2006-01-08

Visual Feedback for Gaming Prevention in Intelligent Tutoring Systems

Jason A. Walonoski
Worcester Polytechnic Institute

VISUAL FEEDBACK FOR GAMING PREVENTION IN INTELLIGENT TUTORING SYSTEMS

by

Jason A. Walonoski

A Thesis

Submitted to the Faculty

of

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

December 2005

____
Professor Neil Heffernan
Advisor

____
Professor Carolina Ruiz
Reader

____
Professor Michael Gennert
Department Head

Abstract

A major issue in Intelligent Tutoring Systems is off-task student behavior, especially performance-based gaming, where students systematically exploit tutor behavior in order to advance through a curriculum quickly and easily, with as little active thought directed at the educational content as possible. The goal of this research was to explore the phenomena of off-task gaming behavior within the *Assistments* system, as well as to develop a passive visual indicator to deter and prevent off-task gaming behavior without active intervention via graphical feedback to the student and teachers. Traditional active intervention approaches were also constructed for comparison purposes, and machine-learned gaming-detection models were developed as a potential invocation and evaluation mechanism. Passive graphical interventions have been well received by teachers, and results are suggestive that they are effective at reducing off-task gaming behavior.

Table of Contents

List of Figures

1 Introduction

Intelligent Tutoring Systems (ITS) have been shown to have a positive effect on student learning [1], however these effects may be negated by a lack of student motivation or student misuse. One area of research examining these issues involves studying student "gaming" of the system, especially recognition of gaming behavior [2]. A student is gaming if they are attempting to systematically use the tutors feedback and help methods as a means to obtain a correct answer with little or no work, in order to advance through the curriculum as fast (or as easily) as possible. Student gaming has been correlated with substantially less learning [3] therefore it is of particular importance to understand in order to maximize tutor effectiveness.

One objective of this proposed research is to apply existing methodologies of gaming behavior detection to the *Assistments* intelligent tutoring system, which was developed jointly between Worcester Polytechnic Institute (WPI) and Carnegie Mellon University (CMU) [4]. These methods involve the construction of machine-learned models to identify gaming behavior. Although gaming behavior has only two hallmark appearances (help abuse and systematic guessing and checking), there may be various hidden factors at work. These hidden factors may result in the same usage patterns or surface characteristics, but affect the students in different ways: some students are harmed by gaming while others are not. Machine learning has been shown to be able to differentiate between these two types of gamers [2].

A second objective was to develop three gaming interventions, the first two being more traditional active interventions, and the third being a more novel passive deterrent or prevention mechanism. To accomplish the third part of this objective, a graphical software component was designed and implemented featuring visual indicators of student actions over time, which was featured prominently on screen, for the student and any observing teacher to easily see and interpret. Effectiveness of the passive graphical component on gaming behavior was compared to the traditional active approaches and was evaluated using both subjective and objective methods.

2 Background

This research was primarily focused within the domain of intelligent tutoring systems, specifically the *Assistments* system that was developed jointly between WPI and CMU. However, within the ITS domain, two areas of interest that merit specific consideration and attention are the topics of gaming behavior detection and the application of machine learning within ITS.

2.1 Intelligent Tutoring Systems

The range of modern educational computer software systems is quite extensive. Some systems focus on assessing student knowledge (such as computerized versions of pencil and paper tests) and others focus on assisting students to learn some particular educational content (learning mathematics or a foreign language, for example). The *Assistments* system attempts to focus on both assisting and assessment.

One type of system that is primarily focused on assessing student knowledge is the group of Computer Adaptive Testing (CAT) systems. CAT systems assess a student's knowledge of the material by presenting problems of varying difficulty, starting with easy problems first, and then adjusting the difficulty based on the correctness of the student's response, and continuing until the system approaches the students estimated knowledge level. If a response is correct, a CAT system will choose a problem with slightly higher difficulty to ask next. Likewise, if a response is incorrect, a CAT system will choose a problem with slightly lower difficulty to ask next. After each question a CAT system hones in on the area estimated to be the true knowledge level of the student, which usually means that the system oscillates around a certain difficulty level for a particular student.

Another type of system, one that is focused on assisting students to learn some educational content is the group of Computer Aided Instruction (CAI) systems. These systems typically only respond by informing the student if their response was right or wrong and perhaps offer a general hint or broad overview of the problem. Intelligent Tutoring Systems (ITS) take this feedback and help to the next level, providing specific hints tailored to the student and their answers, generally by building a cognitive model of the student using knowledge tracing. Knowledge tracing is the process of tracking the knowledge and skills a student would need to possess to solve a given problem. If the problem is solved correctly, the student is given some credit toward the probability that they possess the appropriate knowledge and skills, while if the problem is not solved correctly then the student's probability of possessing that knowledge and skills is slightly reduced. The specific mechanisms to isolate particular knowledge or skills, and the algorithms for adjusting probabilities and performing the knowledge tracing vary from system to system.

2.2 Related Prior Work

Previous work at CMU by Baker et al [2][3] has resulted in documentation of the phenomenon of gaming within intelligent tutoring systems, the development of methodologies for the detection of gaming behavior, and theories about why students game. For a gaming detection task similar to what was undertaken in this research, data mining [2] utilized a machine-learned Latent Response Model (LRM), an approach leveraged from psychometrics. Their LRM was able to distinguish between two types of gamers, those whose actions lead to poor learning and those whose actions did not (colloquially referred to as "gamed-hurt" and "gamed-not-hurt," respectively). Learning was measured with pre and post-tests, and in certain cases gaming was negatively correlated to post-test scores. These two studies identified gaming behavior with either a weak prerequisite knowledge of the educational content, a performance-based mentality on the part of the offending students, or with motivational issues. This also revealed that students who engage in off-task gaming are most likely to so when they are on the most difficult steps of a problem, a likely explanation of their low learning results. One objective of this research was to provide more data for consideration and the possible refinement of these hypotheses.

Additionally, various intervention mechanisms have been proposed, but they either alter the system interfaces and introduce time delays (to forcibly slow down student actions) or they are susceptible to an interesting "arms race" condition, where gaming users attempt to game the intervention techniques. These techniques tend to be active interventions by the tutoring system, which may inhibit legitimate learning efforts of on-task (non-gaming behavior) students by inappropriate invocation [3]. This research was also partially aimed at exploring alternative methods to avoid these issues.

Before this research was completed the *Assistments* project provided no summary feedback information to students in regards to either "skill" or "effort" (two metrics used or suggested by other research projects [3]). However, summary performance information was available as live ad hoc reports to teachers [5]. Preliminary investigation suggested that our system under-predicts student performance on the mathematics portion of the MCAS examination (the targeted domain of the *Assistments* project) when suspected gaming behavior is occurring [6]. The focus of this research was not to provide a comprehensive skills and effort summary to the students, but to focus on the limited area of graphically summarizing actions and progress in a manner that reveals on-task and off-task behavior for the purposes of providing feedback to students and the prevention of off-task gaming behavior.

3 Objectives

This research primarily aimed at achieving two main objectives. The first objective was automating the detection of gaming within the *Assistments* system, and the second objective was to make an evaluation of and comparison between different types of mechanisms to remediate, deter, or prevent gaming behavior.

3.1 Detection of Gaming

Before a student's off-task gaming behavior can automatically be addressed, it first must be detected. Therefore, the first goal of this work was the construction of a model that can reliably identify off-task gaming behavior within the *Assistments* system. Rather than manually constructing a model by authoring rules based on the surface features of gaming (systematic guessing and checking and consistent hint requests until answers are directly supplied), machine-learning methods are employed to potentially identify the underlying hidden variables that lead students to game and illustrate how they are affected by their behavior. A prior study by Baker et al has shown that gaming behavior can be reliably detected with machine-learned models [2], so part of this research was to adapt those results towards application within the *Assistments* system.

The prior work of Baker et al is adapted for two main reasons. The first reason to adapt those results is for verification of the findings, to determine if gaming behavior has the same causes, appearances, and resulting effects in different intelligent tutoring systems. The second, and perhaps a more immediate reason to construct a detection model, especially one based on a known methodology with promising results, is to maximize the effectiveness of the *Assistments* tutoring. If the tutoring software is outfitted with a model that can reliably identify gaming behavior, then strategies to stop this behavior can be developed and deployed with reasonable assurance that any active interventions are invoked at the proper (or at least reasonable) times.

3.2 Prevention of Gaming

Once a reliable mechanism is in place for the identification of off-task gaming behavior then strategies can be devised to address it. Therefore, the second goal of this research was the prevention of off-task gaming behavior within the *Assistments* intelligent tutoring system. Within the ITS field there have been a variety of approaches towards remediation of this undesirable behavior in students [3], which are mostly active interventions focused on combating student gaming, with few approaches focused on prevention.

Active interventions can be informally categorized as either static or dynamic. Static interventions apply to all students and are inherent in the design of the tutor, while dynamic interventions are only invoked when triggered by some mechanism. For example, one approach to static interventions on help seeking has been the removal of all bottom-out hints (the hints that directly supply the answer to a question) or only supplying them after a variable time delay. On the other hand, with active interventions that are dynamic, the system dynamically detects occurrences of gaming behavior in real-time and actively intervenes by identifying and explaining this behavior to the offending student [3].

However, each of these traditional active intervention approaches has two drawbacks. The first drawback is the potential to fuel gaming arms races, where gaming students adapt to the interventions and even attempt to game them. And the second drawback is the unfair penalization of non-gaming students by inappropriate invocation of interventions, thereby inhibiting legitimate learning efforts. For example, by not allowing a student to be able to ask for hints when they are truly needed. Given that students who engage in off-task gaming behavior are such a small percentage of all users, the unfairness resulting from improper application of active interventions (dynamic or static) can heavily unbalance the usefulness of the tutoring software against the majority of on-task students [3].

As an illustration of the infrequency of gaming, in the *Assistments* system, the observed rate of gaming across eight 50-minute class periods with students who were experienced with the tutoring software (these students had been using the system twice a week for most of the school year) was roughly 2 percent of all activity (19 out of 850 observations), while the number of distinct students in that same group who were ever observed gaming was roughly 5 percent (8 out of 145 students).

Considering the drawbacks of active interventions, this research sought to explore the possibility of pioneering dynamic passive interventions within the field of ITS. One of the hypotheses behind this research was that a dynamic yet passive intervention could be developed with none of the drawbacks of active interventions, but with all of their benefits. Informally, an intervention is defined as being passive if it does not alter the operation or behavior of the tutoring software in any functional way, but effectively alters the behavior of the students. Of course, then the objective becomes the creation of a passive intervention that alters student behavior in a positive manner (if at all), such as preventing or eliminating off-task gaming. For clarification purposes, an illustration of the differences between active and passive, and static and dynamic strategies is illustrated in Figure 1.

	Static	**Dynamic**
Active	Supplying multi-level hints with a variable time delay at each successive level	Intervening after rapid and repeated hinting, with explanation and encouragement messages
Passive	Complete removal of bottom-out hints	Graphical plot of the students tutoring session illustrating progress with indicators of actions

Figure 1 – Two Dimensions of Gaming Strategy with Examples

If dynamic passive interventions were realizable in an effective and positive manner, they should offer several hypothetical advantages over existing active approaches. The first advantage being that dynamic passive interventions should require no modification of the tutoring software's functional behavior, which would effectively eliminate the gaming arms race while on-task students would no longer be unfairly penalized by improper invocation of the intervention mechanisms. The second advantage would the visualization of student actions and progress itself, which should provide a vehicle for the summarization of student behavior through emergent visual patterns. Coupled with these emergent visual patterns, when featured prominently on-screen for easy viewing by the student and teachers, this mechanism might prevent off-task gaming behavior through Panopticon-like paranoia (when a fear of being watched, without knowing whether one is being watched at any given moment, causes self-corrective behavior) [7].

Therefore, in order to explore the hypothetical advantages of dynamic passive interventions over traditional active interventions, a passive yet prominent graphical component was developed with the goal of preventing gaming behavior among students who gamed (and other off-task students), but ignored by non-gaming and on-task students. Such a component should (1) allow teachers to easily identify gaming behavior via emergent visual patterns, (2) thereby correcting and preventing gaming behavior in the students by the students themselves, and (3) providing a launching point for teacher intervention where gaming behavior is identified or student misunderstandings are shown.

4 Methods and Results

The methodologies employed during this research can be summarized and separated into three parts according to the objectives of the work. The first part being the methods used for achieving the first objective: the detection of gaming. The second part therefore being the methodology used for the prevention of gaming. And the third part was the overall analysis and evaluation methodology used to interpret the combined results of all the work.

The detection of gaming was essentially a repeating four step process: classroom observation, dataset construction, model construction, and analysis. The prevention of gaming was also a four-step process: design and development of interventions, deployment of interventions across control groups, gathering data and feedback, and analysis. The particular details and outcomes of each of these processes are contained in the following sections.

4.1 Detection of Gaming

Our first objective was to build a model that can reliably identify off-task gaming behavior. Armed with a reliable prediction model, strategies can be developed to stop this undesirable behavior, with the goal of maximizing tutor effectiveness. The methodology used here was adapted from a prior study by Baker et al [2], and is essentially a four-step process: (1) recording classroom observations of students using the tutoring software, (2) dataset creation based upon those observations to be used by machine-learning algorithms, (3) the construction of classifiers (prediction models) using the datasets, and (4) analysis of the results.

4.1.1 Classroom Observation

In order to construct a machine-learned model to detect gaming it is necessary to have suitable training instances. The observation methodology and coding scheme were adapted from two prior studies [2][3] for use with students using the *Assistments* tutoring system.

Classroom observation generally involved observing students using the *Assistments* tutoring system in real classes in the Worcester Public school district. The order of observations was pre-determined by the layout of the computer lab and the seating order of the students, as not to introduce an observer selection bias. Observations were recorded as unobtrusively as possible, without the students being aware that observation was taking place (students treated observers as assistant teachers and displayed no knowledge that they were being systematically observed). Each recorded observation was a triple of the observation time, the student's identity (alias), and recorded behavior.

After the class was logged into the system, the observer walked around the room and wrote down the student screen-names that are displayed above the logout button, one user per row on the observation sheet. The layout of a given schools computer lab determined how observations were made and in what order. Observations were made at two Worcester area schools, Worcester East Middle School and North High School, which both have computers that are positioned in back-to-back rows. Clearly, reliable observations cannot be gathered on a given student if the observer is standing immediately behind them (students easily detect the presence of the observer in that case and become self-consciously more aware). Therefore, observations were taken from a modest distance. While attempting to observe a student in row B, for example, the observer might be standing in row A (an adjacent parallel row) or at the head of the rows (somewhat diagonally to the student). The positioning and height of the computer monitors, height of the computer cases, and the seating position of the student, all of which have a tendency to obscure line of sight, also effect positioning of an observer. Since the observer needs this line of sight to the target student's monitor and actions, positioning during a given observation is often highly constrained.

Watching a given student non-stop for 20 or 30 seconds is often an issue for the following reasons: the students (including the observed student), perhaps due to the lack of motivation or focused discipline on the program, are often watching the observer and modify their behavior or alert other students to the observation – thus creating potential for inaccurate or tainted observations. Therefore, to avoid having observations scuttled in this way, the observer must be somewhat more sneaky and calculating in their methods. Rather than attempting the risky prospect of observing one student directly for 20 or 30 seconds in such an undisciplined environment, and because of the positioning constraints mentioned previously, observers made an adjustment and attempted to observe groups of students simultaneously. In this approach, a group of adjacent students is observed for a longer time period, watching them out of the corner of the eyes and floating back and forth from student to student to give the appearance of merely casually looking at what a group students are doing versus carefully observing what a particular student is doing.

One strength of this approach is that a more accurate observation can often be made for the entire time period. For example, a student may be talking for a moment at first, but another moment of observation reveals that they are actually working diligently on task.

A copy of the observation directions and measurement definitions (adapted from measurements in [3]) that were given to observers appear in the listings below.

1. Record the date, teacher, class period, and period start time.
2. Record the screen-names of the students (as the rows) in the order that they are sitting around the room, and demarcate the observation groups by separator lines.
3. Create columns for each observation period.
4. As each group is observed, record the start time at the top of the column. Observe the entire group of students for approximately 20 to 30 seconds per student.

- For example, a group of 3 students should be observed for 60 to 90 seconds. The possible variation of observation times is left to the observer depending on the consistency or deviation in the students' behavior in order to get a representative measurement.

5. At the end of the observation period and after all measurements are made, record the end time at the bottom of the column.
6. Move to the next group and repeat the process.

Listing 1 – Observation Directions

One of the following *numerically* coded measurements was recorded for each student during a particular group observation (organized into categories for clarity):

Category A: On Task
1. On Task, Tutor
2. On Task, Paper or Teacher (including talking about the problem)
3. On Task, Talking (talking while working, subject matter of conversation is irrelevant)

Category B: Off Task

4. Off Task, Talking
5. Off Task, Inactive (including web-surfing, staring into space, sleeping, et cetera)

Category C: Gaming

6. Gaming (guessing and checking or bottom-out hinting)

Listing 2 – Measurement Definitions

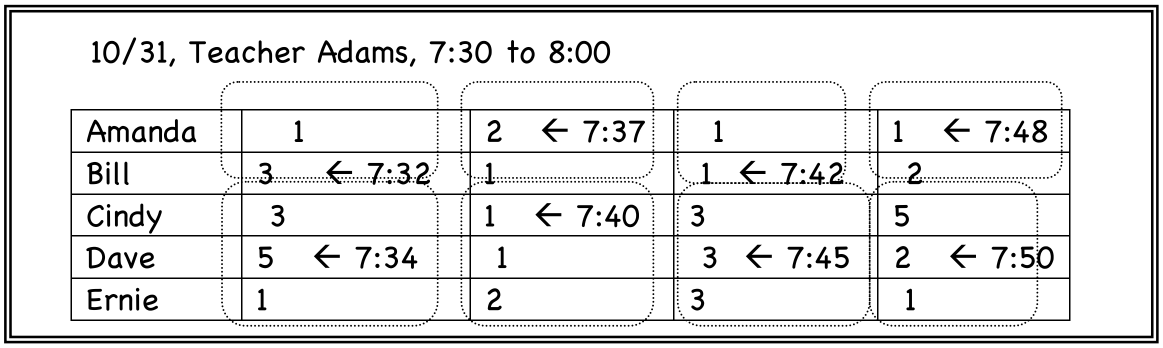
During a given observational period, a student might exhibit multiple behaviors (the numerical codes 1 through 6 in Listing 2). In that case, rather than record all the behaviors, observers were instructed to give priority in the following way. Off-task gaming behavior was the highest priority observation, followed by the three on-task behaviors (sorted by seemingly least engaged to most engaged), followed by the remaining two off-task behaviors (sorted by seemingly most active to least active). A copy of the priority table distributed to observers is shown below in Figure 2.

Priority (Highest to Lowest)	Measurement
1	6 – Gaming
2	3 – On Task, Talking
3	2 – On Task, Paper or Teacher
4	1 – On Task, Tutor
5	4 – Off Task, Talking
6	5 – Off Task, Inactive

Figure 2 – Observation Measurement Priorities

So, for example, if a given student is observed both *talking off task* (observation number 4, priority number 5) and *talking on task* (observation number 3, priority number 2) then the observation is recorded numerically as 3 (*On Task, Talking*) as it has the highest priority. Numerical codes were used to prevent anyone who might accidentally see the sheet from interpreting it.

