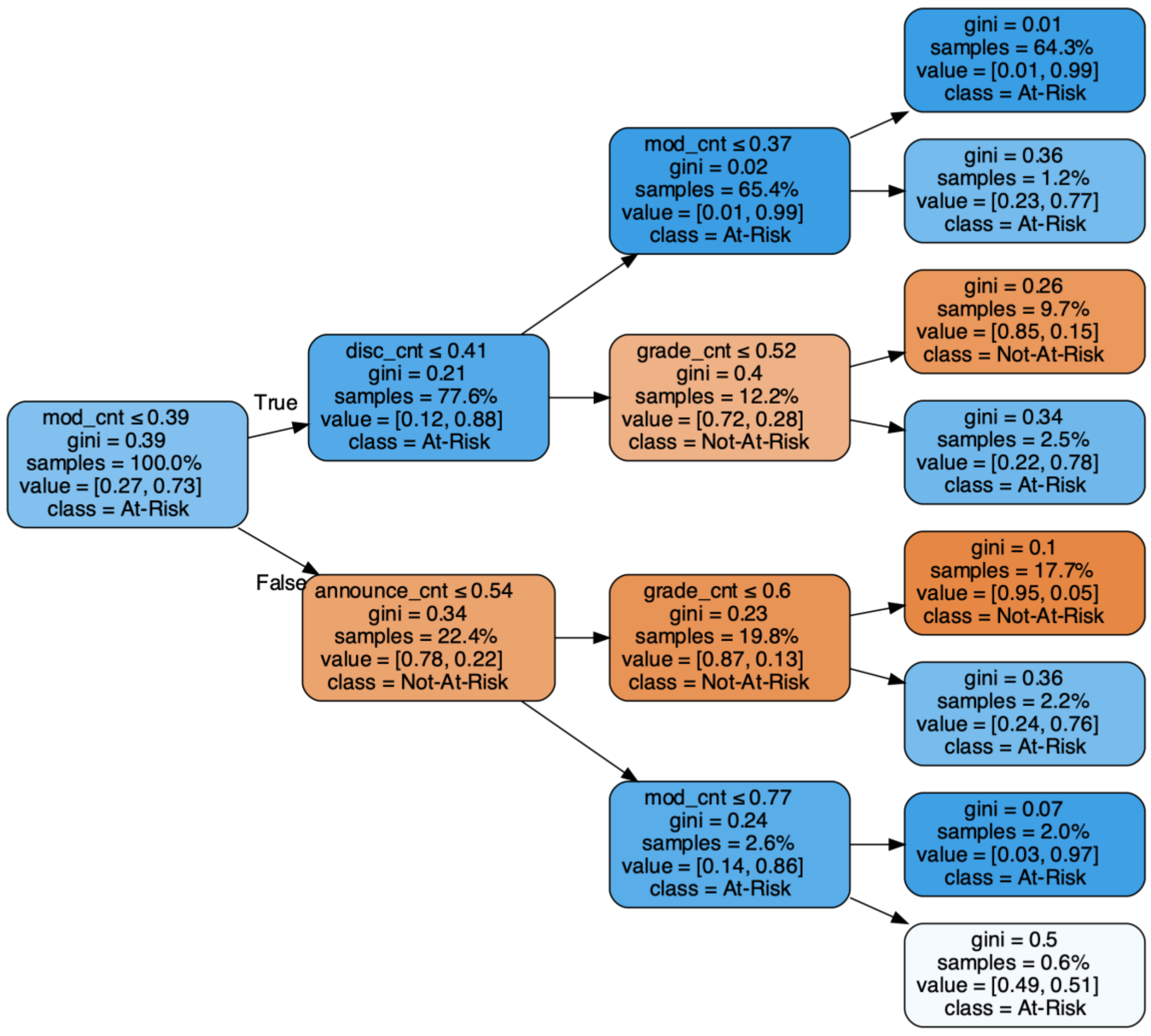
To improve the explainability of the decision tree, the maximum tree depth was limited to three for the same analysis. In addition, Scikit-Learn's graphviz (2022) package was used to format the nodes and provide additional labeling to improve explainability and readability. Figure 52 shows the updated decision tree, which shows that limiting the maximum tree depth has significantly improved the explainability of the decision tree. In addition, there was only a slight reduction in performance as the model had a precision, recall, and f1-score of .93, .94, and .94 respectively (only a 3% reduction). Note that each node provides several explainable information:

- The Boolean expressions provide a characterization of the child nodes. A complete characterization of at-risk activity behaviors can be described by the associated expressions along the path traversed to the leaf node, which identifies the final class predicted for a particular sample.

- The ratio of total samples that are associated with each node provides insight into the number of at-risk students.
- The associated ratios of samples partitioned into at-risk or not-at-risk groups (i.e., the first and second entry of the value array) provides insight into the number of correct or incorrect classifications, which is selected based on the group with the highest proportion.
- The *Gini* index provides a measure of classification confidence and quality. Lower values of Gini indicate more homogeneity in the classified distribution (i.e., fewer classification errors).
- Class labels identify the classified group based on the value array where the class with the highest ratio is selected.

Figure 52

Binary-Class Decision Tree (Max Depth of 3)



By analyzing the interpretable metrics in Figure 52, the following observations and explanations can be made (refer to Figure 40 for references to prior clustering analysis):

- There are eight total groups (two not-at-risk and six at-risk types) represented by the leaf nodes and the Boolean expressions associated with the path traversed.

- The first not-at-risk group (9.7% of total samples) is characterized by lower module, higher discussion, and lower grade activity. 85% of the samples were accurately classified. This is consistent with cluster 1 in prior clustering analysis.
- The second not-at-risk group (17.7% of total samples) is associated with a higher module, lower announcement, and lower grade activity. 95% of the samples were correctly classified in this group. This group is consistent with cluster 1 in prior clustering analysis.
- The first at-risk group (64.3% of total samples) is characterized by lower module and lower discussion activity. 99% of the samples were correctly classified. When mod_cnt is not less than .37 (the second at-risk group), there appears to be a decreased probability (77%) that the students are at-risk. This first group is similar to clusters 0, 2, and 4 of the prior cluster analysis. The second at-risk group is a new finding.
- The third at-risk group (2.5% of total samples) is associated with a lower module, higher discussion, and higher grade activity. 78% of samples were correctly classified. This at-risk group is a new finding.
- The fourth at-risk group (2.2% of the total samples) is associated with a higher module, lower announcement, and higher grade. 76 percent were correctly classified. This group is a new finding.
- The fifth at-risk group (2% of total samples) is associated with a higher module, higher announcement, and lower module activity. 97% of samples were correctly classified. This group is a new finding.

- Finally, the sixth at-risk group (0.6% of total samples) is associated with the higher module and higher announcement activity. Only 51% were correctly classified. This group is a new finding. However, the accuracy is too low so this group should be excluded from use.

As demonstrated, the decision tree was able to provide significant insight into various types of at-risk and not-at-risk groups. Furthermore, the value of correctly classified samples provides an additional measure of confidence that can be used in practice to assess model trustworthiness. The percent of the total also helps identify the effect size of the at-risk problems. In this example, it is clear that the first at-risk group represents the bulk of at-risk students as it comprises over two/thirds of the total samples. In summary, the decision tree binary classifier, when limited to a max depth of three, resulted in superior explainability compared to both the GPLearn binary tree classifier and M4GP stack-based classifier. Concerning prediction performance, it outperformed GPLearn and was slightly less accurate than M4GP. Considering the enhanced explainability, the decision tree is the preferred choice for practical applications. This is primarily attributed to the explainable information offered by Scikit-Learn’s graphviz package.

Explainability of Decision Trees (Multi-Classifier). In the last section, an analysis was performed to assess the explainability of the decision tree binary classifier. Similarly, this section explores the explainability of the multi-class decision tree. Figure 53 gives the visual representation of the decision tree trained and fitted to classification labels obtained from prior K-Means classification analysis. Similar to the binary decision tree, using default settings resulted in an overly complex tree that is arguably complex and not interpretable.

Figure 53

Multi-Class Decision Tree Classifier

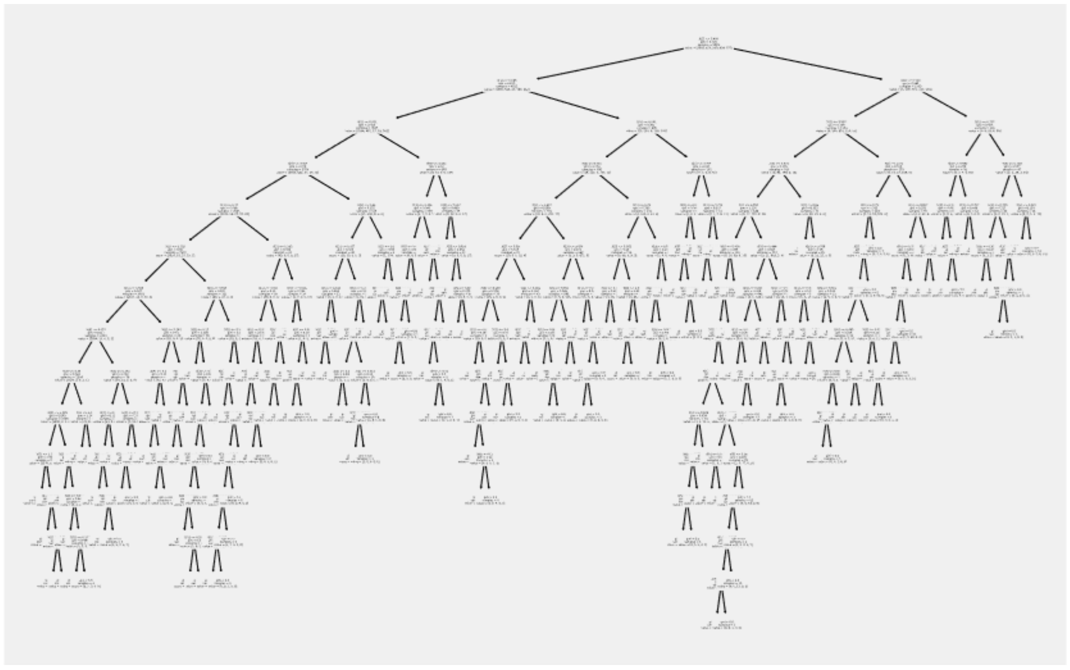
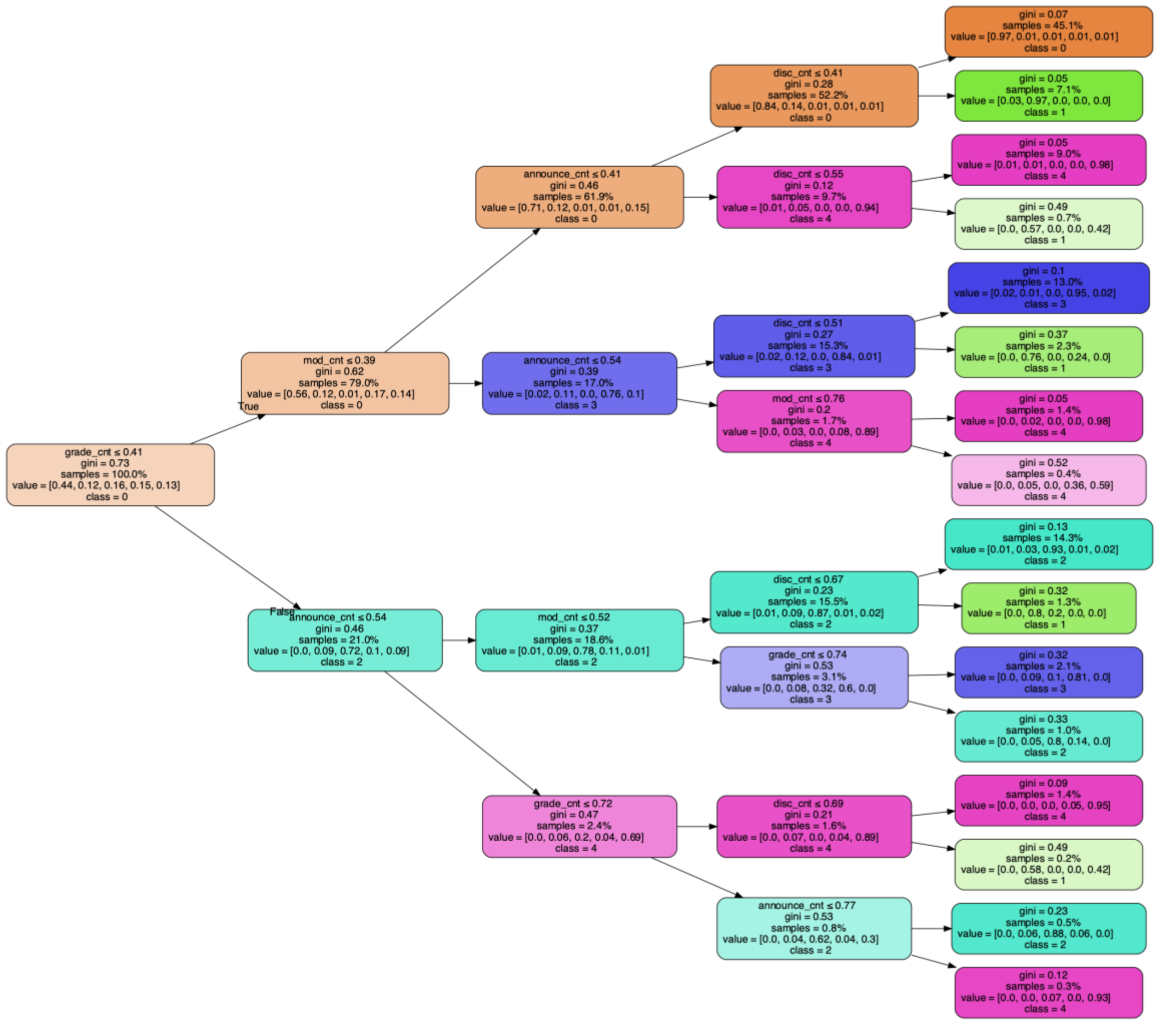


Figure 54 gives an updated decision tree when max depth is set to four, which was identified optimal level to achieve a balance between explainability and prediction performance (f1-score of 91.8%). In contrast, a max depth of 3 resulted in a simpler but less accurate model (f1-score of 71.6%) while a max depth of 5 resulted in an overly complex model with only marginal improvements in accuracy (f1-score of 92.8%). Explainability was also improved by color coding of the nodes to indicate class numbers with an intensity of color indicating the classification accuracy. Compared to GPLearn, the multi-class decision tree is more explainable as the cluster class is explicitly identified. The class provides a one-to-one mapping to labels obtained from prior clustering analysis (see Figure 40). Compared to the prior decision trees, this model provides more insight into other influential features as well as several traversal variations.

Figure 54

Multi-Class Decision Tree Classifier (Max Depth = 3)



Although there is a larger number of items in the decision tree, the color coding and interpretable metrics help alleviate the cognitive load. Traversing any path is a simple exercise to characterize the associated groups, which are defined at the leaf level. In addition, the Gini index and color intensity also help provide a measure of confidence or trustworthiness where a lower index associates with a deeper color to signify higher classification accuracy at the associate node. Using this model, at-risk students (identified as clusters 0, 2, and 4 in prior classification analysis) with high predictive performance (greater than 90% accurate). For example, the largest at-risk group was classified as cluster 0, which comprised 45.1% of the samples and is associated with levels of activities across all features. Furthermore, prediction accuracy was high (.97) for this group. Thus, the decision trees, like K-Means clustering, can provide more fine-grain explanations by identifying specific levels of features to classify at-risk students. Specific interventions can be planned to improve student interactions in features with lower activities.

