Navigating the Data-Rich Landscape of Online Learning: Insights and Predictions from ASSISTments

Aswani Yaramala *, Soheila Farokhi *, and Hamid Karimi
Data Science and Applications lab[†], Department of Computer Science, Utah State University
aswani.yaramala@usu.edu, soheila.farokhi@usu.edu, hamid.karimi@usu.edu

ABSTRACT
This paper presents an in-depth analysis of student behavior and score prediction in the ASSISTments online learning platform. We address four research questions related to the impact of tutoring materials, skill mastery, feature extraction, and graph representation learning. To investigate the impact of tutoring materials, we analyze the influence of students requesting hints and explanations on their performance in end-of-unit assignments. Our findings provide insights into the role of guidance in learning and inform the development of superior tutoring strategies. Additionally, we explore the correlation between mastery/non-mastery of specific skills during in-unit problems and performance in corresponding end-of-unit assignments, shedding light on the efficacy of standard-aligned curricula. In terms of feature extraction, we extract relevant features from extensive student activity data and determine their importance in predicting assignment grades. Furthermore, we employ graph representation learning techniques to model the complex relationships between different entities in the dataset. This yields a more nuanced modeling of factors influencing student performance and facilitates the development of more accurate predictive models. Overall, our study contributes to the practical application of data mining techniques in online learning contexts, with implications for personalized learning, interventions, and support mechanisms. The code is publicly available in https://github.com/DSAatUSU/EDMCup2023.

Keywords
Online Learning, Skill Mastery, Student Performance, Score Prediction, Machine Learning, Graph Representation Learning, Tutoring Materials, Common Core State Standards (CCSS)

[*]Co-first authors and equal contributions.
[†]https://dsa.cs.usu.edu/

A. Yaramala, S. Farokhi, and H. Karimi. Navigating the data-rich landscape of online learning: Insights and predictions from assistments. In B. Paaßen and C. D. Epp, editors, *Proceedings of the 17th International Conference on Educational Data Mining*, pages 321–331, Atlanta, Georgia, USA, July 2024. International Educational Data Mining Society.

https://doi.org/10.5281/zenodo.12729826

1. INTRODUCTION
Emerging technologies and evolving societal needs have propelled the digital transformation in education [1, 2, 3, 4, 5, 6, 7, 8]. A significant catalyst in this digital revolution is the rise of online learning platforms [9, 10]. Recent global events, such as the shift to remote learning, have further accelerated this digital transformation [11, 12]. These platforms aim to make quality education accessible to anyone, regardless of their location, thus democratizing education [13]. They also reach underserved communities and individuals with limited mobility, ensuring equal learning opportunities for all [14]. Online learning platforms generate abundant data, including detailed clickstream and assessment data. This data provides a unique opportunity to understand student behavior and enhance learning outcomes using data mining and machine learning techniques [15]. By predicting student performance and tailoring learning experiences, these platforms can create a student-centric educational environment. This paper focuses on two key areas within this data-rich landscape: analyzing student behavior and predicting exam grades. Understanding student behavior helps uncover patterns in engagement, motivation, and learning strategies [16, 17]. Predicting exam grades based on behavior data is crucial for identifying at-risk students and adjusting teaching strategies [16, 17]. In this study, we present our analysis, machine learning models, and results from the EDM CUP 2023 competition. The competition involved predicting students' scores on end-of-unit assignments using clickstream data and additional curriculum information. A detailed explanation of the dataset can be found on the website [18] (see Appendix A.1). We aim to answer the following research questions:

RQ1: How does requesting tutoring materials impact students' performance in end-of-unit assignments, and what are its implications for guidance and learning strategies? **RQ2:** What patterns exist in the correlation between mastery of specific skills during in-unit problems and performance in corresponding end-of-unit assignments, and how does this relate to educational standards? **RQ3:** How can we extract relevant features from extensive student activity data within the ASSISTments platform for predicting assignment grades, and what is the importance of these features? **RQ4:** How can graph representation learning be used to model complex relationships in the dataset, and can features extracted from these relationships improve assignment grade prediction?

Our contributions in this paper can be summarized as follows:

- We thoroughly investigate the impact of requesting tutoring materials on student performance in an online learning setting. Our quasi-experimental approach provides valuable insights into the causal effects of these elements on end-of-unit assignments.
- We establish a clear link between the mastery of specific Common Core State Standards (CCSS)-aligned skills during in-unit problems and performance on end-of-unit assignments. This contributes to a better understanding of the effectiveness of standard-aligned curricula.
- Extensive feature extraction from various dataset attributes helps us determine their importance in predicting grades. This approach enhances our ability to predict student performance and identify at-risk students early.
- We apply graph representation learning algorithms to model complex dataset relationships, offering a more nuanced understanding of factors influencing student performance.
- Our comprehensive approach not only addresses important research questions in online learning but also provides a methodological framework that can be applied in similar studies, contributing to the field of educational data mining.

2. STUDENT BEHAVIOR ANALYSIS AND ACADEMIC ACHIEVEMENT

To address **RQ1** and **RQ2**, we embark on a comprehensive analysis of student behaviors and their impact on academic outcomes. In Section 2.1, we explore the relationship between tutoring requests and the end-of-unit student performance, considering various tutoring alternatives. Following this, in Section 2.2, we employ association rule mining to extract meaningful patterns (rules). These rules serve to illuminate the relationship between two key concepts: a) the mastery or non-mastery of CCSS-related skills during in-unit assignments and b) the mastery or non-mastery of CCSS-related skills during end-of-unit problems.

