Behavioral Pattern Mining and Modeling in Programming Problem Solving

by

Partho Pratim Mandal

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2017 by the
Graduate Supervisory Committee:

I-Han Hsiao, Chair
Hasan Davulcu
Hanghang Tong

ARIZONA STATE UNIVERSITY

December 2017

ABSTRACT

Online learning platforms such as massive online open courses (MOOCs) and intelligent tutoring systems (ITSs) have made learning more accessible and personalized. These systems generate unprecedented amounts of behavioral data and open the way for predicting students' future performance based on their behavior, and for assessing their strengths and weaknesses in learning.

This thesis attempts to mine students' working patterns using a programming problem solving system, and build predictive models to estimate students' learning. QuizIT, a programming solving system, was used to collect students' problem-solving activities from a lower-division computer science programming course in 2016 Fall semester. Differential mining techniques were used to extract frequent patterns based on each activity provided details about question's correctness, complexity, topic, and time to represent students' behavior. These patterns were further used to build classifiers to predict students' performances.

Seven main learning behaviors were discovered based on these patterns, which provided insight into students' metacognitive skills and thought processes. Besides predicting students' performance group, the classification models also helped in finding important behaviors which were crucial in determining a student's positive or negative performance throughout the semester.

DEDICATION

To my parents Mausumi and Arup Kumar Mandal.

ACKNOWLEDGMENTS

I would like to thank my parents for all their love and support. I am also grateful to my advisor, Dr. Sharon Hsiao, who has always been approachable and eager to assist me. Her motivation, guidance and support has helped me throughout my graduate studies. I am also grateful to Dr. Hasan Davulcu and Dr. Hanghang Tong for reviewing my work and advising me on how improve it. Lastly, I would like to thank Julio Guerra and Dr. Philippe Fournier-Viger for helping me with ideas at the implementation level.

Further thanks go to members of the CSI lab, and to my roommates and friends for their companionship.

TABLE OF CONTENTS

Page

LIST OF TABLES vi

LIST OF FIGURES vii

CHAPTER

1 INTRODUCTION 1

1.1 Motivation 2

1.2 Research Questions 3

1.3 Organization 4

2 BACKGROUND AND LITERATURE REVIEW 5

2.1 Student Assessments Based on Online Learning Systems 5

2.2 Sequential Pattern Mining to Study Learning Behaviors 7

3 METHODOLOGY 10

3.1 Research Platform 10

3.2 Data Collection 11

3.3 Building Action Sequences 12

3.4 Differential Pattern Mining 14

3.5 Building Predictive Models 18

4 EVALUATION 20

4.1 Exploring Learning Strategies with Behavioral Patterns 20

4.2 Exploring Models Based on Behavioral Patterns 25

4.2.1 Individual Pattern Based Predictive Models 26

4.2.2 Learning Behavior Based Predictive Models 31

4.2.3 Model Comparison 35

5 CONCLUSION 37

CHAPTER Page

5.1 Summary 37

5.2 Limitations and Future Work 39

REFERENCES 40

LIST OF TABLES

Table Page

1 Patterns Filtered by CFS Algorithm 18

2 Patterns Classified into Various Learning Strategies 20

3 Coefficients of Models Based on Individual Patterns 26

4 Coefficients of Models Based on Learning Behavior 31

5 Comparison of Metrics in LR and RF Models 35

6 Mean and Standard Deviation of Metrics of All Classifiers 36

LIST OF FIGURES

Figure Page

1 QuizIT Daily Programming System Interface 11

2 Differential Sequential Mining Algorithm 16

3 Learning Behavior Distribution across the Two Performance Groups 23

4 Learning Behavior by Students' Final Grades 24

5 Validation Curve for Pattern-Based LR Model 27

6 Logistic Regression Coefficients for Individual Patterns 28

7 Probability of Patterns for High-Performance in LR Model 29

8 Importance of Features Defined by Pattern-Based RF Classifier 30

9 Validation Curve for Learning Behavior Based LR Model 32

10 Logistic Regression Coefficients for Learning Behaviors 33

11 Probability of Learning Behaviors for High-Performance in LR Model 33

12 Importance of Features Defined by Behavior-Based RF Classifier 34

13 Comparison of Classification Model Metrics 36

Chapter 1

INTRODUCTION

Course management systems (CMSs) and online teaching systems have traditionally been the main sources of data on students' learning activities [11]. These data have led to an array of research directions in educational data-mining, aimed at improving the understanding of how students interact with such systems and use them to improve their learning skills. Cognitive scientists have shown that self-regulation and metacognition are the key components for developing effective learning, whether in a classroom or using online resources [7, 32]. Because teaching is complex and open-ended in nature, students must apply their cognitive skills to achieve success.

These skills are difficult to learn in a classroom environment [20, 31], but an open-ended learning environment can help students to learn and to practice such skills on an online teaching platform. In recent years, online platforms like EdX and Practical Algebra Tutor have given researchers scope to analyze the rich sources of data generated [5]. These studies have focused on examining students' backgrounds, the time spent on problems, and learning patterns that might contribute to success in the course. Knowledge about students' learning behavior and patterns aids instructors in providing feedback to students to help them understand the subject matter. For instance, Brown et al. [6] showed that through feedback, younger students can acquire cognitive skillsets essential for learning, such as strategizing their learning path and using self-monitoring techniques.

Predicting students' success has been a cornerstone of educational research for the last two decades [4, 29]. Several studies have focused on identifying students' performance predictors through information about their past academic performance, screening tests, and questionnaires regarding cognitive behavior, background or expectations. The development of online tutoring systems, complete with educational data-mining and learning analytics, has enabled researchers to track students' performance, learning patterns, and interactions with the system [28]. Based on these strategies, predictive models can be built.

1.1 Motivation

Researchers such as Biswas et al. [3] used hidden Markov models (HMMs) [23] for probabilistic representation of metacognitive strategies. They used HMM because the hidden states were representations of students' mental states, and the observable output was equivalent to student actions. This technique helped in identifying, interpreting and comparing students' learning patterns at an aggregated level. However, averaged descriptions do not provide a full picture about specific learning strategies that students employ while interacting with the system. This lack fueled the need to employ sequential mining techniques to yield a more precise analysis of learning patterns.

Analyzing student patterns through sequential pattern-mining [1] helps in identifying relevant patterns from students' action sequences. This information aids in evaluating and comparing students' learning behaviors and cognitive skills, across different groups such as higher and lower performing students. However, the data can produce a huge number of patterns when gaps between the learning sequences are taken into consideration. The challenge is to limit these large-scale results to a set of the most important patterns that differ across various learning groups [15].

The work in this thesis was motivated by the intuition that the efficacy of daily quizzing systems could be enhanced by a guiding mechanism that tracks students' daily activity. This system should predict productive and unproductive behavior and recommend more efficient learning strategies. Predicting student behavior based on random patterns is difficult. However, if specific patterns are known to be associated with students' traits, knowledge, or learning curves then there is a better chance to learn the underlying associations and use them for prediction [9].

This study examines data retrieved from QuizIT, a programming concepts' problem solving system. The identification of relevant traits among higher and lower performing students was based on creating action sequences from QuizIT data, using mining algorithms to find a set of frequent student patterns that affect student behavior. Then a predictive model was built to try and predict students' performances in a subject, based on these traits.

1.2 Research Questions

This thesis evaluates two main research questions:

Q1 - What are the possible learning strategies adopted by a student working on programming problems in QuizIT?

Q2 - Is it possible to build predictive models based on the patterns identified as being relevant to students' learning performance?

1.3 Organization

This introduction (Chapter 1) is followed by a discussion of the background and literature review to provide in-depth understanding of the context of the study (Chapter 2). The methodology is presented in Chapter 3. That chapter explains the research platform, data collection, and differential mining techniques used to extract frequent learning patterns. The results are discussed in Chapter 4; the evaluation of the results explores the learning behaviors exhibited by students and the performance of the predictive models based on these behaviors and patterns. The conclusion (Chapter 5) summarizes the approach and findings of this thesis.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Student Assessments Based on Online Learning Systems

One of the main research areas in educational data mining is understand students' behavior during learning, and the need to understand parameters that affect this behavior. The relationship between students' performance and gaps in duration, and other factors, has always been of interest to researchers [22, 25]. However, to date it has been difficult to manually assess student performances individually, due to time constraints. Researchers thus tended to rely on students' self-reporting of results. However, the risk of response bias has led to the development of online learning systems. These systems enable researchers to collect not only responses but also to log information regarding duration, course details, topics, difficulty levels and so on. Such information enables research on various aspects of the data.

One of the first major systems developed for this purpose was Computer-Assisted Personalized Approach (CAPA) [14]. CAPA was a network-based personalized assignment system that enabled students to discuss the solving strategy for assignments having similar concepts. They could also submit the solutions without any restriction on revisions before the due date. The same technology was used in a more advanced version, the Learning Online Network with CAPA (LON-CAPA) [17]. These systems provided strong evidence that personalized systems can enhance student performance, offer in-depth understanding of topics, and reduce plagiarism. However, they were mainly parameter-based question-and-answer (QA) systems restricted to the physics and mathematics domain.

Researchers were inspired by such systems to develop their own versions of parameterized QA systems to teach programming. One such system is QuizJET for teaching Java [13], which uses parameterized multiple-choice questions for programming concepts. An interesting feature of this system is that students participated in the system voluntarily. It was found that students were on average 2.5 times more likely to answer questions that offered adaptive navigation support than those without such support. This system helped both weak and strong students to learn concepts based on a gradual increase in difficulty levels, and it led to a high success rate. However, around 25% of students felt that feedback on the quizzes was insufficient.

Perera et al. [21] were interested in understanding the learning behavior of students in group projects and its impact on individual students. A related system was developed, called TRAC [27]. Student teams' log data were captured in TRAC whenever a user created or modified a wiki text page, a new ticket, or a subversion repository. The student groups were ranked by performance and were clustered based on their ticketing behavior. The results illustrated certain sequences pertaining to strong and weak students, related to leadership qualities and monitoring behavior. The sequences helped students obtain feedback on whether their progress was likely to yield a positive or negative outcome at the end of the semester. The main problem with this work was that the dataset was correlated, noisy, and incomplete.