Given the instructions discussed above, a resulting observation sheet might appear similar to the mock-up provided below in Figure 3.



10/31, Teacher Adams, 7:30 to 8:00

Amanda	1	2 ← 7:37	1	1 ← 7:48
Bill	3 ← 7:32	1	1 ← 7:42	2
Cindy	3	1 ← 7:40	3	5
Dave	5 ← 7:34	1	3 ← 7:45	2 ← 7:50
Ernie	1	2	3	1

Figure 3 – Example Observation Sheet Mock-up

Figure 3 shows a hypothetical class on October 31st for teacher "Adams" that lasted between 7:30 and 8 am. The example shows five students, who were each observed four times. The cells of the table that are circled (with the dashed line) demarcate the observation groups and the time within that circle denotes the observation time for the group.

Observations for this research spanned across two academic years (2004-2005 and 2005-2006), two schools (Worcester East Middle School and North High School), two grades (8th and 9th graders), and two experience levels of users (experienced and novice). All of the observations recorded for this research are summarized below in Figure 4.

	Dataset 1	**Dataset 2**
Year	2004-2005	2005-2006
School	Worcester East Middle	North High School
Number of Classes	8	10
Average Class Time	50 minutes	10-20 minutes
Observations	850	297
Student Experience with Tutor	End of year, experienced users	Beginning of year, novice first-time users

Figure 4 – Summary of Classroom Observations

Because of the variation in our two datasets, and because of the anecdotal impression that novice users quickly become experienced users of the *Assistments* system, Dataset 1 was chosen as the primary dataset for modeling gaming detection (see section 4.1.3).

After all observations were recorded, the sheets were collected and transcribed into a spreadsheet and then loaded into a database. The results of the observational method were then analyzed for inter-rater reliability before utilizing the observations themselves in the construction of our detection model.

To ensure that the observation methodology employed was reasonable and not subjective to observer bias, an inter-rater reliability study was performed. Two observers (one of which was the author) were provided with the observation instructions and then observed two classes, with students observed in the exact same order and at the exact same time. The two observers made 71 observations each. The reliability across all behaviors (the six numerical behaviors enumerated in Listing 2) had 77% accuracy (57 out of 71 observations matched). This accuracy was acceptable, but fell short of our desired consistency across observers. Thankfully, the reliability across the categorical behaviors (the three alpha-encoded behaviors in Listing 2) had 97% accuracy (69 out of 71 observations matched), while there was 100% agreement in the identification of gaming behavior. Since our classifier was aimed purely at the identification of off-task gaming behavior, as opposed to the total differentiation of all behaviors, those results suggest a suitable level of consistency.

4.1.2 Dataset Creation

The next step in detecting off-task gaming behavior is the construction of datasets for input into machine-learning algorithms. Dataset construction required joining the recorded classroom observations with database logs of the student's actions within the *Assistments* system.

The *Assistments* system automatically records all user actions and events except mouse movements into an action log. From the action log we can distill information such as a student's number of attempts (including whether the attempt was correct or incorrect, or if it was the first attempt on a given problem), numbers of hint requests (including bottom-out hint requests), and action response time in milliseconds (captured through a client-side script). Each row of the action log is a separate student action. Actions were joined to the recorded classroom observations by user identification and time.

But given the length of time spent observing particular students, it is not immediately clear which actions should be matched with a particular observation. To resolve this issue, actions were joined to the observations using an "unsupervised action filter" based on a variable "time window." Informally, a time window is defined here as a dilation of time around the recorded observation time. Being unsure as to what time window size would be reasonable, 5 sizes were utilized: 30 seconds, 1 minute, 2 minutes, 4 minutes, and 6 minutes. For example, given an observation and a 2 minute time window, all actions made within 1 minute prior to and 1 minute after the observation were included in the generation of the final dataset for training detection models. Additionally, the filter is considered "unsupervised" because no attempt was made to filter in or out actions based on their applicability to the recorded observation value.

One question that results from this method of generating the datasets is how the size of the time window effects the classification of student's behavior. It turns out that the size of the window has only minor impacts on the predicted classification value (i.e. whether a student was gaming or not), but does impact the resulting generated rules (in the case of human-readable models) in curious ways. These results are discussed further in section 4.1.4.

Using the two data sources (recorded observations and action logs), a number of datasets were generated via unsupervised action filtering using time windows. The composition of the datasets was modified after the initial experiments, however in the final versions of the datasets, each instance had 1430 attributes and 1 classification value (gaming, *true* or *false*). Since the objective of the model being built is the detection of gaming behavior, and as the inter-rater reliability results discussed in the previous section were best suited towards differentiating observations at the categorical level, the six observed classifications shown in Listing 2 were rolled up into either gaming is *true* (observation number 6) or gaming is *false* (all other observations). The remaining attributes are itemized and defined below in Listing 3. Additionally, it is worth noting that the success of any machine-learning algorithm is dependent on a number of factors, one of which is the relevancy of the attributes. Therefore, the selection of attributes to include or exclude within a dataset is an important exercise. Because the work of Baker et al [2] was used as a partial guide for this process (as discussed in section 3.1), some of the attributes used are very similar to theirs, only adapted to the particulars of the *Assistments* system and the variable time windows.

- Actions: the total number of all actions for the observation and within the time window
- Attempts: six separate attributes for the total number of all attempts, correct attempts, incorrect attempts, correct first attempts, and incorrect first attempts for the observation and within the time window

- Attempt Time: five separate attributes for the sum, minimum, maximum, average, and standard deviation of all attempt times in milliseconds. Also four Boolean attributes were included indicating whether attempt times were slow, extra-slow, quick, or extra-quick, which were calculated by comparing the student response time with the average response time of all students on the given problems (and plus or minus the standard deviation of all student response times for the extra-slow and extra-quick attributes)
- Hints: two separate attributes for the total number of hint requests and bottom-out hint requests for the observation and within the time window
- Hint Time: five separate attributes for the sum, minimum, maximum, average, and standard deviation of all hint request times in milliseconds. Also four Boolean attributes were included indicating whether hint request times were slow, extra-slow, quick, or extra-quick, which were calculated by comparing the student response time with the average response time of all students on the given problems (and plus or minus the standard deviation of all student response times for the extra-slow and extra-quick attributes)
- Problems: two separate attributes for the total number of top-level problem questions, and the total number of follow-through helping questions, for the observation and within the time window
- User-Interfaces: two separate attributes for the total number of questions that featured a multiple-choice user-interface and another for the total number of questions that featured a textbox user-interface for the observation and within the time window
- Replays: the total number of times a problem was "replayed" by the *Assistments* tutor runtime [8] for the observation and within the time window (this generally indicates that the student tried to exit the system and the runtime had to "replay" the students attempts on a given problem to reconstruct the tutors agenda *exactly* for the given problem)
- *pmpKnow*: "poor man's prior knowledge," the probability that the student possesses the prior knowledge required to answer the given question correctly. Prior knowledge in ITS is often determined by knowledge tracing, however the *Assistments* system currently lacks dynamic knowledge model tracing, so as a substitute we use the poor man's version: the student's percent correct across all previous problems. Also four Boolean attributes were included indicating whether the prior knowledge was high, extra-high, low, or extra-low in comparison to the average prior knowledge of all students and in combination with the standard deviation of that average (for the extra-high and extra-low variables)
- Problem-Difficulty: four separate attributes for the minimum, maximum, average, and standard deviation of problem difficulties for all problems covered by the observation within the time window. Problem difficulty is a number between 0 (easy) and 1 (hard) that is the percent correct on first-attempt by all previous students. The combination of these values would hopefully represent the range of difficulty during the observation.
- Ratios: six separate attributes representing the following ratios: the number of attempts per problem, the number of correct attempts per problem, the number of

incorrect attempts per problem, the number of hints per problem, the number of bottom hints per problem, and the number of replays per top-level problem.
- Pair-wise Interaction Effects: approximately 1400 separate attributes representing the quadratic effects between any two original attributes (one of the attributes previously listed above). For example, the total number of hints times the average problem difficulty. The list of pair-wise interaction effect attributes is comprehensive (all the original attributes have a pair-wise interaction effect attribute with every other original attribute, including itself)

Listing 3 – Attribute Definitions

After computing the values for all the original attributes (everything except the pair-wise interaction effect attributes) within the *Assistments* database, the results were exported as a comma-separated text file. The text-file was fed into a small program that computed all the pair-wise interaction effects and appended them to each row, and then converted the file format into an ARFF file for use with the WEKA machine-learning system [9].

4.1.3 Algorithms and Experiments

The final datasets generated via the process described in the previous section were then used within the WEKA machine-learning system. WEKA is an open source environment that implements a wide variety of common and not-so-common machine learning algorithms, meta-classifiers, and preprocessing routines [9]. Given of the availability of WEKA, there was no need to proverbially "reinvent the wheel" by coding any algorithms by hand.

The generation of classifiers using WEKA was accomplished in two phases. There was an initial exploratory phase (a "fishing expedition" to initially compare various algorithm results over the data, determine the appropriateness of the time windows, and see whatever else might percolate to the surface) and a redesigned final phase (a more focused phase with a limited number of algorithms using datasets that had been refined based on findings from the exploratory phase). A high-level summarization of the differences between the two phases is outlined in Figure 5.

	Exploratory Phase	**Final Phase**
Years	{2004-2005, 2005-2006}	{2004-2005}
Time Windows	30 seconds, {1, 2, 4, 6} minutes	30 seconds, {1, 2, 4, 6} minutes
Classification Schemes	{all, categorical, gaming, active, on-task}	{gaming}
Number of Attributes	29	1431
Number of Datasets	100	10
Number of Algorithms	12	6
Testing Method	10-fold cross-validation	10-fold cross-validation, and leave-one-out cross-validation (LOOCV)
Attempted Algorithms	ZeroR, OneR, J48, ID3, IBk, Naïve Bayes, BayesNet, LWL, MultiLayerPerceptron, PRISM, Logistic Regression, Ada-Boost	ZeroR, OneR, J48, PRISM, MultiLayerPerceptron, Logistic Regression

Figure 5 – Summary of Machine-Learning Experiment Phases

The initial exploratory phase used datasets from both observed academic years (2004-2005 and 2005-2006), all five time windows (30 seconds, and 1, 2, 4, and 6 minutes), and various classification schemes. The various classification schemes explored included using all six behaviors (*on-task*, *on-task-talking*, *on-task-paper*, *off-task-talking*, *inactive*, and *gaming*), the three categorical behaviors (*on-task*, *off-task*, and *gaming*), gaming as a Boolean (*true* or *false*), active as a Boolean (whether the student was actively using the system, *true* or *false*), and on-task as a Boolean (*true* or *false*). This resulted in over 100 distinct datasets when preprocessing considerations were included (some algorithms required attributes to be discretized, meaning that numeric values were converted into nominal values using entropy calculations). Additionally, the following attributes were not present in the datasets during the initial phase, as they were only added later for the final phase (to strengthen the classifiers): prior knowledge, user-interface attributes, standard deviations from the average response times of all other students on the given problems, and the pair-wise interaction effects.

Using 10-fold cross-validation as the testing method, 12 different algorithms were used to generate models including decision tree methods, lazy methods (k-nearest neighbors), locally weighted learning, Bayesian methods (naïve Bayes and Bayesian networks), a neural network (feed-forward multi-layer perceptron using gradient descent back-propagation with the number of hidden layers equal to the number of attributes), a sequential-covering propositional-logic rule generation algorithm (PRISM) [10], as well as logistic regression (approximation of a logistic function). A large number of algorithms were used out of curiosity because each has advantages and disadvantages (which are outside of the scope of this document) that could potentially reveal different kinds of relationships within the data. However, it should be noted that some of the algorithms generate human-readable rules while others produce mathematical models that are often difficult to interpret by humans.

The Cartesian product of datasets and algorithms resulted in literally thousands of experiments in the initial phase alone. Most of the initial results were discouraging, in that the classification accuracy was very poor (often worse than ZeroR, which means worse than simply guessing the classification that is most probable according to the distribution of the training instances), and the confusion matrices revealed various amounts of inconsistency or "confusion" from minor to intense (a confusion matrix is a method of tabulating the number true and false positives and true and false negatives broken down by true classification value and predicted classification value).

Following these initial poor results, a period of critical re-examination followed, where the following questions were explored: why was the performance of the algorithms so poor? Was it because of incorrect or unreasonable classification schemes? Was it because of incorrect, biased, or poor observations? Was it because of incorrect recording of observation timestamps? Was it because of incorrect or unreasonable time windows? Or was it because of unsuitable, inappropriate, or missing attributes? Each of these questions is answered in turn. The gaming/not-gaming classification scheme has been used in at least two prior studies [2][3] with reasonable success, so that does not seem as though it should be a major problem. However, given the results of our inter-rater reliability study, and the fact that we only want to detect gaming behavior, most of the less reliable classification schemes seem bound to create noisy data and therefore probably should not be utilized. The timestamps were not an issue after the timing device was synchronized with the database times to account for delays in the logging process. The complete unsuitability of all the time windows seems unlikely, especially since the 30 second window corresponds almost exactly to the period of observation, and the other longer windows were merely used to test if behavior could be generalized as the period of time around an observation dilates. Re-examination of [2] revealed that perhaps there were other important attributes that should be considered. The following attributes were not used in the initial phase, but were added to the final phase: prior knowledge, user-interface attributes, standard deviations from the average response times of all other students on the given problems, and the pair-wise interaction effects. Additionally, it was decided to only attempt to classify gaming as *true* or *false* for a given observation. With the addition of these attributes and the narrowing of the classification objective, the second phase began.

The redesigned second phase (the final phase) only used the observations from 2004-2005 academic year (as explained in section 4.1.1), with all five time windows, and a single classification scheme (gaming as *true* or *false*). This reduced the number of datasets used in the first phase from 100 down to 10 (including discretization of numeric values for PRISM). Additionally, the prior knowledge, user-interface, standard time deviations, and pair-wise interaction effect attributes were included. Finally, the number of algorithms used was trimmed down to only those that could generate human readable rules (OneR, decision-trees, PRISM) and those that involved functional approximation, such as logistic regression. Initially, each algorithm was run over each dataset using 10-

fold cross-validation. Unfortunately, all the functional approximation algorithms (Bayesian methods, neural network, and logistic regression) did not complete due to memory limitations (1.25 gigabytes) of the host machine (the memory efficiencies of these algorithms were insufficient for processing 1400+ attributes), sometimes crashing after several days of processing. The results were then examined for (1) classification accuracy, (2) accuracy of the confusion matrices, and (3) reasonable rules, especially those that might corroborate expected finding based on previous studies, or other interesting results.

4.1.4 Results and Discussion

The results from the second phase were remarkably improved over the exploratory phase in terms of classification accuracy and confusion matrices, however the algorithms did not significantly outperform each other (according to statistical tests automatically performed by WEKA). Therefore, choosing a final model rested on a selecting a classifier that generated reasonable rules that corroborated both the surface-level hallmark characteristics of gaming (rapid fire guessing-and-checking and bottom-out hinting) and the findings of previous studies (such as students with low prior knowledge were more likely to game on difficult problems [2][3]).

The classifier that was ultimately selected as the preferred model was generated using the J48 decision tree algorithm [9] (based on Quinlan's C45 algorithm [11]), using the default algorithm settings except that the minimum number of instances per leave node was decreased from 2 to 1 (since the number of *false* gaming instances far outweighs the number *true* gaming instances), and the confidence factor was increased from 0.25 to 0.5 (which resulted in slightly less pruning). Although there are algorithms with faster classification times and slightly higher accuracies, this model was chosen because across all training and testing folds it produced reasonably clean confusion matrices, generated human-readable rules that were relatively easy to interpret and even loosely corroborated findings from past studies. Some sample rules are shown and translated into English in Figure 6, while others are included as Appendix 1.

Sample Rule, 30 Second Time Window	

Rule Sampled from the Decision Tree	`CORRECT_ATTEMPTS*PMPKNOW_XLOW <= 0: FALSE`
	`CORRECT_ATTEMPTS*PMPKNOW_XLOW > 0`
	`|   TOTAL_HINTS*TEXTBOX_PROBLEMS <= 6`
	`|   |   HINT_TIME_SHORT*PMPKNOW <= 0.16: FALSE`
	`|   |   HINT_TIME_SHORT*PMPKNOW > 0.16`
	`|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS <= 1: TRUE`
	`|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS > 1: FALSE`
	`|   TOTAL_HINTS*TEXTBOX_PROBLEMS > 6: TRUE`

English Interpretation	If the student has prior knowledge, then gaming is probably false. Otherwise, when the student has little prior knowledge, then we need to look at the rate of hints on textbox problems. If the rate of hints on textbox problems is high, then the student is gaming. If the rate is low, then we additionally need to consider then number of total attempts per *Assistment*. A student with less than 1 attempt per *Assistment* is gaming (probably because they asked for a hint without even attempting the problem), otherwise gaming is probably not occurring.

Figure 6 (a) – Sample Rule, 30 Second Time Window

Sample Rule, 1 Minute Time Window	

Rule Sampled from the Decision Tree	`SDV_HINT_MS*PMPKNOW_XLOW > 525`
	`|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY <= 0.59: TRUE`
	`|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY > 0.59`
	`|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY <= 11.84: FALSE`
	`|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY > 11.84`
	`|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS <= 4: TRUE`
	`|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS > 4: FALSE`

English Interpretation	If the student has extremely low prior knowledge and there is more than half a second of deviation in hint times, and if the problems are moderately easy or if there has been at least one very difficult item and the student has taken many actions but few correct attempts on textbox problems, then the student is gaming. This rule reflects the gaming tactic of guessing and checking.

Figure 6 (b) – Sample Rule, 1 Minute Time Window

Sample Rule, 2 Minute Time Window

Rule Sampled from the Decision Tree

```
CORRECT_ATTEMPTS*PROB_HINT_RATIO > 5.4
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS <= 0
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO <= 0.37
|   |   |   PMPKNOW_HIGH <= 0: TRUE
|   |   |   PMPKNOW_HIGH > 0: FALSE
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO > 0.37:
FALSE
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS > 0: TRUE
```

English Interpretation

If the student has a high hint-to-problem ratio, a low correct attempt ratio in terms of average problem difficulty, and low prior knowledge then the student is gaming. If there is a high hint-to-problem ration and incorrect first attempts and replays then the student is gaming. These rules reflect the tactic of executive help-seeking and bottom-out hinting.

Figure 6 (c) – Sample Rule, 2 Minute Time Window

Sample Rule, 4 Minute Time Window

Rule Sampled from the Decision Tree

```
CORRECT_ATTEMPTS*HINT_TIME_XLONG > 1
|   ATTEMPT_TIME_LONG*MULTIPLE_CHOICE_PROBLEMS <= 1
|   |   TOTAL_HINTS*PMPKNOW_HIGH <= 4: FALSE
|   |   TOTAL_HINTS*PMPKNOW_HIGH > 4
|   |   |   ACTIONS <= 22: FALSE
|   |   |   ACTIONS > 22: TRUE
|   ATTEMPT_TIME_LONG*MULTIPLE_CHOICE_PROBLEMS > 1
|   |   ACTIONS <= 17: TRUE
|   |   ACTIONS > 17: FALSE
```

English Interpretation

If the student is has extra long hint times, but is still asking for more than 1 per minute, has high prior knowledge, and is taking lots of actions (more than 5 per minute) then the student is gaming. If the there are extra long hint times, and long attempts with multiple-choice items, and less than 5 actions per minute, then the student is gaming. These rules reflect a student with high knowledge engaging in a mixture of guessing-and-checking and bottom-out hinting, suggesting low motivation, and perhaps a “gamed-not-hurt” status.

