Predicting Quitting in Students Playing a Learning Game

Shamya Karumbaiah
University of Pennsylvania
3700 Walnut St
Philadelphia, PA 19104
+1 877 736-6473
shamya@upenn.edu

Ryan S. Baker
University of Pennsylvania
3700 Walnut St
Philadelphia, PA 19104
+1 215 573-2990
ryanshaunbaker@gmail.com

Valerie Shute
Florida State University
1114 West Call Street
Tallahassee, FL 32306-4453
+1 850 644-8785
vshute@fsu.edu

ABSTRACT

Identifying struggling students in real-time provides a virtual learning environment with an opportunity to intervene meaningfully with supports aimed at improving student learning and engagement. In this paper, we present a detailed analysis of quit prediction modeling in students playing a learning game called Physics Playground. From the interaction log data of the game, we engineered a comprehensive set of aggregated features of varying levels of granularity and trained individualized level-specific models and a single level-agnostic model. Contrary to our initial expectation, our results suggest that a level-agnostic model achieves superior predictive performance. We enhanced this model further with level-related and student-related features, leading to a moderate increase in AUC. Visualizing this model, we observe that it is based on high-level intuitive features that are generalizable across levels. This model can now be used in future work to automatically trigger cognitive and affective supports to motivate students to pursue a game level until completion.

Keywords

disengagement, learning games, quit prediction, adaptive intervention, personalized learning, physics education

1. INTRODUCTION

In the past couple of decades, education researchers and developers have looked into using digital games as vehicles for learning in a range of domains [19]. Learning games are designed with the goal of keeping students engaged in a fun experience while also focusing on their learning. Well-designed games help build intrinsic motivation in players, which they sustain throughout the process by keeping the player in a state of deep engagement or flow [7].

For a successful learning experience, Gee [14] emphasizes that the game must focus on the outer limits of the student's abilities, making it hard yet doable – Csikszentmihalyi similarly suggests that optimal flow is achieved when student ability is matched with game difficulty [7]. Although some researchers have argued that the difficulty associated with the highest engagement is different than the difficulty associated with the highest learning [20], the goal of good game design must be to promote both engagement and learning.

The challenge, then, must be to maintain the high difficulty associated with learning without compromising engagement to a degree that the student becomes highly frustrated or worse, gives up [e.g. 20]. After all, if a student gives up, they typically do not continue learning from the game (at least, not in the absence of reflective or teacher-driven discussion of the game – e.g. [28, 22]). Some students may quit a game level (or the entire game) only after protracted struggle. Others may quit the level immediately and search for an easier level, a behavior tagged as the "soft underbelly strategy" [1]. Both responses to difficulty should be addressed in an optimal learning game.

To prevent students from giving up, most serious games in education include immediate feedback and interventions aimed to improve the learner experience [26]. When the student is struggling, a relevant and timely intervention could keep the student motivated and prevent frustration from leading the student to give up. A struggling student may also benefit from an intervention that prevents them from wheel-spinning [4], playing for substantial amounts of time without making progress.

However, even though scaffolding may be beneficial to a struggling student, it may be undesirable – even demotivating and harmful to learning – if the student is provided with scaffolding when he or she does not need it [9]. As such, it may be valuable to detect struggling during games that can benefit from an intervention. In that fashion, scaffolding can be provided to students who need it but withheld where it is unnecessary and may be counterproductive. The goal of this paper, then, is to detect whether a student is likely to give up and quit a level in progress. We do so in the context of Physics Playground [27], a game where students learn physics concepts through interactive gameplay.

1.1 Related Work

There has been considerable interest in developing automated detectors of disengagement over the last decade. This work includes detectors for off-task conversation [2], mind wandering while reading [8], and gaming the system - where the student exploits the system to complete the task [3]. In the specific case of games, researchers have developed detectors for a variety of disengagement-related constructs, including whether the learner is engaging in behaviors unrelated to the game's learning goals [23], whether the student is genuinely trying to succeed in the game [10], and whether the learner is gaming the system [29]. One inherent challenge to much of the work to detect disengagement is the dependence on subjective human judgement for ground truth labels such as field observations, self-reports, and retrospective judgement. This makes it challenging to validate the model beyond the context of data collection. By contrast, predicting whether a student will quit has the advantage of only needing an objective ground truth label. This aspect of quit prediction makes it relatively less labor-intensive to validate a model in newer settings and diverse student population.

There has been past work to predict whether a student will quit within other types of online learning environments. In a lab

experiment with a simple reading interface, an interaction-based detector was developed to predict if a student would quit an upcoming text based on the reading behavior of the student in the past text [21]. There has also been considerable attention to the issue of quit prediction (sometimes referred to as dropout or stop-out) in the context of massive open online courses (MOOC), due to the high attrition rate in MOOCs. In one of the studies [31], researchers conducted social network analysis (based on discussion forum participation) and survival analysis to predict student dropout from an ongoing Coursera class. Another study [15] detected at-risk students based on their engagement with video lectures and assignments and their performance in the assignments. One important aspect to some of this MOOC work is that the detectors have been used to drive interventions. For instance, an automatic survey intervention was built based on a MOOC dropout classifier by researchers at HarvardX [30]. They observed that the surveys appeared to increase the proportion of students thought to have dropped out who chose to return to the course.

1.2 Context/Setting

Physics Playground (PP; formerly known as Newton's Playground) [27] is a two-dimensional game, developed to help secondary school students understand qualitative physics related to Newton's laws of force and motion, mass, gravity, potential and kinetic energy, and conservation of momentum. The player creates simple machines or agents (i.e., ramp, springboard, pendulum, and lever) to guide a green ball to hit a red balloon (goal) by using a mouse and drawing directly on the screen. Any solution that solves the problem receives a silver badge; a solution that solves the problem with a minimal number of objects receives a gold badge. Problems are designed so that receiving a gold badge typically requires a specific application of an agent or simple machine. Laws of physics apply to the objects drawn by the player. There are seventy-four levels in total across seven playgrounds. Each level contains fixed and movable objects. The player analyzes the givens (what he/she sees on the screen) and sketches a solution by drawing new objects on the screen (see Figure 1). All objects in the game obey the basic rules of physics relating to gravity and Newton's laws, and each level is designed to be optimally solved by particular agents.