Implications of Explainable Results. This section discusses the explainability analysis results and their applications to educational practice. Topics discussed include identifying key explanations about at-risk students activity behaviors provided by the XAI models as well as benefits that can be leveraged. In addition, recommendations are provided on how these explanations can be used by instructors, students, and/or university administrators.

What do the models explain? The explainability analysis compared and contrasted the interpretability of stack-based GP, tree-based GP, and Decision Trees models trained from clustered labels of prior classification analysis. The objective was to determine how the models can explain student at-risk behaviors based on logged LMS activity (i.e., access frequency to discussion forums, module content, grade pages, and announcement pages). The results indicate that different models had different levels of explainability (i.e., complexity and ease of human understanding).

First, the analysis found that the stack-based GP exhibited the highest predictive performance but the poorest explainability. Although stack-based GP produced simple stack-based solutions, interpretability was not obvious due to the complexity of dimensional space transformations involved. The stack-based algorithm, known as M4GP, first transforms the feature space to a new space of different dimensionality before applying classification based on nearest centroid and Mahalanobis distance similar to k-Means Clustering (La Cava et al., 2019). As a result, the model produced is a stack comprised of synthetic features mapped to an unfamiliar dimensional space. Although the solutions are simple and contain features as variable elements, the mapping back to the original feature space is not obvious nor trivial. What can be inferred is that the synthetic features, which may contain activity features, contribute to new synthetic characteristics, that must be transformed back for meaningful insight. Unfortunately, there is a lack of literature research on explaining or describing stack-based genetic programming. Most studies identified stack-based GP as a simple and interpretable model but did not provide practical explanations, examples, or visualizations (La Cava et al., 2017, 2019; Perkis, 1994; Stoffel & Spector, 1996). Therefore, the explainability of stack-based GP is assessed as poor.

On the other hand, the GPLearn binary tree classifier provided better explanations as the output model was a single binary tree that could be represented as a first-order algebraic equation. The model was simply an equation that contained the features (discussion and module activity) as variables forming a mathematical relationship that could be analyzed and interpreted. Whether a sample was labeled at-risk or not depends on the outputs of equation (7), where low values of discussion and module activity would result in mappings to an at-risk group via a sigmoid function. Conversely, high values of discussion and module activity would result in a not-at-risk mapping. However, there was still limited practical explainability as there was little insight for mapping results back to the original at-risk groups discovered during prior classification analysis.

Surprisingly, decision trees were found to have the best explainable results. Initially, decision tree models developed were accurate but overly complex as there were too many nodes for practical interpretation. It was later discovered that by limiting a max depth, the explainability of the tree could be improved. Optimal values were found for both binary and multi-class decision trees. Furthermore, the Scikit-Learn graphviz package provided additional formatting capabilities along with interpretable metrics such as color coding by class type, a Gini and probability metric to assess node-level classification accuracy, and sample partition. The color intensity of each node also maps to the classification accuracy demonstrated by each node.

As a result, various types of behaviors could be explained by traversing the tree from the root node to the leaf node and examining the Boolean expression results to characterize at-risk behaviors. Leaf nodes represent the final classified groups where the traversed Boolean expressions describe feature traits (i.e., whether particular features were above or below a threshold). Hence, formative feedback could be provided to instructors and students to explain why they were identified as at-risk. The model not only demonstrated consistency with prior K-Means Classification clusters but also revealed new groups and insights that could be accurately predicted. The Gini metric, color intensity, and partition probability also helped to provide a measure for confidence and trustworthiness of predictions. Using these metrics, one could ignore inaccurate nodes of the model while making effective use of the more accurate predictions.

What are key XAI benefits found? So, what are the real benefits that XAI can offer? In this study, XAI was implemented as a post-hoc analysis of existing data in an LMS system. The research provided a proof of concept to explore and investigate what could be possible. In practice, these models could be implemented in a real-time environment to provide automated analysis, alerting, and intervention guidance to students, instructors, and administrators with live actionable data. This saves instructors and institutions time and effort from manual analysis and unguided interventions. As an early warning system, XAI may be able to detect early signs of at-risk students that were not possible with late periodic assessments. This research found that at-risk behaviors can be predicted with very high accuracy in the early weeks of instruction, which is a critical time for interventions.

This research also demonstrated how different XAI models, such as stack-based GP, tree-based GP, and decision trees exhibited different levels of explainability. Unfortunately, GP explainability is still limiting due to the lack of literature and supporting visualization technologies. In contrast, decision trees had extensive visualization support from the Scikit-Learn python application programming interface that greatly enhanced explainability. This research discovered that default settings of decision trees could result in overly complex models. With some tuning of max depth functions, highly accurate and explainable models could be achieved. The results of the explainability analysis show that decision trees are superior in explainability. When used to train on prior labeling, decision trees can be valuable post-hoc explainable techniques for describing complex models. This study demonstrated how XAI models found consistency with prior K-Means analysis.

Finally, another key benefit that XAI offers is the ability to improve the trust, adoption, and use of emerging AI (Adadi & Berrada, 2018; Crowe et al., 2017; Sun & Medaglia, 2019). Without proper reasoning behind at-risk predictions, instructors may hesitate to act on alerts or indications. Without information feedback, they are left to investigate the underlying reasons, which can take extensive time and effort assuming the predictions are accurate. Additional explanations can provide confidence for decision-making. For example, the decision tree’s explainable metrics such as Gini index, probability proportions, and color-coded intensities help provide confidence metrics for different parts of the model, which are easy to interpret and use for practical applications. This information, when provided to students and teachers, can provide context and awareness that is critical for decision-making (Alonso & Casalino, 2019). At-risk indications with lower accuracy (e.g., high Gini value or low classification accuracy) may be ignored while those with higher accuracy could be addressed.

Key Takeaways for Educational Stakeholders. From the K-Means Clustering analysis, at-risk students were found to exhibit lower levels of LMS interaction across all features, such as discussions, modules, grades, and announcements. Of those, discussions and module activities were found to have the highest correlations to final grades. This was later confirmed by the explainability analysis where consistent findings were found by the XAI models’ at-risk explanations. These results are further supported by the developed Theory-Guided Feature Selection (TGFS) model, which identifies discussions and modules as important factors related to Social Learning and Fink’s Significant

Learning foundational knowledge concepts. Finally, these findings were also consistent with the XAI and EDM literature review, which identified the significance of student LMS activity in predicting and explaining student outcomes (Chamizo-Gonzalez et al., 2015; Xing et al., 2015). This triangulation of results provides robust support that: (1) student LMS activity can impact student outcomes and (2) student activity can be used to predict and explain at-risk behaviors.

These findings can prove useful for instructors, students, and university administrators. For instance, instructors can leverage this information to place additional focus on tracking and engaging students and their discussion forums and module interaction early in the semester. Those with a lack of interaction across all LMS functions such as discussion forums, module content, grade pages, and announcement are key indicators of at-risk students. The result of the study shows that even an increase in any single LMS interaction can promote an increase in learning outcomes. The institution can also create programs to promote early student interaction by providing preliminary training, resources, or material to raise awareness of the importance of early interaction, collaboration, and participation. This help prepare learners by identifying the importance of participation, which has been shown to promote their learning success.

Institution administrators can implement supporting technologies that facilitate and promote student discussion engagement and content participation within the LMS environment. Examples include better discussion forums with collaborative real-time capabilities. New technologies may increase interest and student engagement. In addition, administrators could work with institution instructional designers to implement an early warning system that implements XAI models to provide at-risk indications with explainable feedback, such as reasons behind those predictions as well as a confidence measure of the associate predictions (i.e., how likely the predictions are true). This allows for both instructors and students to use those predictions based on their judgment and decision-making processes. Over time, this will help not only improve student outcomes, but also trust and adoption of XAI systems for educational use.

Chapter Summary

This chapter presented the results of the XAI EDM study, which included a description of the EDM process, theory-guided feature selection, and various analyses performed to answer the four key research questions.

RQ1 Findings

After excluding outliers and correcting for traditional learning biases, final grades were found to have a modest (*r = .22 and .21 respectively)* positive correlation to discussion**, module**, and grade** activity and a weak (*r = .09)* correlation to announcement** page access frequency (** = p < .001). All other feature correlations were statistically insignificant and were excluded for subsequent analysis.

Thus, for online learners, teachers and instructors can promote knowledge and social learning interactions by tracking, monitoring, and soliciting student activity associated with discussions and module access. Furthermore, instructors can promote student engagement by providing more frequent assessments and notifications. By playing an active role in instruction, teachers can improve student activity, which was shown to associate with higher learning outcomes.

RQ2 Findings

There were 6 identifiable groups with differing levels of at-risk patterns. The at-risk group (ILP) had a mean final grade of 0.37 (±.18SD) and exhibited very low activity for all features. The mean final grade was significantly lower than other groups, *H(4) = -954 to -1120, p < .001, and r = -.72 to -.83.* Four groups represented Good Performers and were associated with higher levels of individual activities (AAGP, GAGP, DAGP, and MAGP). There was one confounding group (IHP) characterized as having a high final grade with low activity across all features, which was attributed to a large proportion of the sample corresponding to students who enrolled in traditional learning. This group was excluded to remove traditional course biases where students were not required to interact with the LMS environment. Removing this group also improved the correlation results identified in RQ1.

Based on RQ2 findings, at-risk learners are attributed to those who lack student activity across all features. For online teachers, the lack of engagement and participation should be an early predictor of success, especially in the early weeks when interventions could be the most effective. Interventions employed can include frequent communication and interaction with underactive students.

RQ3 Findings

There were 5 predictable at-risk groups. Three of the groups were low performers (ILP, GALP, and AALP) who were associated with the following conditions: all lower activities; all low activities except for grade access; all low activities except for announcement activity. The remaining two groups (DAHP and GAGP) were characterized as high and good performers who were associated with higher discussion and higher module activities. These results were consistent with the results for RQ1. Prediction performance for all models was high and increased with time. GP and MLP were the most accurate models.

Based on RQ3 findings, discussion and module access were associated with higher learning outcomes whereas grade and announcement activities were associated with lower learning outcomes. This could be attributed to the fact that there is a large ratio of traditional in-classroom students that were not excluded. Unfortunately, classification without final grade as a feature did not reveal the confounding group (higher performer with lower activity), which needed to be identified for exclusion.

Similar to RQ3 findings, lower performers are associated with lower activities across all features and higher performers were associated with higher levels of discussion and module access. As such, online teachers can aim to monitor, track, and improve the discussion and module activity to promote success. In addition, teachers should spend more time engaging students who have little to no interaction across all features.

RQ4 Findings

The prediction performance evaluation comparing various AI/ML models found that stack-based GP performed comparably to MLP and superior to other models. Although stack-based GP is simple, mapping its components back to the features is non-trivial and may be difficult to explain. Comparative explainability analysis between stack-based GP (M4GP), tree-based binary GP (GPLearn), and decision tree (binary and multi-classifier) revealed that stack-based GP had the poorest explainability but best performance. Decision trees with default settings result in accurate but highly complex and non-explainable models. When modified to limit max depth, Decision trees retained high accuracy with a much simpler and more explainable model. Leveraging Scikit-Learn's graphviz package for decision tree resulted in superior explainability. Future work is needed to prove the explainability of stack-based GP.

CHAPTER V

Validity and Reliability

Introduction

This chapter provides a discussion on the validity and reliability of the current research. Validity is concerned with accuracy and credibility whereas reliability is concerned with repeatability and consistency of the research (L. Cohen et al., 2018). Furthermore, Cohen et al. (2018) emphasized that “reliability is a necessary but insufficient condition for validity in research” (p. 245). To ensure quality and worthwhile research, both validity and reliability must be ensured as much as practically feasible within the constraints of time, technology, and resources.

How are validity and reliability addressed? For quantitative research, Cohen et al. (2018 cited Shadish et al. 2002) recommend addressing threats to internal validity, such as low statistical power, violation of assumptions, measurement error, limited data range, variation in treatment procedures, extraneous variables, variability in outcome measures, statistical error, and false assumption of causality. Concerning external validity, the research should address the generalizability of results based on the population representativeness of the data sample (L. Cohen et al., 2018, p. 254). Finally, reliability can be “achieved, in part, by a thorough literature review of the state of the field and how it has been researched to date” (L. Cohen et al., 2018, p. 181).

This research implements many of Cohen et al.’s (L. Cohen et al., 2018, p. 201) for improving the validity and reliability of the current research, which are summarized in the proceeding sections. In addition, limitations and associated mitigation strategies are identified concerning specific threats to internal validity.

Literature and Theoretical Supports

In sections *Statement of the Problem, Significance of Study,* and *Literature Research,* evidence was provided for the importance of explainable artificial intelligence and its potential uses in predicting at-risk students for educational improvement purposes. The literature review identified both benefits and challenges revolving around XAI in education, which provided the motivation and guidance for the study. Furthermore, meta-analysis research found provided triangulation of results that help identify research consensus as well as non-findings.

Sample Size

A primary benefit of EDM is the ability to obtain large sample sizes thanks to automation and tooling. This study was able to collect the data from the entire population for a period of interest, which enabled sufficient random sampling of a statistically representative sample size for mean difference testing. Furthermore, the determination of sample size was guided by an a priori power analysis.

Power Analysis

Validity of sample size was ensured by conducting an ad prior power analysis, which was described in the Methods section of this study. The results of the power analysis indicated that the minimum sample size required to detect small effects was 368 across test types, assuming equal group size, power of 0.8, and alpha of 0.05. For this study, a total of 58,924 samples were collected from the target population, which represent unique pairs of student/course enrollments (3,043 courses and 23,862 students). Groups of equal sample size (N=500) were randomly selected for post-hoc statistical tests (e.g., correlation and difference testing), which should provide sufficient power to detect small or greater differences.

Generalizability

Since the scope of this study targets online learners from a large four-year dual-mode higher education institution, generalizations are limited to those of similar size, setting, and demographic makeup. Differences in race, gender, background, and instruction delivery could introduce variances across institutions. Additionally, the consistency and repeatability of this study may be limited to institutions delivering LMS like Blackboard. Unfortunately, reliability remains difficult for educational studies as populations from different institutions are inherently different and challenging to control.

Missing Data

Missing data can negatively impact the accuracy and precision of analysis, as well as invalidate tests that assume no missing data (L. Cohen et al., 2018). This study addresses missing data by identifying instances encountered as well as associated mitigations or decisions made.

During the importing of the raw activity table data into the local SQL database, a parsing error resulted in a loss of approximately 1.54% (70,724 of 4.58 million) of the first week's records, or approximately 0.029% (70,724 of 243.1 million) of the cumulative 15-week records. Since the amount of data lost is negligible, there is minimal impact on the study. In addition, the erratic nature of the associated data was out of the researcher's control.

Missing data was also encountered during the data processing stage in which null values were present for both activity features and final grades. For activity features, missing data equates to a lack of logged activity, and are addressed by setting their values equal to zero. For final grades, missing data corresponds to the lack of entry by the instructor, which implies a withdrawal of the student from the course. As such, assigning the missing final grade to a score of zero was insufficient. To address this, an additional labeled column was created to differentiate between those with and without final grades (e.g., complete, and incomplete status respectively).