2.2 Sequential Pattern Mining to Study Learning Behaviors

The development of online-based learning systems gave rise to a new research interest in educational data-mining, namely to mine sequential patterns of actions performed by students using the system. Researchers hoped that studying such action sequences would provide insight into students' learning behavior and help to give students feedback regarding their strengths and weaknesses.

As discussed in the previous section, Perera et al. [21] clustered student groups according to their ticketing behavior. The researchers transformed the logged data into list of events, with each event consisting of an event type, ticket number, author, and timestamp. Events were encoded into items, using alphabets to form action sequences which represented sequential patterns of group sessions. Frequent patterns were mined using a modified version of the generalized sequential mining (GSM) [26] algorithm. The most frequent patterns were used to distinguish stronger groups from weaker ones.

Maldonado et al. [18] used a pen-based tabletop to study the problem-solving behavior of students in groups, when they were asked to solve picture-based mysteries. All actions performed by the pen were logged. These actions were then codified into events as in [21]; each event consisted of details about time, author, action, and object. The researchers used both raw human–computer interactions and the compact logged actions for mining and clustering frequent patterns. They also used variable-order n-gram-based mining algorithms [19] for both methods, and compared the results to evaluate the strategies of high- and low-performing groups.

One of the most popular sequential pattern algorithms used in educational data-mining is bitmap based sequential pattern mining (SPAM) [2]. This is a DFS-based pruning algorithm which tracks candidates using a vertical bitmap data structure. SPAM has proven to be time-efficient compared with other algorithms for identifying frequent patterns among long sequential patterns. However, a shortcoming was it did not support user-defined constraints.

Joshua Ho et al. [12] developed a generalized version of the SPAM algorithm, called the Pex-SPAM algorithm, which introduced gap constraints. Pex-SPAM was developed to deal with protein sequences consisting of in-between noises to extract high-quality transmembrane helix features. However, the algorithm is general in nature and can be used in other pattern mining problems. It is necessary to incorporate such constraints because action sequences are noisy. Students display patterns that relate to their learning behavior but they also perform additional tasks in-between, which are irrelevant to the pattern.

Guerra et al. [9] used a dataset from QuizJET to create students' action sequences using two alphabets, s and f to denote success or failures respectively. These alphabets in lowercase and uppercase denoted short and long duration while solving subsequent problems. The researchers used the Pex-SPAM algorithm to find frequent patterns and termed these "problem solving genomes". The genomes were then employed to analyze the stability of the learning behavior of students at the temporal and spatial levels. The researchers also analyzed the effect of the difficulty level of a problem on the genome. The results of that research cannot indicate whether problem-solving genomes in other subject domains might show similar properties.

Kinnebrew et al. [15] developed an ITS called Betty's Brain [16] and created action sequences of each student based on Betty's Brain logged details. The researchers used gap-constraint-based SPAM to find common patterns among all students. To identify the most interesting patterns, they developed a novel technique called differential sequential mining. This technique uses t-tests to find differentially frequent patterns and examines whether there is a statistically significant difference in a pattern's frequency for each sequence, for higher and lower performing students. This analysis distinguishes the learning behaviors of high and low performing students. Kinnebrew et al. also analyzed the reading behaviors of high and low performing students during their productive and unproductive phases of work [16].

Herold et. al [11] collected students' actions which were handwritten using a digital pen. Variables such as ink color and the duration of strokes were also logged. Action sequences were developed from the raw data, with each action representing an assignment number, topic type, and duration. Differential sequential mining technique was used to find patterns for strong and weak students, and a linear model was proposed to predict students' performance in the course based on these patterns.

Online learning platforms enabled researchers to gather behavioral data about users, and inspired the study of behaviors related to problem-solving – both individually and in groups. As these data were sequential in nature, researchers developed more powerful sequential mining algorithms to extract patterns to represent cognitive skills. Improvements in techniques over the years encouraged people to use these representations to build systems to predict the strengths and weaknesses of users, with the aim of helping users address their issues and improve their efficiency in learning tasks. This thesis builds on those prior efforts by applying differential sequential mining to identify higher and lower performing students' habits, gain insight into their metacognitive processes, and use these processes to build predictive models.

Chapter 3

METHODOLOGY

3.1 Research Platform

QuizIT is a daily quiz system that generates a “quiz of the day” for programming concepts. It was developed by the CSI research team of the School of Computing, Informatics, and Decision Systems Engineering (CIDSE) at Arizona State University. Students are not required to use QuizIT but are encouraged to do so and are awarded extra credits if they do. Each day, QuizIT generates a random question with a multiple-choice answer; only one of the options presented is correct. There is no time limit on choosing the answer. The QuizIT system evaluates the student’s response and reports whether the answer is correct or incorrect. Figure 1 shows the user interface of QuizIT. The left panel shows the question of the day, and the right panel shows the student discussion board related to questions that were previously attempted by other students. The discussion board also provides a platform to post comments.

The student has the freedom to attempt the quiz of the day as well as previous questions (which were either attempted or not by the student). Each question is marked as a difficulty level of *Easy*, *Moderate*, or *Difficult*. If the question is correctly answered, the student is invited to attempt other questions. The student is not provided with the solution if the answer is incorrect but is rather prompted to redo the question until they give the correct answer. The goal is to make the student reflect on their mistakes to rectify them and find the solution on their own. This enables instructors to gain insight into the cognitive skills students are applying while attempting new questions or re-attempting incorrectly answered questions. These activities leave a trace of students' successes and failures over time.

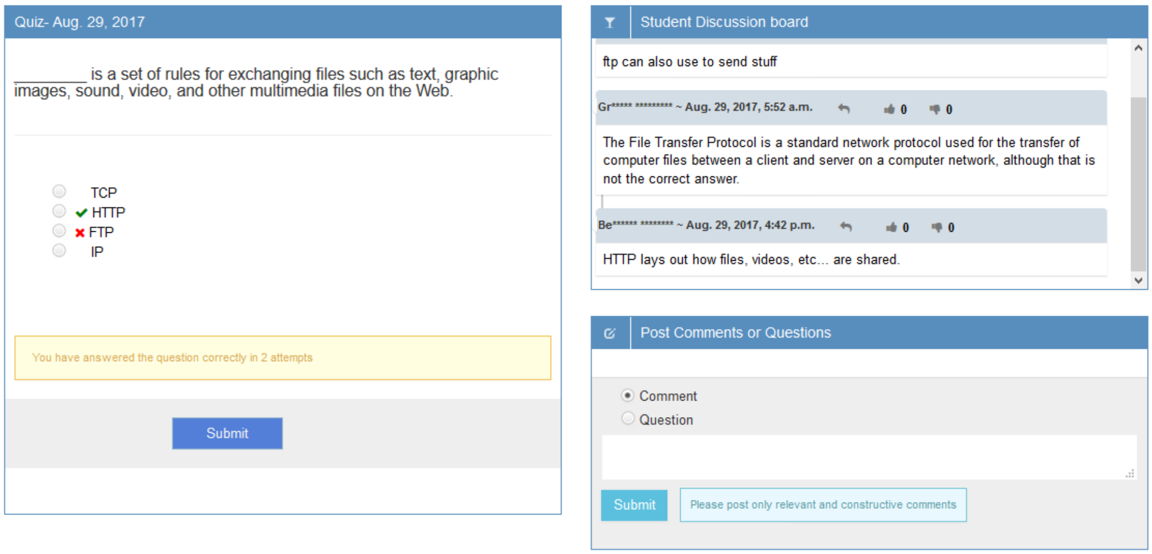

Figure 1: QuizIT daily programming system interface

3.2 Data Collection

The data of students enrolled for CSE 110 in Fall 2016, and who used QuizIT, were collected. CSE 110 is an introductory object-oriented Java programming course at undergraduate level in CIDSE at Arizona State University. QuizIT has 110 exercises organized across 18 topics (such as strings, expression, and method), each of which is labeled as either easy (71 exercises), moderate (30), or difficult (9). In 2016, 375 students were enrolled in CSE 110 but QuizIT was used by 187 students; 5963 correct attempts were recorded, and 4094 incorrect attempts. Exercises at the *easy* level were attempted 5562 times, *moderate* questions were attempted 3150 times, and *difficult* questions were attempted 1345 times. The timestamp at which the questions were attempted was also recorded.

3.3 Building Action Sequences

In this section, the conversion of each data point to an action sequence is discussed. The creation of action sequences was crucial because they represented discrete actions, which are suitable for differential pattern mining. Each action is a set of alphabets representing the particular event. The event is characterized using the difficulty level, correct or incorrect attempts, and duration between successive actions.

The process of building sequences of actions required rearranging the data belonging to the same student. The data were first sorted and grouped by student ID, then by the question ID (i.e., the questions attempted by the same student), and finally by timestamp sorted within each question ID. Then the difference in time for consecutive attempts on a specific question was computed. This measure represented the duration of each successive attempt if the student re-attempted the question.

After this, each event segment was labeled using alphabets and numbers denoting a particular action. Each action segment was labeled with a triple $\{L, F, D\}$, where L $\in$ $\{$*Easy*, *Moderate*, *Difficult*$\}$ represents the difficulty level, F $\in \{1, 2\}$ represents the correctness flag where 1 denotes a correct attempt and 2 denotes an incorrect one; and D $\in \{F, S, M, L, VL, XL\}$ represents the duration of the action. If the student attempted a particular question for the first time, it was denoted as first occurrence (F). For successive re-attempts, the terms *S*, *M*, *L*, *VL* and *XL* were used to indicate an action of small, medium, long, very long, and extremely long durations, respectively. For instance, *<Easy-1-F>* represents a question of easy difficulty that was answered correctly on the first attempt.

Most of the researchers in their prior work on behavioral mining divided the raw data into sessions to form action sequences. However, it was interesting to examine the changes in learning behavior over a semester, considering the intermittent breaks between the action sequences of a student. Therefore, a different way of labeling the durations was devised. To determine the cut-off points for each duration category, univariate k-means clustering in 1D using dynamic programming was used [30]. Among the 10057 data entries, 5344 entries were non-zero (i.e., there were 5344 sequential time data points which were not a first occurrence). Among these, 5299 data points fell below 1 hour; the rest of the data were treated as being of too long a duration and were excluded as outliers to achieve better convergence in the k-means.