2.1 Tutoring Request and Student Performance

When students engage in in-unit assignments, they have the option to request tutoring if it is available for the specific problem they are working on. The available core tutoring options include *hints*, *explanations*, and *answers*. Additionally, there are two auxiliary tutoring options including *skill-related videos*, and *live tutor*. We aim to assess the effectiveness of these tutoring options in improving student performance in end-of-unit problems. To this end, we calculated the percentage of each tutoring option requested for each group of end-of-unit assignment and corresponding problem, which are associated with multiple in-unit problems. The percentage of tutoring requested is determined by dividing the number of in-unit problems for which students requested a particular tutoring option by the total number of in-unit problems where that specific tutoring option was available. This is expressed by the following formula:

$$\text{\% of tutoring option requested} = \frac{\text{\# of in-unit problems with tutoring option requested}}{\text{Total \# of in-unit problems with that tutoring option available}} \times 100 \quad (1)$$

Table 1 presents the results of our examination of the association between students' end-of-unit performance and their requests for the five distinct tutoring options available, as demonstrated in parts (a) through (e). Part (f) of the table aggregates the results for all tutoring options. Additionally, we conducted a grade-specific analysis by segmenting the data based on the first part of the problem skill code, which corresponds to the grade as per the CCSS guidelines. This approach facilitated the computation of the proportion of tutoring requests for each grade and tutoring option independently. However, not all tutoring options were accessible for every grade or had sufficient data to enable experimentation, leading to variations across tables. The second and third columns show the average score and sample size (**SS**) of end-of-unit problems where students requested tutoring more than 70% of the time and less than 30% of the time, respectively. These thresholds were selected to create a clear distinction, facilitating a robust assessment of tutoring's impact. While a binary categorization of 100% or 0% may seem appealing, the infrequency of such extreme cases makes it impractical for addressing **RQ1**. The fourth column, labeled SD, showcases the score difference between students with high tutoring requests (≥ 0.7) and those with low requests (< 0.3). For example, for grade 1 with hint requested, **SD** = **HR** (≥ 0.7) **Score** - **HR** (< 0.3) **Score** = -0.21. This difference helps to evaluate whether requesting a tutoring option frequently within the unit yields a better grade on the end-of-unit assignments (**SD** ≥ 0) or not (**SD** < 0). The fifth column showcases the t-statistic, derived via a Student's t-test. As a measure used in hypothesis testing, the t-statistic determines the likelihood of the observed difference between sample means having occurred by chance, thereby assisting in establishing statistical significance [19]. The final columns exhibit the p-value. The outcomes of these analyses are illustrated in Table 1. We make the following observations based on these results.

- In general, students who requested help less often (< 0.3) performed better than those who requested more often (≥ 0.7). This is observed in all grades for HR, AR, ER, and TTR.
- In most instances across all tutoring options, the t-statistic is significant (p-value < 0.05), indicating a notable difference in means between the groups of students who request help more frequently and those who request it less often. However, for grade 8 in Live Tutor Requested (LTR) and for all grades except grade 7 in Skill Video Requested (SVR), the disparity in means between these two groups is not statistically significant ($p > 0.05$), making it difficult to draw conclusions about these particular groups.
- The largest difference in performance (SD = 0.63) between the two groups is observed in Hint Requested (HR) for grade HSF.
- In the case of Hint Requested (HR), there is a reverse trend in HSG, where those who requested more often

Table 1: The experimental results of investigating the impact of tutoring requests on students' performance

(a) % Hint Requested (**HR**)

Grade	**HR** (≥ 0.7)		**HR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
1	0.65	23	0.86	8438	-0.21	-2.89	$3.93e-3$
2	0.41	49	0.71	97905	-0.30	-4.63	$3.65e-6$
3	0.21	189	0.60	44480	-0.39	-11.00	$0.00e+0$
4	0.42	823	0.53	44879	-0.11	-6.52	$6.93e-11$
5	0.40	261	0.49	60968	-0.09	-2.99	$2.79e-3$
6	0.34	3936	0.59	57721	-0.25	-30.53	$0.00e+0$
7	0.38	5973	0.56	20363	-0.18	-25.03	$0.00e+0$
8	0.45	1604	0.54	15177	-0.09	-7.10	$1.27e-12$
HSA	0.48	31	0.71	869	-0.23	-2.78	$5.52e-3$
HSG	1.00	15	0.79	182	0.21	2.01	$4.56e-2$
HSF	0.00	2	0.63	222	-0.63	-1.84	$6.72e-2$
HSN	0.33	15	0.61	104	-0.28	-2.09	$3.88e-2$
HSS	0.25	4	0.60	244	-0.35	-1.41	$1.61e-1$
All grades	0.38	13122	0.61	388096	-0.22	-51.93	0.00

(b) % Answer Requested (**AR**)

Grade	**AR** (≥ 0.7)		**AR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
1	0.76	1018	0.78	14731	-0.01	-0.84	$3.99e-1$
2	0.60	34310	0.77	39427	-0.17	-50.74	$0.00e+0$
3	0.45	15898	0.67	28104	-0.22	-45.56	$0.00e+0$
4	0.40	18127	0.67	23163	-0.27	-56.94	$0.00e+0$
5	0.32	19129	0.61	46255	-0.29	-69.48	$0.00e+0$
6	0.48	34011	0.67	20481	-0.19	-43.93	$0.00e+0$
7	0.45	19619	0.66	7342	-0.21	-31.66	$5.16e-216$
8	0.41	9015	0.61	7858	-0.20	-26.69	$9.19e-154$
HSA	0.36	1190	0.69	2973	-0.33	-20.55	$1.93e-89$
HSF	0.37	1628	0.69	5613	-0.32	-23.96	$4.04e-122$
HSG	0.34	1402	0.61	5125	-0.27	-18.82	$5.17e-77$
HSN	0.40	139	0.64	624	-0.24	-5.39	$9.56e-8$
HSS	0.35	526	0.49	2147	-0.14	-5.71	$1.25e-8$
All grades	0.47	168651	0.69	230979	-0.22	-145.87	0.00

(c) % Explanation Requested (**ER**)

Grade	**ER** (≥ 0.7)		**ER** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
5	0.17	24	0.41	323	-0.24	-2.35	$1.91e-2$
6	0.27	988	0.64	20069	-0.37	-23.92	$8.92e-125$
7	0.33	943	0.58	16286	-0.25	-15.36	$6.47e-53$
8	0.32	422	0.55	5840	-0.23	-9.25	$2.93e-20$
All grades	0.30	2387	0.61	43168	-0.31	-29.84	$9.47e-194$

(d) % Live Tutor Requested (**LTR**)