Inconsistent Performance Assessment Data

Initially, this study aimed to collect student performance data (e.g., quizzes, assignments, and exams) as features to enhance clustering and prediction accuracy. Student performance data is highly desired as it has been thoroughly researched to improve at-risk student prediction accuracies (see references in Table 4 for the "performance" feature category for support). However, the researcher observed the inconsistent implementation of graded activities and assessments across the 3,043 courses analyzed. Although many courses had assignments, quizzes, and mid/final exams, the exact time for assignments and quizzes were difficult to align, aggregate, and compare. As such, student performance as a feature was excluded from the study.

Although excluding student performance features may limit model performance and follow-on clustering and classification analysis, this research contributes to the lack of literature on exploring activity-only features in EDM for at-risk student prediction. From Table 3, only Xing et al.'s (2015) study investigated the impact of using activity-only features. Their work provided the preliminary groundwork for this research, such as leveraging a theory-guided genetic programming model using student interaction as a feature. They demonstrated that this approach outperformed traditional models in both prediction and explainability (Xing et al., 2015). However, their study was limited to a single math course of 122 students with data collected from learning software as opposed to an LMS system. Therefore, this study extends their work to a wider population and larger sample size, which improves the robustness, reliability, and generalizability of XAI research.

Chapter Summary

This chapter provided a brief definition of validity and reliability as well as recommendations from the literature to address associated threats that could impact the credibility and repeatability of this research. Specific threats to validity and reliability were also identified in addition to discussing the associated decisions and actions taken to mitigate such concerns. In summary, this research addressed various validity and reliability where possible within the scope and constraints of the study.

CHAPTER VII

Future Work

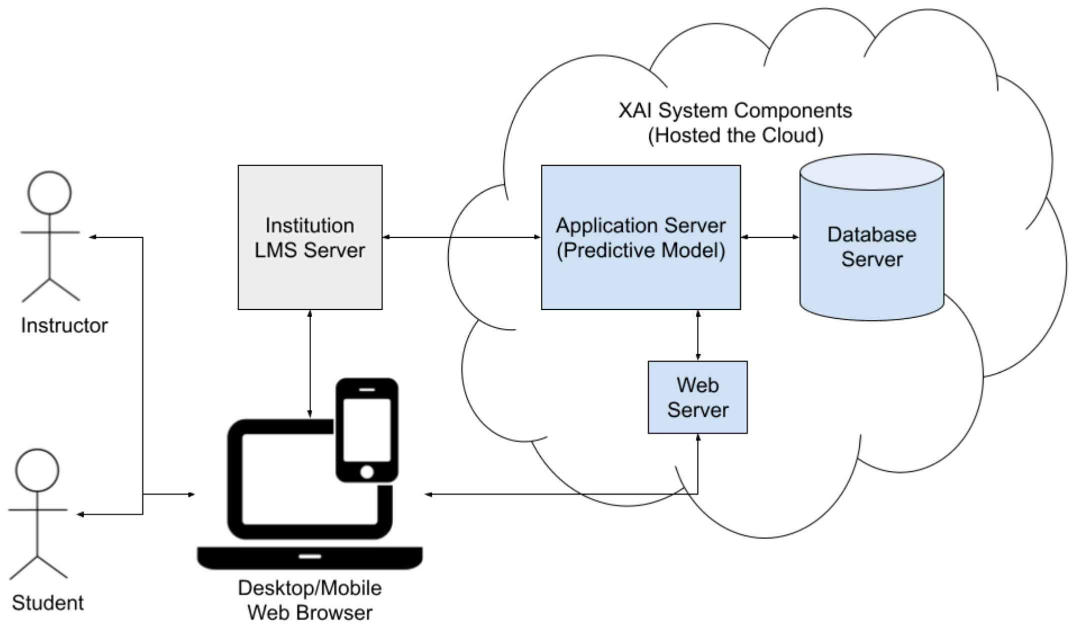
Introduction

XAI Application Design

Future studies can explore the design, development, and practical implementation of an XAI mobile learning system. Figure 11 proposes a conceptual design of the XAI System, which leverages web and cloud technologies to enhance accessibility. The XAI system comprises the Application, Web, and Database Server.

Figure 55

Chart Illustrating the XAI System Architecture Design



The Application Server is responsible for executing functions such as educational data mining and from the LMS, reading from and writing to the Database Server, and responding to Web Server commands. The Application Server also contains the predictive model and at-risk detection logic. The Web Server acts as a front-end graphical user interface between the User (students, educators, and administrators) and the Application/Database Servers.

The Web interface will be accessible to any device with a web browser and will present reports as well as receive information and commands from the users. At-risk alerts are also provided via the Web Server interface, which is retrieved from the Database Server.

The Database server stores data that is processed, formatted, and sanitized by the Application Server, which includes student grades, LMS interaction metrics, and Web Server interaction metrics. The Database server includes at-risk prediction alerts and associated formative feedback, which can be retrieved and formatted by the Web Server and presented to the User-based on their interaction and page requests.

Unlike the current study, which investigates the predictability and explainability of XAI systems, future studies using the XAI systems in a live environment can explore the efficacy and effectiveness of XAI systems in improving student outcomes and retention.

Performance Features

A limitation of this study was the lack of consistent performance metrics across courses and over time. This made it difficult to obtain a representative sample for the population of students in the institution. Since this study aimed to explore the wider population, performance features were excluded. As a result, predictive power and at-risk resolution were reduced due. Future studies can reduce the scope of the target population to investigate the predictability and explainability of XAI systems that include assessment features, such as assignments, quizzes, and/or exam scores.

Intrinsic Features

Although EDM allows for the collection of large sample sizes, the data collected is limited to what is available and logged in the LMS database. This limits the collection of other features such as intrinsic factors, which have been thoroughly researched and determined as critical elements influencing student performance. Cerasoli et al. (2014) conducted a comprehensive meta-analysis and found consensus that intrinsic motivation is a medium to strong predictor of performance. Bandura’s triadic reciprocity within his Social Learning Theory framework identifies the reciprocal interaction between personal factors, behavior, and the environment (Nabavi, 2012). Bandura’s also identified the importance of perceived self-efficacy (i.e., belief in one’s ability) as a “key factor in a generative system of human competence” (Bandura, 1977a, p. 197). When used as features, intrinsic factors such as motivation and self-efficacy may provide additional explanation and predictive power to student behavior, which can be measured by logged activity. Future studies can explore these topics by implementing periodic assessments or surveys within the LMS to solicit student intrinsic factors.

Chapter Summary

This chapter provided directions for future research, such as the implementation of a proposed XAI Application System for exploring the efficacy of at-risk student prediction and explanations. The research focuses on adding performance-based features to the current research, but a smaller scope was recommended to improve prediction performance and at-risk characterization. Finally, intrinsic features are another important topic for future research that can incorporate key factors influencing student beliefs, behaviors, and performance in the context of environmental interactions.

REFERENCES

Abdul, A., Vermeulen, J., Wang, D., Lim, B. Y., & Kankanhalli, M. (2018). Trends and trajectories for explainable, accountable and intelligible systems: An HCI research agenda. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 1–18. https://doi.org/10.1145/3173574.3174156

Abdulhafedh, A. (2021). Incorporating k-means, hierarchical clustering and PCA in customer segmentation. *Journal of City and Development*, *3*(1), 12–30. https://doi.org/10.12691/jcd-3-1-3

Abe, K. (2019). Data mining and machine learning applications for educational big data in the university. *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 350–355. https://doi.org/10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00071

Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, *6*, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052

Al Breiki, B., Zaki, N., & Mohamed, E. A. (2019). Using educational data mining techniques to predict student performance. *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, 1–5. https://doi.org/10.1109/ICECTA48151.2019.8959676

Aliaga, M., & Gunderson, B. (2002). *Interactive statistics*. Sage.

Al-Omar, K. (2018). Evaluating the usability and learnability of the “Blackboard” LMS using SUS and data mining. *2018 Second International Conference on Computing Methodologies and Communication (ICCMC)*, 386–390. https://doi.org/10.1109/ICCMC.2018.8488038

Alonso, J. M., & Casalino, G. (2019). Explainable artificial intelligence for human-centric data analysis in virtual learning environments. In D. Burgos, M. Cimitile, P. Ducange, R. Pecori, P. Picerno, P. Raviolo, & C. M. Stracke (Eds.), *First International Workshop, HELMeTO 2019, Novedrate, CO, Italy, June 6-7, 2019, Revised Selected Papers* (Vol. 1091, Issue September, pp. 125–138). Springer International Publishing. https://doi.org/10.1007/978-3-030-31284-8_10

Alshammari, I. A., Aldhafiri, M. D., & Al-Shammari, Z. (2013). A meta-analysis of educational data mining on improvements in learning outcomes. *College Student Journal*, *47*(2), 326–333. https://www.researchgate.net/publication/273444780_A_META-ANALYSIS_OF_EDUCATIONAL_DATA_MINING_ON_IMPROVEMENTS_IN_LEARNING_OUTCOMES

Anjomshoae, S., Calvaresi, D., Najjar, A., & Främling, K. (2019). Explainable agents and robots: Results from a systematic literature review. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, *2*, 1078–1088. https://dl.acm.org/doi/10.5555/3306127.3331806

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

Arvai, K. (2020). *Kneed: Knee-point detection in Python* (0.7.0) [Python]. https://github.com/arvkevi/kneed

Awaji, M. H. (2018). *Evaluation of machine learning techniques for early identification of at-risk students* [Nova Southern University]. https://nsuworks.nova.edu/gscis_etd/1059/#

Baker, R., & Inventado, P. (2014). Educational data mining and learning analytics. In J. A. Larusson & B. White (Eds.), *Learning analytics: From research to practice* (pp. 61–75). Springer Science. https://doi.org/10.1007/978-1-4614-3305-7_4

Baker, R. S. J. D., Gowda, S. M., & Corbett, A. T. (2011). Automatically detecting a student's preparation for future learning: Help use is key. *EDM 2011 - Proceedings of the 4th International Conference on Educational Data Mining*, 179–188. http://radix.www.upenn.edu/learninganalytics/ryanbaker/PFL-EDM-2011-v19.pdf

Baldelovar, M. (2016). Determinants of students' retention in higher education. *International Journal of Science and Research (IJSR)*, *7*(10), 1460–1462. https://doi.org/10.21275/ART20192308

Bandura, A. (1977a). *Self-efficacy: The exercise of control*. W.H. Freeman and Company.

Bandura, A. (1977b). *Social learning theory*. Prentice-Hall.

Banerjee, P., Dehnbostel, F. O., & Preissner, R. (2018). Prediction is a balancing act: Importance of sampling methods to balance sensitivity and specificity of predictive models based on imbalanced chemical data sets. *Frontiers in Chemistry*, *6*. https://doi.org/10.3389/fchem.2018.00362

Beer, C., & Lawson, C. (2017). The problem of student attrition in higher education: An alternative perspective. *Journal of Further and Higher Education*, *41*(6), 773–784. https://doi.org/10.1080/0309877X.2016.1177171

Beil, D., & Theissler, A. (2020). Cluster-clean-label: An interactive machine learning approach for labeling high-dimensional data. *Proceedings of the 13th International Symposium on Visual Information Communication and Interaction*, 1–8. https://doi.org/10.1145/3430036.3430060

Berens, J., Schneider, K., Görtz, S., Oster, S., & Burghoff, J. (2019). Early detection of students at risk—Predicting student dropouts using administrative student data from German universities and machine learning methods. *Journal of Educational Data Mining*, *11*(3), 1–41. https://jedm.educationaldatamining.org/index.php/JEDM/article/view/389

Berrar, D. (2018). Cross-validation. In *Encyclopedia of bioinformatics and computational biology* (Vol. 1, pp. 542–545). Elsevier. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

Bienkowski, M., Feng, M., & Means, B. (2012). Enhancing teaching and learning through educational data mining and learning analytics: An issue brief. *USDOE*. https://tech.ed.gov/wp-content/uploads/2014/03/edm-la-brief.pdf

Broad agency announcement: Explainable artificial intelligence (XAI) DARPA-BAA-16-53. (2016). DARPA. https://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf

Cano, A., & Leonard, J. D. (2019). Interpretable multiview early warning system adapted to underrepresented student populations. *IEEE Transactions on Learning Technologies*, *12*(2), 198–211. https://doi.org/10.1109/TLT.2019.2911079

Cava, W. L. (2017). *Ellyn* [C++]. https://github.com/lacava/ellyn

Cerasoli, C. P., Nicklin, J. M., & Ford, M. T. (2014). Intrinsic motivation and extrinsic incentives jointly predict performance: A 40-year meta-analysis. *Psychological Bulletin*, *140*(4), 980–1008. https://doi.org/10.1037/a0035661

Chamizo-Gonzalez, J., Cano-Montero, E. I., Urquia-Grande, E., & Muñoz-Colomina, C. I. (2015). Educational data mining for improving learning outcomes in teaching accounting within higher education. *The International Journal of Information and Learning Technology*, *32*(5), 272–285. https://doi.org/10.1108/IJILT-08-2015-0020

Chitti, M., Chitti, P., & Jayabalan, M. (2020). *Need for interpretable student performance prediction*. https://doi.org/10.1109/DeSE51703.2020.9450735

Clow, D. (2013). An overview of learning analytics. *Teaching in Higher Education, 18*(6), 683–695. https://doi.org/10.1080/13562517.2013.827653

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Lawrence Elrbaum Associates.

Cohen, L., Manion, L., & Morrison, K. (2018). *Research methods in education* (8th ed.). Routledge.

Conijn, R., Snijders, C., Kleingeld, A., & Matzat, U. (2017). Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle

LMS. *IEEE Transactions on Learning Technologies*, *10*(1), 17–29. https://doi.org/10.1109/TLT.2016.2616312

CORE. (n.d.). *CORE API*. Retrieved May 9, 2022, from https://core.ac.uk/services/api

Crowe, D., & LaPierre, M. E. (2018). Virtual/mixed reality. *International Journal of Conceptual Structures and Smart Applications*, *6*(1), 33–47. https://doi.org/10.4018/IJCSSA.2018010103

Crowe, D., LaPierre, M., & Kebritchi, M. (2017). Knowledge based artificial augmentation intelligence technology: Next step in academic instructional tools for distance learning. *TechTrends*, *61*(5), 494–506. https://doi.org/10.1007/s11528-017-0210-4

Du, X., Yang, J., Hung, J.-L., & Shelton, B. (2020). Educational data mining: A systematic review of research and emerging trends. *Information Discovery and Delivery*, *48*(4), 225–236. https://doi.org/10.1108/IDD-09-2019-0070

Duval, A. (2019). Explainable artificial intelligence (XAI). *University of Warwick*, 53. https://doi.org/10.13140/RG.2.2.24722.09929

Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis* (5th Eds.). John Wiley & Sons, Ltd.

Ferreira, L. A., Guimarães, F. G., & Silva, R. (2020). Applying genetic programming to improve interpretability in machine learning models. *2020 IEEE Congress on Evolutionary Computation (CEC)*, 1–8. https://doi.org/10.1109/CEC48606.2020.9185620

Field, A. (2018). *Discovering statistics using IBM SPSS statistics* (5th edition). SAGE Publications.

Fink, L. D. (2003). *Creating significant learning experiences: An integrated approach to designing college courses*. Jossey-Bass.