The threshold was set at 3600 seconds to exclude more points as outliers and to prevent overfitting and retain the cut-off boundaries. The Bayesian information criterion (BIC) [24] was used for model selection, with models having lower BIC being preferred. The best BIC for four cluster centers was calculated at -39192.96. The resulting thresholds were 6.3287 seconds for small duration, 121.0022 seconds for medium duration, 394.4931 seconds for long duration, and 914.5216 seconds for very long duration. Finally, to create action sequences from these action segments for each student, the data were first sorted by timestamp and then grouped by student IDs.

Students received percentages and were graded on their performance in assignments, midterms, and finals in CSE 110. Using these data, each student's action sequence was assigned to a performance group. An action sequence was assigned to the high-performance group if the student's percentage fell above the median for the class percentage. Similarly, the action sequence was assigned to the low-performance group if the student's percentage fell below the median of the class percentage. The data were divided into these two groups because the differential sequential mining that would be employed uses two databases as inputs. Hence the division would help to highlight the difference between the learning patterns of higher and lower performing students.

3.4 Differential Pattern Mining

To identify patterns that were distinctive to either higher or lower performing students, the differential pattern mining technique was used. This technique was developed by Kinnebrew and Biswas [15] and it uses two sequence databases, known as the left database and the right database. The order of the left and right groups did not matter. The algorithm uses two important metrics to measure the support of patterns: *s-support* and *i-support*.

- **s-support**: This stands for "sequence support". Sequence support of a pattern is defined as the number of sequences that contain a particular pattern. Patterns that meet a set threshold for s-support are known as s-frequent patterns.
- **i-support**: This stands for "instance support". Instance support of a pattern is defined as the maximum number of times the pattern appears within a sequence without any overlap. The i-support of a pattern for a sequence database is commonly used as it basically represents the mean i-support value of the pattern for all sequences in both databases.

An example will help to illustrate the concepts of s-support and i-support. A sequence database has 5 sequences, with the first 3 sequences showing 1 instance of a specific pattern, and the last 2 sequences showing 4 instances of the same pattern. In this case, the pattern has an s-support of 5 and an i-support of 1 and 4 in the first and last sequence respectively. The mean i-support of the pattern for this sequence database will be 2.2.

The differential sequential mining algorithm shown in Figure 2 was implemented in Python to extract the differential patterns. The algorithm begins by finding all the patterns in the left and right sequence databases that meet the s-support constraint. To find the initial set of s-frequent patterns, an open-source data-mining library called SPMF [8] was used for the SPAM algorithm, which takes into account the gap constraint. The gap constraint means that between each pair of action segments in a frequent pattern, additional action segments up to the gap limit can be accommodated.

For instance, if the gap constraint is 2 and the sequence is *P*-*B*-*X*-*Q*-*S*-*R*, a frequent pattern $<P\text{-}Q\text{-}R>$ can be extracted. Incorporating gap constraints in mining algorithms is crucial because action sequences are noisy. A student's learning behavior is reflected in the action sequences, but the student also performs other actions in-between the relevant pattern – such as randomly attempting a different question out of boredom. This action is out of context of the student's learning behavior. A maximum gap constraint of 2 was used in this work.

In the next step, to compare frequent patterns across both databases, the mean i-support of each pattern was computed in the left and right databases. Computation of i-support also considers the maximum gap constraint of 2 to allow for noise interspersed between the action sequences. After this, the t-test was performed for all the s-frequent patterns. This step determined whether an i-support value for a particular pattern in the left database differed significantly from the i-support value of the same pattern in the right database. If the p-value of the t-test result was below the set p-value threshold (based on confidence intervals), that pattern was considered to be differentially frequent.

Input : $left$ – Left dataset (set of action sequences)
$right$ – Right dataset (set of action sequences)
Parameters: s_{thresh} – s-frequency (support) cutoff for frequent patterns
gap – maximum gap allowed between actions in a pattern
$regex$ – regular expression to match for limiting patterns
p_{thresh} – t-test p value for differential patterns
Output : Ordered lists of patterns in four categories by differential frequency

$sFreqPtrns_{left} \leftarrow \texttt{SPAMc}(left, s_{thresh}, gap, regex)$
$sFreqPtrns_{right} \leftarrow \texttt{SPAMc}(right, s_{thresh}, gap, regex)$
for all $ptrn \in sFreqPtrns_{left} \cup sFreqPtrns_{right}$ **do**
  $iSupport_{ptrn}(left) \leftarrow$ Count regular expression matches of $ptrn$ in each sequence in $left$
  $iSupport_{ptrn}(right) \leftarrow$ Count regular expression matches of $ptrn$ in each sequence in $right$
  if $\texttt{t-test}(iSupport_{ptrn}(left), iSupport_{ptrn}(right)) \leq p_{thresh}$ **then**
    if $ptrn \in sFreqPtrns_{left}$ AND $ptrn \in sFreqPtrns_{right}$ **then**
      if $\texttt{Mean}(iSupport_{ptrn}(left)) > \texttt{Mean}(iSupport_{ptrn}(right))$ **then**
        $ptrns_{bothLeft} = ptrns_{bothLeft} + ptrn$
      else
        $ptrns_{bothRight} = ptrns_{bothRight} + ptrn$
      end if
    else if $ptrn \in sFreq_{left}$ **then**
      $ptrns_{left} = ptrns_{left} + ptrn$
    else
      $ptrns_{right} = ptrns_{right} + ptrn$
    end if
  end if
end for
{Sort categorized patterns by i-frequency difference (left - right)}
$\texttt{SortDesc}(ptrns_{Left})$; $\texttt{SortDesc}(ptrns_{bothLeft})$; $\texttt{SortAsc}(ptrns_{bothRight})$; $\texttt{SortAsc}(ptrns_{Right})$;
return $ptrns_{Left}, ptrns_{bothLeft}, ptrns_{bothRight}, ptrns_{Right}$

Figure 2: Differential Sequential Mining Algorithm [15]

Although this approach employs multiple t-test comparisons between the databases, it is important to note that the t-test was not used to statistically prove that the left and right databases differ. Also, neither Bonferroni or other corrections were used to determine the p-value threshold for rejecting the null hypothesis. Rather, it was used as an exploratory-analysis heuristic to extract more interesting patterns for specific characteristics that were relevant to the two student groups. After that, the mean values for i-support for the left and right databases were compared to see which patterns emerged more in one group. This comparison yielded four types of differentially frequent patterns:

- s-frequent in both databases but mean i-support higher in left database ($ptrns_{bothLeft}$)
- s-frequent in both databases but mean i-support higher in right database ($ptrns_{bothRight}$)
- s-frequent only in left database ($ptrns_{Left}$)
- s-frequent only in right database ($ptrns_{Right}$)

Only the last two cases ($ptrns_{Left}$ and $ptrns_{Right}$) were considered as they were crucial for distinguishing high-performing students from the weaker ones. The high-performing students were used as the left database and low-performing students as the right one. The s-support was set at 0.3 (i.e., for a pattern to be s-frequent, it must appear in at least 30% of the sequences in a database). The p-value threshold was set at 0.05, or 95% confidence level.

3.5 Building Predictive Models

Using differential sequential mining, 35 s-frequent patterns were found. Among these, 23 patterns belonged to higher performing students and 12 to lower performing students. The aim was to use these 35 patterns to build a predictive model that could predict the performance of a student. The patterns were first represented as features with binary coding: the feature was labeled as 1 if the pattern was present in a student's action sequence and as 0 if not. To remove patterns that were strongly correlated with each other and to avoid overfitting, an algorithm named correlation-based feature selection (CFS) [10] was used with 10-fold cross-validation. The CFS algorithm helps in identifying feature subsets having the most weights. Features selected by CFS in more than 5 out of 10 folds were included in the final features for training the predictive model. From CFS, 8 high-performing patterns and 2 low-performing patterns were retrieved. Table 1 shows all the patterns with their p-values and mean i-support values.

Table 1: Patterns filtered by CFS algorithm

Pattern	**Category**	**Mean i-support**	**P-value**
<Easy-1-F> <Medium-2-F> <Medium-1-M>	HIGH	0.0107	0.0001
<Medium-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	HIGH	0.0072	0.0002
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	HIGH	0.0221	0.0001
<Easy-1-F> <Easy-1-F> <Difficult-1-F>	HIGH	0.0101	0.0005
<Easy-1-M> <Medium-1-F>	HIGH	0.0093	0.0005
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Medium-1-F>	HIGH	0.0154	0.0008
<Medium-1-F> <Easy-1-F> <Easy-1-F>	HIGH	0.0286	0.0002
<Easy-1-F> <Easy-1-F> <Medium-1-F>	HIGH	0.0259	0.0041
<Medium-2-F> <Medium-2-S> <Medium-1-S> <Medium-2-F>	LOW	0.0086	0.0082
<Easy-2-S> <Medium-2-F>	LOW	0.0086	0.0452

These 10 features were used to build a linear regression model that can predict a student's performance in the course. Random forest and logistic regression classifiers were used to predict whether a student belonged to the high-performing group or the low-performing group. Linear models and random forest classifier are relatively easy to interpret, as the weights of the features provide deep understanding of the features' importance. The results of the models are discussed in the next chapter.

Chapter 4

EVALUATION

4.1 Exploring Learning Strategies with Behavioral Patterns

Among the extracted 10 patterns, seven learning behaviors were observed, based on students' problem-solving patterns. They were labelled as follows: persistent-practicing, jump-forward-progression, steady-progression, experimental-progression, jump-backward-progression, struggling, and withdrawal. The pattern behaviors are summarized in Table 2.

