WORCESTER POLYTECHNIC INSTITUTE

DEPARTMENT OF COMPUTER SCIENCE

MAJOR QUALIFYING PROJECT

FINAL REPORT

SEARCHING FOR LEARNING

June 13, 2011

Submitted by

MELISSA GAVARNY

Approved by

PROFESSOR JOSEPH E. BECK

1 ABSTRACT

The simple knowledge tracing model assumes that students learn at a constant rate and uses a static probability to update the probability that a student knows a skill which is constant across all students and problems defying common expectations and prior research! Therefore, the model does not accurately model student learning and this view of student learning requires improvement. Using a contextualized learning rate created a model of student learning that outperformed the original model.

2 ACKNOWLEDGEMENTS

I would like to thank Joseph E. Beck for his continued assistance and guidance on this project.

3 TABLE OF CONTENTS

1 ABSTRACT I

2 ACKNOWLEDGEMENTS II

3 TABLE OF CONTENTS III

4 LIST OF FIGURES V

5 LIST OF TABLES VI

6 INTRODUCTION 7

7 BACKGROUND 9

7.1 STUDENT MODELING 9

7.2 KNOWLEDGE TRACING 10

7.2.1 The Knowledge Tracing Model 10
7.2.2 Fitting the Model 11
7.2.3 Tracking Student Knowledge 12
7.2.4 Applications of Knowledge Tracing 16

8 MOTIVATION 19

9 METHODOLOGY 22

9.1 DATA DESCRIPTION 22

9.1.1 Original Data Set 22
9.1.2 Detection of Learning State 26

9.2 INFRASTRUCTURE 45

9.2.1 BNT-SM 47
9.2.2 Python Script: runAfterBNT_SM.py 50
9.2.3 SPSS Statistics 54
9.2.4 Python Script: runAfterSPSS.py 55

10 RESULTS 57

10.1 MODEL FIT 57

10.2 UNDERSTANDING STUDENT LEARNING 61

10.3 IMPACT OF STUDENT LEARNING ON KNOWLEDGE TRACING MODEL 71

11 FUTURE WORK AND CONCLUSIONS 74

12 BIBLIOGRAPHY 76

4 LIST OF FIGURES

FIGURE 7.1: KNOWLEDGE TRACING MODEL 11
FIGURE 7.2: PSEUDO CODE TO CALCULATE STUDENT KNOWLEDGE AT TIME T (GONG, BECK, & HEFFERNAN, 2010B) 13
FIGURE 9.1: ASSISTMENTS PROBLEM #27366 23
FIGURE 9.2: ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM 24
FIGURE 9.3: ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM HINT 1 24
FIGURE 9.4: AVAILABLE HINTS FOR ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM 25
FIGURE 9.5: ASSISTMENTS PROBLEM #27366 SECOND SCAFFOLD PROBLEM 25
FIGURE 9.6: ASSISTMENTS PROBLEM #27366 LAST SCAFFOLD PROBLEM 26
FIGURE 9.7: KNOWLEDGE TRACING MODEL WITH LEARNING STATE 44
FIGURE 9.8: PROCESS TO INFER MODEL PARAMETERS AND ESTIMATE P(LS) VALUES 46
FIGURE 10.1: TREND OF R^2 ERROR ACCROSS ALL ITERATIONS 60
FIGURE 10.2: TREND FOR AVERAGETIMETOCOMPLETE DERIVED FIELD 67
FIGURE 10.3: TREND FOR DIRECTTOANSWERAFTERLASTHINT DERIVED FIELD 69
FIGURE 10.4: TREND FOR DIRECTTOANSWERAFTERBOTTOMOUTHINT DERIVED FIELD 71

5 LIST OF TABLES

TABLE 9.1: SAMPLE DERIVED FIELD VALUES SCORES AND WEIGHTS 28
TABLE 9.2: SAMPLE GAMELASTX DATA 31
TABLE 9.3: SAMPLE GAMELASTSIMILARX DATA 32
TABLE 9.4: SAMPLE LASTXWRONGCURRENTRIGHT DATA 34
TABLE 9.5: SAMPLE LASTXOFYWRONGCURRENTRIGHT DATA 35
TABLE 9.6: SAMPLE AVERAGETIMETOCOMPLETE DATA 38
TABLE 9.7: SAMPLE DIRECTTOANSWERAFTERLASTHINT DATA 41
TABLE 9.8: SAMPLE DIRECTTOANSWERAFTERBOTTOMOUTHINT DATA 44
TABLE 9.9: INITIAL VALUES OF MODEL PARAMETERS 49
TABLE 9.10: PERCENT DELTA VALUE FOR P(LS) LOOKUP TABLE BY POSITION AND STREAK FOR ITERATIONS 1-14 51
TABLE 9.11: PERCENT DELTA VALUE FOR P(LS) LOOKUP TABLE BY POSITION AND STREAK FOR ITERATIONS 15-50 51
TABLE 9.12: PARTIAL RUNAFTERBNT_SM.PY OUTPUT FILE 52
TABLE 10.1: DERIVED FIELDS COEFFICIENTS 63
TABLE 10.2: POSSIBLE SCORES FOR EACH MODIFICATION OF AVERAGETIMETOCOMPLETE DERIVED FIELD 66
TABLE 10.3: POSSIBLE SCORES FOR EACH MODIFICATION OF DIRECTTOANSWERAFTERLASTHINT DERIVED FIELD 68
TABLE 10.4: POSSIBLE SCORES FOR EACH MODIFICATION OF DIRECTTOANSWERAFTERBOTTOMOUTHINT DERIVED FIELD 70
TABLE 10.5: INITIAL AND FINAL VALUES OF THE INFERRED MODEL PARAMETERS 71