Figure 6 (d) – Sample Rule, 4 Minute Time Window

Sample Rule, 6 Minute Time Window

Rule Sampled from the Decision Tree

```
TOTAL_HINTS > 18
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO <= 0.47
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS <= 5937981
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG <= 2: TRUE
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG > 2: FALSE
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS > 5937981:
FALSE
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO > 0.47: TRUE
```

English Interpretation

If a student is averaging 3 or more hints per minute and their prior knowledge is extremely low with a low bottom-out-hint-to-problem ratio, but the hints are taken quickly, then gaming is true. If a student is averaging 3 or more hints per minute and prior knowledge is extremely low and about half of all problems have been bottom-out hinted, then gaming is true. These rules detect the rapid and systematic exploitation of hinting over a long period of time.

Figure 6 (e) – Sample Rule, 6 Minute Time Window

The sample rules in Figure 6 (a) through (e) appear to have some reasonability given what is known about off-task gaming behavior. These rules in particular offer further support to the hypotheses of Baker et al [2][3] that suggest that one cause of gaming is low prior knowledge combined with problem difficulty. The rules for the four-minute window also could be interpreted to identify the class of “gamed-not-hurt” students, which would lend a bit more credence to the validity of the differentiation between students whose learning is affected by gaming and those who are not. Finally, the success and reasonableness of the four and six minute rules suggests that using longer time windows does not adversely effect the detection of gaming, given relatively few observations, and in fact improves as those students who game tend to make a habit of it, and identifying them becomes easier and easier as they continue their off-task behavior.

The J48 algorithm that generated our preferred model was then rerun using leave-one-out cross-validation (LOOCV). LOOCV is a more robust method of training and testing a classifier, which involved the decision tree classifier being generated 850 times, each time using 849 of the 850 instances for training purposes and leaving out one instance (a different instance on each iteration) for testing, and then using the model to predict whether the 850th instance (the one left out of the training) was gaming or not. This process was repeated for each of the datasets (one for each time window). The results of the LOOCV are summarized in this section and enumerated in Appendix 2. Constructing the decision trees with LOOCV resulted in a majority of models having 100% classification accuracy (and therefore completely clean confusion matrices), which might suggest that the models would be applicable to new instances of reasonable similarity.

In this case, the phrase "reasonable similarity" should cover new instances generated under similar circumstances and conditions to those used as training data. Probably the most important similarity requirement for applicability would be that the new instances were from experienced users, as novice users (for example, the instances in Dataset 2, from the 2005-2006 academic year) had different reaction times and slower progression rates through the tutoring content than the experienced users did. The discrepancy in pace could have been a result of either familiarity (or lack thereof) with the user interface and the system's regularities or with student's prior knowledge. Most of the novice users had not completed enough problems by the time of writing this document to effectively determine their knowledge level for verification of this hypothesis, nor were they given pre or post-tests. On the other hand, one could argue that the difference in pace between novices and experienced users is *not* an issue, for two reasons. First, not all the generate rules rested on student speed or pace. And second, the average response times for problems which were used to calculate whether given students were fast or slow were computed using paces from both novice and experienced users, so any possible discrepancy might automatically be effectively smoothed out in that way.

While most of the models resulting from LOOCV had 100% classification accuracy, averaging out the results of all models paints a slightly less optimistic portrait – about 96% accuracy. Given the low rate of observed gaming (19 out of 850 observations, ~2.2%), the effectiveness of the models becomes questionable. Analysis of the confusion matrices helps our understanding of how the models perform. On average, the models tend to correctly identify non-gaming instances about 98% of the time, while correctly identifying gaming instances only about 19% of the time. Clearly, this is far worse than the ideal result – 100% positive identification of all gaming instances. However, if we consider that gaming is much more harmful to learning than other behaviors and it is such an infrequent behavior, then 19% of gaming instances may seem better than what might be expected from chance alone. So, while the model accuracy leaves something to be desired, I am at least satisfied in the general reasonability of the resulting "rules" given what is known about gaming behavior.

Ultimately, the results of the final model were satisfactory for the needs of the project for the following reasons. As a data-mining process, construction of the model helped verify or identify some of the underlying hidden variables that lead students to game (low prior knowledge, for example). And, as an operational classifier, a practical number of instances were reliably and accurately classified (when training and testing using LOOCV), the generated rules were human-readable, and they reasonably captured the hallmark surface-level characteristics as well as other known causes of off-task gaming behavior. Now that such a model is available, it can be made use of in several different ways: (1) it could be outfitted into the *Assistments* tutoring software to dynamically detect gaming behavior and be used as the driving force for the invocation of various intervention strategies, and (2) it can be used as an objective evaluator of various intervention strategies within controlled experiments. This research partially utilized the model for the second task only.

4.2 Prevention of Gaming

The second objective of this research was to develop, deploy, and then compare active and passive interventions. Active interventions combat gaming behavior, potentially unfairly penalizing on-task students, while our hypothesis is that passive interventions can prevent gaming entirely. The methodology employed here was a four-step process: (1) design and development of interventions, (2) deployment of interventions across control groups, (3) gathering data and feedback, and (4) analysis.

4.2.1 Design and Development of Interventions

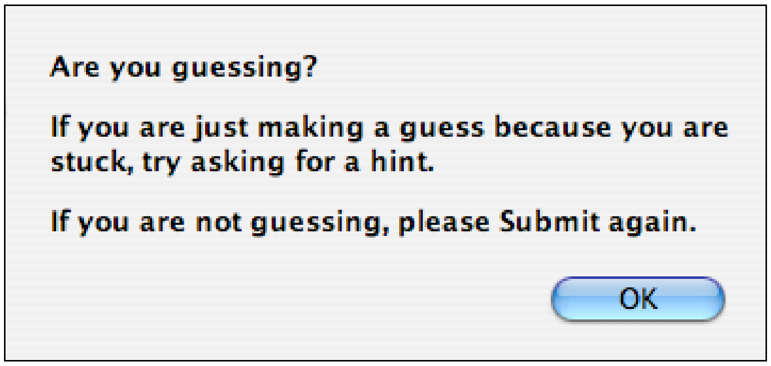
In order to intervene with off-task student behavior, intervention mechanisms and triggers to invoke them are required, necessitating that they be designed, implemented, and deployed. Since one objective of this research was to compare dynamic active and dynamic passive interventions (see section 3.2 for a detailed explanation of the differences), the development of interventions focused on those two types. Since off-task gaming behavior has two hallmark appearances (rapid fire guessing-and-checking and bottom-out hinting), two active interventions were separately developed to respond individually to each type of gaming behavior. Only one passive intervention was developed, with the intention of preventing both types of gaming simultaneously. Additionally, it was hypothesized that that passive intervention would eliminate the active intervention gaming arms race (where students game the gaming interventions in an act of meta-gaming) by not altering the functional behavior of the tutoring system in any way that students could take advantage of.

The two active interventions were not invoked by the machine-learned detection model (detailed in section 4.1) as it was not completed at the time of implementation, but were triggered by much simpler algorithms that marked a student as guessing and checking or bottom-out hinting *prima facie* of the appropriate surface-level characteristics. Once a student was marked for a particular intervention, the student would encounter the intervention if during their very next action they attempted to answer a question (if they were suspected of rapid guessing) or they requested a hint (if they were suspected of bottom-out hinting), until the suspicion of gaming sufficiently decayed. On the other hand, the passive intervention (when turned on in an appropriate control group) had no triggering mechanism, as it was always visible and potentially passively influencing the tutoring session at all times.

Before discussing the active intervention algorithms, it is necessary to explain some terminology rooted in the *Assistments* project. In the *Assistments* system, a problem to be solved is called an "Assistment," which includes the original question, all the follow-up tutoring questions, and hints necessary to solve the problem. The individual follow-up tutoring questions are colloquially referred to as "scaffolding" questions. A student generally has one try (attempt) to answer an original question, but an infinite number of attempts to answer "scaffolding" questions. If a student correctly answers the original question, the "scaffolding" questions are skipped and the student goes onto the next

"Assistment." However, if the original question is answered incorrectly, the student must sequentially work through all the relevant "scaffolding" questions [4].

The guessing-and-checking detection algorithm has a few simple rules based only on student actions. If a student incorrectly answers any "scaffolding" question on three consecutive attempts, they are credited as "shot-gunning" (or guessing-and-checking) the question. If a student shotguns three or more different "scaffolding" questions, they are marked as a shot-gunner (i.e. as a gamer) and will then be eligible for the guess-and-check gaming intervention. The guess-and-check intervention will then be fired on the next problem attempt. If a student completes an entire "Assistment" without shot-gunning a single "scaffolding" question, then the student's total "shotgun count" is reduced by 1. When the shotgun count is greater than or equal to 3 for a given student, they are considered to be a guessing-and-checking gamer. When the shotgun count is less than 3, or is reduced to be less than 3, the student is considered to be a non-gamer. The guess-and-check gaming intervention itself, was a JavaScript alert message, worded as carefully as possible as not to unintentionally insult the student or unjustly accuse them of doing anything improper. A screen capture of the alert message appears in Figure 7.

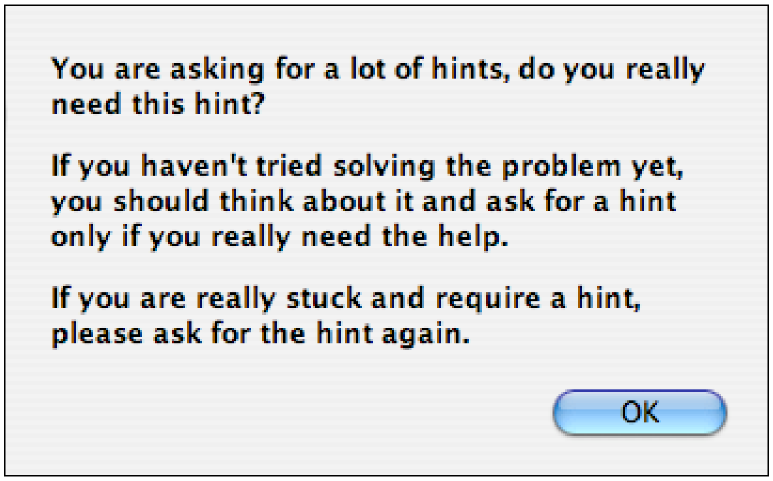

Are you guessing?

If you are just making a guess because you are stuck, try asking for a hint.

If you are not guessing, please Submit again.

Figure 7 – Guess-and-Check Gaming Intervention

The bottom-out hinting detection algorithm is very similar to the guessing-and-checking detection algorithm, and also only has a few simple rules based on student actions. If a student requests a bottom-out hint (the final hint that directly supplies that correct answer for a given question), they are credited as bottom-out hinting the question. If a student bottom-out hints three or more different "scaffolding" questions, they are marked as a "bottom-feeder" (i.e. as a gamer) and will then be eligible for the bottom-out hinting gaming intervention. The bottom-out hinting intervention will then be fired on the next hint request (regardless if it is a bottom-out hint or not). If a student completes an entire "Assistment" without bottom-out hinting a single "scaffolding" question, then the student's total "bottom-feeding count" is reduced by 1. When the bottom-feeding count is greater than or equal to 3 for a given student, they are considered to be a bottom-out hinting gamer. When the bottom-feeding count is less than 3, or is reduced to be less than 3, the student is considered to be a non-gamer. The bottom-out hinting gaming intervention itself, was a JavaScript alert message, worded as carefully as possible as not to unintentionally insult the student or unjustly accuse them of doing anything improper. A screen capture of the alert message appears in Figure 8.


You are asking for a lot of hints, do you really need this hint?

If you haven't tried solving the problem yet, you should think about it and ask for a hint only if you really need the help.

If you are really stuck and require a hint, please ask for the hint again.

Figure 8 – Bottom-out Hinting Gaming Intervention

It should be noted that the informal usage of the words “shotgun” or “shot-gunner” and “bottom-feeder” or “bottom-feeding” are never seen by students or teachers and are merely preferred terms used by the author.

The third intervention is a somewhat novel intervention mechanism, the dynamic yet passive intervention. The hypothesis is that rather than combating gaming with active interventions that risk unfairly impacting on-task students, we should be able to prevent gaming with a passive deterrent that does not penalize students (gaming or otherwise), while giving them performance and progress feedback. It was hypothesized that this could be accomplished with a moderately simple graphical plot of the student’s tutoring session. Featuring visual indicators of student actions and progress over time, the graphical plot would provide a summarization of student behaviors through emergent visual patterns. Prominently featured on-screen for easy viewing by the student and teachers, it was hoped that this passive method would prevent gaming through Panopticon-like paranoia [7] among gaming students (and other off-task students), but ignored by non-gaming and on-task students. And, with no modification of the tutor’s functional behavior, the gaming arms race might also be effectively eliminated.

Unlike the two active interventions, which are visually simplistic and functionally simple, the passive intervention is visually complex and has a more sophisticated generation mechanism. The component graphically plots (as opposed to enumerating as text) in a horizontal timeline all recorded student actions (such as problem attempts, hint requests, bottom-out hints), the amount of time each action took, and the outcome of the action. An example of the component after a few minutes of on-task use is shown in Figure 9 and a full screen-shot of the Assistments system with the embedded component is shown in Figure 10.

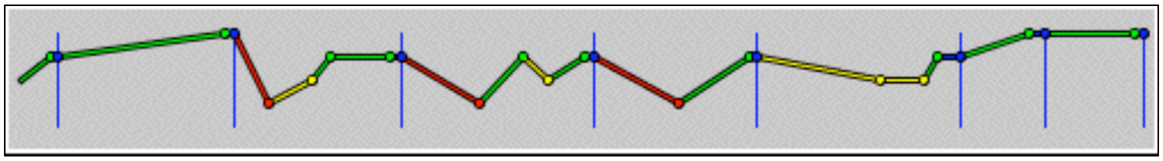
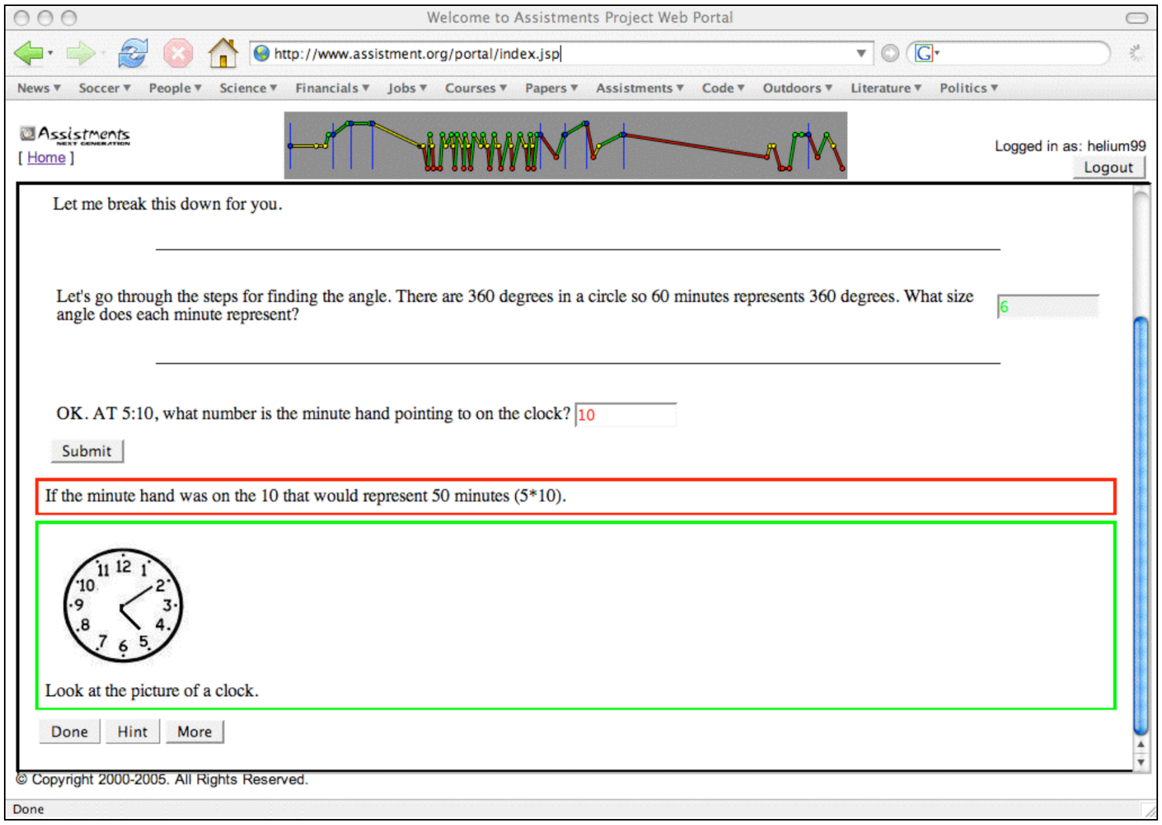


Figure 9 – Passive Intervention Example, On-Task

Figure 9 shows the passive intervention component. The design is basically a timeline (time progresses from left to right) that charts actions, where each action is indicated by a small colored point, and sequential actions are connected via colored lines. Each point has associated summary text (not shown) that identifies the action and relevant details and results of the action on mouse-over. The horizontal distance between points represents the amount of time between the actions (the graphic scales as time passes, so the distance is dependent on the length of the current session). The type of action and its result determines the vertical distance between points, on a range where correct first attempts are the highest and bottom-out hints are the lowest. Furthermore, the action result is also represented by the color of the point and the color of the line connecting it to the previous point. The ubiquitous traffic-light color conventions of modern society are used here, where green is implicitly "good", yellow is "caution", and red is interpreted as "bad." Green is used when questions are answered correctly, yellow is used on hints, and red is used on incorrect answers and bottom-out hints. Additionally, blue points with blue vertical lines are used to mark the transition between problems, while pink points with accompanying pink vertical lines are used to mark a problem replays. As a summary estimate of the student's performance, the background color of the graphic (light grey in Figure 9) changes on a gradient from white to black based on the percentage correct of attempts (at one end of the spectrum, the color white is shown on greater than 90% correct, and on the other end of the spectrum, the color black is shown on less than 10% correct). Figure 10 shows another example of the component as it is situated within the *Assistments* ITS.


Welcome to Assistments Project Web Portal

http://www.assistment.org/portal/index.jsp

News ▾ Soccer ▾ People ▾ Science ▾ Financials ▾ Jobs ▾ Courses ▾ Papers ▾ Assistments ▾ Code ▾ Outdoors ▾ Literature ▾ Politics ▾

Assistments [Home]

Logged in as: helium99

Let me break this down for you.

Let's go through the steps for finding the angle. There are 360 degrees in a circle so 60 minutes represents 360 degrees. What size angle does each minute represent? 6

OK. AT 5:10, what number is the minute hand pointing to on the clock? 10

If the minute hand was on the 10 that would represent 50 minutes (5*10).