Table 2: Patterns classified into various learning strategies

Patterns	**Behavior**
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	Persistent-practicing
<Easy-1-M> <Medium-1-F>	Steady-progression
<Easy-1-F> <Medium-2-F> <Medium-1-M>	Experimental-progression
<Easy-1-F> <Easy-1-F> <Medium-1-F>	Jump-forward-progression
<Easy-1-F> <Easy-1-F> <Difficult-1-F>	Jump-forward-progression
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Medium-1-F>	Jump-forward-progression
<Medium-1-F> <Easy-1-F> <Easy-1-F>	Jump-backward-progression
<Medium-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	Jump-backward-progression
<Medium-2-F> <Medium-2-S> <Medium-1-S> <Medium-2-F>	Struggling
<Easy-2-S> <Medium-2-F>	Withdrawal

The learning behavior are further described in detail as follows:

1. Persistent-practicing is shown as *<Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>*. This behavior is displayed when a student repeatedly solves one or more problems of the same difficulty level correctly, all on the first attempt. This behavior aligns with QuizIT's design rationales to help students space their learning opportunities, by practicing simple problems distributively.

2. Jump-forward-progression behavior, such as *<Easy-1-F>* *<Easy-1-F>* *<Medium-1-F>*, is seen among three patterns. In this behavior, students repeatedly solve one or more problems of the same difficulty level correctly; then they progressively solve a problem of higher difficulty level, all on the first attempt. This behavior demonstrates the testing effects of QuizIT in preparing students to solve harder problems in future after practicing simpler ones. Among 10 observed patterns, three jump-forward-progression patterns represented this behavior.
3. Steady-progression, *<Easy-1-M>* *<Medium-1-F>*, illustrates that students take their time when attempting a question, although they did not answer it correctly on the first try. Eventually they solved the problem correctly and moved on to another more complex problem.
4. Experimental-progression, *<Easy-1-F>* *<Medium-2-F>* *<Medium-1-M>*, occurs when a student correctly solves a question of a certain difficulty level but makes mistakes when attempting a question of a higher difficulty level. In this case, the student re-attempts the incorrectly solved question, takes a while to comprehend the question, and finally solves the question correctly. This behavior is particularly interesting as it shows the student's effort to learn from mistakes while solving a tougher problem and eventually solving it.
5. Jump-backward-progression behavior, such as *<Medium-1-F>* *<Easy-1-F>* *<Easy-1-F>*, is seen among two patterns. The student correctly solves a relatively difficult question and then correctly solves one or more problems of a lower difficulty level, all on the first attempt. This behavior suggests that some student may not understand the medium level problem. Their first attempt for medium problem can be a guess, so they jump backward to restart practicing. This

pattern is a good evidence to show that these students are working on their concepts, and the system can adapt to different learning pace of students.

6. Struggling, *<Medium-2-F> <Medium-2-S> <Medium-1-S> <Medium-2-F>*, expresses the behavior in which the student solves questions of same difficulty level repeatedly. Among all the attempts, there are only limited correct ones in the middle. This behavior suggests that the student struggled to get the answer right, and may have applied trial-and-error strategy to get the limited correct answer. This is because from the pattern it can be seen that the student takes too short a time to re-attempt the questions. This is particularly a poor learning behavior as the student does not try to think and work out on the problem.

7. Withdrawal, *<Easy-2-S> <Medium-2-F>*, is a behavior where the student attempts a question they attempted before, but then they skip the question if their answer is incorrect so as to move on to the next question of a different difficulty level, which is again solved incorrectly. This is another undesirable behavior as it shows that students simply give up on the problem if they are unable to solve it. This learning pattern can prove detrimental to performance in the long run and this needs to be addressed by the instructor.

The distribution of learning behaviors across the two performance groups is shown in Figure 3. All learning strategies are used by students in both groups. However, the strategies of persistent-practicing, jump-forward-progression, steady-progression, experimental-progression, and jump-backward-progression are mainly used by higher performing students. Struggling and withdrawal are strategies used by lower performing students.The most observed behavior among high performers is persistent-practicing, which was used by 63.29% of top students. Jump-backward-progression and jump-forward-progression were used by 50% and 47.25% of high performers, respectively.

This finding shows that half of the top students tended to correctly solve an easier question after attempting a tougher question, which suggests a confidence gain regarding specific topics. Withdrawal was the behavior seen most often among low-performing students, at 39.24%, followed by struggling, at 34.17%. Interestingly, the third most used strategy in the lower performing group was persistent-practicing, at 29.11%. This finding supports the design rationale of QuizIT regarding students' need to practice solving simple questions.

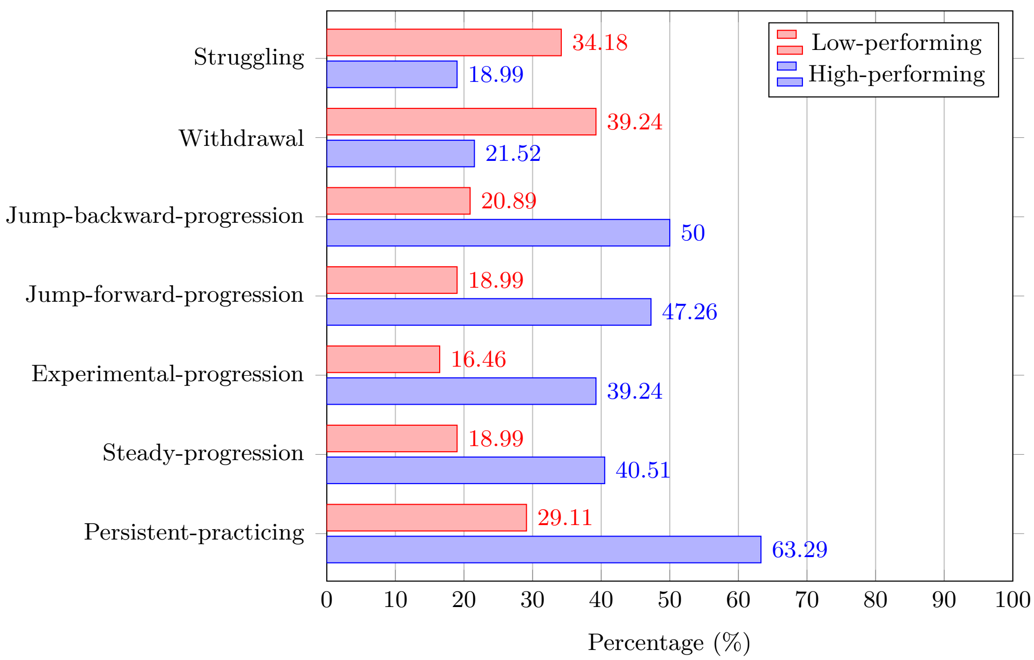

Figure 3: Learning behavior distribution across the two performance groups

The distribution of learning behaviors with respect to final grades is shown in Figure 4. The students' grades were classified according to four main grade lists, namely A, B, C and D. The bar chart shows that 64.47% of A-grade students adopted persistent-practicing. Jump-backward-progression and jump-forward-progression were

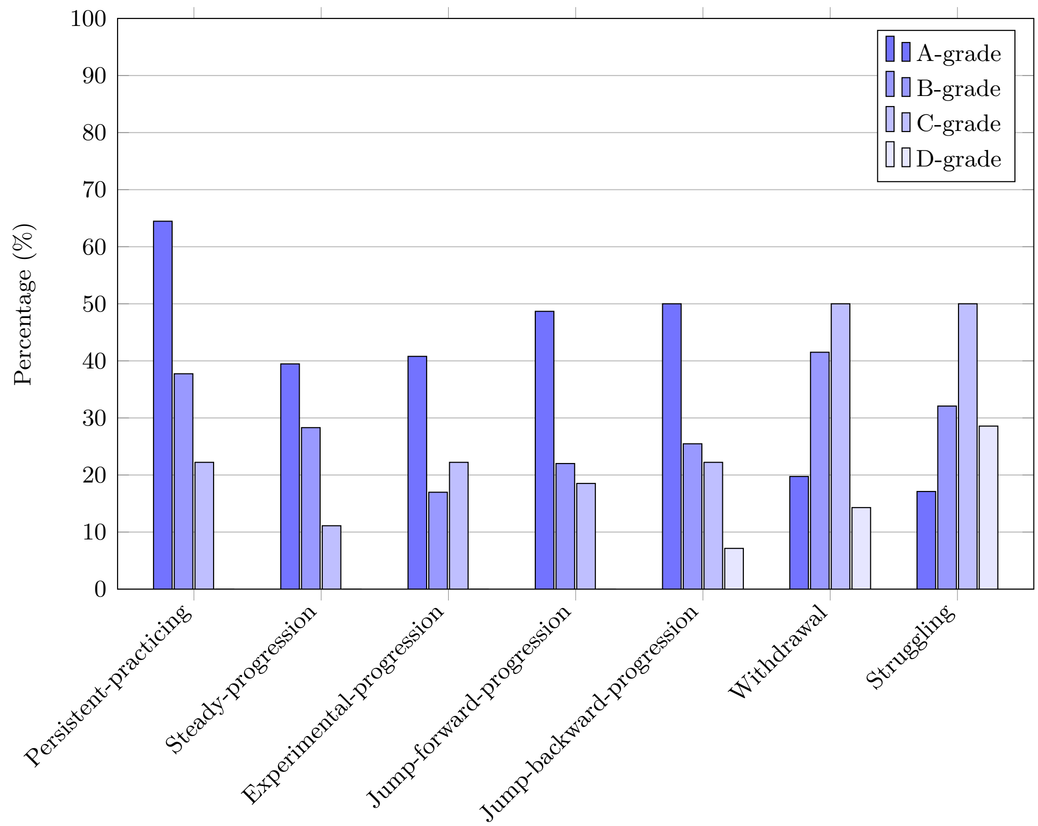

Figure 4: Learning behavior by students' final grades

used by 50% and 48.68% of A-graders respectively. This result shows that most A-graders were using five strategies pertaining to the highly performing group.

Among the B-graders, 37.7% of students showed persistent-practicing. However, withdrawal was the most prominent behavior in this group (41.5%) and struggling was the third most prominent behavior (32%). This finding shows that more B-grade students employed negative behaviors than positive ones. Hence, most of them were average students and fell into the group of lower performing students. Among the C-graders, 50% of the students who withdrew also struggled, which means that both these behaviors contributed negatively to the students' performance. Interestingly, more C-graders showed experimental-progression behavior than B-graders. With regard to the D-graders, most both withdrew and struggled, although one D-grader had adopted jump-backward-progression as a learning strategy. This suggests that student should have followed QuizIT's progression instead of venturing too fast too soon, especially for lower-performing students.