The goal of quit prediction is to identify potential learning moments for a struggling student in the game where a cognitive support could support the student in developing their emerging understanding of key concepts and principles.

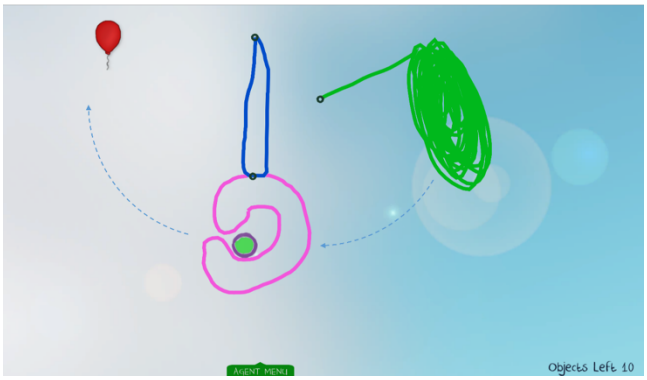

Figure 1. An example level in physics playground being solved with a pendulum agent (drawn in green by the student). The dashed blue (marked for illustration; not shown in the game) line traces the trajectory of the pendulum when released and that of the ball to the balloon after the pendulum strikes.

2. METHODS

2.1 Data Collection

Participants consisted of 137 students (57 male, 80 female) in the 8th and 9th grades enrolled in a public school in a medium-sized city in the southeastern U.S. The study was conducted in a computer-enabled classroom with 30 desktop computers over four consecutive days. On the first day, an online physics pretest was conducted, followed by two consecutive days of gameplay and a posttest on the fourth day. The pre-test and the post-test measured students' proficiency in Newtonian physics. The software logged all the student interactions in a log file. In this paper, we focus on the data collected during second and the third days (where students were playing Physics Playground).

Physics Playground log data capture comprehensive information on student actions and game screen changes as a time series with millisecond precision. One of the important fields in the log data is the *event*. It is used to construct most of the features used in our model. The value of this field categorizes the game moments into – a) game-related events like game start, and end; b) level-related events like start, pause, restart, and end; c) agent creation events like drawing of ramp, pendulum, level, springboard; d) play-related events like object drop, object erase, collision and nudge; e) between-level navigation events like menu-focus. We focus on level-related events, agent creation events, and play-related events for predicting whether a student will quit a specific level.

Some levels in PP can be solved by multiple agents (ramp, lever, pendulum, and springboard). For each of the relevant agent, students can get a silver or a gold badge based on how efficient their solution is. Hence, a student could be playing a level for the first time, replaying using a different agent, or replaying to get a better badge. We consider each of these visits to a level as separate instances of gameplay on that level and predict whether a student will quit the level during the student's current visit. Each time a student exits a level, the log data marks the end of the visit with a level end event. This event can occur either when the student solves the level successfully (earns a badge; quit=0) or when the student exits a level without solving it (doesn't earn a badge; quit=1). Within each visit, a student can restart a level multiple times without quitting the level. Restarting a level erases all the student-created objects and resets the ball and the other level-given objects back to their default positions. The ball also resets back to its original position each time it drops out of the screen. We identify this as a ball reset event.

2.2 Data Preparation

2.2.1 Data Pre-processing

Among the total of seventy-four levels in this version of the game, only thirty-four levels had data for at least fifty students. These levels were used for modelling (Table 1); the other levels did not have enough data to build level specific models (explained in section 2.3.1). Also, these higher levels are only reached by the most successful students, making this data of less interest for our research goal. After data pre-processing, we have 390,148 relevant events across all the students playing the chosen levels.

2.2.2 Feature Engineering

Feature engineering is an important step in the modeling pipeline that converts raw log data to a set of meaningful features. Many argue that the success of data mining approaches relies on thoughtful feature engineering [24]. For each data sample, we have engineered a total of 101 features of the four types listed below. In

designing features, we endeavor to avoid using data about the student's future to interpret their behavior, since our goal is to predict their future outcome. Hence, all the features at any time step solely include the information from the past and the present.

a) *Student+Level+Visit* related features define a student's progress in their current visit to a level. They are recalculated at each event within the logs, and each row represents a single event or student action. There are multiple kinds of student+level+visit features: 1) A set of binary features denote the occurrence of an event (e.g., level restart, ball reset, and, the creation of an object). For these features, each row in the data represents a single event, so only one binary feature will have a value of 1 in any row; 2) A set of numerical features represent the current counts of all the actions taken by the student since the beginning of the visit. These include counts of objects and agents drawn and other relevant events (e.g., the number of springboards, freeform objects, pins, and ball nudges); 3) A set of features track higher-level game activities since the start of the visit (e.g., the number of level restarts and ball resets); 4) A set of temporal features (e.g., the time elapsed in the visit so far and the time elapsed since the last restart); and 5) A set of features that maintain the counts of currently active objects on screen since the drawn objects could drop off the screen or be erased by the student. There are a total of 27 student+level+visit related features. All of these features are updated after each relevant event (see section 2.1). In most cases, only a small subset of feature values change between consecutive data samples.

b) *Student+Level* related features define the student's experience with the level so far, across all the previous visits (recall that a student can replay a previously solved or unsolved level; see section 2.1). This includes high-level features like the number of visits to the level, the number of badges received in the past visits, the number of visits quit without solving, the overall number of pauses, and the total pause duration in the level overall. This also includes cumulative features that indicate past solution approaches (e.g., the total number of pendulums drawn in the past visits). There is a total of 17 such features. These are set to 0 for the first visit and is updated at the end of each consecutive visit to the level by the student.