Frost, N., Moshkovitz, M., & Rashtchian, C. (2020). ExKMC: Expanding explainable k-means clustering. *ArXiv:2006.02399 [Cs.LG]*. https://arxiv.org/abs/2006.02399v2

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2019). Explaining explanations: An overview of interpretability of machine learning. *ArXiv:1806.00069 [Cs, Stat]*. http://arxiv.org/abs/1806.00069

Google Trends. (n.d.). Retrieved July 13, 2020, from https://trends.google.com/trends/?geo=US

*G*Power: Statistical power analyses for Windows and Mac*. (2021). https://www.psychologie.hhu.de/arbeitsgruppen/allgemeine-psychologie-und-arbeitspsychologie/gpower.html

Gunning, D., & Aha, D. (2019). DARPA's explainable artificial intelligence (XAI) program. *AI Magazine*, *40*(2), 44–58. https://doi.org/10.1609/aimag.v40i2.2850

Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., & Yang, G.-Z. (2019). XAI—Explainable artificial intelligence. *Science Robotics*, *4*(37), eaay7120. https://doi.org/10.1126/scirobotics.aay7120

He, F., Mazumdar, S., Tang, G., Bhatia, T., Anderson, S., Dew, M., Krafty, R., V.L., N., Deshpande, S., Hall, M., & C., R. (2016). Nonparametric MANOVA approaches for non-normal multivariate outcomes with missing values. *Communications in Statistics - Theory and Methods*, *46*. https://doi.org/10.1080/03610926.2016.1146767

Imran, M., Latif, S., Mehmood, D., & Shah, M. S. (2019). Student academic performance prediction using supervised learning techniques. *International Journal of Emerging Technologies in Learning (IJET), 14*(14), 92. https://doi.org/10.3991/ijet.v14i14.10310

Kalles, D., & Pierrakeas, C. (2006). Analyzing student performance in distance learning with genetic algorithms and decision trees. *Applied Artificial Intelligence*, *20*(8), 655–674. https://doi.org/10.1080/08839510600844946

Karpatne, A., Atluri, G., Faghmous, J., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., & Kumar, V. (2017). Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, *29*(10), 2318–2331. https://doi.org/10.1109/TKDE.2017.2720168

Khare, K., Stewart, B., & Khare, A. (2018). Artificial Intelligence and the student experience: An institutional perspective. *IAFOR Journal of Education*, *6*(3), 63–78.

Knight, W. (2017). The dark secret at the heart of AI. *MIT Technology Review*, *120*(3), 11. https://www.technologyreview.com/2017/04/11/5113/the-dark-secret-at-the-heart-of-ai/

Knoth, P., & Zdrahal, Z. (2012). CORE: Three access levels to underpin open access. *D-Lib Magazine*, *18*(11/12). https://doi.org/10.1045/november2012-knoth

Kondo, N., Okubo, M., & Hatanaka, T. (2017). Early detection of at-risk students using machine learning based on LMS log data. *Proceedings - 2017 6th IIAI*

International Congress on Advanced Applied Informatics, IIAI-AAI 2017, 198–201. https://doi.org/10.1109/IIAI-AAI.2017.51

Kwak, S. G., & Kim, J. H. (2017). Central limit theorem: The cornerstone of modern statistics. *Korean Journal of Anesthesiology*, *70*(2), 144. https://doi.org/10.4097/kjae.2017.70.2.144

Kwon, O. (2020). Very simple statistical evidence that AlphaGo has exceeded human limits in playing GO game. *ArXiv:2002.11107 [Cs]*. http://arxiv.org/abs/2002.11107

La Cava, W., Silva, S., Danai, K., Spector, L., Vanneschi, L., & Moore, J. H. (2019). Multidimensional genetic programming for multiclass classification. *Swarm and Evolutionary Computation*, *44*, 260–272. https://doi.org/10.1016/j.swevo.2018.03.015

La Cava, W., Silva, S., Vanneschi, L., Spector, L., & Moore, J. (2017). Genetic programming representations for multi-dimensional feature learning in biomedical classification. In G. Squillero & K. Sim (Eds.), *Applications of Evolutionary Computation* (Vol. 10199, pp. 158–173). Springer International Publishing. https://doi.org/10.1007/978-3-319-55849-3_11

Layton, R. (2017). *Learning data mining with Python: Use Python to manipulate data and build predictive models* (2nd Eds.). https://learning.oreilly.com/library/view/-/9781787126787/?ar

Long, M., Ferrier, F., & Heagney, M. (2006). Stay, play or give it away? Students continuing, changing or leaving university study in first year. *Monash University*, 253. https://files.eric.ed.gov/fulltext/ED505791.pdf

Manyika, J., Chui, M., Bughin, J., Dobbs, R., Bisson, P., & Marrs, A. (2013). *Disruptive technologies: Advances that will transform life, business, and the global economy* (pp. 1–162). McKinsey Global Institute. https://www.mckinsey.com/~/media/McKinsey/Business Functions/McKinsey Digital/Our Insights/Disruptive technologies/MGI_Disruptive_technologies_Full_report_May2013.ashx

Matplotlib: Python plotting—Matplotlib 3.4.3 documentation. (n.d.). Retrieved September 5, 2021, from https://matplotlib.org/

McCoy, S., & Byrne, D. (2017). Student retention in higher education. In *Economic Insights on Higher Education Policy in Ireland: Evidence from a Public System* (pp. 111–141). https://doi.org/10.1007/978-3-319-48553-9_5

McCrum-Gardner, E. (2008). Which is the correct statistical test to use? *British Journal of Oral and Maxillofacial Surgery*, *46*(1), 38–41. https://doi.org/10.1016/j.bjoms.2007.09.002

Michael, W. (2021). *Seaborn: Statistical data visualization*. https://seaborn.pydata.org/index.html

Moyer, C. (2016, March 28). *How Google's AlphaGo beat a go world champion*. The Atlantic. https://www.theatlantic.com/technology/archive/2016/03/the-invisible-opponent/475611/

Muijs, D. (2004). *Doing quantitative research in education with SPSS*. Sage Publications.

Nabavi, R. T. (2012). Bandura's social learning theory & social cognitive learning theory. *Research Gate,* 24.

https://www.researchgate.net/publication/267750204_Bandura%27s_Social_Learning_Theory_Social_Cognitive_Learning_Theory

NCES. (2020, April). *The Condition of Education—Postsecondary Education—Programs, Courses, and Completions—Undergraduate Retention and Graduation Rates—Indicator April (2020)*. https://nces.ed.gov/programs/coe/indicator_ctr.asp

NCES. (2021). *College Navigator—National Center for Education Statistics*. https://nces.ed.gov/collegenavigator/

NCES. (2022). *Trend Generator*. https://nces.ed.gov/ipeds/TrendGenerator/app/

Ndou, N., Ajoodha, R., & Jadhav, A. (2020). Educational data-mining to determine student success at higher education institutions. *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 1–8. https://doi.org/10.1109/IMITEC50163.2020.9334139

OECD. (2020). *Education at a glance 2020: OECD indicators*. OECD. https://doi.org/10.1787/69096873-en

Ostler, D. E. (n.d.). *Guidelines for writing research proposals, reports, theses, and dissertations*. 13.

Pandas: Powerful python data analysis toolkit. (2021). [Python]. pandas. https://github.com/pandas-dev/pandas (Original work published 2010)

Peikari, M., Salama, S., Nofech-Mozes, S., & Martel, A. L. (2018). A cluster-then-label semi-supervised learning approach for pathology image classification. *Scientific Reports, 8*(1), 7193. https://doi.org/10.1038/s41598-018-24876-0

Perera-Diltz, D. M., & Moe, J. L. (2014). Formative and summative assessment in online education. *Journal of Research in Innovative Teaching*, *7*(4), 14. https://digitalcommons.odu.edu/chs_pubs/37

Perkis, T. (1994). Stack-based genetic programming. *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, 148–153 vol.1. https://doi.org/10.1109/ICEC.1994.350025

Pillay, N. (2020). The impact of genetic programming in education. *Genetic Programming and Evolvable Machines*, *21*(1–2), 87–97. https://doi.org/10.1007/s10710-019-09362-4

Raju, R., Kalaiselvi, N., M., A. S., I., D., & A., S. (2020). Educational data mining: A comprehensive study. *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, 1–5. https://doi.org/10.1109/ICSCAN49426.2020.9262399

Ramalingam, V. V., Pandian, A., Chetry, P., & Nigam, H. (2018). Automated essay grading using machine learning algorithm. *Journal of Physics: Conference Series*, *1000*(1), 012030. https://doi.org/10.1088/1742-6596/1000/1/012030

Reiser, R. A., & Ely, D. P. (1997). The field of educational technology as reflected through its definitions. *Educational Technology Research and Development*, *45*(3), 63–72. https://doi.org/10.1007/BF02299730

Şahin, M., & Yurdugül, H. (2020). Educational data mining and learning analytics: Past, present and future. *Eğitsel Veri Madenciliği ve Öğrenme Analitikleri: Dünü, Bugünü ve Geleceği.*, *9*(1), 121–131. https://doi.org/10.14686/buefad.606077

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., & Müller, K.-R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE, 109*(3), 247–278. https://doi.org/10.1109/JPROC.2021.3060483

Samek, W., & Müller, K.-R. (2019). Towards explainable artificial intelligence. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 11700 LNCS* (pp. 5–22). https://doi.org/10.1007/978-3-030-28954-6_1

Satopaa, V., Albrecht, J., Irwin, D., & Raghavan, B. (2011). Finding a "kneedle" in a haystack: Detecting knee points in system behavior. *2011 31st International Conference on Distributed Computing Systems Workshops*, 166–171. https://doi.org/10.1109/ICDCSW.2011.20

Schunk, D. H., & Pajares, F. (2009). Self-efficacy theory. In *Handbook of motivation at school.* (pp. 35–53). Routledge/Taylor & Francis Group.

Scikit-learn: Machine learning in python—Scikit-learn 0.24.2 documentation. (n.d.). Retrieved August 28, 2021, from https://scikit-learn.org/stable/index.html

SciPy.org—SciPy.org. (n.d.). Retrieved August 28, 2021, from https://www.scipy.org/

Shin, D. (2021). The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable AI. *International Journal of Human-Computer Studies*, *146*, 102551. https://doi.org/10.1016/j.ijhcs.2020.102551

Shrestha, N. (2020). Detecting multicollinearity in regression analysis. *American Journal of Applied Mathematics and Statistics*, *8*(2), 39–42. https://doi.org/10.12691/ajams-8-2-1

Siemens, G., & Baker, R. S. J. d. (2012). Learning analytics and educational data mining: Towards communication and collaboration. *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, 252–254. https://doi.org/10.1145/2330601.2330661

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., & Bolton, A. (2017). Mastering the game of go without human knowledge. *Nature*, *550*(7676), 354–359.

Sklearn.tree.export_graphviz. (2022). Scikit-Learn. https://scikit-learn/stable/modules/generated/sklearn.tree.export_graphviz.html

Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006). Beyond accuracy, f-score and ROC: A family of discriminant measures for performance evaluation. In A. Sattar & B. Kang (Eds.), *AI 2006: Advances in Artificial Intelligence* (Vol. 4304, pp. 1015–1021). Springer Berlin Heidelberg. https://doi.org/10.1007/11941439_114

Song, D. (2021a). *Advance statistics 10: Classification 2*. https://shsu.blackboard.com

Song, D. (2021b). *Advance statistics 11: Clustering 1*. https://shsu.blackboard.com

Song, D. (2021c). *Advance statistics 12: Clustering 2*. https://shsu.blackboard.com

Song, D. (2021d, April 7). *2021 Spring COUN 7374 Week 12. Machine Learning—Clustering Part 2*. https://www.youtube.com/watch?v=MC1KbJ4TAUw

Song, D., Rice, M., & Oh, E. Y. (2019). Participation in online courses and interaction with a virtual agent. *The International Review of Research in Open and Distributed Learning*, *20*(1). https://doi.org/10.19173/irrodl.v20i1.3998

Stephens, T. (2016). *Introduction to GP — gplearn 0.4.2 documentation*. https://gplearn.readthedocs.io/en/stable/intro.html

Stoffel, K., & Spector, L. (1996). High-performance, parallel, stack-based genetic programming. In J. R. Koza, D. E. Goldberg, D. B. Fogel, & R. L. Riolo (Eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference* (pp. 224–229). The MIT Press. http://faculty.hampshire.edu/lspector/pubs/HiGP-gp96-e.pdf

Stolk, C. van, Tiessen, J., Cliff, J., & Levitt, R. (2007). *Student retention in higher education courses: International comparison* (p. 94). Rand Corporation. https://www.rand.org/content/dam/rand/pubs/technical_reports/2007/RAND_TR482.pdf

Subbiah, U., Kumar, R. V., Panicker, S. A., Bhalaje, R. A., & S, P. (2020). An enhanced deep learning architecture for the classification of cancerous lymph node images. *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 381–386. https://doi.org/10.1109/ICIRCA48905.2020.9183250

Sun, T. Q., & Medaglia, R. (2019). Mapping the challenges of artificial intelligence in the public sector: Evidence from public healthcare. *Government Information Quarterly*, *36*(2), 368–383. https://doi.org/10.1016/j.giq.2018.09.008

The Texas State University System. (2022, February). *Workbook: TSUS enrollment explorer*. https://public.tableau.com/views/TSUSEnrollmentExplorer-aggregatev3/SelectStudents?%3Adisplay_static_image=y&%3AbootstrapWhenNotified=true&%3Aembed=true&%3Alanguage=en-US&:embed=y&:showVizHome=n&:apiID=host0#navType=0&navSrc=Parse

Tinto, V. (2006). Research and practice of student retention: What next? *Journal of College Student Retention: Research, Theory & Practice*, *8*(1), 1–19. https://doi.org/10.2190/4YNU-4TMB-22DJ-AN4W

Veerasamy, A. K. (2020). *Predictive models as early warning systems for student academic performance in introductory programming* [University of Turku]. https://core.ac.uk/reader/347181188

Vilone, G., & Longo, L. (2020). *Explainable artificial intelligence: A systematic review*. *Dl*. http://arxiv.org/abs/2006.00093

Visualizing cross-validation behavior in scikit-learn—Scikit-learn 0.24.2 documentation. (n.d.). Retrieved August 29, 2021, from https://scikit-learn.org/stable/auto_examples/model_selection/plot_cv_indices.html

Vygotsky, L. S. (1985). Mind in society: The development of higher psychological processes. In M. Cole, V. John-Steiner, S. Scribner, & E. Souberman (Eds.), *Harefuah* (Vol. 108, Issues 3–4).