Grade	**LTR** (≥ 0.7)		**LTR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
6	0.00	1	0.54	52018	-0.54	-	-
7	0.86	36	0.53	22684	0.33	4.01	$6.13e-5$
8	0.80	5	0.54	9459	0.26	1.16	0.25
HSF	0.89	99	0.65	248	0.24	4.67	$4.36e-6$
All grades	0.87	141	0.54	206946	0.33	7.83	$5.03e-15$

(e) % Skill Video Requested (**SVR**)

Grade	**SVR** (≥ 0.7)		**SVR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
2	0.83	18	0.77	7487	0.07	0.68	0.50
4	0.64	33	0.58	25560	0.05	0.63	0.53
5	0.43	77	0.50	75651	-0.07	-1.31	0.19
6	0.63	38	0.54	51835	0.09	1.11	0.27
7	0.12	26	0.53	22599	-0.41	-4.22	$2.5e-5$
8	0.50	20	0.54	9237	-0.04	-0.35	0.73
HSF	0.75	8	0.64	210	0.11	0.62	0.54
All grades	0.52	241	0.54	205939	-0.03	-0.79	0.43

(f) % Total Tutoring Requested (**TTR**)

Grade	**TTR** (≥ 0.7)		**TTR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
1	0.39	176	0.80	18200	-0.40	-13.18	$1.67e-39$
2	0.32	2088	0.74	90748	-0.42	-43.45	0.00
3	0.25	3271	0.65	44481	-0.40	-46.45	0.00
4	0.27	4228	0.62	37544	-0.35	-44.22	0.00
5	0.24	8036	0.59	58846	-0.35	-59.95	0.00
6	0.33	8646	0.64	48147	-0.31	-56.11	0.00
7	0.32	6190	0.63	19655	-0.31	-44.64	0.00
8	0.34	3557	0.60	13268	-0.25	-27.57	$1.25e-163$
HSA	0.31	1068	0.69	3148	-0.38	-22.99	$2.84e-110$
HSF	0.37	1591	0.68	5629	-0.32	-23.92	$9.15e-122$
HSG	0.34	1387	0.62	5283	-0.28	-19.52	$1.39e-82$
HSN	0.27	85	0.63	660	-0.36	-6.58	$8.89e-11$
HSS	0.30	395	0.50	2301	-0.20	-7.36	$2.49e-13$
All grades	0.31	44089	0.67	385830	-0.36	-152.69	0.00

Table 2: The experimental results of investigating the impact of percentages of correct and wrong responses during in-unit assignments on the end-of-unit problems

(a) % Correct Response (**CR**)

Grade	**CR** (≥ 0.7)		**CR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
1	0.79	10985	0.64	588	0.15	8.83	$1.20e-18$
2	0.75	76748	0.50	3065	0.25	31.28	$1.78e-213$
3	0.61	34542	0.48	1345	0.12	9.13	$7.30e-20$
4	0.54	40967	0.35	2118	0.19	17.19	$5.38e-66$
5	0.51	57950	0.42	6165	0.09	12.78	$2.47e-37$
6	0.51	17264	0.48	3873	0.04	4.14	$3.56e-5$
7	0.56	17980	0.30	1516	0.26	19.55	$2.80e-84$
8	0.50	6462	0.40	2399	0.09	7.89	$3.48e-15$
HSA	0.61	953	0.43	666	0.18	7.26	$6.06e-13$
HSF	0.58	731	0.52	821	0.06	2.57	$1.03e-2$
HSG	0.52	2570	0.50	957	0.03	1.40	0.16
HSN	0.71	311	0.43	42	0.28	3.68	$2.72e-4$
HSS	0.55	305	0.43	418	0.13	3.43	$6.33e-4$
All grades	0.62	303094	0.45	25268	0.17	54.11	0.00

(b) % Wrong Response (**WR**)

Grade	**WR** (≥ 0.7)		**WR** (< 0.3)		**SD**	**t-stat**	**p-value**
	Score	**SS**	**Score**	**SS**			
1	0.57	382	0.81	17455	-0.24	-11.47	$2.29e-30$
2	0.50	5026	0.75	77445	-0.25	-39.49	0.00
3	0.39	5153	0.64	35325	-0.25	-35.47	$2.30e-271$
4	0.39	7789	0.62	26086	-0.23	-36.92	$1.55e-292$
5	0.38	13036	0.57	44434	-0.19	-38.84	0.00
6	0.41	9332	0.63	43403	-0.23	-40.95	0.00
7	0.38	5918	0.60	18708	-0.22	-29.70	$1.70e-190$
8	0.41	3192	0.55	13591	-0.14	-14.31	$3.55e-46$
HSA	0.40	452	0.61	3843	-0.21	-8.63	$8.41e-18$
HSF	0.47	391	0.63	7232	-0.17	-6.71	$2.14e-11$
HSG	0.35	1560	0.61	4866	-0.26	-18.11	$1.53e-71$
HSN	0.50	86	0.63	614	-0.13	-2.24	0.03
HSS	0.29	278	0.49	3168	-0.20	-6.31	$3.25e-10$
All grades	0.42	59280	0.67	322948	-0.24	-114.70	0.00

performed better. A similar reverse trend is observed in grade 7, 8 and HSF for Live Tutor Requested (LTR).

- In the case of Live Tutor Requested (LTR), there is a large difference (SD = 0.54) in grade 6, but the t-statistic is not reported, suggesting an insufficient sample size.

Additionally, we explored the percentage of correct and wrong responses in the action logs of in-unit assignments to assess their impact on student's scores in end-of-unit problems. Table 2 shows the results. The data from Table 2 offers several key observations about the impact of students' correct and wrong response rates on in-unit assignments on their final grades.

- Students with higher percentages of correct responses (CR $\geq$ 0.7) consistently achieve higher scores across all grades, with the most significant difference seen in grade 2.
- Conversely, students with lower percentages of wrong responses (WR $<$ 0.3) consistently attain higher scores, particularly notable in grade 3, where students with lower WR had a 0.25 higher grade.
- These trends are statistically significant ($p < 0.05$), suggest that the frequency of correct and wrong responses during in-unit assignments is predictive of a student's final grade.