4.2 Exploring Models Based on Behavioral Patterns

The patterns and learning behaviors (discussed in the previous section) were used to build models to predict a student's performance group and final scores. The purpose of the models was to gain in-depth knowledge about individual learning patterns and learning behaviors which are influential in determining students' performances. This section discusses the different classifiers used with hyperparameter settings. The individual models are evaluated in terms of assigning importance to patterns and behaviors, and the models are compared to assess the best predictor.

The R^2 score was used as the metric for assessing the linear regression model. Precision, recall, and F1 scores were used to assess the logistic regression (LR) and random forest (RF) classifiers. For all models, stratified 10-fold cross-validation was performed because the dataset was small and to prevent overfitting while training the models. The R^2, F1, precision, and recall scores were yielded by the weighted average of all the scores for 10-fold cross-validation. No class imbalance affected the models as the data were divided according to the median. The models were trained for both patterns and behaviors to see whether different models imparted different information.

4.2.1 Individual Pattern Based Predictive Models

For linear regression, the student's percentage was rounded to a whole number and used as the dependent variable; the student's patterns (obtained through CFS) were used as the independent variables. The linear regression model did not work well, as R^2 was negative (-3.6473). Hence, the linear regression model was not suitable for predicting students' percentage scores. The coefficients of linear regression are shown in Table 3. The patterns with **bold** coefficients had the maximum weight for the model and patterns with ***italicized bold*** had minimum weights for the model. For both LR and RF classifiers, the independent variables are same as those used in the linear regression model. However, the dependent variable was a binary label that represented which group the student belonged to in terms of performance.

Table 3: Coefficients of models based on individual patterns

Pattern	Linear Regression	Logistic Regression	Random Forest
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	**6.0481**	**0.2899**	**0.1549**
<Easy-1-M> <Medium-1-F>	3.4994	0.1768	***0.0400***
<Easy-1-F> <Medium-2-F> <Medium-1-M>	2.7910	0.2457	0.0687
<Easy-1-F> <Easy-1-F> <Medium-1-F>	2.2614	0.1847	0.0990
<Medium-1-F> <Easy-1-F> <Easy-1-F>	2.0549	0.2224	0.0961
<Easy-1-F> <Easy-1-F> <Easy-1-F> <Medium-1-F>	1.2447	0.2721	0.1335
<Easy-1-F> <Easy-1-F> <Difficult-1-F>	-1.2560	0.2328	0.0617
<Medium-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F> <Easy-1-F>	-1.5811	0.2835	0.1220
<Easy-2-S> <Medium-2-F>	-2.6576	***-0.4972***	0.1261
<Medium-2-F> <Medium-2-S> <Medium-1-S> <Medium-2-F>	***-5.5565***	-0.4261	0.0981

For LR, a liblinear solver was used with L2 penalty and 100 iterations. The logistic model was trained with various C-values using grid-search cross-validation with 10 folds. The training and validation scores with different C-values are plotted in Figure 5. From the validation curve it can be observed that for C of < 0.1, both training and validation scores were low, which caused underfitting. For values greater than

0.1, the training scores increased but the validation scores decreased. Therefore, the best model was for C = 0.1 with an F1 score of 0.6715.

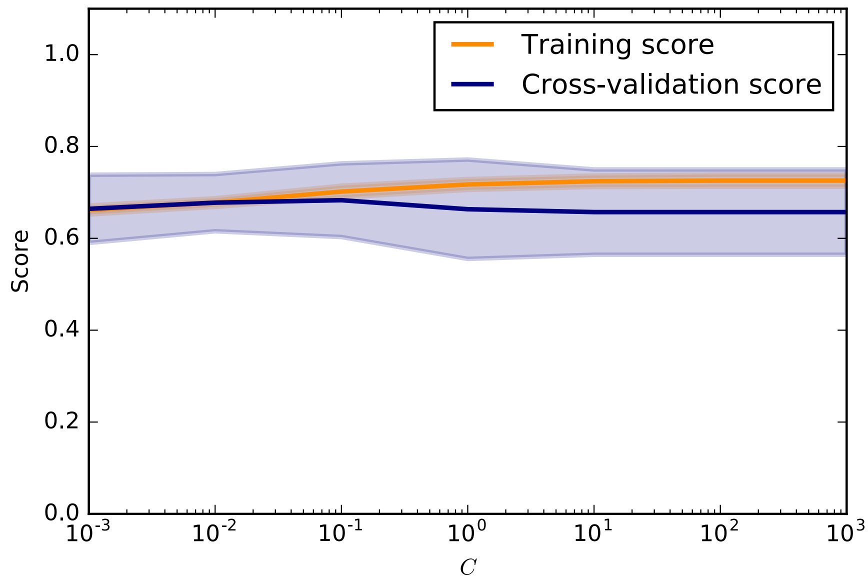

Figure 5: Validation curve for pattern-based LR model

The coefficients of LR, also known as *logits*, are shown in Table 3 and are graphically represented in Figure 6. For better interpretation of these coefficients, the logits were converted to probabilities using the following equation:

$$Probability = \frac{\exp(logit)}{1 + \exp(logit)}$$

The probabilities are shown in Figure 7. The difficulty level for patterns in Fig. [6, 7, 8] are denoted by the initials $\{E, M, D\}$ for conciseness. This model is a predictor of whether a student belonged to a high-performing group or not. As expected, the coefficients for the patterns used by most high-performing students were weighted as positive, and those for patterns used by low-performing students were negative while training the model.

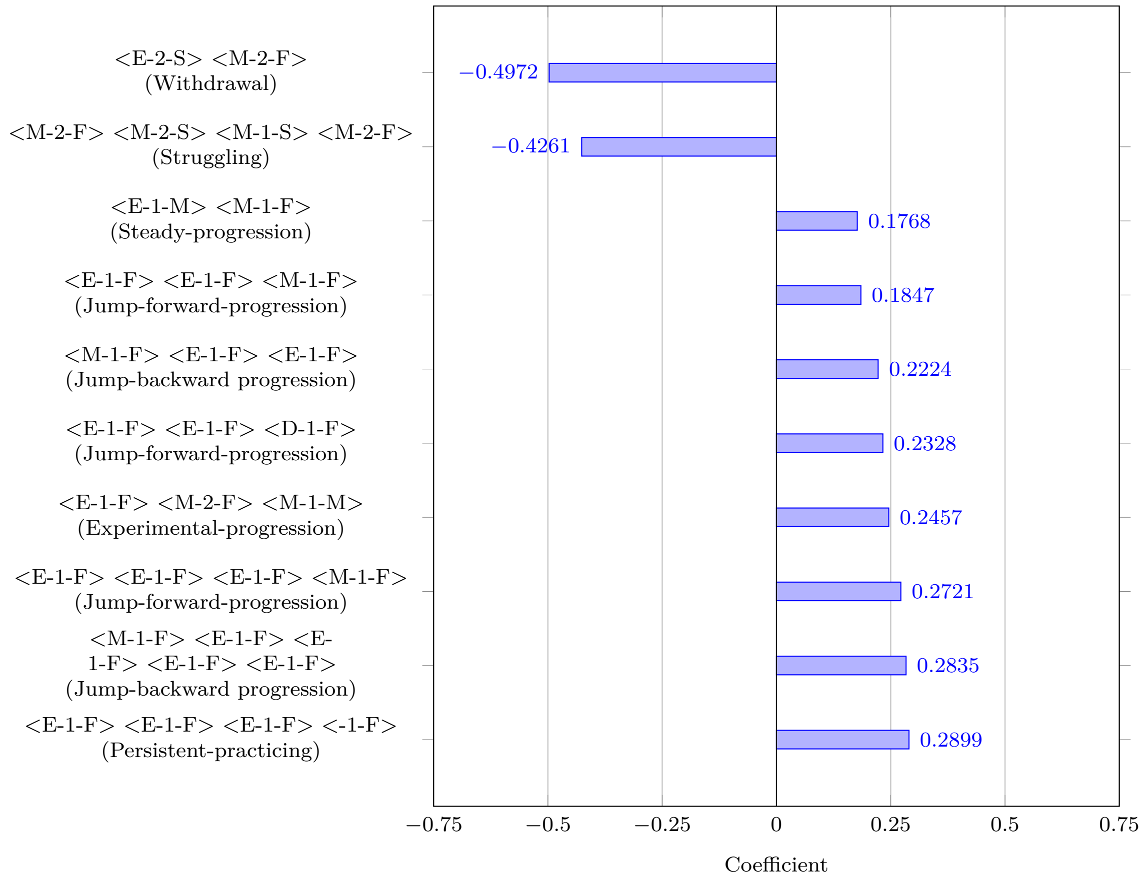

Figure 6: Logistic Regression coefficients for individual patterns

Persistent-practicing obtained the highest logit weight, 0.2899, and the greatest probability (0.572) for predicting student belonging to high-performing group. This result shows that the design rationale of QuizIT, namely to provide students with learning opportunities by solving simpler questions, was given the greatest importance in the model. Withdrawal and struggling were two patterns with negative logits, at -0.4972 and -0.4261 respectively. Withdrawal had the highest probability (0.6218), followed by struggling (0.6049), for predicting which students belonged to the low-performing group.