Wellman, J., Johnson, N., & Steele, P. (2012). *Measuring (and managing) the invisible cost of postsecondary attrition* (p. 12). American Institute for Research. https://files.eric.ed.gov/fulltext/ED536120.pdf

Wills, R., Elder, A., & Molina, D. (2018). What matters in college student success? Determinants of college retention and graduation rates. *Education*, *138*(4), 309–322. https://eric.ed.gov/?id=EJ1180297

Xiao, Y., & Watson, M. (2019). Guidance on conducting a systematic literature review. *Journal of Planning Education and Research*, *39*(1), 93–112. https://doi.org/10.1177/0739456X17723971

Xing, W., Guo, R., Petakovic, E., & Goggins, S. (2015). 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. https://doi.org/10.1016/j.chb.2014.09.034

Yang, Y. C. C., Chen, I. Y. L., & Hiroaki, O. (2021). Toward precision education: Educational data mining and learning analytics for identifying students learning patterns with ebook systems. *Educational Technology & Society*, *24*(1), 152–163. https://ezproxy.shsu.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.26977864&site=eds-live&scope=site

Yuan, C., & Yang, H. (2019). Research on k-value selection method of k-means clustering algorithm. *J*, *2*(2), 226–235. https://doi.org/10.3390/j2020016

Zhai, X., Chu, X., Chai, C. S., Jong, M. S. Y., Istenic, A., Spector, M., Liu, J.-B., Yuan, J., & Li, Y. (2021). A review of artificial intelligence (AI) in education from 2010 to 2020. *Complexity*, *2021*, e8812542. https://doi.org/10.1155/2021/8812542

Zheng, Y. (2020). Predicting personality traits by student learning behaviors on Blackboard systems. *SHS Web of Conferences*, *77*, 01003. https://doi.org/10.1051/shsconf/20207701003

APPENDIX

Analysis Scripts

/data

```
486K Jun 16 00:11  act01.csv.gz
297K Jun 16 00:11  act02.csv.gz
330K Jun 16 00:11  act03.csv.gz
328K Jun 16 00:11  act04.csv.gz
327K Jun 16 00:11  act05.csv.gz
310K Jun 16 00:11  act06.csv.gz
310K Jun 16 00:11  act07.csv.gz
287K Jun 16 00:11  act08.csv.gz
296K Jun 16 00:11  act09.csv.gz
261K Jun 16 00:11  act10.csv.gz
213K Jun 16 00:11  act11.csv.gz
175K Jun 16 00:11  act12.csv.gz
166K Jun 16 00:11  act13.csv.gz
287K Jun 16 00:11  act14.csv.gz
280K Jun 16 00:11  act15.csv.gz
486K Jun 16 00:11  cumulative_act01.csv.gz
525K Jun 16 00:11  cumulative_act02.csv.gz
582K Jun 16 00:11  cumulative_act03.csv.gz
630K Jun 16 00:11  cumulative_act04.csv.gz
665K Jun 16 00:11  cumulative_act05.csv.gz
693K Jun 16 00:11  cumulative_act06.csv.gz
702K Jun 16 00:11  cumulative_act07.csv.gz
708K Jun 16 00:11  cumulative_act08.csv.gz
713K Jun 16 00:11  cumulative_act09.csv.gz
716K Jun 16 00:11  cumulative_act10.csv.gz
719K Jun 16 00:11  cumulative_act11.csv.gz
722K Jun 16 00:11  cumulative_act12.csv.gz
725K Jun 16 00:11  cumulative_act13.csv.gz
745K Jun 16 00:11  cumulative_act14.csv.gz
763K Jun 17 02:57  cumulative_act15.csv.gz
 94K Jun 16 00:11  cumulative_act15_clustered.csv.gz
 11M Jun 16 00:28  data-1624239985814_course_user.csv.gz
9.1M Jun 16 00:28  data-1624240128001_course_user.csv.gz
1.5M Jun 16 00:28  data-1624240236386_course_user.csv.gz
 27M Jun 16 00:30  gradebook_grade_calc_210223_1108.csv.gz
 20M Jun 16 00:30  gradebook_main_data-1624482074159.csv.gz
 16M Jun 16 00:11  schema.zip
```

Code for Exploratory Clustering (RQ1 & RQ2)

/src/cluster_analysis_cumulative_act15_exploratory.py

```
#!/usr/bin/env python
# coding: utf-8

# # Cluster Analysis

# ## Imports

# In[1]:


# Datasets
import pandas as pd
import numpy as np

# sklearn for clustering analysis
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import RobustScaler
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import MinMaxScaler

# scipy
from scipy.stats import pearsonr
from scipy.io import arff    # for working with Weka

# Kneed imports
from kneed import KneeLocator

# Data Visualizations
import matplotlib.pyplot as plt
import matplotlib as mpl
from matplotlib import colors
import seaborn as sns
from ellyn import ellyn


# In[2]:
```

```
rCuffOff = 0.6
alpha = 0.05


# ## Import Data

# In[3]:


from pathlib import Path
df = pd.read_csv(Path("../data/cumulative_act15.csv.gz"))
len(df)


# ## Feature Exclusion
# - Exclude irrelevant features
# - Exclude features with p > .05
# - Exclude redundant features
# - Exclude outliers

# In[4]:


# exclude irrelevant features
df = df.drop(columns=['course','student'])
len(df)


# In[5]:


pvalue = df.corr(method=lambda x, y: pearsonr(x, y)[1]) -
np.eye(len(df.columns))
alphaMax = alpha
mask = np.zeros_like(pvalue)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(7, 5))
    ax = sns.heatmap(pvalue, mask=mask, square=True, linewidths=.01,
vmax=alphaMax)


# In[6]:
```

```
# exclude insignificantly correlated features
col = [i for i in pvalue.columns if pvalue[i].final_grade > alphaMax]
df = df.drop(columns=col)
print("Excluding features: " + str(col))
len(df)


# In[7]:


corr = abs(df.corr(method='spearman'))
corrMax = rCuffOff
mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style("white"):
    f, ax = plt.subplots(figsize=(7, 5))
    ax = sns.heatmap(corr, mask=mask, vmin=0, vmax=corrMax, square=True,
linewidths=.01)


# In[8]:


# Exclude highly correlated and missing values
def excludeHighlyCorrelatedAndMissingValues(df, r):
    temp = []
    corr = abs(df.corr(method='spearman'))
    for y in corr.columns:
        for x in corr.index.values:
            if(x != y):
                c = corr[y][x]
                if(c > r):
                    if [y,x] not in temp:
                        temp.append([x,y])
                        print("Highly correlated pair found: " +
str([x,y]))

    if(len(temp) > 0):
        drop = []
        for i in temp:
            a = i[0]
            b = i[1]
            dA = df.drop(columns=a).dropna()
            dB = df.drop(columns=b).dropna()
            print("Size after dropping " + a + ": " + str(dA.size))
            print("Size after dropping " + b + ": " + str(dB.size))
            if(dA.size > dB.size):
                print("\tDropping " + a)
                drop.append(a)
            else:
                print("\tDropping " + b)
                drop.append(b)
        for i in drop:
            df = df.drop(columns=i)
        return df.dropna()
    else:
        return df.dropna()

df = excludeHighlyCorrelatedAndMissingValues(df, corrMax)


# In[9]:


df


# In[10]:


fig, ax1 = plt.subplots()
ax1.set_xlabel('Feature')
ax1.set_ylabel('Activity Frequency')
ax1.set_title('Activity Summary (N=' + str(len(df)) + ')')

temp = df.drop(columns=['final_grade'])
ax1.boxplot(temp, labels=temp.columns)

temp2 = df['final_grade']
ax2 = ax1.twinx()
ax2.boxplot(temp2, labels=['final_grade'],
positions=[len(temp.columns)+1])


ax2.set_ylabel('Final Grade', color='purple')
ax2.tick_params('y',colors='purple')

# ask matplotlib for the plotted objects and their labels
lines, labels = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
```

```
# In[11]:


# Exclude outliers
def iqr_outliers(df, feature, factor):
    Q1= df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    upper_limit = Q3 + factor * IQR
    lower_limit = Q1 - factor * IQR
    return upper_limit, lower_limit

def std_outliers(df, feature, factor):
    upper_limit = df[feature].mean() + factor * df[feature].std()
    lower_limit = df[feature].mean() - factor * df[feature].std()
    return upper_limit, lower_limit

# lastSize = 0
# while lastSize != len(df):
#     lastSize = len(df)
for c in df.columns:
    upper, lower = iqr_outliers(df, c, 1.5)
    df = df[(df[c] > lower) & (df[c] < upper)]


# In[12]:


len(df)


# In[13]:


df


# In[14]:


fig, ax1 = plt.subplots()
ax1.set_xlabel('Feature')
ax1.set_ylabel('Activity Frequency')
ax1.set_title('Activity Summary (N=' + str(len(df)) + ')')

temp = df.drop(columns=['final_grade'])
```

```
ax1.boxplot(temp, labels=temp.columns)

temp2 = df['final_grade']
ax2 = ax1.twinx()
ax2.boxplot(temp2, labels=['final_grade'],
positions=[len(temp.columns)+1])


ax2.set_ylabel('Final Grade', color='purple')
ax2.tick_params('y',colors='purple')

# ask matplotlib for the plotted objects and their labels
lines, labels = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()


# In[15]:


# Linear scalers
def scaleMinMax(df):
    from sklearn.preprocessing import MinMaxScaler
    scaler = MinMaxScaler()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

def scaleRobust(df):
    from sklearn.preprocessing import RobustScaler
    scaler = RobustScaler()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

def scaleNormalizer(df):
    from sklearn.preprocessing import Normalizer
    scaler = Normalizer()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

# Nonlinear transformers
def transformQuantile(df):
    from sklearn.preprocessing import QuantileTransformer
    qt = QuantileTransformer()
    return pd.DataFrame(qt.fit_transform(df), columns=df.columns)

def transformPower(df):
    from sklearn.preprocessing import PowerTransformer
    pt = PowerTransformer()
    return pd.DataFrame(pt.fit_transform(df), columns=df.columns)
```

```
# Scale/Transform
df = scaleMinMax(df)


# In[16]:


df


# ## Cluster Analysis

# In[17]:


def clusterKMeans(df):
    kmeans_kwargs = {
        "init":"k-means++",
        "n_init": 1,
        "max_iter": 500,
        "random_state": 0
    }

    sse = []
    N = 21
    for k in range(1,N):
        kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
        kmeans.fit(df.drop(columns='final_grade'))
        sse.append(kmeans.inertia_)
    k1 = KneeLocator(range(1,N), sse, curve="convex",
direction="decreasing")

    plt.style.use("fivethirtyeight")
    plt.plot(range(1,N), sse, 'ko-', label='SSE (Week 1-15 Cumulative
Activity)')
    plt.xticks(range(1,N))
    plt.xlabel("Clusters (K)")
    plt.ylabel("SSE")
    plt.title('k-Means SSE (Exploratory)')
    k = k1.elbow
    plt.plot([k], sse[k-1], 'X', ms=10, color='red', label='Elbow Point')
    plt.legend()
    plt.show()

    print("Cluster size: " + str(k1.elbow))
    kmeans = KMeans(n_clusters=k1.elbow, **kmeans_kwargs)
    kmeans.fit(df.drop(columns='final_grade'))
    return df.assign(cluster=kmeans.labels_)

df = clusterKMeans(df)
df


# In[18]:


df['cluster'].unique()


# In[19]:


df.to_csv("../data/cumulative_act15_clustered.csv")


# In[20]:


df.cluster.nunique()


# In[21]:


fig, ax1 = plt.subplots()
ax1.set_xlabel('Feature')
ax1.set_ylabel('Activity Frequency')
ax1.set_title('Activity Summary (N=' + str(len(df)) + ')')

temp = df.drop(columns=['final_grade', 'cluster'])
ax1.boxplot(temp, labels=temp.columns)

temp2 = df['final_grade']
ax2 = ax1.twinx()
ax2.boxplot(temp2, labels=['final_grade'],
positions=[len(temp.columns)+1])


ax2.set_ylabel('Final Grade', color='purple')
ax2.tick_params('y',colors='purple')
```

```
# ask matplotlib for the plotted objects and their labels
lines, labels = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()


# g = sns.PairGrid(df, hue="cluster", palette='tab10')
# g.map_diag(sns.kdeplot, linewidth=3, common_norm=False)
# g.map_lower(sns.scatterplot)
# g.map_upper(sns.kdeplot, alpha=0.9)
# g.add_legend()

# In[22]:


means = df.groupby(['cluster']).mean()
means


# In[23]:


std = df.groupby(['cluster']).std()
std


# In[24]:


fig, ax1 = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax1.set_title('K-Means Cluster Summary (Classification)')
ax1.set_ylabel('Normalized Frequency')
ax1.set_xlabel('Cluster')
j = 0
for i in df.drop(columns=['cluster','final_grade']).columns:
    ax1.plot(means[i], marker[j], ms=10, label=i)
    j += 1

ax2 = ax1.twinx()
ax2.plot(means['final_grade'], marker[j], color='purple', ms=10,
label='final_grade')
ax2.set_ylabel('Final Grade', color='purple')
ax2.tick_params('y',colors='purple')
ax2.grid(True,linestyle='--')
```

```
# ask matplotlib for the plotted objects and their labels
lines, labels = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax2.legend(lines + lines2, labels + labels2, bbox_to_anchor=(1.15, 1.0),
loc='upper left')


# In[25]:


from sklearn import metrics
metrics.silhouette_score(df.drop(columns='cluster'), df['cluster'],
metric='euclidean')


# In[38]:


def getAtRiskBooleansGroup(df, cutoff):
    # get boolean array where at-risk students are defined as final grade
is less than cutoff value
    temp =
pd.DataFrame(df.get(['cluster','final_grade'])).groupby(['cluster'])
    at_risk_groups = []
    for i in range(0,len(temp)):
        mean = temp.get_group(i).mean()['final_grade']
        if(mean < cutoff):
            at_risk_groups.append(i)

    y = np.zeros(len(df), dtype=bool)
    for i in at_risk_groups:
        y = y | (df['cluster'] == i)

    return y


# In[45]:


def getAtRiskBooleansSamples(df, cutoff):
    # get boolean array where at-risk students are defined as final grade
is less than cutoff value
    temp =
pd.DataFrame(df.get(['cluster','final_grade'])).groupby(['cluster'])
    mean = df['final_grade'].mean()
    y = np.zeros(len(df), dtype=bool)
    y = df['final_grade'] < mean

    return y


# In[70]:


def evaluateEllynGP(df, folder):
    from ellyn import ellyn
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    import statistics
    model = ellyn(
        g = 100, classification = True, class_m4gp = True,
op_list=['n','v','+','-','*','/'],
         # ================ Results and printing
         resultspath= './' + folder,
         #savename
         savename="gp",
         #print initial population
         print_init_pop = True,
         #print last population
         print_last_pop = True,
         #print best individual at end
         print_best_ind = True)

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    y_pred = cross_val_predict(model, x, y, cv=10)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

results = evaluateEllynGP(df, 'gp_cumulative_act15_classification')
print(results)


# In[ ]:


import statistics
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import classification_report
```

```
def evaluateNB(x, y):
    gnb = GaussianNB()
    y_pred = cross_val_predict(gnb, x, y, cv=10)
    return classification_report(y, y_pred)
x = df.drop(columns=['cluster', 'final_grade'])
y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
feature_names = df.drop(columns=['cluster', 'final_grade']).columns
c = evaluateNB(x, y)
print(c)


# In[ ]:


df


# In[ ]:


def evaluateBinaryGP(df):
    from gplearn.genetic import SymbolicClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import classification_report
    import statistics
    from sklearn import tree
    import graphviz
    x = df.drop(columns=['cluster', 'final_grade'])
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    feature_names = df.drop(columns=['cluster', 'final_grade']).columns
    gp = SymbolicClassifier(parsimony_coefficient=.01,
                            feature_names=feature_names,
                            random_state=1, verbose=True,
                            generations=100)
    gp.fit(x,y)
    y_pred = cross_val_predict(gp, x, y, cv=10, verbose=True, n_jobs=15)

    dot_data = gp._program.export_graphviz()

    graph = graphviz.Source(dot_data)

    return gp, y_pred, classification_report(y, y_pred), graph

gp, y_pred, report, g = evaluateBinaryGP(df)
```

```
# In[ ]:


print(report)
print(gp._program)
g


# In[75]:


import graphviz
import statistics
from pandas import DataFrame

def evaluateDT(x:DataFrame,y:DataFrame):
    from sklearn import tree
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    from sklearn.tree import export_text

    model = tree.DecisionTreeClassifier(max_depth=3)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True,
n_jobs=15)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    model.fit(x,y)

    dot_data = tree.export_graphviz(model, out_file=None,
feature_names=x.columns, class_names=['Not-At-Risk','At-Risk'],
filled=True, rounded=True, special_characters=True, proportion=True,
precision=2, rotate=True)
    graph = graphviz.Source(dot_data)

    return precision, recall, f1, len(y), graph
x = df.drop(columns=['cluster', 'final_grade'])
y = getAtRiskBooleansGroup(df, statistics.mean(df['final_grade']))
results = evaluateDT(x,y)
print(results)
results[4]


# In[78]:
```

```
import graphviz
import statistics
from pandas import DataFrame

def toStrArray(numArray):
    return [str(i) for i in numArray]

def evaluateDTMulti(df):
    from sklearn import tree
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    from sklearn.tree import export_text

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    model = tree.DecisionTreeClassifier(max_depth=4)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True,
n_jobs=15)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    model.fit(x,y)

    dot_data = tree.export_graphviz(model, out_file=None,
feature_names=df.drop(columns=['cluster','final_grade']).columns,
class_names=toStrArray(df['cluster'].unique()), filled=True, rounded=True,
special_characters=True, proportion=True, precision=2, rotate=True)
    graph = graphviz.Source(dot_data)

    return precision, recall, f1, len(y), graph

results = evaluateDTMulti(df)
print(results)
results[4]


# In[ ]:


import arff

arff.dump('test.arff', df.values, relation='relation
name',names=df.columns)
```

```
# In[ ]:


def evaluateEllynGP(df, folder):
    from ellyn import ellyn
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    model = ellyn(
        g = 100, classification = True, class_m4gp = True,
op_list=['n','v','+','-','*','/','<','>','<=','>='],
         # ================ Results and printing
         resultspath= './' + folder,
         #savename
         savename="gp",
         #print initial population
         print_init_pop = True,
         #print last population
         print_last_pop = True,
         #print best individual at end
         print_best_ind = True)

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade'])) #binary
classification
    y_pred = cross_val_predict(model, x, y, cv=10)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)


precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateEllynGP(df, 'wk_' + "{:02d}".format(i+1))
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())
```

```
x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Genetic Programming, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Genetic Programming')
```