c) *Student* related features define the student's progress through the game across all the levels played so far. These include counts like the total number of levels played, the number of levels quit, the number of levels involving a particular physics concept played so far (e.g., Newton's first law of motion, energy can transfer, properties of torque), and the number of levels solved using a particular agent. These also include an overall summary of gameplay attributes across the levels played so far (e.g., means and standard deviations of the number of visits, pause duration, time spent, and number of objects used across all the levels played so far). There are a total of 40 such features. The feature values start at zero for a new student and continue to get updated as the student proceeds playing more levels in the game.

d) *Level* related features define the inherent qualities of a particular level. There are two kinds of level-related features – 1) A set of ten features computed by taking averages and standard deviations of student-level features from all students who played that level (e.g., means and standard deviations of number of objects used, time taken, number of level restarts, and badges received in this level); and 2) A set of seven level-related features that do not require past student data. These include binary features for primary physics concept and agent(s) used for solving. There are a total of 17 level-related features. These features are pre-set at the game start and their values remain the same for all the students and all the visits to a particular level.

Upon exploring the relationship between the level-pause and level-end events, we noticed that in order to access the quit button, students need to pause the gameplay. Since level-pause is directly indicative of the outcome variable (though not all pauses lead to quitting), we have discarded any feature that is related to the occurrence of a pause event from the student+level+visit set of features and retained the pause-related features in the student+level set of features.

2.2.3 Aggregations

As we are predicting an outcome (quitting a level) that comes as the culmination of many actions, and that is likely to be predicted by patterns of inter-related actions rather than single actions (such as drawing a single object), we aggregate the data into 60-second clips [24] , [5]. Since only student+level+visit (see Section 2.2.2 a) and student+level (see Section 2.2.2 b) features change with each event, these are the only features to be aggregated at the 60 second interval. The binary student+level+visit features are converted to integer features that count the occurrence of these events over the 60 second interval. For cumulative features like the total number of level restarts in the visit so far, the last value at the end of the 60 second window is retained. Similarly, for features indicating the current object counts and on-screen elements like current number of lever objects, the values of the last data sample in the 60 second interval are retained. The same approach is followed for the features corresponding to elapsed time, like time elapsed since level restart in the visit. After feature aggregation, we have a final sample size of 14,116 data points and a feature space of 101 dimensions.[1]

2.3 Model Training

The next step of the modeling process is to define the quit value for each data sample. We predict a binary label that represents whether the student quit a visit (without solving) or not. This variable can be operationalized in several fashions. One possible way to define the label value would be to only mark the last data sample of a visit before the student quits as representing quitting, as in the research on MOOC stop-out [30]. However, our goal is to be able to detect that a student is likely to quit early enough to prevent this behavior. Therefore, we label every 60-second data clip during a visit that is eventually quit as "quit". The overall class distribution in the data is 28.77% quit and 71.23% not-quit.

2.3.1 Level-specific Models Versus Level-agnostic Model

Within this paper, we consider two possible types of models for detecting quit: a) *level-specific models* which are trained on the data from a single level; and b) a *level-agnostic model* which is a single model trained on the data from all levels. One can see pros and cons to both approaches. We could expect level-specific models to be more accurate as the data is tailored to a narrower prediction context. However, using level-specific models necessitates having enough training data for all the levels. It also implies that detection will be unavailable at first when new levels are designed for the game.

[1] The aggregated data (section 2.2.3) is made available at https://upenn.box.com/s/4ocucflaehd7c51lbxx96heikcjtcwz1

2.3.2 Gradient Boosting Classifiers

Due to the popularity of ensemble methods in classification, gradient boosting classifiers [13] are chosen for quit prediction. Only one other model (random forest) was tried and the results are similar to the gradient boosting classifier. Gradient boosting classifiers combine the predictive power of multiple weak models into a single strong learner, reducing model bias and variance. The ensemble is built in a forward stage-wise fashion where the current model corrects its predecessor model by fitting to its pseudo-residuals. Decision trees are used as the base learners. To avoid overestimation of model generalizability, hyperparameter values are kept at the default specified by scikit-learn, the python machine learning library. These consist of setting the number of estimators to 100, the maximum depth of estimators to 3, the learning rate to 0.1, using a deviance loss function, using the Friedman mean squared error criterion, and setting the subsample value at 1 for a deterministic algorithm.

2.3.3 Model Training Architecture[2]

Five-fold student-level cross-validation is used for evaluation of model performance. In this approach, students are split into folds and a single student's data is only contained in one-fold. To avoid biasing the model, feature selection is repeatedly conducted only on the training fold data. Model-based feature selection approach is used. Based on the model's fit on the training data, features are only selected to be included if their feature importance [6] (see section 3.3.2) is more than the mean of the importance of all features. The reduced-feature training data is used for model training. The performance of the trained model is evaluated on the held-out test set. The same pipeline is followed for all the five non-overlapping folds of train-test splits. Due to the skewness in the data, area under the curve (AUC) is used as the evaluation metric [16]. AUC indicates the probability that the classifier ranks a randomly chosen quit sample higher (more likely to indicate quitting) than a randomly chosen not-quit sample. The corresponding F1 value, giving the harmonic mean between precision and recall at the default threshold between quit/not quit (0.5), is also noted. Finally, precision-recall curves are used to better understand the performance of the model and to choose an appropriate probability threshold for intervention. Feature importance and partial dependence plots (section 3.3.3) are used to interpret the final model.