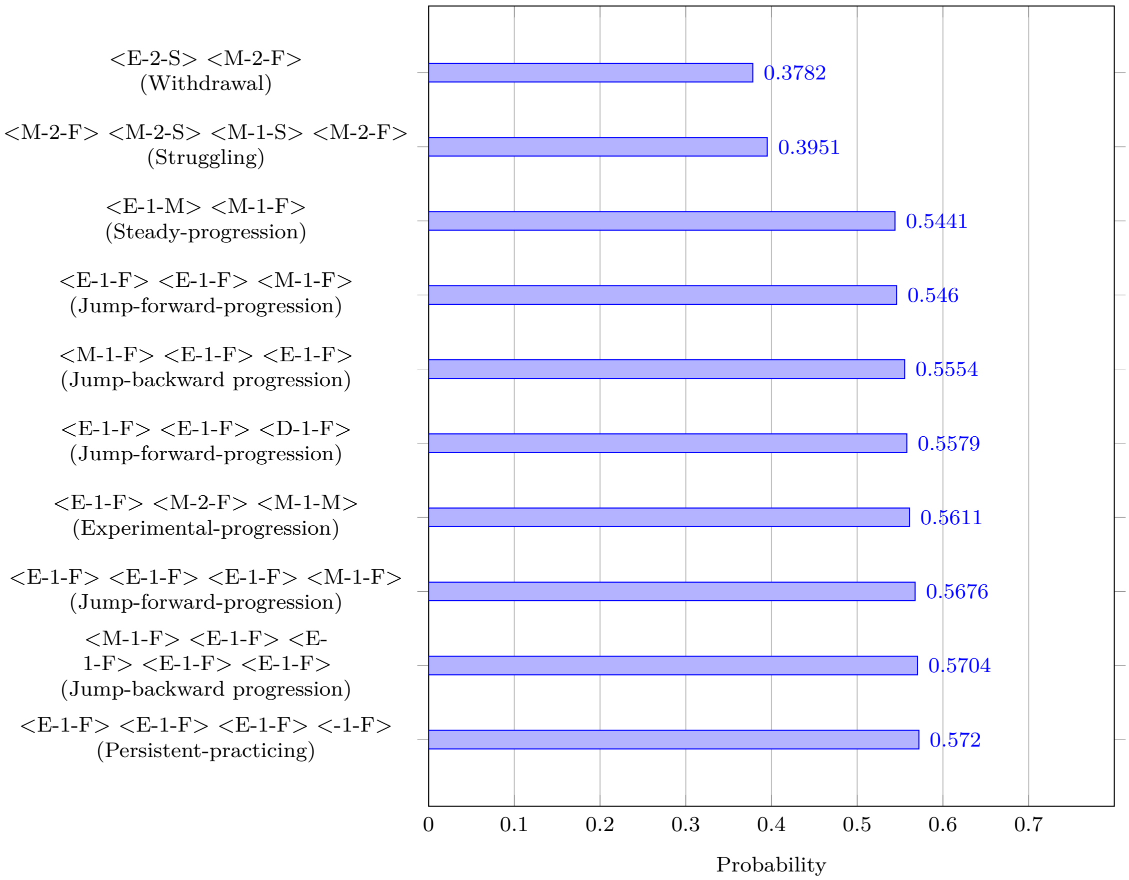

Figure 7: Probability of patterns for high-performance in LR model

A random forest (RF) classifier was also trained to gain deeper insight into which patterns might prove useful for improving students' learning skills. The RF model was hypertuned using grid-search cross-validation with 10 folds on parameters such as the number of trees, maximum depth of tree, and minimum number of samples for leaf node and split internal node. The best configuration was as follows:

- Random forest estimators : 500
- Minimum samples leaf : 7
- Minimum samples split : 2

- Split criterion : gini
- Maximum tree depth : 3

This model achieved an F1 score of 0.6777. The weighted patterns of the RF classifier, ordered by decreasing importance, are shown in Figure 8 and the same coefficients are shown in Table 3. As can be seen, the RF classifier gave most importance to persistent- practicing, as was the case for both logistic and linear regression. Interesting, this classifier also termed withdrawal as the third most important feature. The model attempted to assign high importance to both positive and negative behavior patterns in predicting students' performances.

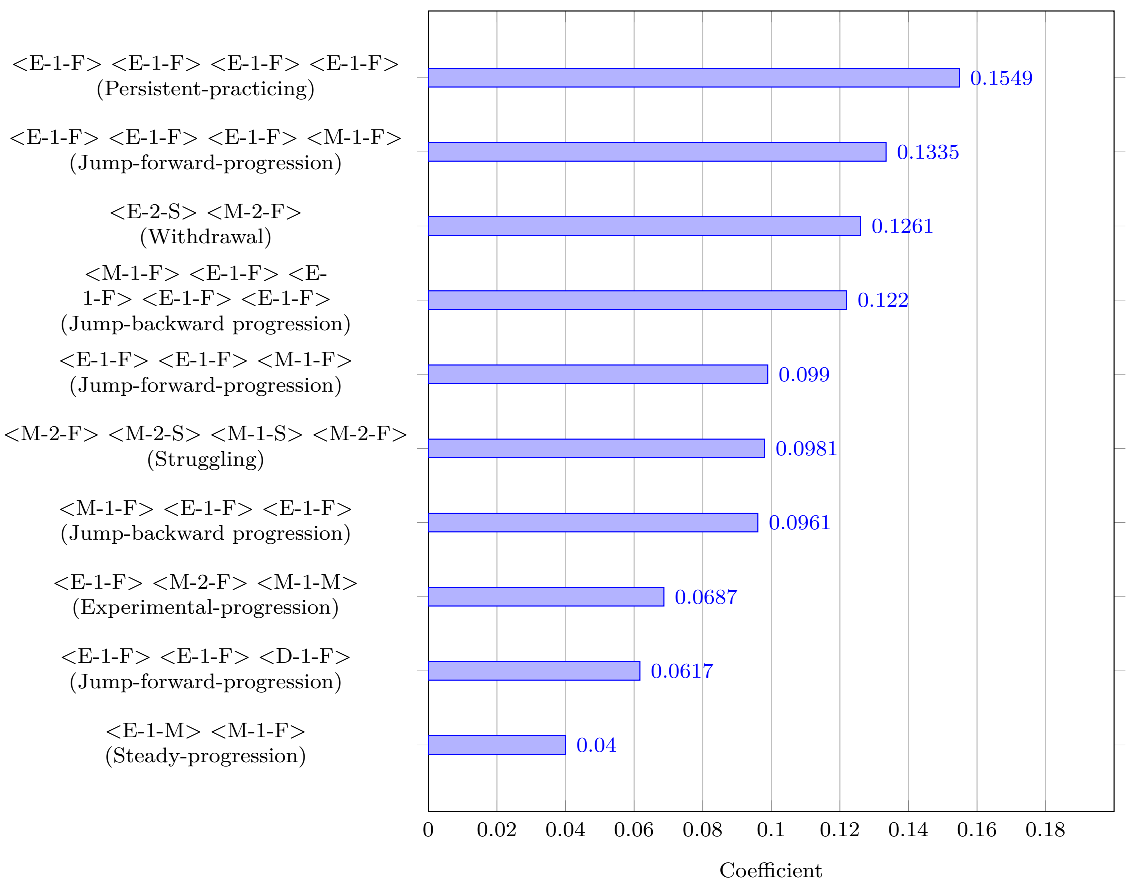

Figure 8: Importance of features defined by pattern-based RF classifier

The jump-forward-progression pattern *<Easy-1-F> <Easy-1-F> <Medium-1-F>* was the fifth most important pattern in the RF model. However, it was the second least important positive pattern in the LR model. Experimental-progression was the fourth most important pattern in LR but the third least important pattern in RF classifier. Steady-progression was assigned the least important positive pattern in both the LR and RF models.

4.2.2 Learning Behavior Based Predictive Models

For linear regression, the student's percentage (rounded to a whole number) was used as the dependent variable and students' behaviors were used as independent variable. The R^2 score was -3.6477, even worse than that of the pattern-based model. Therefore, the linear regression model was not suitable for predicting students' percentage scores using learning behaviors. The coefficients of linear regression are shown in Table 4. (As before, **bold** coefficients represent maximum weights and ***italicized bold*** represent minimum weights for behaviors in the model.)

Table 4: Coefficients of models based on learning behavior

Behavior	Linear Regression	Logistic Regression	Random Forest
Experimental progression	2.6905	**0.8656**	0.1105
Jump-backward progression	2.327	0.3955	0.1375
Jump-forward progression	0.5141	0.6691	**0.2022**
Persistent practicing	**6.2809**	0.6723	0.1981
Steady progression	3.9269	0.5375	***0.0854***
Struggling	***-5.5138***	-1.0988	0.1226
Withdrawal	-2.6402	***-1.1758***	0.1436

The dependent variable for the LR and RF models was a binary label – which represented whether a student showed the particular behavior or not. Liblinear solver was again used with L2 penalty and 100 iterations for LR. As shown by the validation curve in Figure 9, the best C-value was 0.1. The model at C = 0.1 had an F1 score of 0.6854. The coefficients of LR are shown in Table 4, and Figure 10 provides a bar chart representation. The probability for each behavior is represented in Figure 11. The logits of behavioral strategies used by most high-performing students were weighted as positive.

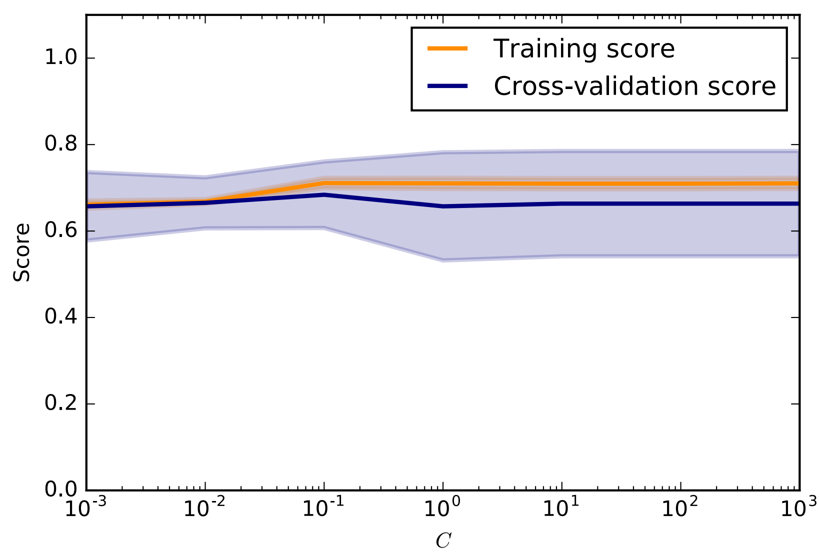

Figure 9: Validation curve for learning behavior based LR model

Interestingly, in this LR model experimental-progression obtained the highest probability (0.7038) for predicting whether a student belonged to the high-performing group. Persistent-practicing was second, whereas it was the most important in the LR model for patterns. Withdrawal and struggling had the lowest probabilities at 0.2358 and 0.25 respectively; these behaviors were better for predicting low-performing students. In addition, jump-backward- progression was termed the least important positive behavior by this LR model, with a probability of 0.3955.