Look at the picture of a clock.

© Copyright 2000-2005. All Rights Reserved.

Done

Figure 10 – Assistments Screenshot featuring the Passive Intervention

The graphical component has two main functions – displaying a summary of user actions over time that would clearly classify the gaming-status of a student to an observer (such as a teacher) by emerging patterns of indictors (see Figures 11 and 12 for example of charts capturing off-task gaming behavior), and allowing a teacher to ask, "what did you do here?" and prompt an investigation into the student actions which could reveal a lack of motivation, or even a fundamental gap in the student's knowledge. The first function is addressed toward the goal of gaming prevention, and the second function is addressed toward a secondary goal of assisting a teacher in identifying student weaknesses or misunderstandings via a trace of actions through the student's session.

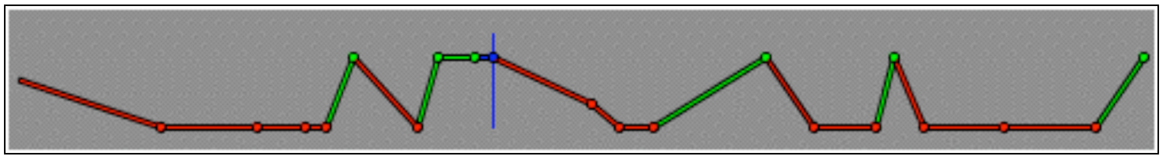


Figure 11 – Passive Intervention Example, Guessing-and-Checking

Both of these functions, classifying gaming-status by an emergent pattern of indicators and the identification of student weaknesses, is captured in Figure 11, which illustrates how a typical guessing-and-checking gaming student's actions would plot. As each red point and line represents an incorrect action, and the green points and lines represent a correct action, interpretation of this chart is fairly straightforward: a large series of rapid incorrect attempts is occasionally interrupted by a lone correct attempt (a lucky guess, perhaps) or the transfer to the next “Assistment,” only to be followed by a new stream of incorrect actions. This chart can only embody two possibilities: the student is engaging in off-task guessing-and-checking gaming behavior, or they have a fundamental lack of knowledge relating to the problem and are making lots of genuine errors without seeking help. Either way, teacher intervention is probably appropriate in such a case.

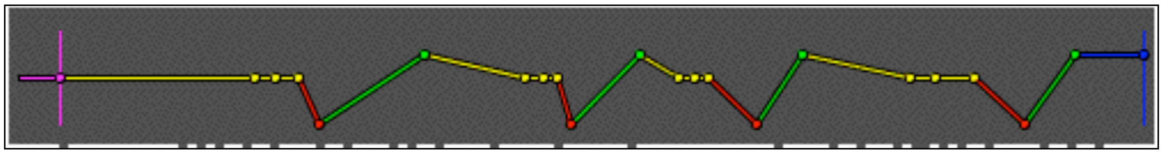


Figure 12 – Passive Intervention Example, Bottom-out Hinting

Similarly, Figure 12 shows the resulting graph of bottom-out hinting gaming. A series of rapid hint requests (the yellow points and lines in the vertical-middle of the component), followed immediately by a steep red drop (the bottom-out hint request) and then an upward green line (answering correctly after the answer was directly supplied by the bottom-out hint). This pattern occasionally appears in on-task usage, but when systematically repeated, is a clear indicator of off-task bottom-out hinting gaming behavior.

Both Figures 11 and 12 show the plots of students who were engaged in off-task gaming behavior after only a few minutes of a tutoring session. Since the graphical component scales as time progresses, the long-term identification of gaming appears slightly differently, but is even more readily apparent. Such an example appears below in Figure 13.

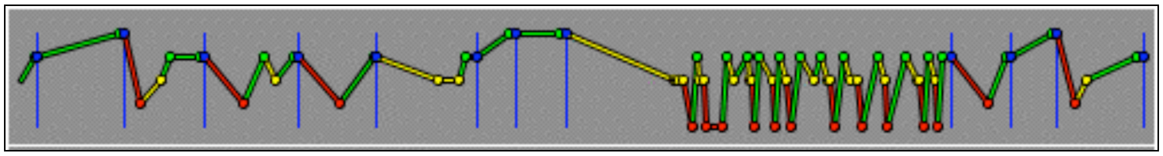


Figure 13 – Passive Intervention Example, Long Session with Partial Gaming

Figure 13 shows a plot of actions from a student who has completed 12 “Assistments” (represented by the thin vertical blue lines). Usage was relatively on-task, except for the ninth “Assistment,” which was completed via off-task bottom-out hinting. It is fairly obvious, even to an uninformed observer, that this portion of the graphical plot is far different than the rest of it. Not only does the graphical component plot the actions in a fairly logical manner (at the very least it is systematic), but it also does so in such a way that conspicuous patterns emerge in the case of off-task gaming behavior, especially as the length of the tutoring session increases.

4.2.2 Deployment of Interventions and Control Groups

With all three interventions designed and implemented, they had to be deployed within the *Assistments* system according to controlled experimental groups. The *Assistments* project has a large user-base of students who are conveniently grouped within the system by school, teacher, and class. These preexisting categorical groupings lend themselves particularly well to controlled studies.

In order to automatically take advantage of any controlled experimental studies that might be designed, the *Assistments* system first needed to be retrofitted with a system to configure and set experimental settings. Experimental settings were stored in the database in two tables, one for the experiment *definitions* and one for the experiment *values*. The experiment *definitions* table included the experiment name, description, the scope, the scale, and the legal values. The "scope" of an experiment was defined to be the range of effect – such as "all groups within Pennsylvania", or "all groups within Worcester County." The "scale" of an experiment was defined as the level that the values of the experiment could be set – such as "at the state level", or "at the school level", or "at the teacher level." The experiment *values* table simply contained a list of experiments (defined in the *definitions* table), with an assigned group (within a valid scope and scale), and an assigned value. This simple data store, along with simple objects to set and retrieve the values, was used to connect the experimental settings to the core *Assistments* runtime [8]. Additionally, a small application was included in the *Assistments* Portal application suite [12] to enable administrators to create, delete, and alter running experimental control groups as necessary. In this way, control groups were configured to receive the active and/or passive interventions.

The original plan for the experiments designed for this research was to have four control groups: one which received only the active interventions, a second which received only the passive interventions, a third that received both interventions, and a fourth that received no interventions. Due to a variety of issues, including the various dates that different schools and teachers started using the *Assistments* system with their classes, as well as teachers requesting the passive intervention be "turned on" for their classes, the control groups were not "controlled" in an ideally disciplined manner. Figure 14 summarizes the different experimental groups created for evaluation purposes of the developed interventions.

Active Interventions	Passive Interventions	Number of Classes	Number of Distinct Students
False	*False*	7	442
False	*True*	4	42
True	*False*	0	0
True	*True*	63	1289

Figure 14 – Summary of Experimental Control Groups

Once the control groups were set, a "toggle experiment" was performed in order to gather a better understanding of the effectiveness of the interventions. The phrase "toggle experiment" simply means that the status of the active and passive interventions was toggled after an initial number of times using the *Assistments* system (between 1 and 5 sessions, depending on the school, teacher, and class a student was in), and the rate of student gaming before and after the toggling is then compared. So, the group that had neither passive nor active interventions before the toggle, afterwards had both turned on. And the group that originally received both interventions had neither after the toggling.

4.2.3 Results, Analysis, and Evaluation

4.2.3.1 Control Group Summary

For the control group with both the active and passive interventions, there were 63 distinct classes that ran across 31 school days with 1 to 12 classes running per day (with an average of about 7 classes per day, with a standard deviation of about 4) over an approximately 1.5 month span (31 October 2005 through 15 December 2005). Some of the classes used the tutoring system more than once during that time span. There was an average of about 78 students using the tutor per school day, with a standard deviation of about 62. The average number of students per class was about 19, with a standard deviation of about 11.

4.2.3.2 Active Interventions

The results of the two active interventions, even though deployed to the control groups as a pair, are evaluated separately since they address different gaming behavior that likely has different causes and affects.

For the active intervention aimed at stopping guessing-and-checking gaming, there was an average of 18 interventions fired per class period (with a standard deviation of 12). The average number of interventions fired per gamer per day was 4 (standard deviation of 1), with the average number of students who received a stop-guess intervention per day being 18 (standard deviation of 13) out of the average of 78 students per day (standard deviation of 62), which means that roughly 23% of students received the stop-guessing intervention per day (an average of 18 guess-and-check gamers out of an average of 78 students).

For the active intervention aimed at stopping bottom-out hinting, there was an average of 19 interventions fired per class period (with a standard deviation of 14). The average number of interventions fired per gamer per day was 6 (standard deviation of 2), with the average number of students who received a stop-hinting intervention per day being 11 (standard deviation of 7) out of the average of 78 students per day (standard deviation of 62), which means that roughly 14% of students received the stop-hinting intervention per day (an average of 11 bottom-hint gamers out of an average of 78 students).

Overall, in the control group that had both the active and passive interventions, 19.4% (233 out of 1199) of students received stop-hinting interventions and 30.8% (370 out of 1199) of students received stop-guessing interventions. This evidence suggest that according to the *prima facie* active intervention triggering algorithms, students guess more in the *Assistments* system than they abuse help. While more students receive the stop-guessing intervention than the stop-hinting interventions, the average number of stop-guessing interventions per guesser (4, standard deviation of 4) is slightly less than the average number of stop-hinting interventions per hint-abuser (5, standard deviation of 6), weakly suggesting that perhaps the stop-guessing intervention is more effective. However, further analysis reveals that both interventions are for the most part rather ineffective.

Another method of analyzing the effectiveness the active interventions was to examine which types of actions (and the response times) that were executed both before and after each intervention was triggered. If after all the stop-guessing or stop-hinting interventions fired, they were immediately followed by quick attempts or hints, then they probably were not very effective. Figure 15 (a) charts the clustering of actions around the stop-hinting intervention, which is logged in our actions table as "Stop Hint Fired." Likewise, Figure 16 (a) charts the clustering of actions around the stop-guessing intervention, or "Stop Guess Fired." Figure 15 (a) shows the number of occurrences of each type of action at each step of an action sequence. At time zero, there are only "Stop Hint Fired" actions. The actions at time –5 show the distribution of actions 5 steps before the intervention was fired; time –4 shows the distribution of actions 4 steps before the intervention; and so on. Figure 15 (b) provides the data that generated this chart.

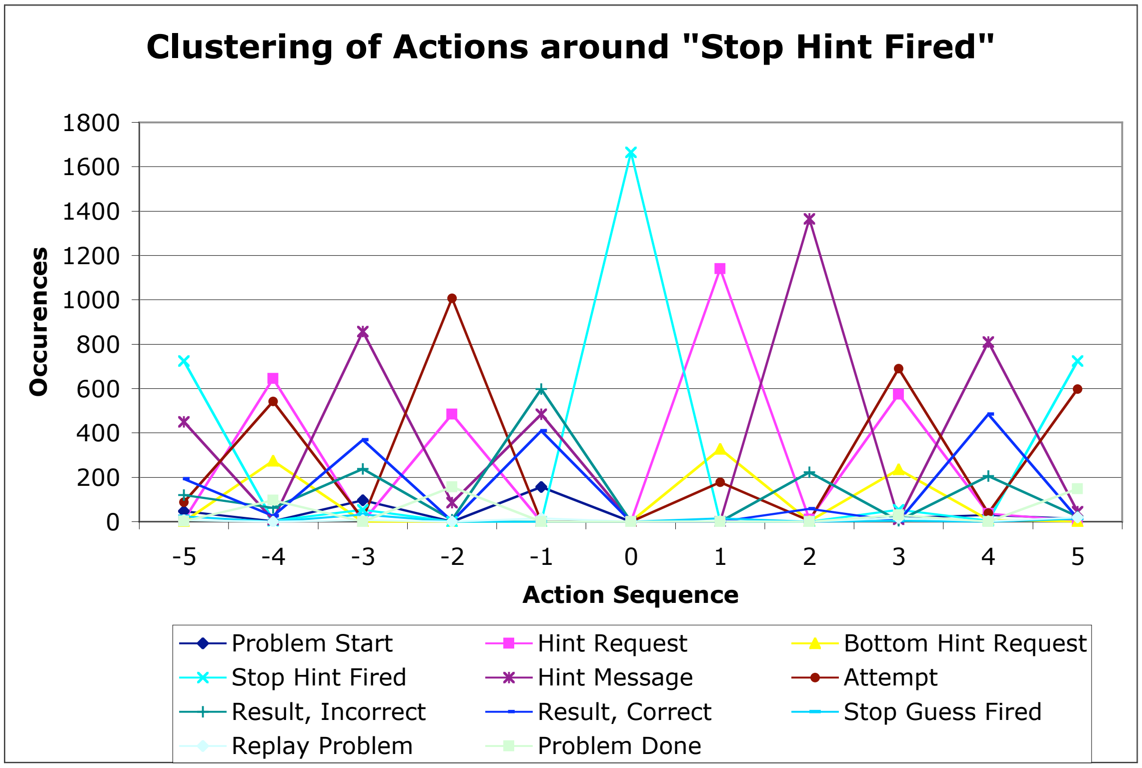


Figure 15 (a) – Clustering of Actions around "Stop Hint" Intervention

	Action Sequence										
Action	**-5**	**-4**	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**	**5**
Problem Start	47	3	96	1	156	0	0	0	17	29	12
Hint Request	7	646	0	484	0	0	1141	10	575	37	5
Bottom Hint Request	1	275	0	0	0	0	327	2	235	9	0
Stop Hint Fired	725	4	54	1	0	1664	0	1	54	4	725
Hint Message	450	8	857	85	484	0	0	1366	11	810	46
Attempt	88	542	12	1007	0	0	178	5	691	38	598
Result, Incorrect	119	62	238	9	598	0	0	223	4	206	24
Result, Correct	193	26	368	3	409	0	0	57	2	485	14
Stop Guess Fired	24	0	32	0	0	0	16	0	5	1	13
Replay Problem	7	1	6	0	14	0	2	0	17	2	19
Problem Done	3	96	0	156	0	0	0	0	29	1	148

Figure 15 (b) – "Stop Hint" Cluster Data

As we can see from Figures 15 (a) and 15 (b), about 88% of the time the stop-hinting intervention was received, it was immediately followed by a hint request or bottom-out hint request, which suggests that the intervention is largely being ignored. Furthermore, the intervention was immediately followed by an attempt approximately 10% of the time, with about a third of those attempts being correct. Another interesting observation is about 9% of all stop-hinting interventions were triggered at the start of a new problem before any attempt was even made. While this data suggests that the stop-hinting intervention was being triggered at reasonable times, it was largely ineffective at stopping hinting.

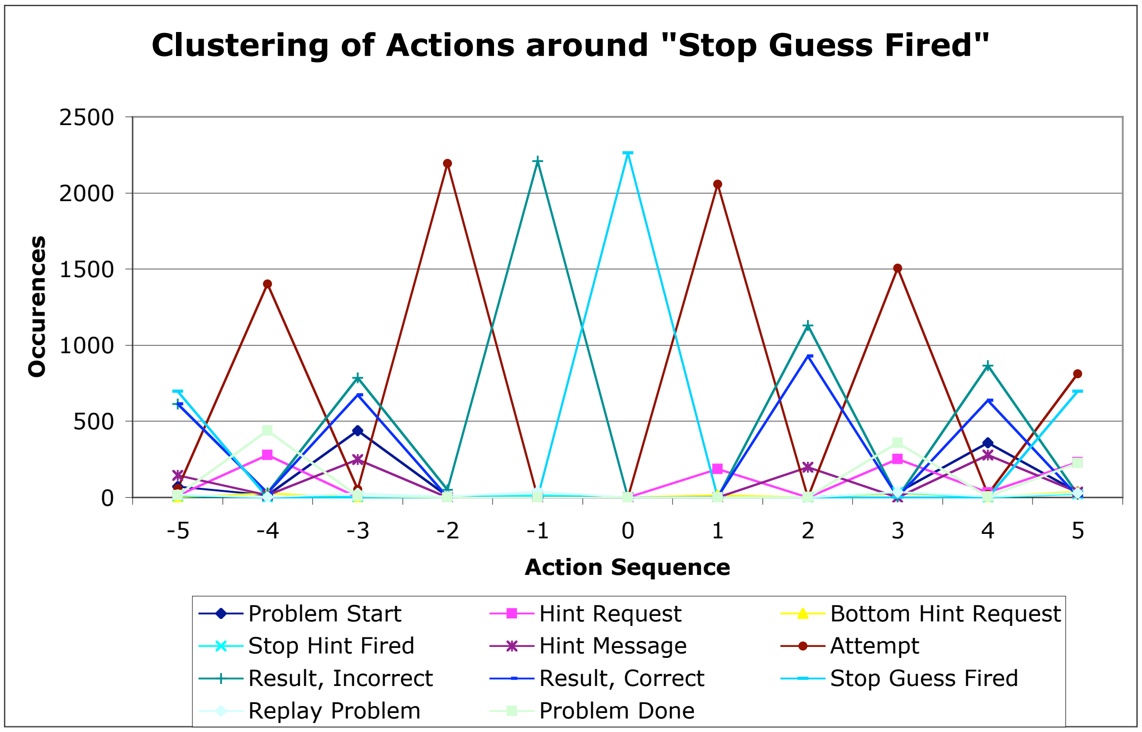


Figure 16 (a) – Clustering of Actions around "Stop Guess" Intervention

	Action Sequence										
Action	**-5**	**-4**	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**	**5**
Problem Start	71	15	440	10	0	0	0	0	32	360	36
Hint Request	12	280	0	0	0	0	186	0	251	34	231
Bottom Hint Request	3	26	0	0	0	0	14	0	27	2	39
Stop Hint Fired	13	1	5	0	16	0	0	0	32	0	24
Hint Message	144	14	250	0	0	0	0	199	0	278	36
Attempt	59	1403	52	2194	0	0	2058	0	1506	21	814
Result, Incorrect	613	29	787	49	2208	0	1	1131	0	867	22
Result, Correct	614	31	672	3	0	0	2	929	4	639	9
Stop Guess Fired	697	0	0	0	0	2262	0	0	0	0	697
Replay Problem	17	4	28	6	38	0	1	1	36	7	30
Problem Done	15	440	10	0	0	0	0	2	358	6	225

Figure 16 (b) – "Stop Guess" Cluster Data

Similar negative results were found for the effectiveness of the stop-guessing intervention. The chart in Figure 16 (a) and the data in Figure 16 (b) can be interpreted in the same manner as Figures 15 (a) and 15 (b), respectively. Approximately 98% of all stop-guessing interventions are fired after incorrect attempts, and the remaining 2% are fired after problems were replayed or a stop-hinting intervention was triggered. The stop-guessing intervention was immediately followed by hint requests about 8% of the time, and attempts the other 92% of the time. About 55% of those attempts were incorrect. This data strongly suggests that the stop-guessing intervention was completely ineffective, as it was being almost totally ignored.