3. RESULTS

3.1 Level-specific Models Versus Level-agnostic Model

3.1.1 Cross-validation Results

The first analysis is aimed at choosing between level-specific and level-agnostic modelling approaches for quit prediction (section 2.3.1). For our first comparison of the model performances, only the aggregations of 49 features corresponding to student+level+visit and student+level attributes were used for training, since the level specific models cannot benefit from level-related features. We add those additional features to the level-agnostic model in a following section. Following the modeling architecture described in section 2.3.3, the five-fold student-level cross-validation results of level-specific models for the 34 unique levels in this dataset are given in Table 1. The average AUC of the level-specific models is 0.68 ($SD = 0.11$), and the average F1 value is 0.39 ($SD = 0.16$). The level-agnostic model has a cross-validated AUC of 0.75 and F1 of 0.41. The AUC of the level-agnostic model is higher than the median and mean and close to the third quartile value of the level-specific AUCs (Figure 2). The F1 value of the level-agnostic model is higher than the median and mean of level-specific F1 values. The level-specific F1 values also have high variance.

Table 1. Cross-validation results of level-specific models for the 34 levels sorted by their order in the game.

Level	#Users	%quit	AUC	F1
downhill	124	8.13	0.93	0.82
lead the ball	123	4.32	0.94	0.33
on the upswing	124	11.82	0.83	0.30
scale	124	8.70	0.92	0.32
spider web	126	15.38	0.62	0.20
sunny day	126	23.25	0.55	0.22
through the cracks	125	13.10	0.77	0.46
wavy	127	20.78	0.54	0.18
around the tree	115	29.45	0.63	0.29
chocolate factory	121	26.11	0.65	0.29
cloudy day	121	33.78	0.65	0.39
diving board	120	32.50	0.61	0.36
jelly beans	122	21.04	0.59	0.29
little mermaid	115	39.46	0.56	0.33
move the rocks	114	16.70	0.60	0.21
need fulcrum	126	42.41	0.55	0.40
shark	111	44.14	0.61	0.46
tricky	107	17.75	0.78	0.32
trunk slide	116	32.56	0.67	0.35
wedge	107	7.86	0.83	0.32
yippie!	123	12.66	0.69	0.28
annoying lever	107	22.41	0.68	0.37
big watermill	101	43.70	0.62	0.46
caterpillar	95	40.24	0.67	0.53
crazy seesaw	92	35.74	0.68	0.38
dolphin show	81	46.78	0.59	0.52

[2] The scripts for feature engineering and modelling is open at https://github.com/Shamya/Quit-Prediction-Physics-Playground.git

flower power	74	35.45	0.65	0.42
heavy blocks	72	18.75	0.61	0.23
Jar of Coins	73	36.14	0.74	0.60
roller coaster	67	45.60	0.64	0.52
stiff curtains	58	26.47	0.45	0.08
tetris	67	39.89	0.74	0.56
work it up	57	73.76	0.68	0.77
avalanche	54	28.85	0.75	0.60

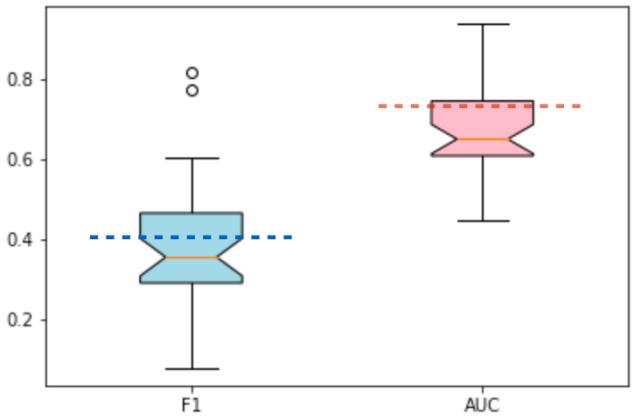

Figure 2. Box plot representing the range of AUC and F1 values of the 34 level-specific models. The box extends from the 25th to 75th percentiles, with a notch at the median. The dashed horizontal lines correspond to the values of the level-agnostic model.

3.1.2 Understanding the Model Differences

The qualitative differences between the two approaches can be explored by contrasting the features selected by each (Table 2). Feature selection for this analysis is done on the full data. The level-agnostic model seems to mainly select general features like past quits, pauses, badges, visits, level restarts, and ball resets which are common across levels. While level-specific models also include these features, to the level-specific models incorporate additional features related to finer-grained aspects of gameplay like the placement of pins and the drawing of specific machines (in the current 60-second time bin, in the current visit, and across visits). For instance, one of the levels named *diving board* is solved using a springboard. Among the ten features selected by this level's specific model, six of them correspond to the specific gameplay actions that one can observe a student take (e.g., total springboards drawn, total pins drawn (pins are used to hold the springboard on the screen), current number of pendulum objects on screen, and total nudges). A similar trend is seen in most level-specific models. Note that the number of level-specific models selecting any specific agent-related feature (as shown in Table 2) is distributed across agents, as most levels can be solved by only a subset of these agents.

Table 2. Comparing top features selected in level-agnostic and level-specific models.

Selected Feature	In level-agnostic model?	In how many level-specific models (out of 34)
Number of visits made by the student to this level so far	Yes	25
Total pause duration in the level so far	Yes	26
Number of past quits by the student in the level	Yes	25
Number of badges received in the level by the student so far	Yes	22
Number of restarts by the student in the level so far	Yes	20
Number of ball resets in the visit so far	Yes	23
Total ball resets in the level so far	Yes	20
Total pins drawn in the visit so far	Yes	20
Total pendulums drawn in the visit so far	Yes	17
Total nudges in the visit so far	No	32
Total nudges in the level so far	No	30
Total pins placed in the level so far	No	28
Total free form objects drawn in the visit so far	No	24
Current number of free form objects on the screen	No	25
Total ramps drawn in the level so far	No	21
Total ramps drawn in the visit so far	No	18
Total free form objects drawn in the level so far	No	17
Total pendulums drawn in the visit so far	No	15
Current number of pendulum objects on the screen	No	10

3.2 Enhancing the Level-agnostic Model

3.2.1 Feature Additions

Counter to the expectation that the individualized models may perform better, the AUC value of the level-agnostic model was 7 percentage points higher than the average AUC of the level-specific models. This could be attributed to the ability of the level-agnostic model to leverage the larger amount of data to identify generalizable features for quit prediction.