The analysis suggests that students who infrequently request tutoring often perform better, indicating self-reliance and problem-solving capability. In contrast, direct interactions via live tutor requests (LTR) appear beneficial, enhancing performance. However, these correlations should not be interpreted as causation due to their context-specific nature. Conversely, frequent requests for hints, explanations, or frequent incorrect responses may indicate struggles with the material, suggesting a need for more teacher support or personalized interventions to improve understanding and performance [20, 21, 22]. Identifying and addressing these areas is essential for educators to effectively support student success.

2.2 CCSS Skill Mastery and Student Performance

Understanding the relationship between mastery of skills in in-unit assignments and end-of-unit outcomes is vital for educators to identify which skills affect students' success or failure in corresponding end-of-unit tasks [23, 24, 25]. Mastery of prerequisite skills in in-unit problems often leads to success in end-of-unit assignments, while failure to master these can result in poor outcomes. As the dataset lacks scores for in-unit problems, we assigned scores based on criteria from end-of-unit problems as described in Section A.1. A correct first attempt scores 1 (success); otherwise, actions like hints request score 0 (failure).

Once we determined the scores for in-unit problems, we needed to extract and fix skill codes since, as mentioned, we are interested in mastery/non-mastery for *skills*, not specific problems. The CCSS skill codes follow a hierarchical structure, where the first level corresponds to the grade, and the second level represents the topic or subject. The later levels of a CCSS skill code provide more specific descriptions, such as specific problem details. However, these detailed levels may not be as useful since they can potentially generate skill-related patterns that are overly specific. Therefore, for the purpose of generating meaningful patterns, we focused on the first two levels of the skill code hierarchy, which capture the broader grade and topic information, respectively.

So far, we have scores (success or failure) for all in-unit and end-of-unit problems as well as their corresponding skill codes. Now, we need to specify how we can extract meaningful "patterns". One effective approach for achieving this is by utilizing association rule learning/mining, which is a data mining technique employed to discover interesting relationships or patterns within extensive datasets [26, 27] such as educational data [28]. They identify frequently occurring *itemsets* and generate rules that describe associations between different items based on their co-occurrence. By leveraging these rules, educators can gain valuable insights into patterns and dependencies among skills. See appendix A.2 for formal definition of the association rule mining process on the CCSS skill codes.

Table 3: Extracted association rules for mastery (score 1) for the entire dataset

Rule	S	C	Rule Description
8.EE → HSA.REI	0.81	1.00	Expressions and Equations → Reasoning with Equations and Inequalities
HSN.RN → HSA.REI	0.93	0.99	The Real Number System → Reasoning with Equations and Inequalities
HSN.RN, HSN.CN → HSA.REI	0.86	0.99	The Real Number System, The Complex Number System → Reasoning with Equations and Inequalities
HSN.CN → HSA.REI	0.92	0.99	The Complex Number System → Reasoning with Equations and Inequalities
HSA.REI → HSN.RN	0.93	0.94	Reasoning with Equations and Inequalities → The Real Number System
HSN.RN → HSN.CN	0.87	0.93	The Real Number System → The Complex Number System

Table 4: Extracted association rules for non-mastery (score 0) for the entire dataset

Rule	S	C	Rule Description
HSA.REI → HSN.CN	0.79	0.93	Reasoning with Equations and Inequalities → The Complex Number System
HSN.RN, HSA.REI → HSN.CN	0.71	0.93	The Real Number System, Reasoning with Equations and Inequalities → The Complex Number System
HSN.RN → HSN.CN	0.81	0.92	The Real Number System → The Complex Number System
HSA.REI → HSN.RN	0.77	0.91	Reasoning with Equations and Inequalities → The Real Number System
HSN.CN, HSA.REI → HSN.RN	0.71	0.90	The Complex Number System, Reasoning with Equations and Inequalities → The Real Number System
HSF.BF → HSF.IF	0.71	0.89	Building Functions → Interpreting Functions

In association rule learning, "strong" rules are captured by two rule-related concepts: support and confidence. **Support** (**S**) is the proportion of transactions in the dataset that contain a particular itemset. **Confidence** (**C**) is a measure of the reliability of the inference made by a rule. To extract rules, we used *mlxtend* python library [29], which uses the famous Apriori algorithm [30, 31] for efficient rule mining. To generate frequent item sets, we identified items that occur frequently by setting a minimum support threshold of 0.8 for mastery (score 1) and 0.7 for non-mastery (score 0). Subsequently, we derived association rules using the confidence metric, with a minimum threshold of 0.9 for mastery (score 1) and 0.8 for non-mastery (score 0). Tables 3 and 4 demonstrate the results of association rule learning for skill mastery and non-mastery, respectively. The first column is the rule, the second column is the support, the third column is the confidence, and the last column is the rule where the codes have been replaced with their English description. We obtained the descriptions for skill codes up to level 2 from CCSS [32] since they were not included in the dataset. Tables 3 and 4 display the most reliable rules among all the transactions with high support and confidence. Based on the results in Tables 3 and 4, we make the following observations:

- The high support and confidence of most rules in both tables indicate that there are strong associations among the different math skills in the dataset. For example, the rule "8.EE → HSA.REI" in Table 3, with a support of 0.81 and confidence of 1.00, suggests that students who have mastered "Expressions and Equations" (8.EE) are also very likely to have mastered "Reasoning with Equations and Inequalities" (HSA.REI).
- The rule "HSN.RN, HSN.CN → HSA.REI" in Table 3 shows that the mastery of the Real Number System (HSN.RN) and the Complex Number System (HSN.CN) is highly associated with the mastery of Reasoning with Equations and Inequalities (HSA.REI). This reflects the progressive complexity in learning mathematics, where advanced concepts often rely on the mastery of more basic concepts.
- There is one unique rule in the non-mastery table, "HSF.BF → HSF.IF", which suggests that students who have not mastered "Building Functions" (HSF.BF) are also likely not to have mastered "Interpreting Functions" (HSF.IF). This could imply that the skills required for interpreting functions are contingent on the ability to build functions, or vice versa.
- Another observation is the high frequency of HSA.REI (Reasoning with Equations and Inequalities), HSN.RN (High School Number and Quantity - The Real Number System), and HSN.CN (The Complex Number System) in the rules, which implies that these concepts might be fundamental to mastery for High School subjects in this ASSISTments platform.
- Certain rules are evident in both Tables 3 and 4, including "The Real Number System → The Complex Number System". Interpreting from Table 3, it is inferred that mastery in the real number system is a strong predictor for mastery in the complex number system. Conversely, Table 4 implies that students who struggle with the real number system are likely to face challenges in understanding the complex number system. These rules underscore the interdependency of

understanding these two mathematical concepts.