Both of these active interventions are triggered by simple *prima facie* algorithms, which have long decay times. Meaning, that once the *prima facie* algorithm classifies a student as a gamer, a long stretch of non-gaming actions are required for the algorithm to consider an offending student as back on-task. This raises the possibility that these two active interventions are being ignored so often because they are being incorrectly triggered during the decay period after a student has reverted to on-task behavior. To evaluate these interventions without being biased by this possibility, another cluster is generated using only the first intervention received by any student. This should reveal whether the interventions are effective on first encounter. Figure 17 (a) charts the cluster of actions around first-time encounters of the stop-hinting intervention, while Figure 17 (b) contains the data for the chart. Interestingly, none of the first-time encounter stop-hinting interventions are preceded by repeated hint requests; rather they trigger immediately after a series of attempts. However, about 71% of these first-time interventions are immediately followed by hint requests (compared to the 88% overall). Figure 18 (a) charts the cluster of actions around first-time encounters of the stop-guessing intervention, while Figure 18 (b) contains the data for the chart. Over 96% of first-time stop-guessing interventions are triggered immediately after incorrect attempts. About 78% of these interventions are immediately followed by attempts (compared to the 92% overall). It is clear, that although they are relatively more effective upon first-encounter, these active interventions are for the most part ignored by students.

Overall, it seems that both of the active interventions were largely ineffective and were for the most part ignored by students. However, results of the toggle experiment (discussed in Section 4.2.3.4) suggest that these interventions might have some minor affect on users when combined with the passive intervention.

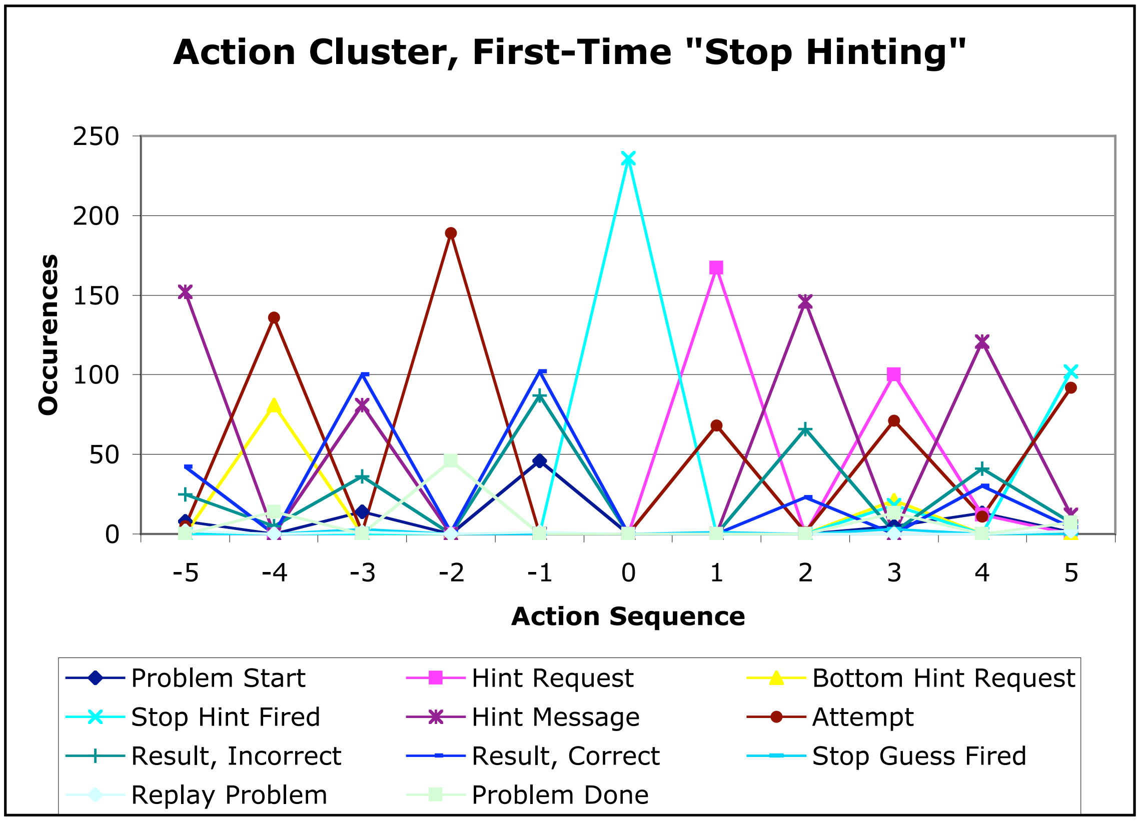


Figure 17 (a) – Cluster of Actions around "Stop Hinting" First-Time Encounters

	Action Sequence										
Action	-5	-4	-3	-2	-1	0	1	2	3	4	5
Problem Start	8	0	14	0	46	0	0	0	5	13	1
Hint Request	0	0	0	0	0	0	167	0	100	12	0
Bottom Hint Request	0	81	0	0	0	0	0	0	21	0	0
Stop Hint Fired	0	0	0	0	0	236	0	0	18	0	102
Hint Message	152	0	81	0	0	0	0	146	0	121	12
Attempt	5	136	1	189	0	0	68	1	71	11	92
Result, Incorrect	25	5	36	0	87	0	0	66	1	41	7
Result, Correct	42	0	100	1	102	0	0	23	0	30	4
Stop Guess Fired	1	0	3	0	0	0	1	0	3	0	0
Replay Problem	2	0	1	0	1	0	0	0	0	0	2
Problem Done	0	14	0	46	0	0	0	0	13	0	7

Figure 17 (b) – "Stop Hinting" First-Time Encounter Cluster Data

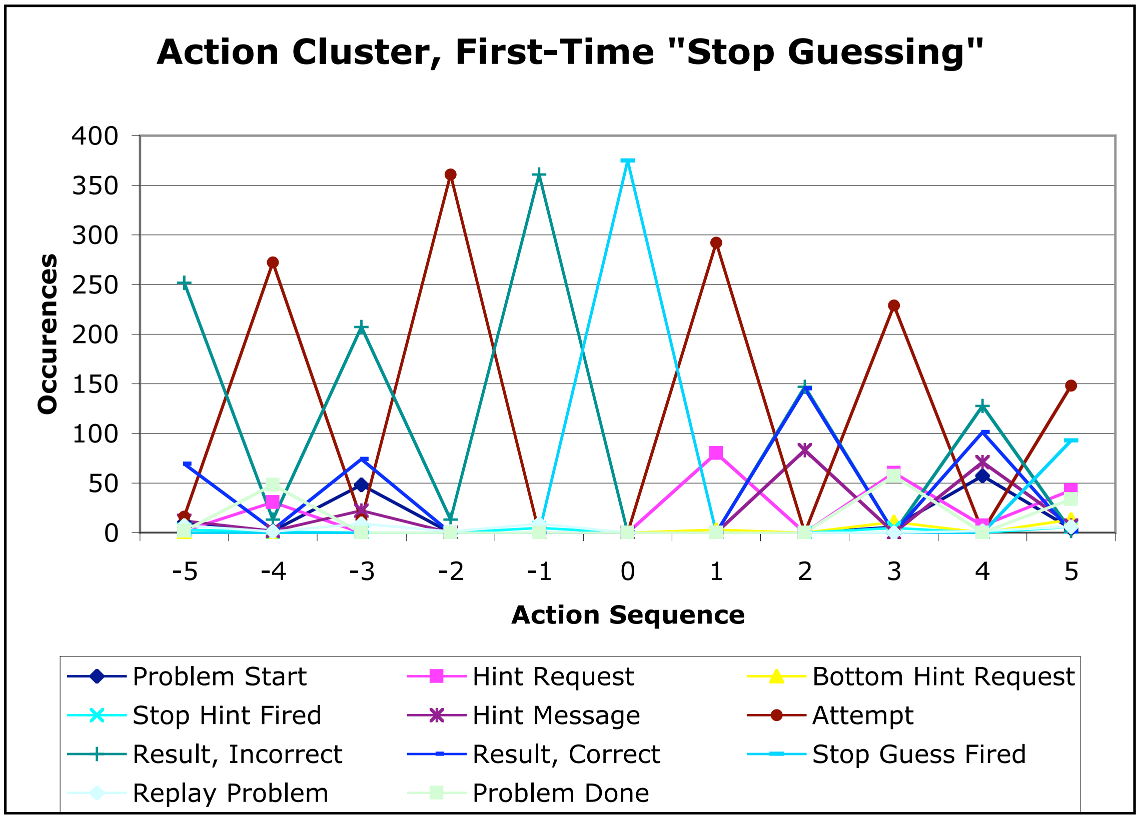


Figure 18 (a) – Cluster of Actions around "Stop Guessing" First-Time Encounters

	Action Sequence										
Action	-5	-4	-3	-2	-1	0	1	2	3	4	5
Problem Start	10	2	48	0	0	0	0	0	6	57	4
Hint Request	2	31	0	0	0	0	80	0	60	7	43
Bottom Hint Request	0	0	0	0	0	0	3	0	11	0	13
Stop Hint Fired	3	1	0	0	5	0	0	0	5	0	6
Hint Message	12	2	22	0	0	0	0	83	0	71	7
Attempt	16	272	13	361	0	0	292	0	229	1	148
Result, Incorrect	252	13	207	13	361	0	0	147	0	128	2
Result, Correct	69	3	74	0	0	0	0	145	0	101	1
Stop Guess Fired	0	0	0	0	0	375	0	0	0	0	93
Replay Problem	7	1	9	1	9	0	0	0	0	2	6
Problem Done	2	48	0	0	0	0	0	0	57	0	34

Figure 18 (b) – "Stop Guessing" First-Time Encounter Cluster Data

4.2.3.3 Passive Intervention

The effectiveness of the passive intervention is somewhat harder to evaluate, since there were no interesting recorded metrics available for latent response or human factors analysis, and there was no meaningful control group (students without the passive intervention) during the same school year. Two methods were used to evaluate the passive intervention: (1) anecdotal teacher surveys and (2) analysis of the "toggle experiment" – although it is hard to separate the effects of the passive and active interventions within the toggle experiment.

The teacher survey was used to gather anecdotal evaluation of the passive intervention and was also meant as a feedback mechanism for suggestions to improve the graphical chart. A similar survey aimed at students was never administered. The teacher survey had 5 questions that were rated on a scale of 1 to 5 (1=strong disagree, 2=disagree, 3=not sure, 4=agree, 5=strongly agree), and one open response question seeking suggestions and feedback. The survey was administered to only 6 teachers who had used the system in both the 2004-2005 and 2005-2006 academic school years. Clearly, 6 teachers is not a significant sample size, but it was a useful means to gather suggestions. Teacher's had several thoughtful suggestions including altering the background gradient scheme from white-to-black to another color gradient scheme over worries of racial bias issues, keeping the background one color at all times to eliminate confusing, altering the vertical heights of plotted points to indicate increasing depth of hints or a progression of incorrect or correct attempts, removing the lines altogether and simply having various colored vertical-bars indicating actions and results, and elimination of the automatic scaling of time for consistency across different student computers. A summary of the questions and responses appears below in Figure 19.

Question	Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
Do you think that the new graphical chart will aid teachers in assessing the progress and performance of their students?					6
Do you think that the new graphical chart will aid students in self-assessing their progress and performance?			1	3	2
Do you think that the new graphical chart will provide students with additional performance-based motivation?				3	3
Do you think that the new graphical chart will provide students with additional learning-based motivation?			1	2	3
Do you think that the new graphical chart will decrease off-task student behavior (talking, inactivity, excessive or unnecessary hinting or guessing)?			1	1	5

Figure 19 – Summary of Teacher Survey Results

Other than suggestions for alterations to the stylization of the graphical plot, most teachers expressed satisfaction with the passive intervention, at least as a feedback mechanism that would assist them in helping to select which students to focus their attention on. Additionally, this satisfaction or excitement with the graphical chart was one cause of many for the imbalance of the control groups, as teachers requested to have it "turned on" in the classes where it was not being displayed.

There are a few alternative methods of evaluating the passive intervention that could have been undertaken but were not. They are discussed briefly here for posterity. I could have calculated the correlations between having the passive intervention (using the 2005-2006 academic year data) or not (using similar data from the 2004-2005 academic year) and a student's number of correct and incorrect attempts, speed or pace of use, and hint requests – all of which are factors that might hypothetically be indicative of a student's motivation level; as well as correlations with learning rates. I could have also used the machine-learned model to compare the predicated rate of off-task gaming behavior with and without the passive intervention (as discussed in Section 4.1). Given the questionable results of the model, the *prima facie* method used in both the invocation of the active interventions and in the toggle experiment seems just as practical, albeit less sophisticated. The results of the toggle experiment are considered to be the final and overall method of evaluation for all the intervention mechanisms.

4.2.3.4 Overall Evaluation

The final evaluation for all the intervention mechanisms was originally planned to be the application of the classification model constructed in Section 4.1 as a predictor of the rates of gaming behavior between the control groups. However, given that the resulting suitability of the model for the task had some questionability, the toggle experiment described in Section 4.2.2 was ultimately the final evaluation method. The results of the toggle experiment are summarized in Figure 20.

The hint and guess scores used in Figure 20 were calculated using the same *prima facie* recognition algorithm that was used to invoke the active interventions. Although not as sophisticated as the machine-learned method and unable to identify gaming based on prior knowledge or other important factors, it does identify suspected gaming behavior based on hallmark usage patterns. It also has the advantage of not being susceptible to the same possible training bias (over-fitting of the data) of the machine-learned model, which may have come into play since the training data was from the 2004-2005 school year and the toggle experiment was from the 2005-2006 school year, which had different characteristics (discussed in Chapter 4.1).

Control Group	Metric	Before Toggle	After Toggle	Gaming Effect	Notes
both_none	Users	82	82	-2.82	This group originally had both the active and passive interventions turned ON, and then turned OFF after the toggling. Most kids had 20 minutes to an hour of use on days after toggling, with an average of about 30 minutes.
	Average Days	2.33	1.01		
	Avg Hint Score	2.13	0.72		
	Avg Guess Score	6.38	0.05		
	Avg Total Gaming Score	8.51	0.77		
	Avg Gaming Score Per Day	3.51	0.70		
none_both	Users	75	75	-5.20	This group originally had both the active and passive interventions turned OFF, and then turned ON after the toggling. Most kids had 20 minutes to an hour of use on days after toggling, with an average of about 30 minutes.
	Average Days	3.92	1.36		
	Avg Hint Score	9.19	1.49		
	Avg Guess Score	12.19	0.04		
	Avg Total Gaming Score	21.37	1.53		
	Avg Gaming Score Per Day	6.16	0.96		
passiveOnly	Users	18	18	-6.58	This group originally had only passive intervention turned ON, but had both active and passive turned OFF after the toggling. Most of the kids had less than 10 minutes of use on days after toggling.
	Average Days	1.33	1.00		
	Avg Hint Score	4.11	0.11		
	Avg Guess Score	4.50	0.00		
	Avg Total Gaming Score	8.61	0.11		
	Avg Gaming Score Per Day	6.69	0.11		

Figure 20 – Raw Results of the Toggle Experiment

Interestingly, before the settings were even toggled, the average guessing and hinting score for the group that had both interventions in place was less than half the scores for the group that had no interventions, suggesting that the interventions were perhaps having some sort of effect. However, in order to show that those differences were not the result of some sort of selection effect in the groups (for example, merely having more students in the first group who were less inclined to participate in off-task gaming behavior), the settings were toggled. The raw results of the toggle experiment indicate that the average gaming score per student decreased after toggling the active and passive intervention settings for all the control groups. However, examination of the data showed that a number of student sessions being considered were less than 10 minutes long. In order to be fair about the calculation of the gaming scores, it was decided to remove any sessions that were less than 20 minutes in length. The filtered results of the toggle experiment are shown in Figure 21.

Control Group	Metric	Before Toggle	After Toggle	Gaming Effect	Notes
both_none	Users	70	70	-2.82	This group originally had both the active and passive interventions turned ON, and then turned OFF after the toggling. Most kids had 20 minutes to an hour of use on days after toggling, with an average of about 30 minutes.
	Average Days	2.357	1.014		
	Avg Hint Score	2.257	0.829		
	Avg Guess Score	6.657	0.057		
	Avg Total Gaming Score	8.914	0.886		
	Avg Gaming Score Per Day	3.62	0.8		
none_both	Users	57	57	-4.45	This group originally had both the active and passive interventions turned OFF, and then turned ON after the toggling. Most kids had 20 minutes to an hour of use on days after toggling, with an average of about 30 minutes.
	Average Days	3.772	1.158		
	Avg Hint Score	8.912	1.754		
	Avg Guess Score	11.61	0.053		
	Avg Total Gaming Score	20.53	1.807		
	Avg Gaming Score Per Day	6.235	1.781		

Figure 21 – Filtered Results of the Toggle Experiment

Filtering out sessions that were under 20 minutes in length totally eliminated the third control group, since all the "after" sessions were only about 10 minutes long. However, the effect in the first two groups is mostly the same as the raw results. Before the toggling, the group that originally received both interventions had a total gaming score that was less than half their non-intervention receiving counterparts. After the toggling, both groups had decreased amounts of gaming. The average student decreased gaming by 3.5 points per day, after the toggling. Those who had been receiving the interventions, and then stopped receiving them, reduced gaming on average by 2.8 points per day. However, the decrease in the group that originally had no interventions and then began receiving them after the toggling, decreased their gaming by an average of 4.4 points per day. While this outcome might simply be the result of chance, it seems to suggest that the combination of the active and passive interventions has a moderately successful effect in the reduction of off-task gaming behavior.

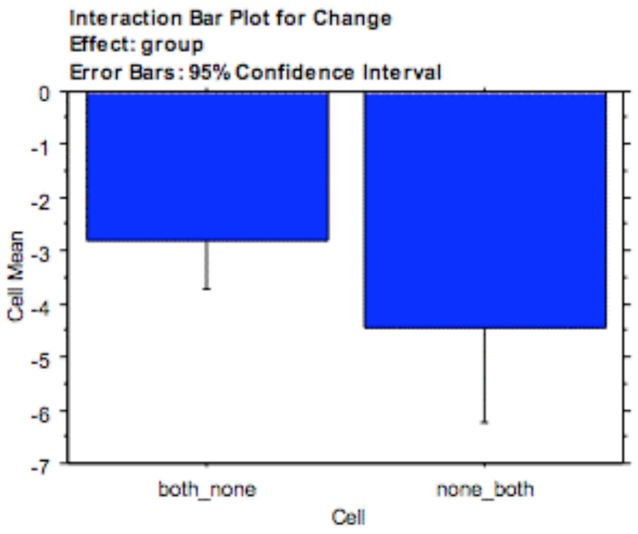
Using SAS statistical software, the statistical significance of these results was analyzed. One-side t-tests were performed on the "both_none" and "none_both" control groups, to see if the resulting change in gaming scores after the toggle was significantly different from zero, and in both cases the answer is YES ($p < 0.0001$, in both tests). The next question to test was whether there really was a bigger impact in the "none_both" group – turning the intervention mechanisms on versus off – and did so partially as a control for other effects such as if students learn to game more or less over time. This was done with an analysis of variance (ANOVA) and those results are contained in Figure 22.