However, it may be possible to achieve even better predictive performance in a level-agnostic model by exploiting the level-related features (section 2.2.2 d). To examine this, we re-fit the level-agnostic model, now also incorporating the level-related features. Recall that there are two kinds of level related features – pre-defined features that can be defined for any new levels (indicating what agents and concepts are involved in solving the level) and features that use past student data to determine average behaviors for other students on the level, such as the number of

objects used. We tested each type of additional features separately (Table 3, model #2 and #3). Adding just the predefined features had very little effect on the output (model #2). By contrast, incorporating the ten level-related features that use past students' data appears to improve the AUC value, though only by a modest 0.04 (model #3).

Table 3. The performance of the original level-agnostic model and various extensions to the model with level-related and student-related features.

#	Feature Set(s)	#Features	AUC	F1
1	Level-agnostic Model	44	0.75	0.41
2	Model 1 + Predefined Level-related Features	51	0.75	0.42
3	Model 2 + Level-related Features from Past Data	61	0.79	0.45
4	Model 1 + Student-related Features (level-agnostic features only)	84	0.79	0.49
5	Model 3 + Student-related Features (all features)	101	0.81	0.51

Finally, we investigated whether we can enhance the model by adding features pertaining to the student's whole history of past play (student-related features; section 2.2.2 c). We see that there is a modest improvement to the AUC values (Table 3, model #4 and #5). Note that model #4 (like model#1) doesn't contain level-related features and hence is level-agnostic. With an AUC of 0.79, model #4 could be used for new levels of the game where we do not have past student data to compute level-related features. For the current levels of the game, the best performing model (model #5) has an AUC of 0.81. Across the five folds, the AUC values of the held-out test sets have a low standard deviation of 0.01.

3.2.2 Understanding Model Performance

The AUC values above show that the best model (#5) is good at distinguishing students who will eventually quit from other students, but the F1 values are surprisingly low, considering the AUC. We can further understand the full model's (model #5) performance for different thresholds by examining a precision-recall (PR) curve (Figure 3) generated for all test set predictions. We see that precision is close to perfect for any threshold where recall is at or below 0.2. Additionally, recall is perfect when precision drops to 0.3. In between these extremes, the relationship between precision and recall is nearly linear, offering a clear trade-off between which of these two metrics is optimized for. Based on the characteristics of an intervention, a custom threshold on the probability can prioritize recall over precision or vice versa.

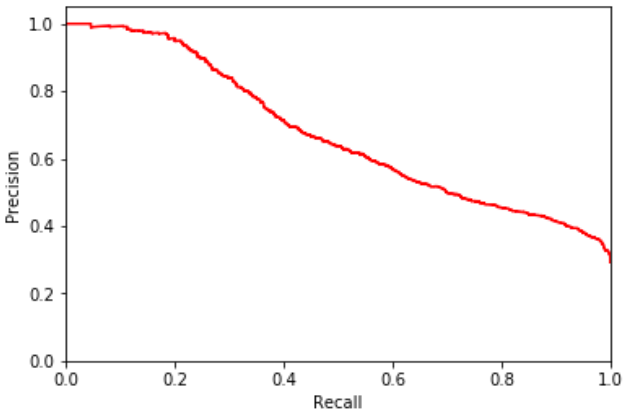

Figure 3. Precision-Recall curve of the final model (model #5).

3.3 Final Model Interpretation

3.3.1 Selected Features

Out of 101 features, a total of 34 features were selected by the final model (model #5). The 21 features are student-related features (out of a possible 40 student-related features), 2 are level-related features (out of a possible 17), 6 are student+level related features (out of a possible 17), and 5 are student+level+visit related features (out of a possible 27). Table 4 lists the top 15 features. Similar to the original level-agnostic model (model #1), the selected features focus on high-level game activities like visits, badges, past quits, time spent, level restarts, and experience with agents across visits and other levels. There is no student+level+visit related feature in the top 15 selected features. The final model (model #5) has 10 student-related features out of the top 15 features; note that these student-related features were not available to the original level-agnostic model. These features continually track the student's progress across all the levels.

Table 4. Top 15 features selected by the final model (model #5). Feature type – SLV=Student+Level+Visit, SL=Student+Level, L=Level, S=Student

Feature ID	Selected Feature	Feature Type
1	Number of visits made by the student to this level so far	SL
2	Standard deviation of the total time spent by the student across levels so far	S
3	Mean number of badges received by all students in this level	L
4	Number of past quits by the student in the level	SL
5	Number of badges received in the level by the student so far	SL
6	Standard deviation of the total pendulums drawn by the student across levels so far	S
7	Standard deviation of total freeform objects drawn by the student across all levels so far	S
8	Mean badges received by the student across levels so far	S

9	Total pause duration in the level so far across all visits	SL
10	Mean time spent by the student in a level	S
11	Standard deviation of the number of ball resets by the student across levels so far	S
12	Standard deviation of the number of visits made by the student across levels so far	S
13	Mean pause duration of the student across levels so far	S
14	Standard deviation of badges received by the student across levels so far	S
15	Mean number of pendulums drawn by the student across levels so far	S

3.3.2 Feature Importance

Feature importance can be thought of as a feature ranking method. The gradient boosting classifier used for modelling quit prediction also provides the feature importance, the contribution of a feature to the predictive power of the trained model. Recall that (section 2.3.2) the gradient boosting classifier uses multiple decision trees as base learners. Decision tress naturally perform feature selection by picking appropriate split points at the nodes. Intuitively, the more often a feature is used for splitting, the more important it is for the model's decision making. In case of an ensemble such as gradient boosting where there are multiple decision trees as base learners, the importance of a feature is obtained by averaging its importance across all the trees. A higher score indicates higher contribution of that feature to the model prediction. The importance of all the features sum to 1. Figure 4 plots the importance of the top 15 features selected by the final model (model #5).