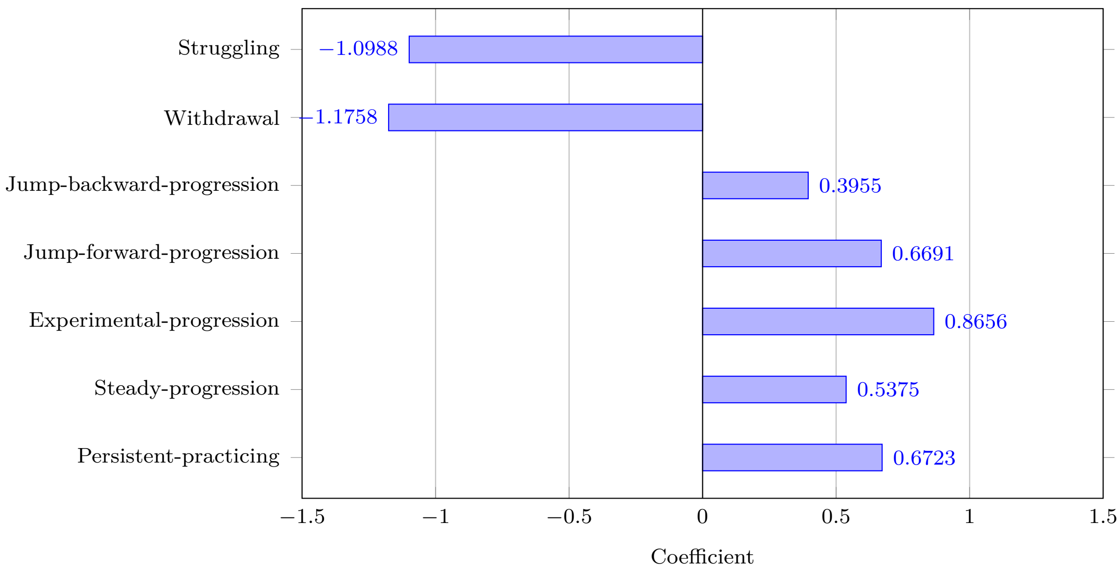

Figure 10: Logistic Regression coefficients for learning behaviors

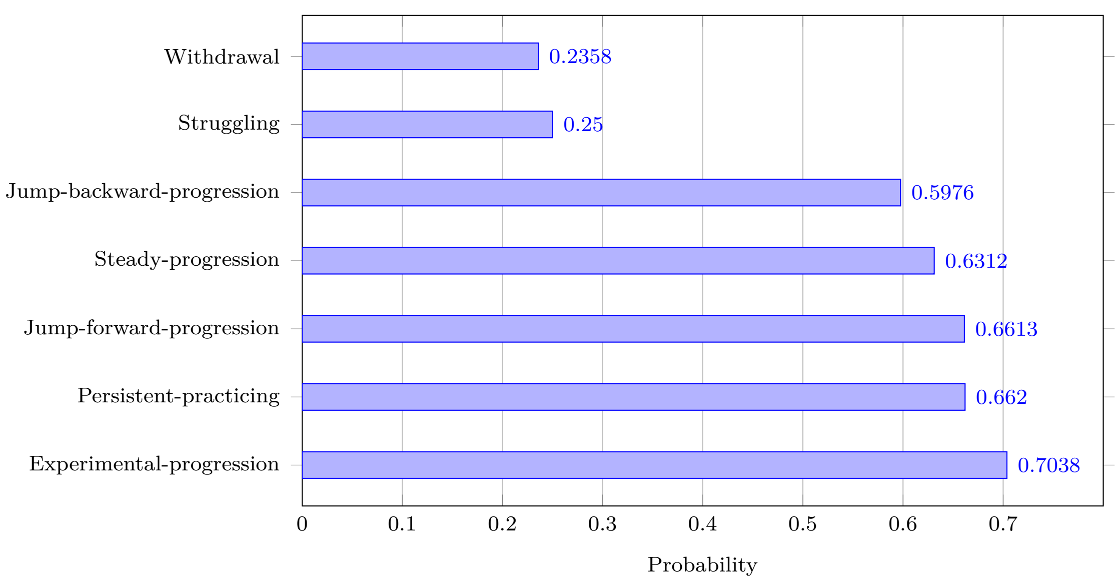

Figure 11: Probability of learning behaviors for high-performance in LR model

For the RF model, the best configuration after hypertuning using a grid search with 10-fold cross-validation was as follows:

- Random forest estimators : 200
- Minimum samples leaf : 3
- Minimum samples split : 3
- Split criterion : gini
- Maximum tree depth : 3

This model achieved an F1 score of 0.6644. The weighted patterns of the RF classifier, ordered by decreasing importance, are shown in Figure 12 and the same coefficients are listed in Table 4.

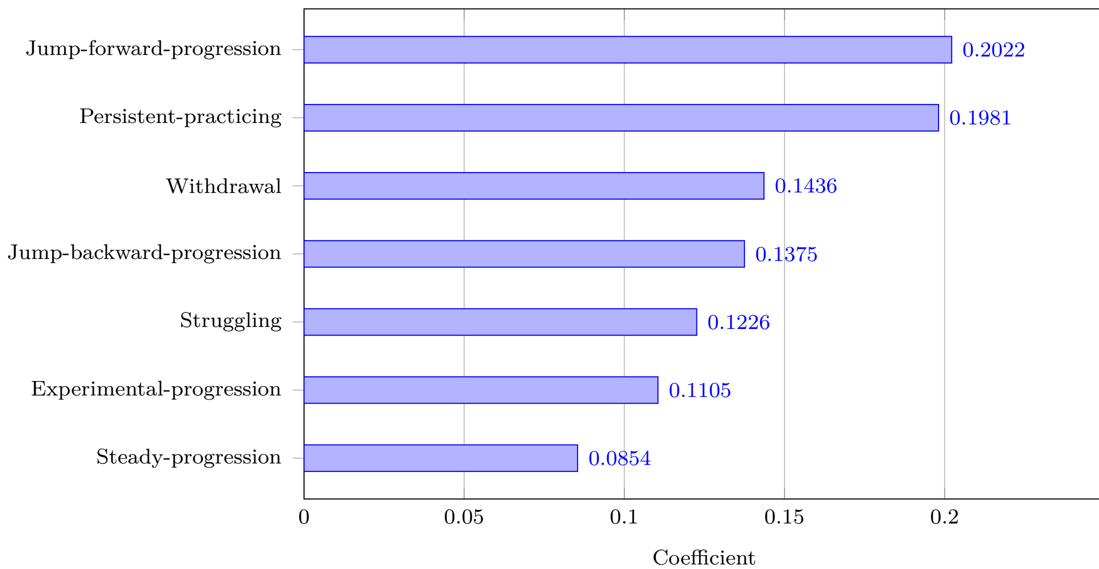

Figure 12: Importance of features defined by behavior-based RF classifier

This RF classifier gave the most importance to jump-forward-progression (compared with persistent-practicing in the pattern-based RF model). Withdrawal was the third most important behavior in this model; steady-progression was the least important pattern (as in the previous RF model). Thus, the top three behaviors were the same in both RF models. This shows that both RF classifiers were relatively consistent in computing the importance of behaviors, compared with their LR counterparts.

Another interesting observation was that experimental-progression was rated second least important in predicting students' behavior, whereas it was the most important in the LR model.

4.2.3 Model Comparison

The LR and RF models, for both individual patterns and learning behaviors, were evaluated in the previous section. Table 5 summarizes the metrics of all the classifiers as well as the means and standard deviations of their validation scores. Figure 13 shows the precision, recall, and F1 scores for all the models. As shown in the table and figure, the LR model using learning behaviors as independent variables was the best classifier model; it achieved the best precision at 69.93%, recall at 69.11%, and F1 score of 0.6854. However, the classifier model with the highest mean validation score and lowest standard deviation of validation scores was the RF model using individual patterns as independent variables. This was the second best model, with an F1 score of 0.6777.

Table 5: Comparison of metrics in LR and RF models

Model	**Precision**	**Recall**	**F1-score**	$\mu_{\textbf{Validation Score}}$	$\sigma_{\textbf{Validation Score}}$
LR - Pattern	0.6912	0.6777	0.6715	0.6840	0.0800
LR - Behavior	0.6993	0.6911	0.6854	0.6840	0.0490
RF - Pattern	0.6979	0.6839	0.6777	0.7110	0.0390
RF - Behavior	0.6837	0.6714	0.6644	0.6840	0.0770

Table 6 provides the statistical measures of all the classifiers. The metric score differences between these classifiers were not significant. The mean F1 score of all the classifiers was 0.6748 and the standard deviation was only 0.0089.

Table 6: Mean and standard deviation of metrics of all classifiers

Measure	**Precision**	**Recall**	**F1-score**
Mean (μ)	0.6930	0.6810	0.6748
Standard Deviation (σ)	0.0072	0.0084	0.0089

This result indicates that the classifiers accurately allocated slightly more than two-thirds of students to the right performance group.