ANOVA Table for Change

	DF	Sum of Squares	Mean Square	F-Value	P-Value	Lambda	Power
group	1	83.887	83.887	3.021	.0847	3.021	.390
Residual	125	3470.881	27.767				

Means Table for Change
Effect: group

	Count	Mean	Std. Dev.	Std. Err.
both_none	70	-2.820	3.705	.443
none_both	57	-4.454	6.713	.889



Fisher's PLSD for Change
Effect: group
Significance Level: 5 %

	Mean Diff.	Crit. Diff.	P-Value
both_none, none_both	1.634	1.861	.0847

Figure 22 – ANOVA of Gaming by Group

The resulting p-value of .08 suggests that turning the interventions on (none_both) makes a bigger impact on *prima facie* gaming than turning them off (both_none). One possible interpretation and explanation of these results would be that with the interventions turned on the students learn not to game, and then they do not start gaming once the interventions are turned off and go away. Further analysis might reveal whether the actual invocation or receiving of the active interventions (when they were turned on) is correlated with this decrease in gaming, as opposed to simply the *possibility* of receiving them (a student might have never seen the active interventions when they were turned on if they weren't gaming). Otherwise, we might be able to conclude that the decrease in gaming was due more to the passive intervention, or perhaps other factors.

4.3 Gaming within the Assistments System

The last portion of this thesis work dealt with the examination of the causes and the results of off-task gaming behavior within the *Assistments* System in general. This examination was undertaken by calculating how much students gamed using the *prima facie* algorithm, and then seeing if those numbers were correlated any survey or learning data.

4.3.1 Assistments System Survey Responses

A survey was administered at the end of the 2004-2005 academic year to students whose classes had been using the *Assistments* system throughout the year on a biweekly basis. Using the same *prima facie* gaming detection algorithms used for the invocation of our active interventions, and the analysis of the toggle experiment, gaming scores were calculated for the students who completed the entire survey. Once a score was generated for each student, they were classified as "Very High Gaming", "Above Average Gaming", "Below Average Gaming", or "Very Low Gaming" students. If a student's score was above the average gaming score then they were "Above Average Gaming". If a student's score was below the average gaming score then they were "Below Average Gaming." If a student's score was greater than the average gaming score plus the standard deviation, then they were classified as "Very High Gaming" students. If the student's score was less than the average gaming score minus the standard deviation, then they were classified as "Very Low Gaming" students. The full breakdown of the number of students in each category is shown below in Figure 23.

Category	Students	Percentage
Very High Gaming	53	0.14520548
Above Average Gaming	91	0.24931507
Below Average Gaming	179	0.49041096
Very Low Gaming	42	0.11506849
Total	365	1

Figure 23 – Classification of Survey Respondents

There were a total of 32 survey questions and the results were broken down by the gaming categorization of the respondents. Thirteen of the questions that had more or less the same distribution of responses regardless of gaming status, and are therefore considered uninteresting. The remaining questions revealed a number of interesting conclusions. According to the survey results grouped by gaming categorization:

- Students who gamed a lot were less likely to have a computer at home.
- Students who gamed were more likely to believe that they were not good at math.

- Students who gamed were less likely to believe they could do well at math if they worked hard.
- Students who gamed said they were less likely to do homework in math class.
- Students who gamed a lot said they were less likely to have trouble concentrating on the computer.
- Students who gamed a lot tended to strongly agree that the items were frustrating because they were too hard, while students who gamed very little were more likely to disagree. This is probably partially related to student prior knowledge.
- Students who gamed a lot agreed more often and more strenuously that they liked learning from a computer than those who gamed very little.
- The more students gamed, the less they said they liked math class.
- Students who tended not to game were more likely to say that they preferred using the *Assistments* system to doing homework. In a similar question, there were no differences between the groups when asked if they would prefer to use the tutor rather than take a test – they mostly all strongly agreed that they would.
- The less a student gamed, the more strongly they would prefer using the *Assistments* system to normal classroom activity.
- Students who gamed a lot had a slight tendency to say that they prefer facts and data to concepts and ideas more than other students.
- The more a student gamed, the more they thought that being told the answer was more helpful than reading the hints.
- The more a student gamed, the more they agreed that the hints aided in their understanding of similar problems.
- Students who gamed a lot were more likely to agree or strongly agree that they tried to get through difficult problems as quickly as possible.
- Students who gamed very little were more likely to strongly agree that they would seek help when they didn't understand something.
- Students who gamed a lot tended more than other students to say that their goal was to get through as many items as possible.
- The more students gamed, the more they tended to strongly agree that their goal was to learn new things.
- Students who gamed a lot were much more likely than other students to strongly agree that their parents thought it important for them to do well in math. This might explain the performance-based motivation behind some gaming behavior.
- The less a student gamed the more they were likely to strongly agree that they would use math in a job when they grew up.

The summarized breakdowns of responses for all these questions are contained in Appendix 4. Some of these results are interesting merely because they either corroborate or disagree with past findings. For example, Baker has reported that students who game do not like computers [13], while our survey suggests that those students who appeared to be heavily gaming *prima facie*, agreed more often and more strenuously that they liked learning from a computer than those who gamed very little. However, our survey results show that gamers were less likely to own a computer at home, and were more likely to dislike math class – another contradiction of previous findings.

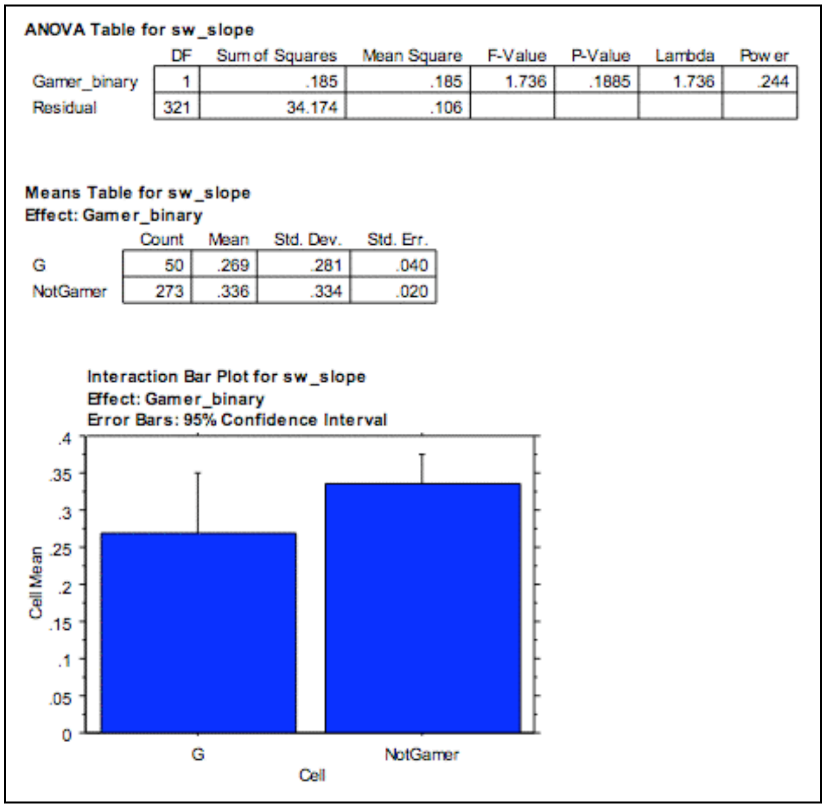
4.3.2 Gaming and Learning

Off-task gaming behavior has been correlated with substantially less learning in several prior studies [3]. In an attempt to validate those findings, learning rates that had been previously calculated for [14] using traditional methods as well as longitudinal data analysis, were grouped by the very high, above average, below average, and very low gaming categories (as described in Section 4.3.1). The results are summarized in Figure 24.

Category	Traditional Slope	Traditional Intercept	SW Slope	SW Intercept	Actual MCAS	Scaled Ans.
Very Low Gaming	1.68	26.92	0.34	24.04	35.17	0.76
Below Avg Gaming	1.62	19.53	0.33	19.31	30.56	0.8
Above Avg Gaming	1.29	15.07	0.33	14.23	24.24	0.7
Very High Gaming	0.95	11.99	0.26	11.71	19.66	0.77
OVERALL AVERAGE	1.44	17.88	0.32	17.25	27.69	0.76

Figure 24 – Learning Rates by Gaming Category

These results seem to corroborate previous findings that indicate that off-task gaming behavior is correlated with substantially less learning.



ANOVA Table for sw_slope

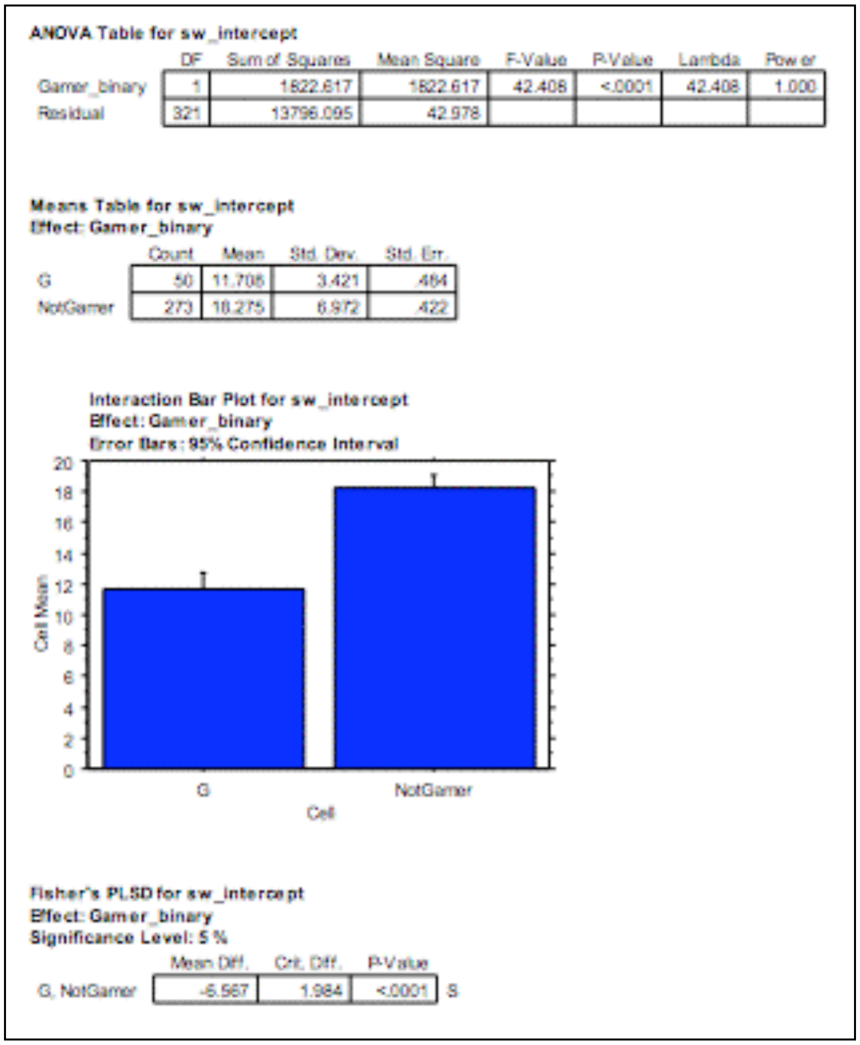
	DF	Sum of Squares	Mean Square	F-Value	P-Value	Lambda	Power
Gamer_binary	1	.185	.185	1.736	.1885	1.736	.244
Residual	321	34.174	.106				

Means Table for sw_slope
Effect: Gamer_binary

	Count	Mean	Std. Dev.	Std. Err.
G	50	.269	.281	.040
NotGamer	273	.336	.334	.020

Figure 25 – ANOVA of Learning by Gaming

However, a few statistical tests were run to examine the significance of these results. Before those tests were run, students were classified as being gamers or not. If a student's score put them at the level of "very high gaming" then they were a considered a gamer, and otherwise they were not. This was done to simplify the results and make them easier to interpret. The first test was an ANOVA of learning (SW slope) by gaming (p < 0.19). The complete results are contained in Figure 25, but they suggest that our learning rates are reasonably different than mere chance alone, and they show that off-task gaming behavior is correlated with less learning.



ANOVA Table for sw_intercept

	DF	Sum of Squares	Mean Square	F-Value	P-Value	Lambda	Power
Gamer_binary	1	1822.617	1822.617	42.408	<.0001	42.408	1.000
Residual	321	13796.095	42.978				

Means Table for sw_intercept
Effect: Gamer_binary

	Count	Mean	Std. Dev.	Std. Err.
G	50	11.708	3.421	.484
NotGamer	273	18.275	6.972	.422

Interaction Bar Plot for sw_intercept
Effect: Gamer_binary
Error Bars: 95% Confidence Interval
Cell Mean
0
2
4
6
8
10
12
14
16
18
20
G
NotGamer
Cell

Fisher's PLSD for sw_intercept
Effect: Gamer_binary
Significance Level: 5 %

	Mean Diff.	Crit. Diff.	P-Value	
G, NotGamer	-6.567	1.984	<.0001	S

Figure 26 – ANOVA of Knowledge by Gaming

The second test was an ANOVA of knowledge (SW intercept) by gaming (p < 0.0001). The complete results are contained in Figure 26, but this very strongly suggest that students who engage in off-task gaming behavior are more likely to come to the *Assistments* system with lower prior knowledge. One last test examined the correlation of gaming with a students actual MCAS score via Bartlett's Test of Sphericity, which very strongly showed that gamers do not perform well on the actual MCAS test (p < 0.0001).

5. Conclusions

Off-task gaming behavior is a major issue within the field of ITS, since it has been correlated with poor learning. The goal of this research was to explore this important phenomenon within the *Assistments* system. Two detection models were developed: a *prima facie* method that was used to invoke the active interventions, and a machine-learned decision-tree model. While the practicality of the machine-learned model was questionable, the resulting rules corroborated the connection of low prior knowledge and problem difficulty with gaming. Three dynamic interventions were designed: two active interventions for hints and guessing, and a novel passive intervention. The interventions were analyzed and evaluated with a variety of methods, but primarily with a toggling experiment with results suggesting that the combination of the active and passive interventions successfully reduces off-task gaming behavior more effectively than no intervention mechanisms. Despite a small sample size and minor suggestions for improvement, teachers were overwhelmingly positive in their reception of the passive graphical component, as much for its instant feedback and as a launching point for targeted instruction as for any other possible benefits. Finally, for further analysis of gaming and its effects specifically within the *Assistments* system, student surveys from 2004-2005 and student learning rates were grouped by gaming status. The student survey results provide some agreement and disagreement with previous studies about the nature of gaming, and the learning rates corroborated with previous findings that indicate that off-task gaming behavior is correlated with substantially less learning.

Acknowledgements

In addition to thanking my advisor, Neil Heffernan, I would like to thank my friends and colleagues at WPI and in the AI Lab for their friendship and support; Cathi Luperchio of Worcester Public Schools for her invaluable assistance; and Ryan Baker for his helpful comments and suggestions.

References

1. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). *Intelligent tutoring goes to school in the big city*. International Journal of Artificial Intelligence in Education, 8, 30-43

2. Baker, R.S., Corbett, A.T., Koedinger, K.R. (2004) *Detecting Student Misuse of Intelligent Tutoring Systems*. Proceedings of the 7th International Conference on Intelligent Tutoring Systems, 531-540.

3. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) *Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System"*. Proceedings of ACM CHI 2004: Computer-Human Interaction, 383-390.

4. Razzaq, L, Feng, M., Nuzzo-Jones, G., Heffernan, N.T. et. al (2005). *The Assistment Project: Blending Assessment and Assisting*. To appear in the Proceedings of the 12th Annual Conference on Artificial Intelligence in Education 2005, Amsterdam

5. Feng, M., Heffernan, N.T. (2005). *Informing Teachers Live about Student Learning: Reporting in the Assistment System.* Submitted to the 12th Annual Conference on Artificial Intelligence in Education 2005, Amsterdam

6. Personal communications with Feng, M., Heffernan, N.T.

7. Bentham, Jeremy. *The Panopticon Writings*. Ed. Miran Bozovic (London: Verso, 1995). (available at http://cartome.org/panopticon2.htm)

8. Nuzzo-Jones, G., Walonoski, J.A., Heffernan, N.T., Livak, T. (2005). *The eXtensible Tutor Architecture: A New Foundation for ITS*. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) Proceedings of the 12th Annual International Conference on Artificial Intelligence In Education, 555-562. Amsterdam: ISO Press.

9. Ian H. Witten and Eibe Frank (2005). "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

10. J. Cendrowska (1987). *PRISM: An algorithm for inducing modular rules*. International Journal of Man-Machine Studies. Vol.27, No.4, pp.349-370.

11. JR Quinlan. "C4. 5: Programs for Machine Learning." The Morgan Kaufmann Series in Machine Learning, San Mateo, 1993.

12. Macasek, Michael A. "Towards Teachers Quickly Creating Tutoring Systems" WPI Computer Science Master's Thesis (2005).

13. Baker, R.S., Roll, I., Corbett, A.T., Koedinger, K.R. (2005) *Do Performance Goals Lead Students to Game the System?* Proceedings of the 12th International Conference on Artificial Intelligence and Education (AIED2005), 57-64.

14. Feng, M., Heffernan, N.T, Koedinger, K.R., *Addressing the Testing Challenge with a Web-Based E-Assessment System that Tutors as it Assesses*, Submitted to WWW2006, Edinburgh, Scotland (2006).

Appendix 1 – J48 Decision Tree Rules

This appendix is not a comprehensive list of WEKA results, nor the complete output of experiments. Several of these runs used all the data for training purposes simply so the resulting rules could be examined for reasonability. They were not used for testing or evaluation purposes, and are only included here to show the rules that would be generated from the rule training data.