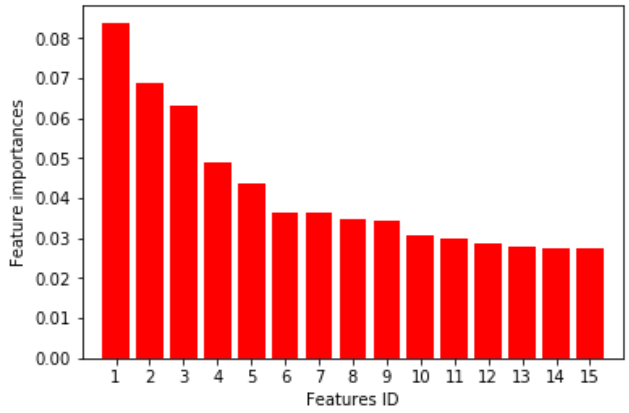

Figure 4. The feature importance of the top 15 features selected by the final model (model #5). The mapping between feature IDs and feature names is given in Table 4.

3.3.3 Partial Dependence Plots

Partial dependence plots (PDP) [13], originally proposed to interpret gradient boosting algorithms, have since been used with many predictive models to understand the dependence of model predictions on the covariates. Intuitively, partial dependence refers to the expected quit probability (*logit(p)*) as a function of one or more features. For example, the top right plot (5B) in Figure 5 gives the partial dependence between the mean number of badges received by students in a level (level-related feature) and the logit of quit probability after controlling for all the other features. Negative partial dependence values (y-axis) imply that for the corresponding value of the feature, it is less likely to predict quit=1. Similarly, a positive partial dependence for a feature value implies that it is more likely to predict quit=1 for that feature value. In our example, levels with mean numbers of badges earned below 0.6 are more likely to be quit by students. In general, as one might expect, there is a negative relationship between quitting and the mean number of badges received by students in a level. The higher the value of partial dependence, the stronger the relationship between the feature value and the outcome of quitting. More generally, the larger the range of the dependence value, the larger the overall influence of that feature on the model prediction.

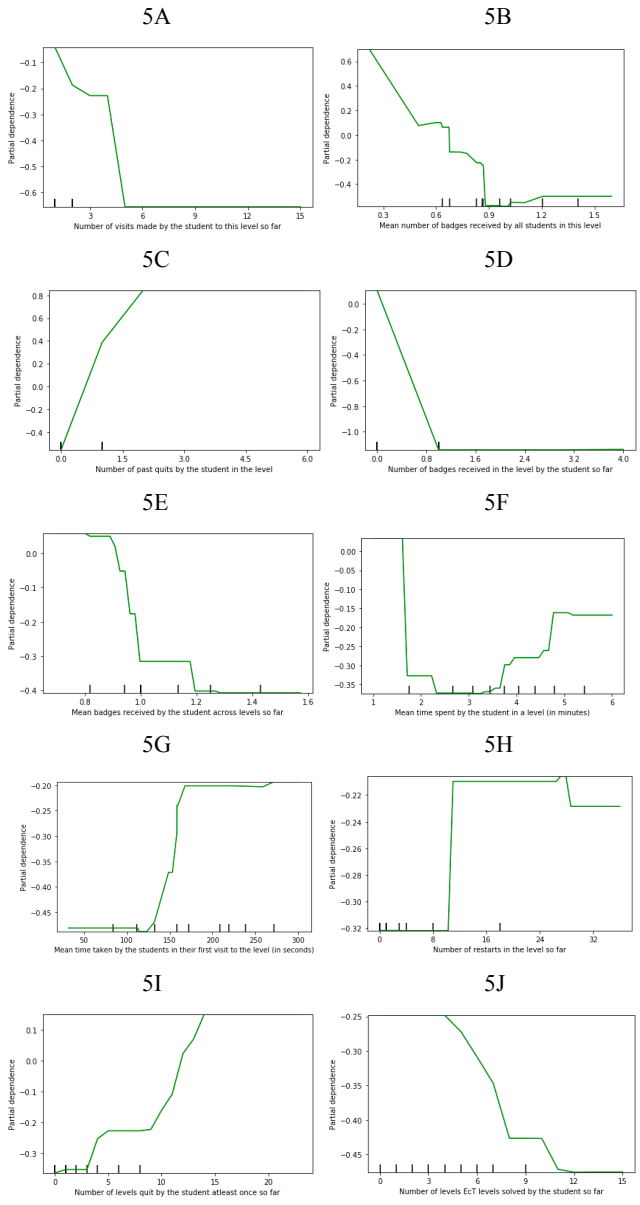

Figure 5. Partial dependence of quit probability on some of the selected features. Note that the range of y axis is different for each plot; larger ranges indicate that the feature is more predictive overall.

Below is the summary of our interpretation of some of the features (Figure 5) selected by the final model. Most of these align with a general intuition of the game attributes and student behavior. Note that this analysis is intended only for a high-level model interpretation. The model decision-making is more complex and involves interactions between different sets of features.

1. A student who revisits a level is less likely to quit the level. This could indicate student interest in solving the level. (Figure 5A; student+level-related feature)
2. A student who has quit a level in the past is more likely to quit the level again. This could indicate that the student is struggling with a concept or how to apply it in a way that is preventing him/her from succeeding in the level. (Figure 5C; student+level-related feature)
3. A student who has previously solved the level is less likely to quit in their revisits to the level. This could indicate that the student generally understands the level and is trying to solve it with different agents. (Figure 5D; student+level-related feature)
4. A student who has restarted a level fewer times is less likely to quit the level. Higher numbers of level restarts could indicate struggle. (Figure 5H; student+level-related feature)
5. A student who has received a higher number of badges (mean badges > 0.9) in the past levels is less likely to quit a future level. This could indicate a student who generally understands the physics concepts better. (Figure 5E; student-related feature)
6. A student who either spends under 2 minutes or over 5 minutes on average across levels is more likely to quit future levels. This feature is discussed in section 4.2. (Figure 5F; student-related feature)
7. A student who has quit more levels in the past is more likely to quit a future level. This could indicate low competence and/or disengagement. (Figure 5I; student-related feature)
8. A student who has solved more number of levels that involve the concept "energy can transfer" (EcT) is less likely to quit a level in the future. EcT is a relatively complex physics concept. In our past research [18] we have seen evidence that levels that include EcT are associated with higher student frustration. (Figure 5J; student-related feature)
9. A level in which students have received fewer badges (mean < 0.6; note that a student may earn multiple badges in a level) is more likely to see quitting behavior in future students. This could indicate the inherent level difficulty. (Figure 5B; level-related feature)
10. A level in which students spend less time in average is more likely to be solved correctly by a future student. This could indicate lower level difficulty. (Figure 5G; level-related feature)