- Finally, it is worth noting that while support and confidence are high for most rules, they are not absolute indicators of causality. The relationships could be affected by other factors such as the order of teaching, student demographics, etc. They do, however, offer a strong basis for further investigation into these associations.

3. STUDENT SCORE PREDICTION

In this section, we primarily address **RQ3** and **RQ4** through the presentation and discussion of our experimental results. The process of feature engineering, which is crucial to our approach, is detailed in Section 3.1. In Section 3.2, we delve into a unique type of feature, the graph representation learning. Following this, we outline the machine learning predictive models used in our study in Section 3.3. Finally, we detail the experimental results of predicting end-of-unit student grades in Section 3.4, where we also discuss the significance and implications of our findings.

3.1 Feature Extraction

The feature extraction process involved combining the training and evaluation data and extracting relevant information mainly from four tables: Action logs, Assignment details, Problem details, and Sequence details– See Appendix A.1. Next, we explain each group of features.

- **Action Log Features.** We utilized the assignment relationships table to retrieve action logs associated with end-of-unit assignments. Given the variation in feature values for different in-unit problems within the same end-of-unit assignment, we aggregated information by selecting the most frequently occurring values for certain features like 'max attempts', 'score viewable', and 'continuous score viewable'. For 'action' and 'available core tutoring' features, we performed one-hot encoding and took the sum of values for the group. This aggregation process resulted in a feature set of 21 dimensions.
- **Assignment Detail Features.** We obtained sequence details for all end-of-unit assignments from the assignment details table. Additionally, we introduced a new feature called '% of assignment not finished', based on the assignment end date. By calculating the percentage of uncompleted in-unit assignments for each end-of-unit assignment, we captured valuable information about student engagement and completion rates.
- **Sequence Detail Features.** We retrieved sequence-related features such as 'sequence folder path levels' 1, 2, 3, and 4 from the sequence details table. These four features were represented using one-hot encoding, adding 172 dimensions to the feature vector due to the unique values in each of the sequence folder path levels.
- **Problem Detail Features.** We extracted all problem-related features for the end-of-unit problems in the training and evaluation sets from the problem details table. This included splitting the feature 'problem skill code' into four parts, one-hot encoding the 'problem type' feature, and representing the 'problem skill description' feature using 32-dimensional BERT PCA embeddings. We also retained existing binary features such as 'problem contains image' and introduced a new feature called 'problem multipart ID frequency' This feature captures the number of occurrences of a specific problem multipart ID within the dataset. Taken together, the size of Problem Detail Features ended up being 210.

3.2 Graph Representation Learning

Graph Representation and Construction. In our study on the ASSISTments learning platform, we combined graph structure and graph construction methods to model and understand student behavior. We utilized graph representation learning, which focuses on capturing relationships within a graph to derive informative representations of nodes and edges [33]. Our hypothesis was that the structural features captured in relationships among various entities could predict outcomes on end-of-unit tests. We identified five key entities as nodes: 'student', 'teacher', 'class', 'problem', and 'sequence'. The connections between these entities were represented by four edge types: 'teacher-class', 'class-student', 'student-problem', and 'problem-sequence'. By employing these specific connections, we managed to maintain essential structural information while avoiding an overly dense graph structure. To construct the graph, we began by adding nodes for students involved in end-of-unit assignments and traced their connected in-unit assignments using the assignment relationships table. Nodes for their classes and teachers were added, linking students to their respective classes and the classes to their teachers. We also added nodes for both in-unit and end-of-unit problems, establishing connections between these problems and the students who tackled them, as well as linking problems to their corresponding sequences based on the action log records. This comprehensive construction allowed us to analyze the interconnected elements within the dataset effectively and explore the predictive power of the graph structure on student performance in end-of-unit assessments. Figure 2 in Appendix A.3 demonstrates the structure of the constructed graph. Also, Table 7 in Appendix A.3 includes the basic properties of the constructed graph.

Graph Representation Learning. We applied the node2vec algorithm, a random-walk-based representation learning technique, to generate low-dimensional vector embeddings for nodes in a graph, capturing their structural and community properties [34]. This method uses random walks to explore local neighborhood information, where nodes close in the graph share similar roles and functions. Node embeddings were learned using a modified Skip-gram model from natural language processing [35]. We utilized these embeddings for the end-of-unit problem to enhance representation and conducted experiments to assess their effectiveness in grade prediction, bypassing the need for hand-crafted features as discussed in Section 3.1.

3.3 Predictive Models

We employ a variety of predictive machine learning models, such as Random Forest, Gradient Boosting, XGBoost, LightGBM, ExtraTrees, and a Mean Ensemble, to forecast end-of-unit grades within the ASSISTments platform. These models are chosen for their robustness, flexibility, and performance, particularly in handling high-dimensional data and providing accurate predictions. The paper details the unique features of each model and explains how the Mean Ensemble model, which averages the predictions of individual models, enhances overall predictive performance. This

Table 5: Performance of developed ML models across three settings and for different evaulation metrics. In each column and for each setting, the bold value indicates the best model according to that metric. The underlined value indicates the best across all settings.