30 Second Window, 10-fold Cross-Validation, High Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     lastyear_30sec_gaming.arff
Test mode:    10-fold cross-validation

J48 pruned tree
------------------
CORRECT_ATTEMPTS*PMPKNOW_XLOW <= 0: FALSE (821.0/13.0)
CORRECT_ATTEMPTS*PMPKNOW_XLOW > 0
|   TOTAL_HINTS*TEXTBOX_PROBLEMS <= 6
|   |   HINT_TIME_SHORT*PMPKNOW <= 0.16: FALSE (21.0/1.0)
|   |   HINT_TIME_SHORT*PMPKNOW > 0.16
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS <= 1: TRUE (3.0)
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS > 1: FALSE (3.0)
|   TOTAL_HINTS*TEXTBOX_PROBLEMS > 6: TRUE (2.0)
```

2 Minute Window, 10-fold Cross-Validation, Moderate Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.5 -M 1
Relation:     lastyear_2min_gaming.arff
Test mode:    10-fold cross-validation

J48 pruned tree
------------------
CORRECT_ATTEMPTS*PROB_HINT_RATIO <= 5.4
|   TOTAL_HINTS*ATTEMPT_TIME_SHORT <= 4
|   |   INCORRECT_FIRST_ATTEMPTS*SDV_PROBLEM_DIFFICULTY <= 0.23
|   |   |   PMPKNOW*PMPKNOW_XLOW <= 0.21: FALSE (734.0/3.0)
|   |   |   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   |   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   |   |   ASSISTMENTS > 0: FALSE (24.0)
|   |   INCORRECT_FIRST_ATTEMPTS*SDV_PROBLEM_DIFFICULTY > 0.23
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0: FALSE (21.0/1.0)
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   INCORRECT_ATTEMPTS*MAX_PROBLEM_DIFFICULTY <= 2.13: TRUE (2.0)
|   |   |   |   INCORRECT_ATTEMPTS*MAX_PROBLEM_DIFFICULTY > 2.13: FALSE (5.0)
|   TOTAL_HINTS*ATTEMPT_TIME_SHORT > 4
|   |   TEXTBOX_PROBLEMS <= 0
|   |   |   HINT_TIME_LONG*PMPKNOW_HIGH <= 0
|   |   |   |   MIN_ATTEMPT_MS*PROB_INCORRECT_RATIO <= 2610.3: FALSE (7.0)
|   |   |   |   MIN_ATTEMPT_MS*PROB_INCORRECT_RATIO > 2610.3: TRUE (2.0)
|   |   |   HINT_TIME_LONG*PMPKNOW_HIGH > 0: TRUE (1.0)
|   |   TEXTBOX_PROBLEMS > 0: FALSE (26.0)
CORRECT_ATTEMPTS*PROB_HINT_RATIO > 5.4
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS <= 0
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO <= 0.37
|   |   |   PMPKNOW_HIGH <= 0: TRUE (5.0)
|   |   |   PMPKNOW_HIGH > 0: FALSE (1.0)
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO > 0.37: FALSE (17.0)
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS > 0: TRUE (3.0)
```

30 Second Window, LOOCV, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_30sec_gaming.arff
Test mode:    849-fold cross-validation

J48 pruned tree
------------------
CORRECT_ATTEMPTS*PMPKNOW_XLOW <= 0
|   CORRECT_FIRST_ATTEMPTS*PMPKNOW_LOW <= 0
|   |   TOTAL_HINTS*ATTEMPT_TIME_SHORT <= 0
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0
|   |   |   |   HINT_TIME_LONG*PMPKNOW_LOW <= 0: FALSE (691.0/6.0)
|   |   |   |   HINT_TIME_LONG*PMPKNOW_LOW > 0
|   |   |   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY <= 0.28: FALSE (27.0)
|   |   |   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY > 0.28
|   |   |   |   |   |   ACTIONS <= 5: TRUE (1.0)
|   |   |   |   |   |   ACTIONS > 5: FALSE (1.0)
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   ACTIONS*SUM_ACTION_TIME_MS <= 85900
|   |   |   |   |   PMPKNOW <= 0.51: TRUE (2.0)
|   |   |   |   |   PMPKNOW > 0.51: FALSE (2.0)
|   |   |   |   ACTIONS*SUM_ACTION_TIME_MS > 85900: FALSE (39.0)
|   |   TOTAL_HINTS*ATTEMPT_TIME_SHORT > 0
|   |   |   CORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS <= 0
|   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO <= 0.5: FALSE (26.0)
|   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO > 0.5
|   |   |   |   |   SUM_ACTION_TIME_MS <= 16210: FALSE (1.0)
|   |   |   |   |   SUM_ACTION_TIME_MS > 16210: TRUE (1.0)
|   |   |   CORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS > 0
|   |   |   |   ACTIONS <= 5: FALSE (1.0)
|   |   |   |   ACTIONS > 5: TRUE (1.0)
|   CORRECT_FIRST_ATTEMPTS*PMPKNOW_LOW > 0
|   |   INCORRECT_ATTEMPTS*SDV_ATTEMPT_MS <= 3147
|   |   |   ATTEMPT_TIME_LONG*TEXTBOX_PROBLEMS <= 0: FALSE (21.0)
|   |   |   ATTEMPT_TIME_LONG*TEXTBOX_PROBLEMS > 0
|   |   |   |   MIN_PROBLEM_DIFFICULTY <= 0.46: TRUE (1.0)
|   |   |   |   MIN_PROBLEM_DIFFICULTY > 0.46: FALSE (4.0)
|   |   INCORRECT_ATTEMPTS*SDV_ATTEMPT_MS > 3147
|   |   |   BOTTOM_HINTS <= 0: TRUE (1.0)
|   |   |   BOTTOM_HINTS > 0: FALSE (1.0)
CORRECT_ATTEMPTS*PMPKNOW_XLOW > 0
|   TOTAL_HINTS*TEXTBOX_PROBLEMS <= 6
|   |   HINT_TIME_SHORT*PMPKNOW <= 0.16
|   |   |   REPLAYS <= 0: FALSE (19.0)
|   |   |   REPLAYS > 0
|   |   |   |   TOTAL_HINTS <= 0: TRUE (1.0)
|   |   |   |   TOTAL_HINTS > 0: FALSE (1.0)
|   |   HINT_TIME_SHORT*PMPKNOW > 0.16
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS <= 1: TRUE (3.0)
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS > 1: FALSE (3.0)
|   TOTAL_HINTS*TEXTBOX_PROBLEMS > 6: TRUE (2.0)
```

1 Minute Window, 10-fold Cross-Validation, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_1min_gaming.arff
]
Test mode:    10-fold cross-validation

J48 pruned tree
------------------
SDV_HINT_MS*PMPKNOW_XLOW <= 525
|   PROBLEMS*PMPKNOW_LOW <= 1
|   |   SDV_PROBLEM_DIFFICULTY*PROB_HINT_RATIO <= 0
|   |   |   INCORRECT_FIRST_ATTEMPTS*TEXTBOX_PROBLEMS <= 0
|   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_XLONG <= 0
|   |   |   |   |   PMPKNOW*PMPKNOW_XLOW <= 0.17
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT <= 0: FALSE
(610.0/2.0)
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT > 0
|   |   |   |   |   |   |   REPLAYS*MIN_ATTEMPT_MS <= 10710: FALSE (26.0)
|   |   |   |   |   |   |   REPLAYS*MIN_ATTEMPT_MS > 10710
|   |   |   |   |   |   |   |   ACTIONS <= 4: TRUE (1.0)
|   |   |   |   |   |   |   |   ACTIONS > 4: FALSE (1.0)
|   |   |   |   |   PMPKNOW*PMPKNOW_XLOW > 0.17
|   |   |   |   |   |   ASSISTMENTS <= 0
|   |   |   |   |   |   |   PMPKNOW*PMPKNOW <= 0.05: FALSE (5.0)
|   |   |   |   |   |   |   PMPKNOW*PMPKNOW > 0.05
|   |   |   |   |   |   |   |   PMPKNOW <= 0.24: TRUE (2.0)
|   |   |   |   |   |   |   |   PMPKNOW > 0.24: FALSE (4.0)
|   |   |   |   |   |   ASSISTMENTS > 0: FALSE (19.0)
|   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_XLONG > 0
|   |   |   |   |   SDV_ATTEMPT_MS*SUM_SDV_PROB_HINT_MS <= 39019968: FALSE (25.0)
|   |   |   |   |   SDV_ATTEMPT_MS*SUM_SDV_PROB_HINT_MS > 39019968
|   |   |   |   |   |   ACTIONS <= 5: TRUE (1.0)
|   |   |   |   |   |   ACTIONS > 5: FALSE (2.0)
|   |   |   INCORRECT_FIRST_ATTEMPTS*TEXTBOX_PROBLEMS > 0
|   |   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0: FALSE (16.0)
|   |   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   |   SUM_ACTION_TIME_MS <= 44330: TRUE (1.0)
|   |   |   |   |   SUM_ACTION_TIME_MS > 44330: FALSE (10.0)
|   |   SDV_PROBLEM_DIFFICULTY*PROB_HINT_RATIO > 0
|   |   |   HINT_TIME_XLONG <= 0: FALSE (24.0)
|   |   |   HINT_TIME_XLONG > 0
|   |   |   |   ACTIONS <= 7: TRUE (1.0)
|   |   |   |   ACTIONS > 7: FALSE (1.0)
|   PROBLEMS*PMPKNOW_LOW > 1
|   |   MULTIPLE_CHOICE_PROBLEMS*PROB_HINT_RATIO <= 1.5
|   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY <= 0.08
|   |   |   |   CORRECT_ATTEMPTS <= 1: FALSE (2.0)
|   |   |   |   CORRECT_ATTEMPTS > 1: TRUE (2.0)
|   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY > 0.08
|   |   |   |   HINT_TIME_SHORT*PROB_ATTEMPT_RATIO <= 1.33: FALSE (64.0)
|   |   |   |   HINT_TIME_SHORT*PROB_ATTEMPT_RATIO > 1.33
|   |   |   |   |   TOTAL_HINTS*ASSISTMENTS <= 1: TRUE (1.0)
|   |   |   |   |   TOTAL_HINTS*ASSISTMENTS > 1: FALSE (3.0)
|   |   MULTIPLE_CHOICE_PROBLEMS*PROB_HINT_RATIO > 1.5
|   |   |   PROB_CORRECT_RATIO <= 0.33: FALSE (2.0)
|   |   |   PROB_CORRECT_RATIO > 0.33: TRUE (2.0)
SDV_HINT_MS*PMPKNOW_XLOW > 525
|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY <= 0.59: TRUE (3.0)
|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY > 0.59
|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY <= 11.84: FALSE (18.0)
|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY > 11.84
|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS <= 4: TRUE (3.0)
|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS > 4: FALSE (1.0)
```

6 Minute Window, 10-fold Cross-Validation, Moderate Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.5 -M 1
Relation:     lastyear_6min_gaming.arff
Test mode:    10-fold cross-validation

J48 pruned tree
------------------
TOTAL_HINTS <= 18
|   PMPKNOW*PMPKNOW_XLOW <= 0.21
|   |   ATTEMPT_TIME_LONG*PROBLEMS <= 4: FALSE (666.0/3.0)
|   |   ATTEMPT_TIME_LONG*PROBLEMS > 4
|   |   |   SDV_HINT_MS*SDV_PROBLEM_DIFFICULTY <= 770.9: FALSE (92.0)
|   |   |   SDV_HINT_MS*SDV_PROBLEM_DIFFICULTY > 770.9
|   |   |   |   ACTIONS*AVG_HINT_MS <= 280633: TRUE (4.0)
|   |   |   |   ACTIONS*AVG_HINT_MS > 280633: FALSE (31.0/1.0)
|   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   ASSISTMENTS > 0: FALSE (29.0/1.0)
TOTAL_HINTS > 18
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO <= 0.47
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS <= 5937981
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG <= 2: TRUE (3.0)
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG > 2: FALSE (1.0)
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS > 5937981: FALSE (17.0)
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO > 0.47: TRUE (5.0)
```

4 Minute Window, 10-fold Cross-Validation, Moderate Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.5 -M 1
Relation:     lastyear_4min_gaming.arff
Test mode:    10-fold cross-validation

J48 pruned tree
------------------
TOTAL_HINTS*PMPKNOW_LOW <= 10
|   SDV_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY <= 0.05
|   |   PMPKNOW*PMPKNOW_XLOW <= 0.21: FALSE (753.0/6.0)
|   |   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   |   ASSISTMENTS > 0: FALSE (28.0)
|   SDV_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY > 0.05
|   |   SUM_SDV_PROB_HINT_MS*PMPKNOW_LOW <= 141790: FALSE (35.0/1.0)
|   |   SUM_SDV_PROB_HINT_MS*PMPKNOW_LOW > 141790: TRUE (2.0)
TOTAL_HINTS*PMPKNOW_LOW > 10
|   REPLAYS*PMPKNOW_XLOW <= 0
|   |   INCORRECT_FIRST_ATTEMPTS*MIN_HINT_MS <= 5550
|   |   |   PMPKNOW*SDV_PROBLEM_DIFFICULTY <= 0.05: FALSE (20.0)
|   |   |   PMPKNOW*SDV_PROBLEM_DIFFICULTY > 0.05
|   |   |   |   PROB_ATTEMPT_RATIO <= 0.9: FALSE (2.0)
|   |   |   |   PROB_ATTEMPT_RATIO > 0.9: TRUE (2.0)
|   |   INCORRECT_FIRST_ATTEMPTS*MIN_HINT_MS > 5550: TRUE (2.0)
|   REPLAYS*PMPKNOW_XLOW > 0: TRUE (4.0)
```

6 Minute Window, Trained with all Data, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_6min_gaming.arff

J48 pruned tree
------------------
TOTAL_HINTS <= 18
|   PMPKNOW*PMPKNOW_XLOW <= 0.21
|   |   ATTEMPT_TIME_LONG*PROBLEMS <= 4
|   |   |   CORRECT_FIRST_ATTEMPTS*PMPKNOW_HIGH <= 0: FALSE (522.0)
|   |   |   CORRECT_FIRST_ATTEMPTS*PMPKNOW_HIGH > 0
|   |   |   |   ASSISTMENTS <= 2
|   |   |   |   |   ATTEMPT_TIME_SHORT*MULTIPLE_CHOICE_PROBLEMS <= 0
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS <= 2: FALSE (31.0)
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS > 2
|   |   |   |   |   |   |   TOTAL_ATTEMPTS <= 5: FALSE (1.0)
|   |   |   |   |   |   |   TOTAL_ATTEMPTS > 5: TRUE (1.0)
|   |   |   |   |   ATTEMPT_TIME_SHORT*MULTIPLE_CHOICE_PROBLEMS > 0
|   |   |   |   |   |   TOTAL_HINTS*TEXTBOX_PROBLEMS <= 0
|   |   |   |   |   |   |   ACTIONS <= 10: FALSE (1.0)
|   |   |   |   |   |   |   ACTIONS > 10: TRUE (2.0)
|   |   |   |   |   |   TOTAL_HINTS*TEXTBOX_PROBLEMS > 0: FALSE (6.0)
|   |   |   |   ASSISTMENTS > 2: FALSE (102.0)
|   |   ATTEMPT_TIME_LONG*PROBLEMS > 4
|   |   |   SDV_HINT_MS*SDV_PROBLEM_DIFFICULTY <= 770.9: FALSE (92.0)
|   |   |   SDV_HINT_MS*SDV_PROBLEM_DIFFICULTY > 770.9
|   |   |   |   ACTIONS*AVG_HINT_MS <= 280633: TRUE (4.0)
|   |   |   |   ACTIONS*AVG_HINT_MS > 280633
|   |   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO <= 0.67: FALSE (29.0)
|   |   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO > 0.67
|   |   |   |   |   |   ACTIONS <= 34: FALSE (1.0)
|   |   |   |   |   |   ACTIONS > 34: TRUE (1.0)
|   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   ASSISTMENTS > 0
|   |   |   CORRECT_FIRST_ATTEMPTS*REPLAYS <= 1: FALSE (27.0)
|   |   |   CORRECT_FIRST_ATTEMPTS*REPLAYS > 1
|   |   |   |   ACTIONS <= 58: FALSE (1.0)
|   |   |   |   ACTIONS > 58: TRUE (1.0)
TOTAL_HINTS > 18
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO <= 0.47
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS <= 5937981
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG <= 2: TRUE (3.0)
|   |   |   BOTTOM_HINTS*HINT_TIME_LONG > 2: FALSE (1.0)
|   |   TOTAL_HINTS*SUM_AVG_PROB_ATTEMPT_MS > 5937981: FALSE (17.0)
|   PMPKNOW_XLOW*PROB_BOTTOM_HINT_RATIO > 0.47: TRUE (5.0)
```

4 Minute Window, Trained with all Data, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_4min_gaming.arff

J48 pruned tree
------------------
TOTAL_HINTS*PMPKNOW_LOW <= 10
|   CORRECT_ATTEMPTS*HINT_TIME_XLONG <= 1
|   |   SDV_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY <= 0.05
|   |   |   CORRECT_FIRST_ATTEMPTS*PMPKNOW_HIGH <= 2
|   |   |   |   PMPKNOW*PMPKNOW_XLOW <= 0.21
|   |   |   |   |   INCORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT <= 0: FALSE
(575.0)
|   |   |   |   |   INCORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT > 0
|   |   |   |   |   |   MIN_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY <= 0.02
|   |   |   |   |   |   |   PMPKNOW_LOW*SDV_PROBLEM_DIFFICULTY <= 0.01
|   |   |   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*SDV_ATTEMPT_MS <= 24269:
FALSE (13.0)
|   |   |   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*SDV_ATTEMPT_MS > 24269:
TRUE (1.0)
|   |   |   |   |   |   |   PMPKNOW_LOW*SDV_PROBLEM_DIFFICULTY > 0.01: TRUE (1.0)
|   |   |   |   |   |   MIN_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY > 0.02:
FALSE (96.0)
|   |   |   |   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   |   |   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   |   |   |   ASSISTMENTS > 0: FALSE (27.0)
|   |   |   CORRECT_FIRST_ATTEMPTS*PMPKNOW_HIGH > 2
|   |   |   |   INCORRECT_ATTEMPTS*ATTEMPT_TIME_XLONG <= 0
|   |   |   |   |   REPLAYS*PROBLEMS <= 10: FALSE (37.0)
|   |   |   |   |   REPLAYS*PROBLEMS > 10
|   |   |   |   |   |   ACTIONS <= 22: TRUE (1.0)
|   |   |   |   |   |   ACTIONS > 22: FALSE (1.0)
|   |   |   |   INCORRECT_ATTEMPTS*ATTEMPT_TIME_XLONG > 0
|   |   |   |   |   INCORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS <= 1: TRUE (1.0)
|   |   |   |   |   INCORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS > 1: FALSE (2.0)
|   |   SDV_PROBLEM_DIFFICULTY*SDV_PROBLEM_DIFFICULTY > 0.05
|   |   |   SUM_SDV_PROB_HINT_MS*PMPKNOW_LOW <= 141790
|   |   |   |   INCORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT <= 2: FALSE (31.0)
|   |   |   |   INCORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT > 2
|   |   |   |   |   ACTIONS <= 12: TRUE (1.0)
|   |   |   |   |   ACTIONS > 12: FALSE (1.0)
|   |   |   SUM_SDV_PROB_HINT_MS*PMPKNOW_LOW > 141790: TRUE (2.0)
|   CORRECT_ATTEMPTS*HINT_TIME_XLONG > 1
|   |   ATTEMPT_TIME_LONG*MULTIPLE_CHOICE_PROBLEMS <= 1
|   |   |   TOTAL_HINTS*PMPKNOW_HIGH <= 4: FALSE (24.0)
|   |   |   TOTAL_HINTS*PMPKNOW_HIGH > 4
|   |   |   |   ACTIONS <= 22: FALSE (1.0)
|   |   |   |   ACTIONS > 22: TRUE (1.0)
|   |   ATTEMPT_TIME_LONG*MULTIPLE_CHOICE_PROBLEMS > 1
|   |   |   ACTIONS <= 17: TRUE (1.0)
|   |   |   ACTIONS > 17: FALSE (1.0)
TOTAL_HINTS*PMPKNOW_LOW > 10
|   REPLAYS*PMPKNOW_XLOW <= 0
|   |   INCORRECT_FIRST_ATTEMPTS*MIN_HINT_MS <= 5550
|   |   |   PMPKNOW*SDV_PROBLEM_DIFFICULTY <= 0.05: FALSE (20.0)
|   |   |   PMPKNOW*SDV_PROBLEM_DIFFICULTY > 0.05
|   |   |   |   PROB_ATTEMPT_RATIO <= 0.9: FALSE (2.0)
|   |   |   |   PROB_ATTEMPT_RATIO > 0.9: TRUE (2.0)
|   |   INCORRECT_FIRST_ATTEMPTS*MIN_HINT_MS > 5550: TRUE (2.0)
|   REPLAYS*PMPKNOW_XLOW > 0: TRUE (4.0)
```