4. DISCUSSION

In this paper, we describe an automated detector we developed to predict if a student will quit a specific level they have started, within the game Physics Playground. Multiple sets of features were engineered to capture student-related, level-related and gameplay-related information over time. We compared the performance of models trained on data from single levels to the performance of a single level-agnostic model trained on the data from 34 levels. Contrary to our initial expectations, the level-agnostic model (#1 above) performed better than almost three-fourths of the level-specific models. After adding level-related features (which cannot be used in level-specific models), the resultant model (#3 above) performed better than 29 (out of 34) level-specific models. Among the five level-specific models that outperform model #3, four of them are the first four levels encountered by the students in the game and are designed to be easy. All five of the outperforming levels have around 10% of student visits ending in quitting whereas the overall incidence of quitting behavior is 28.77%. Comparing the features selected by the two kinds of models reveal the emphasis of the level-agnostic model on generalizable student behavior, while the level-specific models focus on low-level gameplay related features. The performance of the level-agnostic model is further enhanced by adding student-related features (model #4, #5 above). The final combined model (#5) selects 34 out of 101 features, which are interpreted using the feature importance scores and partial dependence plots. Due to the superior performance of the level-agnostic model and its ability to transfer to new levels and the levels with limited data, we recommend its usage over the level-specific models.

Given the model's level of AUC, it appears to be of sufficient quality to use in intervention, identifying a student who is struggling and could benefit from learning supports before they quit the level. Our final model has a clear trade-off between precision and recall, shown in the precision-recall curve in Figure 3. Depending on the properties of a specific intervention, an appropriate threshold could be set on the classifier probability to decide whether a student is sufficiently likely to quit to justify an intervention.

4.1 Limitations

There are some potential limitations to the approach presented here. First of all, there are limitations arising from our choice to label all data in a student's visit to a level as to whether the student eventually quit. By labeling all data in the visit as quit, we may predict quitting before the behaviors have emerged that lead to quitting, and may intervene too early. This also leads to the risk of interfering with student persistence [25][11]. This risk could be mitigated by using interventions that allow the student to continue their efforts if they feel that they are not yet ready for an intervention.

Another limitation is in the generalizability of the model we have developed. Physics Playground is played by students of various age range and representing a diverse range of backgrounds, but the students in this dataset are of similar ages and live in the same region. Hence, it is important to test the generalizability of the model on data from a broader and more diverse range of students. As a next step, we are collecting data from a middle school in New York City where over 80% of students are economically disadvantaged, 97% belong to historically disadvantaged groups and all students enter the school with test scores far below proficiency. We also intend to collect data from a broader range of levels and test model applicability within this broader range of contexts.

4.2 Future Work

The goal of quit prediction is to identify student struggle in real-time to intervene meaningfully. Towards this end goal, the Physics Playground team is building an array of cognitive and affective supports that can be delivered when a student is predicted to be at

risk of quitting to improve students' experience and learning. Ideally, these interventions should be based on an understanding of why a student is likely to quit, which our current model does not yet reveal. For example, a student may quit a level after putting in considerable effort, or rather quickly after minimal effort. A student may quit a level to replay other levels to achieve a gold badge, or may seek to follow a soft underbelly strategy [1], searching for a level easy enough to complete. As reported in section 3.3.3 (Figure 5F), there are two distinct quitting behaviors associated with time spent in a level. A student spending very little time in a level is more likely to quit the level. This may occur when the student is engaging in soft underbelly strategies, or when the student is putting in limited effort. Other students quit a level after considerable time and effort, indicating that they are struggling, possibly in some cases even wheel-spinning [e.g. 4]. Future work to differentiate *why* a student is likely to quit may help an intervention model to differentiate why a specific student needs support and personalize the support delivered to that student.

A learner playing a game experiences a range of emotions while engaging with the game. These can influence learning outcomes by influencing cognitive processes [12]. Knowing students' affective experience could provide deeper insights into the causes of quitting behavior. In past research [17], video-based and interaction-based affect detectors were built for Physics Playground to identify the incidence of affective states like flow, confusion, frustration, boredom, and delight. Combining quit prediction with affect detection could help us make a fuller assessment of the student experience in the learning game to provide more optimal support.

5. ACKNOWLEDGMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A170376 to Valerie Shute. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

6. REFERENCES

[1] Baker, R. S., Mitrović, A., and Mathews, M. (2010, June). Detecting gaming the system in constraint-based tutors. In *International Conference on User Modeling, Adaptation, and Personalization*. pp. 267-278. Springer, Berlin, Heidelberg.

[2] Baker, R.S.J. 2007. Modeling and understanding students' off-task behavior in intelligent tutoring systems. In Proceedings of the *SIGCHI conference on Human factors in Computing Systems*. pp. 1059–1068.

[3] Baker, R.S.J., Corbett, A.T., and Koedinger, K.R. 2004. Detecting student misuse of intelligent tutoring systems. In *Intelligent Tutoring Systems*. pp. 54–76.

[4] Beck, J. E., and Gong, Y. 2013. Wheel-spinning: Students who fail to master a skill. In *International Conference on Artificial Intelligence in Education.* pp. 431-440. Springer, Berlin, Heidelberg.