	Setting	Model	Accuracy	Precision	Recall	F1 score	AUC
(I)	Hand-crafted features (Section 3.1)	XGBoost	0.72238	**0.74441**	0.80658	0.77425	0.78295
		Random Forest	0.71402	0.73045	0.81694	0.77128	0.77005
		Extra Trees	0.69865	0.70906	**0.83001**	0.76479	0.74925
		LightGBM	0.72370	0.74361	0.81177	0.77620	**0.78473**
		Gradient Boosting	0.72257	0.74296	0.81030	0.77517	0.78245
		Mean Ensemble	**0.72437**	0.74043	0.82075	**0.77853**	0.78383
(II)	Student+Problem embedding (Section 3.2)	XGBoost	0.68305	**0.72894**	0.73711	0.73300	0.73251
		Random Forest	**0.69458**	0.72626	0.77444	**0.74958**	**0.74286**
		Extra Trees	0.69094	0.72168	**0.77543**	0.74759	0.74115
		LightGBM	0.68579	0.72203	0.76039	0.74071	0.73338
		Gradient Boosting	0.69068	0.72525	0.76621	0.74517	0.73982
		Mean Ensemble	0.69165	0.72763	0.76329	0.74504	0.74174
(III)	Hand-crafted (Section 3.1) + Problem embedding (Section 3.2)	XGBoost	0.70857	0.74560	0.76843	0.75685	0.76520
		Random Forest	0.71218	0.73662	0.79750	0.76585	0.77084
		Extra Trees	0.71737	0.74689	0.78831	0.76704	0.77279
		LightGBM	0.71967	**0.74990**	0.78780	0.76838	0.78145
		Gradient Boosting	0.71507	0.74887	0.77825	0.76328	0.77443
		Mean Ensemble	**0.72784**	0.74979	**0.80880**	**0.77818**	**0.78977**

approach aims to provide more accurate and robust grade predictions by combining the strengths of different models. Appendix A.4 includes details of implementation settings.

3.4 Experimental Results

Table 5 summarizes the results of our extensive experiments. We define three experimental settings to illustrate the effectiveness of graph embedding techniques in the end-of-unit grade prediction task. **Setting (I):** This setting only uses hand-crafted features, detailed in Section 3.1, as input to the machine learning models. These features are derived from the raw data without the application of graph embedding techniques. The aim here is to evaluate the performance of models based on explicit feature engineering. **Setting (II):** This setting uses a combination of the end-of-unit problem embeddings and student embeddings as inputs to the models. Embedding techniques are used here to represent the problems and students in a high-dimensional space. The purpose of this setting is to investigate the effectiveness of embedding methods in representing students and problems. **Setting (III):** In this setting, the models are trained on data that combines both hand-crafted features and problem embeddings. This setting is designed to study whether the combination of hand-crafted features and embeddings can improve the performance of the models in predicting students' scores. Based on the results presented in Table 5, we make the following observations.

- In setting (I), which uses only hand-crafted features, the Mean Ensemble model performs best in terms of accuracy and F1 score. Notably, Extra Trees model gives the best recall, indicating that it is the best at identifying true positive cases.
- For setting (II), where we only use the embeddings of the student and end-of-unit problem, the Random Forest model seems to outperform the others, providing the highest accuracy and F1 score. However, the Extra Trees model provides the highest recall, similar to setting (I).
- In setting (II), despite only using low-dimensional features and lacking information related to students' actions within the units, a relatively high AUC is achieved. This indicates the power of embedding techniques in capturing complex structural relationships in the data without the need for complicated hand-crafted features, which may not be obtained readily.
- In setting (III), where both hand-crafted features and problem embeddings are used, the Mean Ensemble model outperforms all other models in terms of accuracy, F1 score, and AUC.
- In setting (III), the results not only confirmed the effectiveness of graph embedding but also showed that combining these embeddings with hand-crafted features can further enhance prediction performance. This setting achieved the best results in terms of accuracy, precision and AUC among all settings.
- The Mean Ensemble model consistently demonstrates strong performance across all settings, often achieving the highest or near-highest values for several metrics. This suggests that the ensemble approach, which leverages the strengths of multiple models, can effectively enhance the prediction performance.
- Overall, we observe that combining hand-crafted features with problem embeddings (setting III) provides the highest performance across most metrics, underlining the importance of integrating domain-specific features with learned embeddings for educational grade prediction. Notably, the AUC, the metric used in EDM Cup 2023 Kaggle Competition, is the highest in this setting compared to the other two.

We analyzed the feature importance of all hand-crafted features, identifying the end-of-unit problem text, student actions within the unit, and problem skill descriptions as the top three predictors of student grades– See Appendix A.5).

APPENDIX

A. ADDITIONAL INFORMATION

In this part, we provide supplementary figures, tables, and detailed explanations of dataset and implementation settings used in experiments.

A.1 Dataset

We provide an overview of the dataset from the EDM Cup 2023 Kaggle Competition [18], detailing the tables and their salient attributes. ASSISTments is a digital platform designed for daily math practice and assessment, aiding classroom instruction by offering digital assignments and feedback. Each student is given a set of math problems(sequence) to solve, referred to as an assignment within the dataset. The dataset contains millions of student interactions within the platform, detailing their completion of math assignments. It also encompasses information about the curriculum, assignments, problems, and any tutoring provided to the students. Table 6 shows the basic statistics for all entities in the dataset.

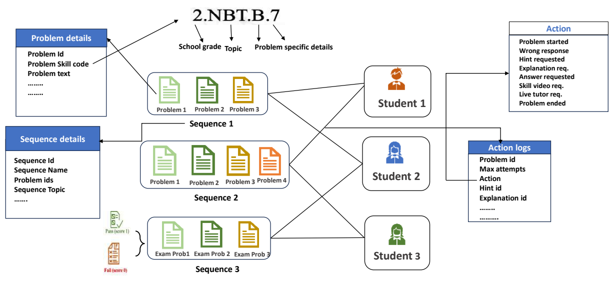

Figure 1: The relationship between different entities in the dataset

- **Action Logs**: Captures student interactions with in-unit assignments, including correctness of responses, hint or explanation requests, and tutoring availability.
- **Training and Evaluation Unit Test Scores**: Contains binary scores for end-of-unit assignments, which serve as training and evaluation data for score prediction models. Score of 1 denotes completion of an open-ended response problem or correct first-attempt response without tutoring, while 0 signifies otherwise.
- **Assignment Details**: Records the details of both in-unit and end-of-unit assignments, including the problem sequences assigned to individual students.
- **Sequence Details**: Represents set of problems structured into sequences, which are further organized into folder paths that denote attributes such as curriculum or grade.
- **Problem Details**: Contains information on both problems, including the Common Core State Standards (CCSS) for Mathematics [32] skill code and BERT embeddings for textual content.
- **Hint and Explanation Details**: Holds information on hints and explanations, including BERT embeddings for textual content.
- **Assignment and Sequence Relationships**: Specifies correspondences between in-unit and end-of-unit assignments, and sequences.