Code for Classification Analysis (RQ1 & RQ3)

/src/cluster_analysis_cumulative_All.py

```
#!/usr/bin/env python
# coding: utf-8

# # Cluster Analysis

# ## Imports

# In[1]:


# Datasets
import pandas as pd
import numpy as np

# sklearn for clustering analysis
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import RobustScaler
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import MinMaxScaler

# scipy
from scipy.stats import pearsonr
from scipy.io import arff   # for working with Weka

# Kneed imports
from kneed import KneeLocator

# Data Visualizations
import matplotlib.pyplot as plt
import matplotlib as mpl
from matplotlib import colors
import seaborn as sns


# In[2]:
```

```
rCuffOff = 0.6
alpha = 0.05


# ## Import Data

# In[3]:


from pathlib import Path
fname = 'cumulative_act'
df = pd.read_csv(Path("../../_Data/act01.csv"))
dfArray = []
dfArray.append(df)
for i in range(2,16):
    df = pd.read_csv(Path("../../_Data/" + fname + "{:02d}".format(i) +
'.csv'))
    dfArray.append(df)


# In[4]:


dfArray


# ## Feature Exclusion
# - Exclude irrelevant features
# - Exclude features with p > .05
# - Exclude redundant features
# - Exclude outliers

# In[5]:


# exclude irrelevant features
for i in range(0,len(dfArray)):
    dfArray[i] = dfArray[i].drop(columns=['course','student'])


# In[6]:


dfArray
```

```
# In[7]:


# exclude insignificantly correlated features
for j in range(0, len(dfArray)):
    pvalue = dfArray[j].corr(method=lambda x, y: pearsonr(x, y)[1]) -
np.eye(len(dfArray[j].columns))
    alphaMax = alpha
    col = [i for i in pvalue.columns if pvalue[i].final_grade > alphaMax]
    dfArray[j] = dfArray[j].drop(columns=col)
    print("Excluding features in Dataset " + str(j) + ": " + str(col))


# In[8]:


# Exclude highly correlated and missing values
def excludeHighlyCorrelatedAndMissingValues(df, r):
    temp = []
    corr = abs(df.corr(method='spearman'))
    for y in corr.columns:
        for x in corr.index.values:
            if(x != y):
                c = corr[y][x]
                if(c > r):
                    if [y,x] not in temp:
                        temp.append([x,y])
                        print("Highly correlated pair found: " + str([x,y]))

    if(len(temp) > 0):
        drop = []
        for i in temp:
            a = i[0]
            b = i[1]
            dA = df.drop(columns=a).dropna()
            dB = df.drop(columns=b).dropna()
            print("Size after dropping " + a + ": " + str(dA.size))
            print("Size after dropping " + b + ": " + str(dB.size))
            if(dA.size > dB.size):
                print("\tDropping " + a)
                drop.append(a)
            else:
                print("\tDropping " + b)
                drop.append(b)
        for i in drop:
                df = df.drop(columns=i)
            return df.dropna()
        else:
            return df.dropna()

corrMax = rCuffOff
for i in range(0, len(dfArray)):
    print("Analyzing correlation in Dataset " + str(i) + " for exclusion.")
    dfArray[i] = excludeHighlyCorrelatedAndMissingValues(dfArray[i], corrMax)


# In[9]:


# Exclude outliers
def iqr_outliers(df, feature, factor):
    Q1= df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    upper_limit = Q3 + factor * IQR
    lower_limit = Q1 - factor * IQR
    return upper_limit, lower_limit

def std_outliers(df, feature, factor):
    upper_limit = df[feature].mean() + factor * df[feature].std()
    lower_limit = df[feature].mean() - factor * df[feature].std()
    return upper_limit, lower_limit

for i in range(0, len(dfArray)):
    df = dfArray[i]
    for c in df.columns:
        upper, lower = iqr_outliers(df, c, 1.5)
        df = df[(df[c] > lower) & (df[c] < upper)]
    dfArray[i] = df


# In[10]:


# Linear scalers
def scaleMinMax(df):
    from sklearn.preprocessing import MinMaxScaler
    scaler = MinMaxScaler()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

# Scale/Transform
```

```
for i in range(0, len(dfArray)):
    dfArray[i] = scaleMinMax(dfArray[i])


# In[11]:


import statistics
nArray = [len(x) for x in dfArray]
print("mean:  " + str(statistics.mean(nArray)))
print("stdev:  " + str(statistics.stdev(nArray)))


# In[12]:


sns.boxplot(nArray)


# ## Cluster Analysis

# In[13]:


def clusterKMeans(df, N, dataSetLabel, lineType, markerType, ax):
    kmeans_kwargs = {
        "init":"k-means++",
        "n_init": 1,
        "max_iter": 500,
        "random_state": 0
    }

    sse = []
    for k in range(1,N):
        kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
        kmeans.fit(df.drop(columns='final_grade'))
        sse.append(kmeans.inertia_)

    ax.plot(range(1,N), sse, label=dataSetLabel, linewidth=2,
linestyle=lineType, marker=markerType)
    ax.legend()
    ax.set_xticks(range(1,N))

    k1 = KneeLocator(range(1,N), sse, curve="convex", direction="decreasing")
    print("Cluster size:  " + str(k1.elbow))
    kmeans = KMeans(n_clusters=k1.elbow, **kmeans_kwargs)
    kmeans.fit(df.drop(columns='final_grade'))
    return df.assign(cluster=kmeans.labels_)

plt.style.use("fivethirtyeight")
fig, ax = plt.subplots()
ax.figure.set_figwidth(8)
ax.figure.set_figheight(6)
ax.set_xlabel("Clusters (K)")
ax.set_ylabel("SSE")
ax.set_title('K-Means SSE (Cumulative Weeks 1-15)')

import itertools
lineTypes = itertools.cycle(['solid', 'dotted', 'dashed', 'dashdot'])
markerTypes = itertools.cycle(['o','v','^','<','>','s','P','*','X','D'])

for i in range(0, len(dfArray)):
    dfArray[i] = clusterKMeans(dfArray[i], 21, 'wk ' + str(i+1),
next(lineTypes), next(markerTypes), ax)


# In[14]:


for i in range(0, len(dfArray)):
    print("wk " + str(i+1) + " cluster size: " +
str(dfArray[i].cluster.nunique()))


# In[15]:


for i in range(0,len(dfArray)):
    print(str(i) + " : " + str(dfArray[i].groupby(['cluster']).mean()))


# In[16]:


std = dfArray[0].groupby(['cluster']).std()
std


# In[17]:
```

```
# global variables for performance comparisons
f1Array = []
precisionArray = []
recallArray = []
f1Labels = []


# In[18]:


def evaluateLR(df):
    from sklearn.linear_model import LogisticRegression
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    model = LogisticRegression(random_state=0)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateLR(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Logistic Regression, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Logistic Regression')

# 7.6s


# In[19]:


def evaluateDT(df):
    from sklearn import tree
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    model = tree.DecisionTreeClassifier()
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateDT(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
```

```
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Decision Tree, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('DecisionTree')

# 1.4s


# In[20]:


from datetime import datetime
dt = datetime.now()
print(dt)


# In[21]:


def evaluateEllynGP(df, folder):
    from ellyn import ellyn
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    model = ellyn(
        g = 100, classification = True, class_m4gp = True,
op_list=['n','v','+','-','*','/','<','>','<=','>='],
         # ================ Results and printing
         resultspath= './' + folder,
         #savename
         savename="gp",
         #print initial population
         print_init_pop = True,
         #print last population
         print_last_pop = True,
         #print best individual at end
         print_best_ind = True)
    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    y_pred = cross_val_predict(model, x, y, cv=10)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateEllynGP(df, 'wk_' + "{:02d}".format(i+1))
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Genetic Programming, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Genetic Programming')

# 178m 45.5s


# In[22]:


def evaluateKNN(df):
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    model = KNeighborsClassifier(n_neighbors=3)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateKNN(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (K-Nearest Neighbors, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('K-Nearest Neighbors')

# 2.2s
# In[23]:


def evaluateNB(x,y):
    from sklearn.naive_bayes import GaussianNB
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    model = GaussianNB()

    y_pred = cross_val_predict(model, x, y, cv=10)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    p, r, f, s = evaluateNB(x,y)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,len(precision)+1)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Naive Bayes, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Naive Bayes')
```

```
# 0.7s


# In[24]:


def evaluateSVM(df):
  from sklearn import svm
  from sklearn.model_selection import cross_val_predict
  from sklearn.metrics import precision_recall_fscore_support

  model = svm.SVC()

  x = df.drop(columns=['cluster', 'final_grade']).values
  y = df['cluster'].values
  y_pred = cross_val_predict(model, x, y, cv=10, n_jobs=15)
  precision, recall, f1, support = precision_recall_fscore_support(y, y_pred,
average='macro')
  return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
  df = dfArray[i]
  p, r, f, s = evaluateSVM(df)
  precision.append(p)
  recall.append(r)
  f1.append(f)
  support.append(s)
  clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Support Vector Machine, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Support Vector Machine')

# 3.9s


# In[25]:


def evaluateMLP(df, hlz):
    from sklearn.neural_network import MLPClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    model = MLPClassifier(solver='lbfgs', hidden_layer_sizes=hlz,
random_state=0, max_iter=10000)
    x = df.drop(columns=['cluster', 'final_grade']).values
    y = df['cluster'].values
    y_pred = cross_val_predict(model, x, y, cv=10, n_jobs=15)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateMLP(df, (10,10))
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Multi-layer Perceptron, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Multi-layer Perceptron')

# 44.4s


# In[26]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('f1-Score')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(f1Array[j])+1), f1Array[j], marker[j], ms=10,
label=i, linewidth=3)
    j += 1

plt.legend(loc='lower right')


# In[27]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('Precision')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
```

```
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(precisionArray[j])+1), precisionArray[j], marker[j],
ms=10, label=i, linewidth=3)
    j += 1

plt.legend(loc='lower right')


# In[28]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('Recall')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(recallArray[j])+1), recallArray[j], marker[j], ms=10,
label=i, linewidth=3)
    j += 1

plt.legend(loc='lower right')


# In[29]:


meanF1 = []
meanPrecision = []
meanRecall = []
for i in f1Array:
    meanF1.append(np.average(i))

for i in precisionArray:
    meanPrecision.append(np.average(i))

for i in recallArray:
    meanRecall.append(np.average(i))

results = {
    'Mean f1-Score':meanF1,
    'Mean Precision':meanPrecision,
    'Mean Recall':meanRecall,
    'Model':f1Labels
}

perfDF = pd.DataFrame(results)

perfDF.plot.bar(x='Model', ylim=[.94,1], title='Mean Performance Scores',
ylabel='Score')
plt.legend(loc='lower right')


# In[30]:


meanF1
```

Code for Binary Classification (RQ4)