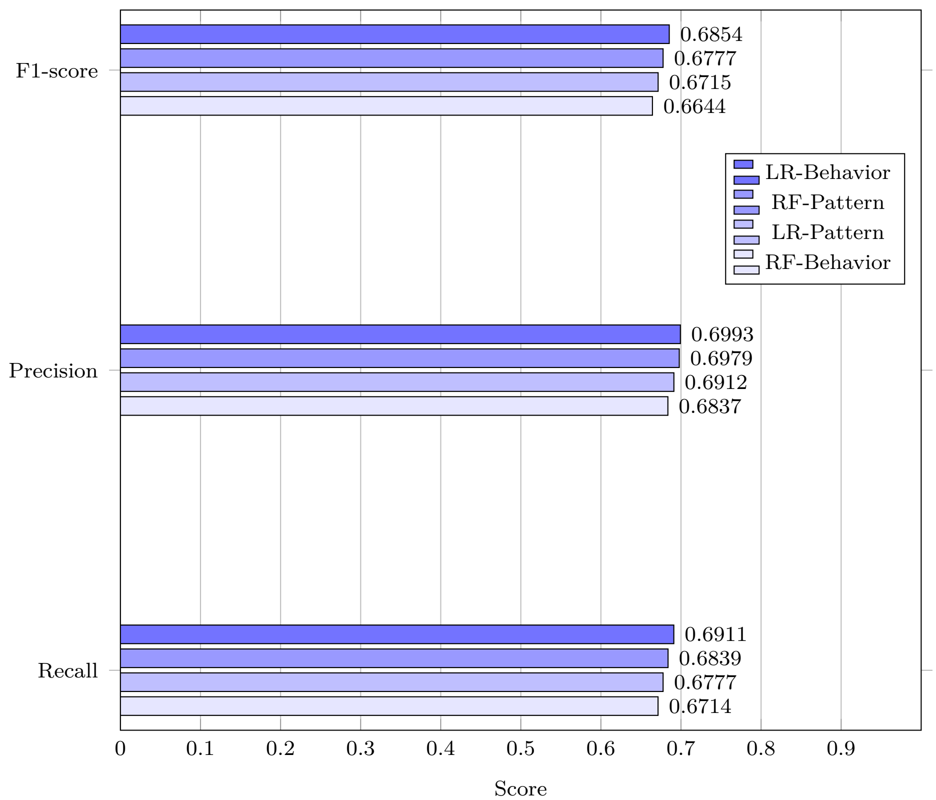

Figure 13: Comparison of classification model metrics

Chapter 5

CONCLUSION

The purpose of this thesis was two-fold. The first aim was to understand the learning strategies adopted by students to solve programming questions in QuizIT, and the second aim was to use data on these behaviors to build predictive models to determine a student's performance. Descriptive analysis of learning behavior was based on class performance to understand the adaptation of behaviors by different levels of students. Classification models were used to gain insight into individual learning patterns and learning behaviors which played a role in determining students' performance. These models will help instructors to build a robust system to address a student's weaknesses. These models can also be used to build recommendation systems customized to students' situations, to provide them with suggested learning paths to enhance their performance.

5.1 Summary

This thesis presents an application of data mining and machine learning techniques to data extracted from QuizIT programming system. These data represented the state of students' programming solving activities while using QuizIT. These states were recorded through the questions that were attempted, their complexity level, the correctness, and the timestamp. Each of the actions were a combination of numbers and letters characterized by its correctness, complexity and duration. These features were used to represent student's problem-solving behavior based on the action sequences. Sequential pattern mining was used to extract frequent patterns in analyzing all these behaviors to uncover students' problem solving patterns over the course of entire semester. Additionally, the action sequences were examined according to students' performances in their course work, labeled as higher and lower performing groups. Differential sequential mining technique was applied on sequences belonging to each groups to identify sequential mined patterns which were more prominent in one group compared to the other.

Seven main learning behaviors were discovered using these patterns. Among these, five behaviors were prominent in the high-performing group and two were prominent in the low-performing group. The patterns were also used as features to build models to predict students' membership of a performance group. Logistic regression (LR) and random forest (RF) classifiers were trained on individual patterns as the independent variables, and thereafter on learning behaviors as the independent variables. The rationale behind choosing these classifiers was that they are good at showing features that are important for the classification of data points. The best model was an LR model trained on behaviors, which achieved an F1 score of 0.6854.

The models provided insight into which patterns and behaviors correlated most strongly with students' performances. They also helped in identifying cognitive tendencies which were detrimental to students in their coursework. This information can help instructors to address the issues faced by their students and to suggest a better learning path.

5.2 Limitations and Future Work

A major limitation of this work is that the discovered patterns and behaviors were not helpful in building a predictive model about students' final percentages. Linear regression models were trained on both patterns and behaviors, but the best R^2 score was only -3.6473. One of the main reasons was not having enough data points and features to define a polynomial relationship between patterns and behaviors, which could have placed students' scores on a continuous range.

More data need to be collected by increasing student participation in the QuizIT platform. Currently, only 49.86% of students enrolled in coursework also use QuizIT. This proportion can be increased by providing incentives for those enrolled in courses and by increasing the interaction with users by sending them weekly newsletters to keep them engaged with the system. If data is collected on a daily basis, a LSTM based model (with time step being one day) can be created to predict students' future performance at any particular time. Different methods for building action sequences can be explored, such as topic difficulty, question IDs, and session IDs. Such details can be explored to build different feature vectors, which might increase the performances of the existing classification models. Other machine learning models, like AdaBoost or support vector machines with different kernels, can also be explored for their ability to predict students' performance groupings. These models can be integrated with the dashboard in the later versions of QuizIT. The students' data could then be used to assess the students' patterns and behaviors so as to notify the students about their weaknesses, and suggest alternatives for improving their performance.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. "Mining sequential patterns". In: *Data Engineering, 1995. Proceedings of the Eleventh International Conference on.* IEEE. 1995, pp. 3–14.

[2] Jay Ayres et al. "Sequential pattern mining using a bitmap representation". In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM. 2002, pp. 429–435.

[3] Gautam Biswas et al. "Measuring self-regulated learning skills through social interactions in a teachable agent environment". In: *Research and Practice in Technology Enhanced Learning* 5.02 (2010), pp. 123–152.

[4] Paulo Blikstein. "Using learning analytics to assess students' behavior in open-ended programming tasks". In: *Proceedings of the 1st international conference on learning analytics and knowledge.* ACM. 2011, pp. 110–116.

[5] Lori Breslow et al. "Studying learning in the worldwide classroom: Research into edX's first MOOC". In: *Research & Practice in Assessment* 8 (2013).

[6] Ann L Brown and Annemarie S Palincsar. "Guided, cooperative learning and individual knowledge acquisition". In: *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (1989), pp. 393–451.

[7] National Research Council et al. *How people learn: Brain, mind, experience, and school: Expanded edition.* National Academies Press, 2000.

[8] Philippe Fournier-Viger et al. "SPMF: A Java Open-Source Pattern Mining Library". In: *Journal of Machine Learning Research* 15 (2014), pp. 3569–3573.

[9] Julio Guerra et al. "The problem solving genome: Analyzing sequential patterns of student work with parameterized exercises". In: *Educational Data Mining 2014.* 2014.

[10] Mark Andrew Hall. "Correlation-based feature selection for machine learning". In: (1999).

[11] James Herold, Alex Zundel, and Thomas F Stahovich. "Mining meaningful patterns from students' handwritten coursework". In: *Proceedings of EDM* (2013).

[12] Joshua Ho, Lior Lukov, and Sanjay Chawla. "Sequential pattern mining with constraints on large protein databases". In: *Proceedings of the 12th International Conference on Management of Data (COMAD)*. 2005, pp. 89–100.

[13] I-Han Hsiao, Sergey Sosnovsky, and Peter Brusilovsky. "Adaptive navigation support for parameterized questions in object-oriented programming". In: *European Conference on Technology Enhanced Learning*. Springer. 2009, pp. 88–98.

[14] Edwin Kashy et al. "Using networked tools to enhance student success rates in large classes". In: *Frontiers in Education Conference, 1997. 27th Annual Conference. Teaching and Learning in an Era of Change. Proceedings.* Vol. 1. IEEE. 1997, pp. 233–237.

[15] John S Kinnebrew, Kirk M Loretz, and Gautam Biswas. "A contextualized, differential sequence mining method to derive students' learning behavior patterns". In: *JEDM-Journal of Educational Data Mining* 5.1 (2013), pp. 190–219.

[16] John Kinnebrew and Gautam Biswas. "Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution". In: *Educational Data Mining 2012*. 2012.

[17] Gerd Kortemeyer et al. "Experiences using the open-source learning content management and assessment system LON-CAPA in introductory physics courses". In: *American Journal of Physics* 76.4 (2008), pp. 438–444.

[18] Roberto Martinez Maldonado et al. "Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop". In: *Educational Data Mining 2011*. 2010.

[19] Hirokazu Masataki and Yoshinori Sgisaka. "Variable-order n-gram generation by word-class splitting and consecutive word grouping". In: *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*. Vol. 1. IEEE. 1996, pp. 188–191.

[20] Scott G Paris and Alison H Paris. "Classroom applications of research on self-regulated learning". In: *Educational psychologist* 36.2 (2001), pp. 89–101.

[21] Dilhan Perera et al. "Clustering and sequential pattern mining of online collaborative learning data". In: *IEEE Transactions on Knowledge and Data Engineering* 21.6 (2009), pp. 759–772.

[22] Kathy Pezdek, Tiffany Berry, and Paul A Renno. "Children's mathematics achievement: The role of parents' perceptions and their involvement in homework." In: *Journal of Educational Psychology* 94.4 (2002), p. 771.

[23] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[24] Gideon Schwarz et al. "Estimating the dimension of a model". In: *The annals of statistics* 6.2 (1978), pp. 461–464.

[25] Kusum Singh, Monique Granville, and Sandra Dika. "Mathematics and science achievement: Effects of motivation, interest, and academic engagement". In: *The Journal of Educational Research* 95.6 (2002), pp. 323–332.

[26] Ramakrishnan Srikant and Rakesh Agrawal. "Mining sequential patterns: Generalizations and performance improvements". In: *Advances in Database Technology—EDBT'96* (1996), pp. 1–17.

[27] *TRAC*. http://trac.edgewall.org/. Accessed: 2017-07-05.

[28] Rémi Venant et al. "Using Sequential Pattern Mining to Explore Learners' Behaviors and Evaluate Their Correlation with Performance in Inquiry-Based Learning". In: *European Conference on Technology Enhanced Learning*. Springer. 2017, pp. 286–299.

[29] Arto Vihavainen. "Predicting students' performance in an introductory programming course using data from students' own programming process". In: *Advanced Learning Technologies (ICALT), 2013 IEEE 13th International Conference on*. IEEE. 2013, pp. 498–499.

[30] Haizhou Wang and Mingzhou Song. "Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming". In: *The R journal* 3.2 (2011), p. 29.

[31] Barry J Zimmerman. "Self-regulating academic learning and achievement: The emergence of a social cognitive perspective". In: *Educational psychology review* 2.2 (1990), pp. 173–201.

[32] BJ Zimmermann. *Theories of self-regulated learning and academic achievement: an overview and analysis. self-regulated learning and academic achievement: theoretical perspectives*. 2001.