In-unit assignments resemble practice problems conducted throughout the unit, while end-of-unit assignments resemble exams administered at the end of each unit. Scores are exclusively provided for end-of-unit assignments, whereas action logs are solely available for in-unit assignments. More details about the dataset can be found in website [18]. Figure 1 shows the connection between different entities in dataset.

Table 6: Basic statistics of the dataset

Entity	Count
# Students	651253
# Teachers	23523
# Classes	47401
# Sequences	10228
# Problems	132738
# Assignments	9319676
# Hints	8381
# Explanations	4132
# Problems per sequence	μ=13
# Unfinished assignments	1878016

Abbreviations: In the rest of manuscript, *HSS* stands for High School Statistics and Probability, *HSF* for High School Functions, *HSG* for High School Geometry, *HSN* for High School Number and Quantity, and *HSA* for High School Algebra.

A.2 Association Rule Mining Problem Statement

We formally define the association rule mining process on the CCSS skill codes as follows: Let $\mathcal{C} = \{C_1, C_2 \cdots C_k\}$ denote a set of CCSS skill code levels e.g., *the Complex Number System*. We've categorized our end-of-unit problems into two groups: P^s for successful outcomes (scored 1) and P^f for unsuccessful outcomes (scored 0). Each successful end-of-unit problem p_i^s in P^s corresponds to a "transaction", labeled as T_i. These transactions contain the skill code of the end-of-unit problem (I_{m_i}) and the skill codes of all related in-unit problems (I_1 to I_{m_i-1}). Similarly, each failed end-of-unit problem p_i^f in P^f is associated with its own transaction. Our goal is to uncover patterns related to skill mastery and non-mastery. For skill mastery, we're interested in strong rules of the form $X \rightarrow y$, where X is a subset of skills (from C) and y represents the skill code of the end-of-unit problem. We're also seeking similar rules for non-mastery.

A.3 Constructed Graph

Figure 2 shows the structure of the final constructed graph. Table 7 shows the basic properties of the constructed graph.

A.4 Implementation settings

After extracting all the features as explained in Section 3.1, we standardized both training and evaluation data using the

Table 7: Properties of the constructed graph.

Attribute	Value
Type	Heterogeneous
# Node types	5
# Students	34652
# Problems	59109
# Sequences	5766
# Teachers	2024
# Classes	3055
# Total nodes	104606
# Edges	5527865
Density	0.001

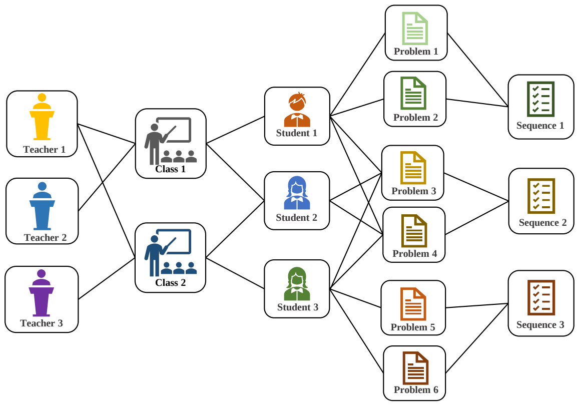

Figure 2: The constructed graph between different entities in the dataset

StandardScaler package from scikit-learn [36]. Standardization helps to bring the features to a common scale, enabling fair comparisons and preventing features with larger magnitudes from dominating the model's learning process. For tuning the hyperparameters of each predictive model mentioned in Section 3.3, we utilized the RandomizedSearchCV from scikit-learn. We performed 10-fold cross-validation while tuning the hyperparameters. Additionally, we used the scikit-learn package to implement the Random Forest, Gradient Boosting, and Extra Trees methods. For the XGBoost model, we employed the XGBoost Python package [37], while we used the lightgbm package [38] for implementing the LGBM model. We made use of the node2vec package (`https://pypi.org/project/node2vec/`) for the node2vec implementation described in Section 3.2. The hyperparameters used in learning node embeddings with node2vec are as follows: embedding dimension: 32, number of walks: 100, walk length: 10, and window size: 10. In the end, we utilized various evaluation metrics, including accuracy, precision, recall, F1-score, and AUC, employing the scikit-learn package.

A.5 Feature Importance

Figure 3 demonstrates the relative importance of different hand-crafted features in predicting the student grade in setting (I). We make the following observations based on the results presented in this figure.

- The most important feature appears to be the end-of-unit problem text, which is provided as a 32-dimensional embedding vector in the dataset.
- Actions that students took within the unit, such as tutoring requests and responses to questions, are shown to significantly affect the student grades, reinforcing our analysis in Section 2.1.
- The problem skill description and problem skill code also feature among the top 5 most important features, reflecting the relationship between students' grades and the skills they are being tested on. This point underscores the importance of our analysis in Subsection 2.2.
- The sequence folder path level 3, which describes what unit the sequence is part of, is among the top 5 important features. More descriptions for other sequence path levels can be found on the dataset website [18].
- The importance of available core tutoring types within the unit in predicting the student's grade is worth noting, implying that instructional methods can be indicative of student performance.

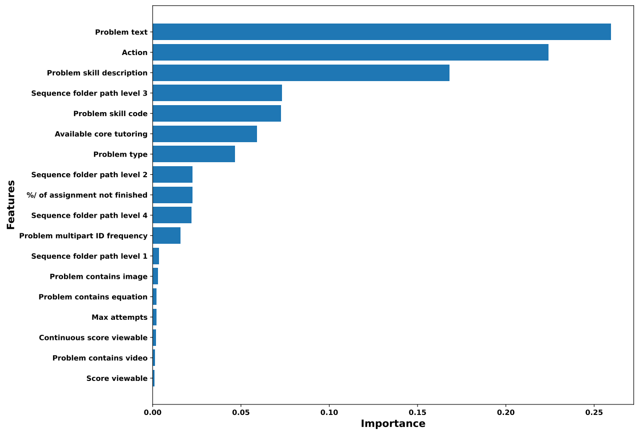

Figure 3: Feature importance of hand-crafted features explained in Section 3.1

5. REFERENCES

[1] Muhammad Fawad Akbar Khan, John Edwards, Paul Bodily, and Hamid Karimi. Deciphering student coding behavior: Interpretable keystroke features and ensemble strategies for grade prediction. In *2023 IEEE International Conference on Big Data (BigData)*, pages 5799–5808. IEEE, 2023.