/src/cluster_analysis_cumulative_All_BinaryClassification.py

```
#!/usr/bin/env python
# coding: utf-8

# # Cluster Analysis

# ## Imports

# In[1]:


# Datasets
import pandas as pd
import numpy as np

# sklearn for clustering analysis
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
from sklearn.datasets import make_blobs
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import RobustScaler
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import MinMaxScaler

# scipy
from scipy.stats import pearsonr
from scipy.io import arff   # for working with Weka

# Kneed imports
from kneed import KneeLocator

# Data Visualizations
import matplotlib.pyplot as plt
import matplotlib as mpl
from matplotlib import colors
import seaborn as sns


# In[2]:
```

```
rCuffOff = 0.6
alpha = 0.05


# ## Import Data

# In[7]:


fname = 'cumulative_act'
df = pd.read_csv("../data/act01.csv.gz")
dfArray = []
dfArray.append(df)
for i in range(2,16):
    df = pd.read_csv("../data/" + fname + "{:02d}".format(i) + '.csv.gz')
    dfArray.append(df)


# In[8]:


dfArray


# ## Feature Exclusion
# - Exclude irrelevant features
# - Exclude features with p > .05
# - Exclude redundant features
# - Exclude outliers

# In[9]:


# exclude irrelevant features
for i in range(0,len(dfArray)):
    dfArray[i] = dfArray[i].drop(columns=['course','student'])


# In[10]:


dfArray


# In[11]:
```

```
# exclude insignificantly correlated features
for j in range(0, len(dfArray)):
    pvalue = dfArray[j].corr(method=lambda x, y: pearsonr(x, y)[1]) -
np.eye(len(dfArray[j].columns))
    alphaMax = alpha
    col = [i for i in pvalue.columns if pvalue[i].final_grade > alphaMax]
    dfArray[j] = dfArray[j].drop(columns=col)
    print("Excluding features in Dataset " + str(j) + ": " + str(col))


# In[12]:


# Exclude highly correlated and missing values
def excludeHighlyCorrelatedAndMissingValues(df, r):
    temp = []
    corr = abs(df.corr(method='spearman'))
    for y in corr.columns:
        for x in corr.index.values:
            if(x != y):
                c = corr[y][x]
                if(c > r):
                    if [y,x] not in temp:
                        temp.append([x,y])
                        print("Highly correlated pair found: " + str([x,y]))

    if(len(temp) > 0):
        drop = []
        for i in temp:
            a = i[0]
            b = i[1]
            dA = df.drop(columns=a).dropna()
            dB = df.drop(columns=b).dropna()
            print("Size after dropping " + a + ": " + str(dA.size))
            print("Size after dropping " + b + ": " + str(dB.size))
            if(dA.size > dB.size):
                print("\tDropping " + a)
                drop.append(a)
            else:
                print("\tDropping " + b)
                drop.append(b)
        for i in drop:
            df = df.drop(columns=i)
        return df.dropna()
    else:
        return df.dropna()

corrMax = rCuffOff
for i in range(0, len(dfArray)):
    print("Analyzing correlation in Dataset " + str(i) + " for exclusion.")
    dfArray[i] = excludeHighlyCorrelatedAndMissingValues(dfArray[i], corrMax)


# In[13]:


# Exclude outliers
def iqr_outliers(df, feature, factor):
    Q1= df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    upper_limit = Q3 + factor * IQR
    lower_limit = Q1 - factor * IQR
    return upper_limit, lower_limit

def std_outliers(df, feature, factor):
    upper_limit = df[feature].mean() + factor * df[feature].std()
    lower_limit = df[feature].mean() - factor * df[feature].std()
    return upper_limit, lower_limit

for i in range(0, len(dfArray)):
    df = dfArray[i]
    for c in df.columns:
        upper, lower = iqr_outliers(df, c, 1.5)
        df = df[(df[c] > lower) & (df[c] < upper)]
    dfArray[i] = df


# In[14]:


# Linear scalers
def scaleMinMax(df):
    from sklearn.preprocessing import MinMaxScaler
    scaler = MinMaxScaler()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

# Scale/Transform
for i in range(0, len(dfArray)):
    dfArray[i] = scaleMinMax(dfArray[i])
```

```
# In[15]:


import statistics
nArray = [len(x) for x in dfArray]
print("mean: " + str(statistics.mean(nArray)))
print("stdev: " + str(statistics.stdev(nArray)))


# In[16]:


sns.boxplot(nArray)


# ## Cluster Analysis

# In[17]:


def clusterKMeans(df, N, dataSetLabel, lineType, markerType, ax):
    kmeans_kwargs = {
        "init":"k-means++",
        "n_init": 1,
        "max_iter": 500,
        "random_state": 0
    }

    sse = []
    for k in range(1,N):
        kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
        kmeans.fit(df.drop(columns='final_grade'))
        sse.append(kmeans.inertia_)

    ax.plot(range(1,N), sse, label=dataSetLabel, linewidth=2,
linestyle=lineType, marker=markerType)
    ax.legend()
    ax.set_xticks(range(1,N))

    k1 = KneeLocator(range(1,N), sse, curve="convex", direction="decreasing")
    print("Cluster size: " + str(k1.elbow))

    kmeans = KMeans(n_clusters=k1.elbow, **kmeans_kwargs)
    kmeans.fit(df.drop(columns='final_grade'))
    return df.assign(cluster=kmeans.labels_)
```

```
plt.style.use("fivethirtyeight")
fig, ax = plt.subplots()
ax.figure.set_figwidth(8)
ax.figure.set_figheight(6)
ax.set_xlabel("Clusters (K)")
ax.set_ylabel("SSE")
ax.set_title('K-Means SSE (Cumulative Weeks 1-15)')

import itertools
lineTypes = itertools.cycle(['solid', 'dotted', 'dashed', 'dashdot'])
markerTypes = itertools.cycle(['o','v','^','<','>','s','P','*','X','D'])

for i in range(0, len(dfArray)):
    dfArray[i] = clusterKMeans(dfArray[i], 21, 'wk ' + str(i+1),
next(lineTypes), next(markerTypes), ax)


# In[18]:


for i in range(0, len(dfArray)):
    print("wk " + str(i+1) + " cluster size: " +
str(dfArray[i].cluster.nunique()))


# In[19]:


for i in range(0,len(dfArray)):
    print(str(i) + " : " + str(dfArray[i].groupby(['cluster']).mean()))


# In[20]:


std = dfArray[0].groupby(['cluster']).std()
std


# In[21]:


def getAtRiskBooleans(df, cutoff):
    # get boolean array where at-risk students are defined as final grade is
less than cutoff value
```

```
    temp =
pd.DataFrame(df.get(['cluster','final_grade'])).groupby(['cluster'])
    at_risk_groups = []
    for i in range(0,len(temp)):
        mean = temp.get_group(i).mean()['final_grade']
        if(mean < cutoff):
            at_risk_groups.append(i)

    y = np.zeros(len(df), dtype=bool)
    for i in at_risk_groups:
        y = y | (df['cluster'] == i)

    return y


# In[22]:


def evaluateNB(df):
    from sklearn.naive_bayes import GaussianNB
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    model = GaussianNB()
    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    y_pred = cross_val_predict(model, x, y, cv=10)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]

    p, r, f, s = evaluateNB(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())
```

```
x = range(1,len(precision)+1)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Naive Bayes, Cumulative Data)')
plt.legend(loc='lower right')
f1Array = []
precisionArray = []
recallArray = []
f1Labels = []
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Naive Bayes')


# In[23]:


f1


# In[24]:


def evaluateKNN(df):
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    model = KNeighborsClassifier(n_neighbors=3)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    return precision, recall, f1, len(y)

precision = []
```

```
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateKNN(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (K-Nearest Neighbors, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('K-Nearest Neighbors')


# In[25]:


def evaluateDT(df):
    from sklearn import tree
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    model = tree.DecisionTreeClassifier()
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
```

```
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateDT(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Decision Tree, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('DecisionTree')


# In[26]:


def evaluateLR(df):
    from sklearn.linear_model import LogisticRegression
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support

    x = df.drop(columns=['cluster', 'final_grade']).values
    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
    model = LogisticRegression(random_state=0)
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True, n_jobs=2)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateLR(df)
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Logistic Regression, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Logistic Regression')


# In[27]:


def evaluateSVM(df):
  from sklearn import svm
  from sklearn.model_selection import cross_val_predict
  from sklearn.metrics import precision_recall_fscore_support

  model = svm.SVC()
  x = df.drop(columns=['cluster', 'final_grade']).values
  y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))
  y_pred = cross_val_predict(model, x, y, cv=10, n_jobs=15)
  precision, recall, f1, support = precision_recall_fscore_support(y, y_pred,
average='macro')
  return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
  df = dfArray[i]
  p, r, f, s = evaluateSVM(df)
  precision.append(p)
  recall.append(r)
  f1.append(f)
  support.append(s)
  clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Support Vector Machine, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Support Vector Machine')


# In[28]:


def evaluateMLP(df, hlz):
    from sklearn.neural_network import MLPClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import precision_recall_fscore_support
    model = MLPClassifier(solver='lbfgs', hidden_layer_sizes=hlz,
random_state=0, max_iter=10000)
    x = df.drop(columns=['cluster', 'final_grade']).values

    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))

    #y = df['cluster'].values
    y_pred = cross_val_predict(model, x, y, cv=10, n_jobs=15)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')
    return precision, recall, f1, len(y)

precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    p, r, f, s = evaluateMLP(df, (10,10))
    precision.append(p)
    recall.append(r)
    f1.append(f)
    support.append(s)
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Multi-layer Perceptron, Cumulative Data)')
plt.legend(loc='lower right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Multi-layer Perceptron')


# In[29]:
```

```
def evaluateGP(df):
    from gplearn.genetic import SymbolicClassifier
    from sklearn.model_selection import cross_val_predict
    from sklearn.metrics import classification_report
    from sklearn.metrics import precision_recall_fscore_support
    from sklearn import tree
    import statistics
    import graphviz
    x = df.drop(columns=['cluster', 'final_grade'])

    y = getAtRiskBooleans(df, statistics.mean(df['final_grade']))

    feature_names = df.drop(columns=['cluster', 'final_grade']).columns
    gp = SymbolicClassifier(parsimony_coefficient=.01,
                            feature_names=feature_names,
                            random_state=1, verbose=True)
    gp.fit(x,y)
    y_pred = cross_val_predict(gp, x, y, cv=2, verbose=True, n_jobs=15)

    dot_data = gp._program.export_graphviz()
    graph = graphviz.Source(dot_data)

    report = precision_recall_fscore_support(y, y_pred, average='macro')

    return gp, y_pred, report, graph

gp, y_pred, report, g = evaluateGP(df)
print(report)
print(gp._program)
g



# In[30]:


precision = []
recall = []
f1 = []
support = []
clusters = []
for i in range(0, len(dfArray)):
    df = dfArray[i]
    gp, y_pred, report, graph = evaluateGP(df)
    precision.append(report[0])
    recall.append(report[1])
    f1.append(report[2])
    support.append(report[3])
    clusters.append(df.cluster.nunique())

x = range(1,16)
mp = 'o-'
mr = 'v--'
mf = 's:'
plt.plot(x, precision, mp, label='precision', ms=8, linewidth=3)
plt.plot(x, recall, mr, label='recall', ms=8, linewidth=3)
plt.plot(x, f1, mf, label='f1', ms=8, linewidth=3)
plt.xlabel('Week')
plt.ylabel('Score')
plt.title('Prediction Performance (Genetic Programming, Cumulative Data)')
plt.legend(loc='upper right')
f1Array.append(f1)
precisionArray.append(precision)
recallArray.append(recall)
f1Labels.append('Genetic Programming')


# In[31]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('f1-Score')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(f1Array[j])+1), f1Array[j], marker[j], ms=10,
label=i, linewidth=3)
    j += 1

plt.ylim([.8,1])
plt.legend(loc='lower right')
```

```
# In[32]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('Precision')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(precisionArray[j])+1), precisionArray[j], marker[j],
ms=10, label=i, linewidth=3)
    j += 1

plt.ylim([.8,1])
plt.legend(loc='lower right')


# In[33]:


fig, ax = plt.subplots()
marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
ax.set_title('Recall')
ax.set_ylabel('Score')
ax.set_xlabel('Week')
ax.figure.set_figwidth(8)
ax.figure.set_figheight(5)

j = 0
cols = f1Labels
for i in cols:
    ax.plot(range(1, len(recallArray[j])+1), recallArray[j], marker[j], ms=10,
label=i, linewidth=3)
    j += 1

plt.ylim([.8,1])
plt.legend(loc='lower right')


# In[34]:
```

```
meanF1 = []
meanPrecision = []
meanRecall = []
for i in f1Array:
    meanF1.append(np.average(i))

for i in precisionArray:
    meanPrecision.append(np.average(i))

for i in recallArray:
    meanRecall.append(np.average(i))

results = {
    'Mean f1-Score':meanF1,
    'Mean Precision':meanPrecision,
    'Mean Recall':meanRecall,
    'Model':f1Labels
}

perfDF = pd.DataFrame(results)

perfDF.plot.bar(x='Model', ylim=[.85,1], title='Mean Performance Scores',
ylabel='Score')

plt.legend(loc='lower left')


# In[35]:


statistics.mean(df['final_grade'])
```