[5] Bergstra, J., Casagrande, N., Erhan, D., Eck, D., and Kégl, B. 2006. Aggregate features and adaboost for music classification. In *Machine learning*. *65*(2-3), 473-484.

[6] Breiman, L., and Friedman, J. *Classification and regression trees*, 1984.

[7] Csikszentmihalyi, M. 1996. *Flow and the psychology of discovery and invention*. New York: Harper Collins.

[8] D'Mello, S., Cobian, J., and Hunter, M.: Automatic Gaze-Based Detection of Mind Wandering during Reading. In *Proceedings of the 6th International Conference on Educational Data Mining*. pp. 364–365. International Educational Data Mining Society.

[9] Daniel, S. M., Martin-Beltrán, M., Peercy, M. M., and Silverman, R. 2016. Moving Beyond Yes or No: Shifting From Over-Scaffolding to Contingent Scaffolding in Literacy Instruction With Emergent Bilingual Students. In *TESOL Journal*. *7*(2), 393-420.

[10] Dicerbo, K., and Kidwai, K. 2013. Detecting player goals from game log files. In *Educational Data Mining 2013.*

[11] Duckworth, A. L., Peterson, C., Matthews, M. D., and Kelly, D. R. 2007. Grit: perseverance and passion for long-term goals. In *Journal of personality and social psychology*, *92*(6), 1087.

[12] Fiedler, K., and Beier, S. 2014. Affect and cognitive processes in educational contexts. In R. Pekrun and L. Linnenbrink-Garcia (Eds.), *International handbook of emotions in education* (pp. 36-56). New York, NY: Routledge.

[13] Friedman, J. H. 2001. Greedy Function Approximation: A Gradient Boosting Machine. In *Annals of Statistics* 29: 1189–1232.

[14] Gee, J. P. 2003. *What digital games have to teach us about learning and literacy*. New York: Palgrave Macmillan.

[15] He, J., Bailey J., Benjamin, Rubinstein, I., and Zhang, R. 2015. Identifying at-risk students in massive open online courses. In *AAAI*.

[16] Jeni, L. A., Cohn, J. F., and De La Torre, F. 2013. Facing imbalanced data–recommendations for the use of performance metrics. In *Affective Computing and Intelligent Interaction.* ACII 2013. Humaine Association Conference on. IEEE, 245–251.

[17] Kai, S., Paquette, L., Baker, R. S., Bosch, N., D'Mello, S., Ocumpaugh, J., Shute, V., and Ventura, M. 2015. A Comparison of Video-Based and Interaction-Based Affect Detectors in Physics Playground. In *International Educational Data Mining Society*.

[18] Karumbaiah, S., Rahimi, S., Baker, R.S, Shute, V. J., and D'Mello, S. 2018. Is Student Frustration in Learning Games More Associated with Game Mechanics or Conceptual Understanding?. In *International Conference of Learning Sciences*. ICLS 2018.

[19] Ke, F. 2009. A qualitative meta-analysis of computer games as learning tools. In *Handbook of research on effective electronic gaming in education*. 1, 1-32.

[20] Lomas, D., Patel, K., Forlizzi, J. L., and Koedinger, K. R. 2013. Optimizing challenge in an educational game using large-scale design experiments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* pp. 89-98. ACM.

[21] Mills, C., Bosch, N., Graesser, A., and D'Mello, S. K. 2014. To Quit or Not to Quit: Predicting Future Behavioral Disengagement from Reading Patterns. In S. Trausan-Matu, K. Boyer, M. Crosby and K. Panourgia (Eds.), *Proceedings*

of the 12th International Conference on Intelligent Tutoring Systems. ITS 2014. pp. 19-28. Switzerland: Springer International Publishing.

[22] Rowe, E., Asbell-Clarke, J., Baker, R. S., Eagle, M., Hicks, A. G., Barnes, T. M., ... and Edwards, T. 2017. Assessing implicit science learning in digital games. In *Computers in Human Behavior*. 76, 617-630.

[23] Rowe, J. P., McQuiggan, S. W., Robison, J. L., and Lester, J. C. 2009. Off-Task Behavior in Narrative-Centered Learning Environments. In *Artificial Intelligence for Education*. pp. 99-106.

[24] Sao Pedro, M., Baker, R.S.J.d., and, Gobert, J. 2012. Improving Construct Validity Yields Better Models of Systematic Inquiry, Even with Less Information. In *Proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization*. UMAP 2012, 249-260.

[25] Shechtman, N., DeBarger, A., Dornsife, C., Rosier, S., and Yarnall, L. 2013. Promoting grit, tenacity, and perseverance: Critical factors for success in the 21st century. Draft released by the *US Department of Education Office of Educational Technology.*

[26] Shute, V. J., and Ke, F. 2012. Games, learning, and assessment. *In Assessment in game-based learning.* pp.43-58. Springer New York.

[27] Shute, V., and Ventura, M. 2013. *Measuring and supporting learning in games: Stealth assessment*. Cambridge, MA: The MIT Press.

[28] Squire, K. D. 2008. Video games and education: Designing learning systems for an interactive age. In *Educational Technology*. 48(2), 17.

[29] Wang, L., Kim, Y. J., and Shute, V. 2013. Gaming the system" in Newton's Playground. In *AIED 2013 Workshops Proceedings Volume 2 Scaffolding in Open-Ended Learning Environments.* OELEs. p. 85.

[30] Whitehill, J., Williams, J. J., Lopez, G., Coleman, C. A., and Reich, J. (2015). Beyond prediction: First steps toward automatic intervention in MOOC student stopout. In *Social Science Research Network*.

[31] Yang, D., Sinha, T., Adamson, D., and Rose, C. P. 2014. "Turn on, tune in, drop out": Anticipating student dropouts in massive open online courses. *In NIPS Workshop on Data-Driven Education*.