2 Minute Window, Trained with all Data, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_2min_gaming.arff

J48 pruned tree
------------------
CORRECT_ATTEMPTS*PROB_HINT_RATIO <= 5.4
|   TOTAL_HINTS*ATTEMPT_TIME_SHORT <= 4
|   |   INCORRECT_FIRST_ATTEMPTS*SDV_PROBLEM_DIFFICULTY <= 0.23
|   |   |   PMPKNOW*PMPKNOW_XLOW <= 0.21
|   |   |   |   CORRECT_ATTEMPTS*ATTEMPT_TIME_XLONG <= 0: FALSE (651.0/1.0)
|   |   |   |   CORRECT_ATTEMPTS*ATTEMPT_TIME_XLONG > 0
|   |   |   |   |   ACTIONS*SUM_ACTION_TIME_MS <= 282720
|   |   |   |   |   |   SDV_ATTEMPT_MS*PMPKNOW_HIGH <= 3571
|   |   |   |   |   |   |   MULTIPLE_CHOICE_PROBLEMS*PMPKNOW_LOW <= 0: FALSE (6.0)
|   |   |   |   |   |   |   MULTIPLE_CHOICE_PROBLEMS*PMPKNOW_LOW > 0: TRUE (1.0)
|   |   |   |   |   |   SDV_ATTEMPT_MS*PMPKNOW_HIGH > 3571: TRUE (1.0)
|   |   |   |   |   ACTIONS*SUM_ACTION_TIME_MS > 282720: FALSE (75.0)
|   |   |   PMPKNOW*PMPKNOW_XLOW > 0.21
|   |   |   |   ASSISTMENTS <= 0: TRUE (2.0)
|   |   |   |   ASSISTMENTS > 0: FALSE (24.0)
|   |   INCORRECT_FIRST_ATTEMPTS*SDV_PROBLEM_DIFFICULTY > 0.23
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0
|   |   |   |   PROBLEMS <= 4: FALSE (19.0)
|   |   |   |   PROBLEMS > 4
|   |   |   |   |   ACTIONS <= 15: TRUE (1.0)
|   |   |   |   |   ACTIONS > 15: FALSE (1.0)
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   INCORRECT_ATTEMPTS*MAX_PROBLEM_DIFFICULTY <= 2.13: TRUE (2.0)
|   |   |   |   INCORRECT_ATTEMPTS*MAX_PROBLEM_DIFFICULTY > 2.13: FALSE (5.0)
|   TOTAL_HINTS*ATTEMPT_TIME_SHORT > 4
|   |   TEXTBOX_PROBLEMS <= 0
|   |   |   HINT_TIME_LONG*PMPKNOW_HIGH <= 0
|   |   |   |   MIN_ATTEMPT_MS*PROB_INCORRECT_RATIO <= 2610.3: FALSE (7.0)
|   |   |   |   MIN_ATTEMPT_MS*PROB_INCORRECT_RATIO > 2610.3: TRUE (2.0)
|   |   |   HINT_TIME_LONG*PMPKNOW_HIGH > 0: TRUE (1.0)
|   |   TEXTBOX_PROBLEMS > 0: FALSE (26.0)
CORRECT_ATTEMPTS*PROB_HINT_RATIO > 5.4
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS <= 0
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO <= 0.37
|   |   |   PMPKNOW_HIGH <= 0: TRUE (5.0)
|   |   |   PMPKNOW_HIGH > 0: FALSE (1.0)
|   |   AVG_PROBLEM_DIFFICULTY*PROB_CORRECT_RATIO > 0.37: FALSE (17.0)
|   INCORRECT_FIRST_ATTEMPTS*REPLAYS > 0: TRUE (3.0)
```

1 Minute Window, Trained with all Data, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_1min_gaming.arff

J48 pruned tree
------------------
SDV_HINT_MS*PMPKNOW_XLOW <= 525
|   PROBLEMS*PMPKNOW_LOW <= 1
|   |   SDV_PROBLEM_DIFFICULTY*PROB_HINT_RATIO <= 0
|   |   |   INCORRECT_FIRST_ATTEMPTS*TEXTBOX_PROBLEMS <= 0
|   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_XLONG <= 0
|   |   |   |   |   PMPKNOW*PMPKNOW_XLOW <= 0.17
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT <= 0: FALSE
(610.0/2.0)
|   |   |   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_SHORT > 0
|   |   |   |   |   |   |   REPLAYS*MIN_ATTEMPT_MS <= 10710: FALSE (26.0)
|   |   |   |   |   |   |   REPLAYS*MIN_ATTEMPT_MS > 10710
|   |   |   |   |   |   |   |   ACTIONS <= 4: TRUE (1.0)
|   |   |   |   |   |   |   |   ACTIONS > 4: FALSE (1.0)
|   |   |   |   |   PMPKNOW*PMPKNOW_XLOW > 0.17
|   |   |   |   |   |   ASSISTMENTS <= 0
|   |   |   |   |   |   |   PMPKNOW*PMPKNOW <= 0.05: FALSE (5.0)
|   |   |   |   |   |   |   PMPKNOW*PMPKNOW > 0.05
|   |   |   |   |   |   |   |   PMPKNOW <= 0.24: TRUE (2.0)
|   |   |   |   |   |   |   |   PMPKNOW > 0.24: FALSE (4.0)
|   |   |   |   |   |   ASSISTMENTS > 0: FALSE (19.0)
|   |   |   |   CORRECT_FIRST_ATTEMPTS*ATTEMPT_TIME_XLONG > 0
|   |   |   |   |   SDV_ATTEMPT_MS*SUM_SDV_PROB_HINT_MS <= 39019968: FALSE (25.0)
|   |   |   |   |   SDV_ATTEMPT_MS*SUM_SDV_PROB_HINT_MS > 39019968
|   |   |   |   |   |   ACTIONS <= 5: TRUE (1.0)
|   |   |   |   |   |   ACTIONS > 5: FALSE (2.0)
|   |   |   INCORRECT_FIRST_ATTEMPTS*TEXTBOX_PROBLEMS > 0
|   |   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0: FALSE (16.0)
|   |   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   |   SUM_ACTION_TIME_MS <= 44330: TRUE (1.0)
|   |   |   |   |   SUM_ACTION_TIME_MS > 44330: FALSE (10.0)
|   |   SDV_PROBLEM_DIFFICULTY*PROB_HINT_RATIO > 0
|   |   |   HINT_TIME_XLONG <= 0: FALSE (24.0)
|   |   |   HINT_TIME_XLONG > 0
|   |   |   |   ACTIONS <= 7: TRUE (1.0)
|   |   |   |   ACTIONS > 7: FALSE (1.0)
|   PROBLEMS*PMPKNOW_LOW > 1
|   |   MULTIPLE_CHOICE_PROBLEMS*PROB_HINT_RATIO <= 1.5
|   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY <= 0.08
|   |   |   |   CORRECT_ATTEMPTS <= 1: FALSE (2.0)
|   |   |   |   CORRECT_ATTEMPTS > 1: TRUE (2.0)
|   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY > 0.08
|   |   |   |   HINT_TIME_SHORT*PROB_ATTEMPT_RATIO <= 1.33: FALSE (64.0)
|   |   |   |   HINT_TIME_SHORT*PROB_ATTEMPT_RATIO > 1.33
|   |   |   |   |   TOTAL_HINTS*ASSISTMENTS <= 1: TRUE (1.0)
|   |   |   |   |   TOTAL_HINTS*ASSISTMENTS > 1: FALSE (3.0)
|   |   MULTIPLE_CHOICE_PROBLEMS*PROB_HINT_RATIO > 1.5
|   |   |   PROB_CORRECT_RATIO <= 0.33: FALSE (2.0)
|   |   |   PROB_CORRECT_RATIO > 0.33: TRUE (2.0)
SDV_HINT_MS*PMPKNOW_XLOW > 525
|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY <= 0.59: TRUE (3.0)
|   ASSISTMENTS*AVG_PROBLEM_DIFFICULTY > 0.59
|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY <= 11.84: FALSE (18.0)
|   |   ACTIONS*MAX_PROBLEM_DIFFICULTY > 11.84
|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS <= 4: TRUE (3.0)
|   |   |   CORRECT_ATTEMPTS*TEXTBOX_PROBLEMS > 4: FALSE (1.0)
```

30 Second Window, Trained with all Data, Low Pruning

```
Scheme:       weka.classifiers.trees.J48 -C 0.75 -M 1
Relation:     lastyear_30sec_gaming.arff

J48 pruned tree
------------------
CORRECT_ATTEMPTS*PMPKNOW_XLOW <= 0
|   CORRECT_FIRST_ATTEMPTS*PMPKNOW_LOW <= 0
|   |   TOTAL_HINTS*ATTEMPT_TIME_SHORT <= 0
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH <= 0
|   |   |   |   HINT_TIME_LONG*PMPKNOW_LOW <= 0: FALSE (691.0/6.0)
|   |   |   |   HINT_TIME_LONG*PMPKNOW_LOW > 0
|   |   |   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY <= 0.28: FALSE (27.0)
|   |   |   |   |   PMPKNOW*MIN_PROBLEM_DIFFICULTY > 0.28
|   |   |   |   |   |   ACTIONS <= 5: TRUE (1.0)
|   |   |   |   |   |   ACTIONS > 5: FALSE (1.0)
|   |   |   ATTEMPT_TIME_LONG*PMPKNOW_HIGH > 0
|   |   |   |   ACTIONS*SUM_ACTION_TIME_MS <= 85900
|   |   |   |   |   PMPKNOW <= 0.51: TRUE (2.0)
|   |   |   |   |   PMPKNOW > 0.51: FALSE (2.0)
|   |   |   |   ACTIONS*SUM_ACTION_TIME_MS > 85900: FALSE (39.0)
|   |   TOTAL_HINTS*ATTEMPT_TIME_SHORT > 0
|   |   |   CORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS <= 0
|   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO <= 0.5: FALSE (26.0)
|   |   |   |   PMPKNOW_LOW*PROB_INCORRECT_RATIO > 0.5
|   |   |   |   |   SUM_ACTION_TIME_MS <= 16210: FALSE (1.0)
|   |   |   |   |   SUM_ACTION_TIME_MS > 16210: TRUE (1.0)
|   |   |   CORRECT_ATTEMPTS*MULTIPLE_CHOICE_PROBLEMS > 0
|   |   |   |   ACTIONS <= 5: FALSE (1.0)
|   |   |   |   ACTIONS > 5: TRUE (1.0)
|   CORRECT_FIRST_ATTEMPTS*PMPKNOW_LOW > 0
|   |   INCORRECT_ATTEMPTS*SDV_ATTEMPT_MS <= 3147
|   |   |   ATTEMPT_TIME_LONG*TEXTBOX_PROBLEMS <= 0: FALSE (21.0)
|   |   |   ATTEMPT_TIME_LONG*TEXTBOX_PROBLEMS > 0
|   |   |   |   MIN_PROBLEM_DIFFICULTY <= 0.46: TRUE (1.0)
|   |   |   |   MIN_PROBLEM_DIFFICULTY > 0.46: FALSE (4.0)
|   |   INCORRECT_ATTEMPTS*SDV_ATTEMPT_MS > 3147
|   |   |   BOTTOM_HINTS <= 0: TRUE (1.0)
|   |   |   BOTTOM_HINTS > 0: FALSE (1.0)
CORRECT_ATTEMPTS*PMPKNOW_XLOW > 0
|   TOTAL_HINTS*TEXTBOX_PROBLEMS <= 6
|   |   HINT_TIME_SHORT*PMPKNOW <= 0.16
|   |   |   REPLAYS <= 0: FALSE (19.0)
|   |   |   REPLAYS > 0
|   |   |   |   TOTAL_HINTS <= 0: TRUE (1.0)
|   |   |   |   TOTAL_HINTS > 0: FALSE (1.0)
|   |   HINT_TIME_SHORT*PMPKNOW > 0.16
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS <= 1: TRUE (3.0)
|   |   |   TOTAL_ATTEMPTS*ASSISTMENTS > 1: FALSE (3.0)
|   TOTAL_HINTS*TEXTBOX_PROBLEMS > 6: TRUE (2.0)
```

Appendix 2 – J48 LOOCV Results

Ideal Confusion Matrix		
TRUE	FALSE	← classified as
19	0	TRUE
0	831	FALSE

Time Window	Average Classification Accuracy	Confusion Matrices		
1 minute	0.958823529	Confusion Matrix		
		TRUE	FALSE	← classified as
		3	16	TRUE
		19	812	FALSE
2 minutes	0.955294118	Confusion Matrix		
		TRUE	FALSE	← classified as
		6	13	TRUE
		25	806	FALSE
30 seconds	0.955294118	Confusion Matrix		
		TRUE	FALSE	← classified as
		0	19	TRUE
		19	812	FALSE
4 minutes	0.965882353	Confusion Matrix		
		TRUE	FALSE	← classified as
		5	14	TRUE
		15	816	FALSE
6 minutes	0.956470588	Confusion Matrix		
		TRUE	FALSE	← classified as
		4	15	TRUE
		22	809	FALSE
All	0.958352941	Average Confusion Matrix		
		TRUE	FALSE	← classified as
		3.6	15.4	TRUE
		20	811	FALSE

Appendix 3 – Teacher Survey

Assistments Survey November 2005

We have recently made some changes to the Assistments program, including a new chart at the top of the webpage that tracks student progress. As teachers who have seen our system last year, and can compare the two systems, we especially value your feedback.

(*Please read all the questions before answering, as there is some slight overlap*)

Do you think that the new graphical chart will:

1. Aid teachers in assessing the progress and performance of their students?

Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
1	2	3	4	5

2. Aid students in self-assessing their progress and performance?

Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
1	2	3	4	5

3. Provide students with additional performance-based motivation?

Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
1	2	3	4	5

4. Provide students with additional learning-based motivation?

Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
1	2	3	4	5

5. Decrease off-task student behavior (talking, inactivity, excessive or unnecessary hinting or guessing)?

Strongly Disagree	Disagree	Not Sure	Agree	Strongly Agree
1	2	3	4	5

General Comments or Suggestions:

Thank you!

Appendix 4 – Student Survey Responses

Do you have a computer at home?

Rating	YES	NO
Very Low Gamers	90	10
Below Average	78	22
Above Average	71	29
Very High Gamers	68	32

- Students who gamed a lot were less likely to have a computer at home.

I am good at math.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	40	31	17	2	10
Below Average	33	34	15	12	6
Above Average	20	31	19	10	21
Very High Gamers	25	19	17	19	19

- Students who gamed were more likely to believe that they were not good at math.

I believe that if I work hard at math I can do well.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	67	21	5	0	7
Below Average	65	23	5	4	1
Above Average	66	20	10	2	1
Very High Gamers	51	32	9	3	4

- Students who gamed were less likely to believe they could do well at math if they worked hard.

I do my homework in math class.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	55	26	7	5	7
Below Average	45	30	8	8	7
Above Average	42	34	9	7	7
Very High Gamers	26	47	6	13	6

- Students who gamed said they were less likely to do homework in math class.

I found it hard to stay concentrated on the computer all the time.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	21	21	7	33	14
Below Average	23	26	12	15	23
Above Average	22	25	11	14	26
Very High Gamers	13	30	11	26	26

- Students who gamed a lot said they were less likely to have trouble concentrating on the computer.

I found many of the items frustrating because they were too hard.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	10	29	21	30	10
Below Average	21	30	21	17	11
Above Average	29	35	18	14	4
Very High Gamers	36	34	15	13	0

- Students who gamed a lot tended to strongly agree that the items were frustrating because they were too hard, while students who gamed very little were more likely to disagree. This is probably partially related to student prior knowledge.

I like learning from a computer.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	36	17	23	14	10
Below Average	39	35	15	7	4
Above Average	31	34	20	10	5
Very High Gamers	45	38	6	9	2

- Students who gamed a lot agreed that they liked learning from a computer, more often and more strenuously, than those who gamed very little.

I like math class.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	36	34	8	12	10
Below Average	36	25	10	15	14
Above Average	30	21	11	15	22
Very High Gamers	19	28	15	9	28

- The more students gamed, the less they said they liked math class.

I liked using the Assistment system better than doing my homework.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	43	17	19	7	12
Below Average	48	23	13	8	7
Above Average	35	21	19	9	15
Very High Gamers	36	32	15	6	11

- Students who tended not to game were more likely to say that they preferred using the *Assistments* system to doing homework. In a similar question, there were no differences between the groups when asked if they would prefer to use the tutor rather than take a test -- they mostly all strongly agreed that they would.

I liked using the Assistment system better than my normal classroom activity.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	50	19	7	10	14
Below Average	41	20	17	9	13
Above Average	33	20	22	10	14
Very High Gamers	38	25	23	4	9

- The less a student gamed, the more strongly they would prefer using the *Assistments* system to normal classroom activity.

I prefer classes that emphasize facts and data over concepts and ideas.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	12	19	45	12	10
Below Average	11	26	42	13	7
Above Average	11	22	45	12	9
Very High Gamers	18	26	47	4	4

- Students who gamed a lot had a slight tendency to say that they prefer facts and data over concepts and ideas more than other students.

I think that being told the answer was more helpful than reading the hints.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	12	19	12	29	29
Below Average	23	20	12	23	22
Above Average	23	23	16	20	18
Very High Gamers	28	23	19	15	15

- The more a student gamed, the more they thought that being told the answer was more helpful than reading the hints.

I think the hints helped me understand how to solve similar problems.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	21	45	19	10	5
Below Average	36	39	12	8	5
Above Average	38	36	14	5	5
Very High Gamers	45	38	8	2	8

- The more a student gamed, the more they agreed that the hints aided in their understanding of similar problems.

I tried to get through difficult problems as quickly as possible.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	21	21	21	26	10
Below Average	26	32	9	18	15
Above Average	22	27	15	18	18
Very High Gamers	30	34	8	19	15

- Students who gamed a lot were more likely to agree or strongly agree that they tried to get through difficult problems as quickly as possible.

I usually seek help when I don't understand something.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	40	26	16	12	5
Below Average	39	41	9	7	5
Above Average	32	42	13	7	5
Very High Gamers	25	47	13	5	7

- Students who gamed very little were more likely to strongly agree that they would seek help when they didn't understand something.

My goal when using the Assistment system was to get through as many items as possible.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	21	31	29	12	7
Below Average	37	31	15	11	4
Above Average	35	34	14	12	4
Very High Gamers	42	34	13	9	1

- Students who gamed a lot tended more than other students to say that their goal was to get through as many items as possible.

My goal when using the Assistment system was to learn new things.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	33	26	21	14	2
Below Average	35	35	19	6	3
Above Average	42	25	19	5	8
Very High Gamers	43	28	23	4	2

- The more students gamed they more they tended to strongly agree that their goal was to learn new things.

My parents think it's very important to do well in math.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	62	29	10	0	0
Below Average	60	21	16	2	1
Above Average	58	20	19	2	0
Very High Gamers	74	13	8	2	2

- Students who gamed a lot were much more likely than other students to strongly agree that their parents thought it important for them to do well in math.

When I grow up I think I will use math in my job.

Rating	Strongly Agree	Agree	Not Sure	Disagree	Strongly Disagree
Very Low Gamers	43	17	29	2	10
Below Average	34	21	27	6	11
Above Average	29	31	26	4	10
Very High Gamers	25	36	23	2	13

- The less a student gamed the more they were likely to strongly agree that they would use math in a job when they grew up.