Helper Code for Other Scripts

/src/helper.py

```python
# Imports
from datetime import date, timedelta
from email.utils import format_datetime
from gplearn.genetic import SymbolicClassifier
from math import nan
from matplotlib import cm
from matplotlib import colors
from matplotlib.colors import Colormap
from matplotlib.colors import ListedColormap
from pandas import DataFrame
from random import seed
from scipy.stats import pearsonr
from scipy.stats import spearmanr
from sklearn import metrics
from sklearn import tree
from sklearn.metrics import classification_report
from sklearn.metrics import precision_recall_fscore_support
from sklearn.model_selection import cross_val_predict
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import PowerTransformer
from sklearn.preprocessing import QuantileTransformer
from sklearn.preprocessing import RobustScaler
from sklearn.tree import export_text
from unittest import skip
from sklearn.cluster import KMeans
import colorsys
import graphviz
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import statistics
from kneed import KneeLocator

def print_samples_info(df:DataFrame):
    print("Samples: ", df.shape[0])
    print("unique courses: ", df['course'].nunique())
    print("unique students: ", df['student'].nunique())
    print("unique course/student pairs: ",
len(df[['course','student']].drop_duplicates()))

def get_outlier_limits(df:DataFrame, feature:str):
    Q1= df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    upper_limit = Q3 + 1.5 * IQR
    lower_limit = Q1 - 1.5 * IQR
    return upper_limit, lower_limit

def add_enrollment_timedelta(df:DataFrame, startDate:date, endDate:date) -
> DataFrame:
    """Adds enrollment_timedelta feature that equals the number of days
the students registered before the last day of registration.
       Note that this performs an inner join, which effectively excludes
samples, which did not enroll between the start and end date."""
    dir = "../data/"
    ed =           pd.read_csv(dir + "data-
1624239985814_course_user.csv.gz", dtype={'pk1':int, 'crs_main_pk1':int,
'data_src_pk1':int, 'role':str, 'users_pk1':int, 'enrollment_date':str})
    ed = pd.concat([ed,pd.read_csv(dir + "data-
1624240128001_course_user.csv.gz", dtype={'pk1':int, 'crs_main_pk1':int,
'data_src_pk1':int, 'role':str, 'users_pk1':int, 'enrollment_date':str})])
    ed = pd.concat([ed,pd.read_csv(dir + "data-
1624240236386_course_user.csv.gz", dtype={'pk1':int, 'crs_main_pk1':int,
'data_src_pk1':int, 'role':str, 'users_pk1':int, 'enrollment_date':str})])
    ed.columns =
['pk1','course','data_src','role','student','enrollment_date']
    ed['enrollment_date'] = pd.to_datetime(ed['enrollment_date']).dt.date
    ed = ed[  (ed['enrollment_date'] >
startDate)  &  (ed['enrollment_date'] <
endDate.__add__(timedelta(days=1)))  ]
    ed['enrollment_timedelta'] = (ed['enrollment_date'] -
startDate).dt.days.astype(int)
    return df.merge(ed[['course','student','enrollment_timedelta']],
on=['course','student'], how='inner')

def add_missing_final_grade(df:DataFrame) -> DataFrame:
    """Adds missing_final_grade feature column where 1 = missing and 0 =
not missing."""
    df = df.assign(missing_final_grade=[1 if fg == True else 0 for fg in
df['final_grade'].isnull()])
    return df
```

```
def add_courses_enrolled(df:DataFrame) -> DataFrame:
    """Adds courses_enrolled feature that equals the number of concurrent
courses enrolled by the associated student."""
    courses_enrolled =
df.groupby('student').count()['course'].rename("courses_enrolled")
    return df.join(courses_enrolled,on='student')

def add_class_size(df:DataFrame) -> DataFrame:
    """Adds class_size feature that equals the number of students enrolled
in the associated course."""
    class_size =
df.groupby('course').count()['student'].rename("class_size")
    return df.join(class_size,on='course')

def get_at_risk_level(fg:float) -> int:
    """Returns at_risk_level as an integer"""
    arl = 0
    if(fg >= 0.9):
        arl = 1 # very_low
    elif(fg >= 0.8):
        arl = 2 # low
    elif(fg >= 0.7):
        arl = 3 # medium
    elif(fg >= 0.6):from datetime import date, timedelta

def get_at_risk_booleans(df, cutoff):
    at_risk_groups = []
    for i in df['cluster'].unique(): # for each numbered cluster
        mean = statistics.mean(df.query("cluster == " +
str(i))['final_grade']) # compute the mean final_grade
        if(mean < cutoff): # if it's less than the cutoff
            at_risk_groups.append(i) # add it to the at-risk group

    # create a new array and assign at-risk == true for those in the at-
risk group
    y = np.zeros(len(df), dtype=bool)
    for i in at_risk_groups:
        y = y | (df['cluster'] == i)

    return y

def add_at_risk_levels(df:DataFrame) -> DataFrame:
    """Adds at-risk levels based on final grades."""
    df = df.assign(at_risk_level=[get_at_risk_level(fg) for fg in
df['final_grade'].values])
```

```
    return df

def pad_zeroes(n:int, i:int) -> str:
    """Pads n zeros to i and returns the result."""
    return ("{:0" + str(n) + "d}").format(i)

def get_cmap_r(ratio:float) -> Colormap:
    """Creates a reverse gray scale color map from 0 to 1 with pink
highlighting lower range of the scale from 0 to ratio."""
    binary = cm.get_cmap('binary_r',100)
    newcolors = binary(np.linspace(0,1,100))
    pink = np.array([96/100, 9/100, 58/100, 1])
    newcolors[:int(ratio*100), :] = pink
    newcmp = ListedColormap(newcolors)
    return newcmp

def get_cmap(ratio:float) -> Colormap:
    """Creates a gray scale color map from 0 to 1 with pink highlighting
the upper range of the scale from ratio to 1."""
    binary = cm.get_cmap('binary',100)
    newcolors = binary(np.linspace(0,1,100))
    pink = np.array([96/100, 9/100, 58/100, 1])
    newcolors[int(ratio*100):, :] = pink
    newcmp = ListedColormap(newcolors)
    return newcmp

def get_mean_random_samples(df:DataFrame, N:int, K:int) -> DataFrame:
    """Obtains N mean samples of K random samples. When N and K are
'large' (greater than 30), a normally distribution of samples of the means
can be obtained regardless of underlying distribution shape."""
    dfsMeans = pd.DataFrame(columns=df.columns)
    for i in range(0,N):
        dfs = [x for x in df.sample(n=K).mean()]
        dfsMeans =
pd.concat([dfsMeans,pd.DataFrame(data=dfs,index=df.columns).transpose()],a
xis=0)
    dfsMeans.reindex()
    return dfsMeans

def plot_histogram(df:DataFrame,col:str,bins:int):
    sns.histplot(df[col], bins=bins).set_title(col + ' (N=' + str(len(df))
+ ")")

def plot_corr_matrix(df:DataFrame, R:float, L:int, W:int):
    corr = abs(df.corr(method='pearson'))
    mask = np.zeros_like(corr)
    mask[np.triu_indices_from(mask)] = True
    with sns.axes_style("white"):
        f, ax = plt.subplots(figsize=(W,L))
        plt.title(label="Correlation Matrix (Pearson's R)")
        ax = sns.heatmap(corr, mask=mask, square=True, linewidths=.01,
vmax=R, annot=True)
    return corr

def plot_pval_matrix(df:DataFrame, ALPHA:float, L:int, W:int):
    pvalue = df.corr(method=lambda x, y: pearsonr(x, y)[1]) -
np.eye(len(df.columns))
    mask = np.zeros_like(pvalue)
    mask[np.triu_indices_from(mask)] = True
    with sns.axes_style("white"):
        f, ax = plt.subplots(figsize=(W,L))
        ax.set_title("P-Value Matrix (Pearson's R)")
        ax = sns.heatmap(pvalue, mask=mask, square=True, linewidths=.01,
vmax=ALPHA, annot=True)
    return pvalue

# sns.histplot(data=df,
x='final_grade',hue='at_risk_level',bins=10,palette=sns.color_palette("bri
ght",6)).set_title('Final Grade (N=' + str(len(df)) + ")")

# Linear scalers
def scaleMinMax(df):
    scaler = MinMaxScaler()
    col = df.drop(columns=['course','student']).columns
    df[col] = scaler.fit_transform(df[col])
    return df

def scaleRobust(df):
    scaler = RobustScaler()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

def scaleNormalizer(df):
    scaler = Normalizer()
    return pd.DataFrame(scaler.fit_transform(df), columns=df.columns)

# Nonlinear transformers
def transformQuantile(df):
    qt = QuantileTransformer()
    return pd.DataFrame(qt.fit_transform(df), columns=df.columns)
```

```
def transformPower(df):
    pt = PowerTransformer()
    return pd.DataFrame(pt.fit_transform(df), columns=df.columns)

# Exclude outliers
def iqr_outliers(df, feature, factor):
    Q1= df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    upper_limit = Q3 + factor * IQR
    lower_limit = Q1 - factor * IQR
    return upper_limit, lower_limit

def std_outliers(df, feature, factor):
    upper_limit = df[feature].mean() + factor * df[feature].std()
    lower_limit = df[feature].mean() - factor * df[feature].std()
    return upper_limit, lower_limit

def remove_outliers(df:DataFrame) -> DataFrame:
    for c in df.columns:
        if ((c == 'course') or (c == 'student')):
            continue
        upper, lower = iqr_outliers(df, c, 1.5)
        df = df[(df[c] > lower) & (df[c] < upper)]
    return df

def remove_outliers_recursive(df:DataFrame) -> DataFrame:
    n = 0
    while(True):
        n = len(df)
        df = remove_outliers(df)
        if(n == len(df)):
            break
    return df

def plot_activity_summary(df:DataFrame, w:int):
    fig, ax1 = plt.subplots()
    ax1.set_xlabel('Feature')
    ax1.set_ylabel('Activity Frequency')
    ax1.set_title('Activity Summary (N=' + str(len(df)) + ')')
    ax1.figure.set_figwidth(w)
    temp = df.drop(columns=['final_grade'])
    ax1.boxplot(temp, labels=temp.columns)
    temp2 = df['final_grade']
    ax2 = ax1.twinx()
    ax2.boxplot(temp2, labels=['final_grade'],
positions=[len(temp.columns)+1])
    ax2.set_ylabel('Final Grade', color='purple')
    ax2.tick_params('y',colors='purple')

# Exclude highly correlated and missing values
def excludeHighlyCorrelatedAndMissingValues(df:DataFrame, r:float) ->
DataFrame:
    temp = []
    corr =
abs(df.drop(columns=['student','course']).corr(method='spearman'))
    for y in corr.columns:
        for x in corr.index.values:
            if(x != y):
                c = corr[y][x]
                if(c > r):
                    if [y,x] not in temp:
                        temp.append([x,y])
                        print("Highly correlated pair found: " +
str([x,y]))

    if(len(temp) > 0):
        drop = []
        for i in temp:
            a = i[0]
            b = i[1]
            dA = df.drop(columns=a).dropna()
            dB = df.drop(columns=b).dropna()
            print("Size after dropping " + a + ": " + str(dA.size))
            print("Size after dropping " + b + ": " + str(dB.size))
            if(dA.size > dB.size):
                print("\tDropping " + a)
                drop.append(a)
            else:
                print("\tDropping " + b)
                drop.append(b)
        for i in drop:
            df = df.drop(columns=i)
        return df.dropna()
    else:
        return df.dropna()

def evaluateGP(df:DataFrame, at_risk_cutoff:float):
    x = df.drop(columns=['cluster','final_grade'])
    y = get_at_risk_booleans(df, at_risk_cutoff)
    feature_names = x.columns
    gp = SymbolicClassifier(parsimony_coefficient=.01,
                            feature_names=feature_names,
                            random_state=1, verbose=True)
    gp.fit(x,y)
    y_pred = cross_val_predict(gp, x, y, cv=2, verbose=True, n_jobs=15)

    dot_data = gp._program.export_graphviz()
    graph = graphviz.Source(dot_data)

    return gp, y_pred, classification_report(y, y_pred), graph

def evaluateNB(df):
    x = df.drop(columns=['cluster']).values
    y = df['cluster'].values
    gnb = GaussianNB()
    y_pred = cross_val_predict(gnb, x, y, cv=10)
    return classification_report(y, y_pred)

def evaluateDT(df):
    x = df.drop(columns=['cluster']).values
    y = df['cluster'].values
    model = tree.DecisionTreeClassifier()
    y_pred = cross_val_predict(model, x, y, cv=10, verbose=True,
n_jobs=15)
    precision, recall, f1, support = precision_recall_fscore_support(y,
y_pred, average='macro')

    model.fit(x,y)
    tree.plot_tree(model)

    return precision, recall, f1, len(y)

def clusterKMeans(df):
    kmeans_kwargs = {
        "init":"k-means++",
        "n_init": 1,
        "max_iter": 500,
        "random_state": 0
    }

    sse = []
    N = 21
    for k in range(1,N):
        kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
        kmeans.fit(df.drop(columns=['course','student']))
        sse.append(kmeans.inertia_)
    k1 = KneeLocator(range(1,N), sse, curve="convex", 
direction="decreasing")

    plt.style.use("fivethirtyeight")
    plt.plot(range(1,N), sse, 'ko-', label='SSE (Week 1-15 Cumulative 
Activity)')
    plt.xticks(range(1,N))
    plt.xlabel("Clusters (K)")
    plt.ylabel("SSE")
    plt.title('K-Means SSE (Exploratory)')
    k = k1.elbow
    plt.plot([k], sse[k-1], 'X', ms=10, color='red', label='Elbow Point')
    plt.legend()
    plt.show()

    print("Cluster size: " + str(k1.elbow))

    kmeans = KMeans(n_clusters=k1.elbow, **kmeans_kwargs)

    kmeans.fit(df.drop(columns=['course','student']))
    return df.assign(cluster=kmeans.labels_)

def plot_cluster_summary(df:DataFrame):
    df = df.drop(columns=['course','student'])
    fig, ax1 = plt.subplots()
    marker = ['o-', 'v--', 'D-.', 's:', 'X-', 'P--', 'p-', '>:']
    ax1.set_title('K-Means Cluster Summary (Exploratory)')
    ax1.set_ylabel('Normalized Frequency')
    ax1.set_xlabel('Cluster')
    j = 0
    cols = df.drop(columns=['cluster','final_grade']).columns
    means = df.groupby(['cluster']).mean()
    for i in cols:
        ax1.plot(means[i], marker[j], ms=10, label=i)
        j += 1

    ax2 = ax1.twinx()
    ax2.plot(means['final_grade'], marker[j], color='purple', ms=10, 
label='final_grade')
    ax2.set_ylabel('Final Grade', color='purple')
    ax2.tick_params('y',colors='purple')
    ax2.grid(True,linestyle='--')
    # ask matplotlib for the plotted objects and their labels
    lines, labels = ax1.get_legend_handles_labels()
    lines2, labels2 = ax2.get_legend_handles_labels()
    plt.legend()
    ax2.legend(lines + lines2, labels + labels2, bbox_to_anchor=(1.15,
1.0), loc='upper left')
```

VITA

Ngoc Van P. Bui

OBJECTIVE

- Obtain challenging and rewarding opportunities to deliver student-centered and research-driven instruction
- Augment teaching and learning with open and innovative technologies rooted in theory and evidence-based research to improve learning performance
- Improve educational outcomes by promoting social learning and intrinsic value such as inspiration, motivation, and self-efficacy
- Continuously learn, grow, professionally develop, and advance research in the field of Instruction Systems Design and Technology

EDUCATION

Ed.D in Instructional Systems Design and Technology Aug 2022
Sam Houston State University. Huntsville, TX

Master of Education - Instructional Technology May 2015
Georgia College & State University. Milledgeville, GA

Bachelor of Arts - Computer Science Dec 2009
Mercer University. Macon, GA

SKILLSET

Instructional Methods	**Learning Technologies**
- Quantitative & Qualitative Research	- Web Design & Development
- Evidence-based Learning Principles	- Database Design & Administration
- Learning/Content Management Systems	- Software Programming Languages
- Computational Learning Analytics	- IBM SPSS Statistics
- Educational Data Mining	- Unsupervised/Supervised AI/ML

SERVICE

- Association for Education Communication Technology, Reviewer	Jan 2020 - Present
- Southwest Educational Research Association, Reviewer	Sep 2020 - Present
- Honor Society of Phi Kappa Phi, Member	Apr 2015 - Present
- EDUCAUSE, Member	Jan 2015 - Present

RESEARCH

Dissertation - Explainable AI (XAI): A Theory-based Genetic Programming Model for Predicting At-risk Students. 2021 - Present

- Performed literature review, educational data mining (EDM), and computational learning analytics researching Explainable AI (XAI) models for predicting and explaining at-risk students using Blackboard student performance and activity data.
- Explore the efficacy of a Theory-based Genetic Programming Model for improving and explaining at-risk student predictions using Blackboard student behavior and performance features.

Grant proposal - Technology sustainability and Explainable AI (XAI) research: The efficacy of a mobile learning assistant implementing theory-based genetic programming model for predicting at-risk learners 2020 - 2021

- Performed preliminary literature research and developed a plan and research design to investigate a XAI mobile learning assistant for improving student retention and learning outcomes.

CONFERENCE PAPERS/PRESENTATIONS

Bui, N. & Donggil, S. (2021). Theory-Guided XAI: Improving Performance and Explainability of At-Risk Student Predictions. AECT Nov 2021

- Performed educational data mining (EDM) on Blackboard LMS Activity data and implemented Theory-Guided XAI Models to explore at-risk student behaviors and prediction performance from a large online higher education 4-year institution.

Bui, N. (2021). Explainable Artificial Intelligence: A Theory-Guided Genetic Programming Model for At-Risk Student Prediction. SERA Feb 2021

- Performed initial literature research to design research for addressing student retention with Explainable AI (XAI).

Bui, N., Collier, J., Guled, A., & Song, D. (2020). Effectiveness of Conversational Virtual Agents in Learning Support: A Systematic Literature Review. AECT Nov 2020

- Principal lead researcher who planned, coordinated, and tracked work tasks that lead to the successful completion of research tasks.
- Used scripting language tools such as python to automate file management renaming and allocation tasks.
- Use spreadsheet tools for data aggregation and analysis.
- Performed data collection, reduction, synthesis, and charting for visualization and analysis.

PROFESSIONAL EXPERIENCE

Graduate Research Assistant — Apr-Oct 2021; Mar-Aug 2020

- Sam Houston State University Online
- Performed literature reviews, educational data mining (EDM), and learning analytics (LA) using a variety of programming and AI/ML tools (MySQL, Python, Jupyter notebooks, Weka, Seaborn, Scikit-Learn, Matplotlib, Ellyn GP, GPLearn).
- Performed unsupervised and supervised learning to identify, classify, and evaluate trends and patterns in student activity behaviors and their relationship to performance outcomes.

New Media Instructional Specialist Intern — Jan-Apr 2015

- Mercer University, Academic Technology Services Department
- Identified problems associated with the underutilization of technology resources in higher academia stemming from insufficient faculty training.
- Conducted training workshops and seminars to raise faculty awareness of the technology available at Mercer that was being underutilized.
- Trained faculty and students on the benefits and proper use of LMS and Media tools for instruction and learning.

University Instructional Technology Coordinator — Aug 2010 to Dec 2012

- Mercer University, Psychology Department
- Supported undergraduate studies with the development of research involving computational media
- Provided IT service support to students and faculty in troubleshooting computing systems
- Updated, maintained, and resolved technical issues for efficient operation of computer lab systems

Mercer on Mission Vietnam Coordinator — Summers of 2009 - 2013

- Mercer University, International Programs
- Coordinated, planned, and organized 5 years of international missionary trips for faculty and students.
- Designed, developed, and instructed a preliminary course in the Vietnamese language and culture.
- Designed, developed, and maintained an online website for tracking and content management of student journal entries while abroad.

IT Help Desk – Computer Tech — 2006 - 2009

- Mercer University, Computer Science & Engineering Departments
- Monitored and diagnosed computer lab issues as they arrive and provide technical support to students and faculty
- Obtained valuable interpersonal communication skills with people of diverse backgrounds (students, faculty, and IT service technicians)

DOSSIER

Please visit my Dossier site (https://sites.google.com/view/ngocvan-bui-dossier) for a comprehensive collection of research papers, presentations, and work samples, such as complete course designs using Google Docs, Moodle, and Blackboard LM