[2] Muhammad Fawad Akbar Khan, Max Ramsdell, Erik Falor, and Hamid Karimi. Assessing the promise and pitfalls of chatgpt for automated code generation. *arXiv preprint arXiv:2311.02640*, 2023.

[3] Hamid Karimi, Jiangtao Huang, and Tyler Derr. A deep model for predicting online course performance. *Association for the Advancement of Artificial Intelligence*, 2020.

[4] Hamid Karimi, Tyler Derr, Kaitlin T. Torphy, Kenneth A. Frank, and Jiliang Tang. A roadmap for incorporating online social media in educational research. volume 121, pages 1–24, 2019.

[5] Hamid Karimi, Tyler Derr, Kaitlin Torphy, Kenneth A Frank, and Jiliang Tang. Towards improving sample representativeness of teachers on online social media:

A case study on pinterest. pages 130–134. Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21, 2020.

[6] Kaitlin Torphy Knake, Hamid Karimi, Sihua Hu, Kenneth A Frank, and Jiliang Tang. Educational research in the twenty-first century: Leveraging big data to explore teachers' professional behavior and educational resources accessed within pinterest. *The Elementary School Journal*, 122(1):86–111, 2021.

[7] Hamid Karimi, Jiliang Tang, Xochitl Weiss, and Jiangtao Huang. Automatic identification of teachers in social media using positive unlabeled learning. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 643–652, 2021.

[8] Sakshi Solanki, Kiana Kheiri, Marissa A Tsugawa, Hamid Karimi, et al. Leveraging social media analytics in engineering education research. In *2023 ASEE Annual Conference & Exposition*, Baltimore , Maryland, June 2023. ASEE Conferences. https://peer.asee.org/43472.

[9] Soheila Farokhi, Aswani Yaramal, Jiangtao Huang, Muhammad Fawad Akbar Khan, Xiaojun Qi, and Hamid Karimi. Enhancing the performance of automated grade prediction in mooc using graph representation learning. In *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2023.

[10] Hamid Karimi, Tyler Derr, Jiangtao Huang, and Jiliang Tang. Online academic course performance prediction using relational graph convolutional neural network. ERIC, 2020.

[11] AA Bilyalova, DA Salimova, and TI Zelenina. Digital transformation in education. pages 265–276, 2020.

[12] Zi-Yu Liu, Natalya Lomovtseva, and Elena Korobeynikova. Online learning platforms: Reconstructing modern higher education. volume 15, pages 4–21. International Journal of Emerging Technology in Learning, 2020.

[13] Tawanna Dillahunt, Zengguang Wang, and Stephanie D Teasley. Democratizing higher education: Exploring mooc use among those who cannot afford a formal education. volume 15, pages 177–196. Érudit, 2014.

[14] Todd JB Blayone, Roland vanOostveen, Wendy Barber, Maurice DiGiuseppe, and Elizabeth Childs. Democratizing digital learning: theorizing the fully online learning community model. volume 14, pages 1–16. Springer, 2017.

[15] Zehra Akyol and D Randy Garrison. Understanding cognitive presence in an online and blended community of inquiry: Assessing outcomes and processes for deep approaches to learning. volume 42, pages 233–250. Wiley Online Library, 2011.

[16] Kay Livingston and Rae Condie. The impact of an online learning program on teaching and learning strategies. volume 45, pages 150–158. Taylor & Francis, 2006.

[17] Katrina A Meyer. Student engagement in online learning: What works and why. volume 40, pages 1–114. Wiley Online Library, 2014.

[18] Ethan Prihar and III Heffernan, Neil T. Edm cup 2023, Jun 2023.

[19] William Mendenhall, Robert J Beaver, and Barbara M Beaver. *Introduction to Probability and Statistics*. Cengage Learning, 2012.

[20] Elena Barberà Gregori, Jingjing Zhang, Cristina Galván-Fernández, and Francisco de Asís Fernández-Navarro. Learner support in moocs: Identifying variables linked to completion. volume 122, pages 153–168. Elsevier, 2018.

[21] Jacob Whitehill, Joseph Williams, Glenn Lopez, Cody Coleman, and Justin Reich. Beyond prediction: First steps toward automatic intervention in mooc student stopout. 2015.

[22] Thanasis Daradoumis, Roxana Bassi, Fatos Xhafa, and Santi Caballé. A review on massive e-learning (mooc) design, delivery and assessment. In *2013 eighth international conference on P2P, parallel, grid, cloud and internet computing*, pages 208–213. IEEE, 2013.

[23] Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. volume 4, pages 253–278. Springer, 1994.

[24] Kurt VanLehn. The behavior of tutoring systems. volume 16, pages 227–265. IOS Press, 2006.

[25] Neil Heffernan and Cristina Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. In *International Conference on Intelligent Tutoring Systems*, pages 494–504. Springer, 2014.

[26] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, 1993.

[27] Trupti A Kumbhare and Santosh V Chobe. An overview of association rule mining algorithms. volume 5, pages 927–930. Citeseer, 2014.

[28] Cristóbal Romero, José Raúl Romero, Jose María Luna, and Sebastián Ventura. Mining rare association rules from e-learning data. In *Educational Data Mining 2010*. ERIC, 2010.

[29] Sebastian Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. volume 3. The Open Journal, April 2018.

[30] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499. Santiago, Chile, 1994.

[31] Markus Hegland. The apriori algorithm–a tutorial. pages 209–262. World Scientific, 2007.

[32] Murat Akkus. The common core state standards for mathematics. volume 2, pages 49–54. ERIC, 2016.

[33] William L Hamilton. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

[34] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality.

volume 26, 2013.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. volume 12, pages 2825–2830, 2011.

[37] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[38] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. volume 30, pages 3146–3154, 2017.