6 INTRODUCTION

The standard knowledge tracing model assumes that students learn at a constant rate as a result of practice (Corbett & Anderson, 1995). The model uses a static probability to estimate the probability that a student learns a given skill at each time step. Therefore, the probability a student learns the given skill at this time step is constant across all students and problems, and is independent of any other information such as whether the student answered the problem correctly or incorrectly or whether the student was gaming. Using a constant rate of learning not only defies common expectations that different students learn at different rates and these rates change from day to day and problem to problem, but it also defies all prior research on the subject. A student's learning rate is affected by context, the individual student, the emotional state and behavior of the student, and more.

Since the current knowledge tracing model uses this constant rate of learning across all students and problems, it therefore does not accurately model student learning. Thus, improvement in how the knowledge tracing model views learning is required in order to create a more accurate model of the student's learning process. Using a contextualized, dynamic learning rate in the knowledge tracing model, where the learning rate is conditioned on certain characteristics which have been shown to influence student learning, such as whether or not a student is gaming, can create a more accurate model of the student learning process. A more accurate model of student learning not only allows tutoring systems, such as cognitive tutor systems, to adapt their instruction in order to nudge students into certain states which have been shown to lead to learning, but gives more accurate information about the student learning process to the cognitive psychology community which can be used to improve classroom teaching methods as well. This new information about student learning can be used for a wide range of tasks, such as pinpointing practice problems that have statistically been seen to produce the greatest gain in learning (Feng, Heffernan, & Beck, 2009) or improving the tailoring of practice problems to students based on the way they learn, among others. Through this project, I created a set of derived fields which were used to generate a value indicating the likelihood that a student was in a state where he or she was more likely to acquire knowledge of a skill. This value was given as input to the current knowledge tracing model which created a model of student learning that outperformed this simple knowledge tracing model. The results I collected about the derived features I selected to generate this probability largely followed my initial hypotheses about the effect each would have on the likelihood of student learning and therefore it can be concluded that these features do affect student learning.

7 BACKGROUND

As Intelligent Tutoring Systems (ITS) become more widely used both inside and outside of the classroom to aid student learning, there is an increased need for these computer based tutoring environments to increase and accelerate learning. Already a type of ITS, Cognitive Tutors, have been shown to be an effective teaching tool enabling students to complete assignments in significantly less time (Cen, Koedinger, & Junker, 2007) and score as well if not better than control students on posttests (Koedinger & Anderson, 1997; Corbett & Anderson, 1995). ITS rely on student modeling in order to predict future student behavior, such as future student performance. This model can then be studied using Educational Data Mining to generate information about student knowledge or used in a variety of applications such as the creation of cognitive tutors or creating processes to facilitate Mastery Learning. Cognitive tutors, for example, use this model to determine which skills the student knows and which need to be practiced, which enables these tutors to assign practice problems tailored specifically to the current knowledge of each student. Recently work has been done on applying knowledge engineering methods and educational data mining techniques to the modeling process to produce a more accurate model of student knowledge.

7.1 STUDENT MODELING

Student modeling is the general process by which observations of student behavior are recorded and information about the student is inferred based on these observations. The most common observation used when creating a student model is whether the student responded correctly or incorrectly. Any observed information can be used to generate a student model; for example, the difficulty of the task or time data. One modeling technique, created by John Stamper and Tiffany Barnes, in order to provide useful hint information to students studying mathematical proofs used observations of which action the student took, in other words, which axiom the student applied and to what the student applied this axiom (Barnes & Stamper, 2008).

These observations help to create a student model that attempts to infer a characteristic of the student. Typically, student models infer information about student knowledge, the misconceptions students have, or what approach the student is using to solve a given problem or complete a specified task. For the proof model example, John Stamper and Tiffany Barnes initially used the information gathered for numerous students which detailed which axiom the student applied and to what the student applied this axiom in order to automatically build a tree of all probable paths students take while solving a given proof. They were able to create a new student model which would use this tree and would attempt to determine which approach the student was using to complete the proof. Because this new model had information about which approach the student was likely taking, it could be used to provide more specific automatically generated hints catered to the method the student was attempting to use (Barnes & Stamper, 2008).

7.2 KNOWLEDGE TRACING

7.2.1 THE KNOWLEDGE TRACING MODEL

Knowledge Tracing (Corbett & Anderson, 1995) is a specific type of student modeling technique that uses observations of whether or not the student performed correctly on a given task in order to infer whether or not the student knew the skill required to complete the task at that time step. The knowledge a student has or does not have about a given skill will affect his or her performance on a set of problems which test that skill (P(correct | knows skill) != P(correct | doesn't know skill)). Student performance is used to infer knowledge because knowledge is a *latent* property and therefore cannot be directly measured.

As a real-world example, consider having a bag of candies where each candy had a flavor of either cherry or lime and a wrapper of red or green (Russell & Peter, 2010). Most cherry candies in the bag have red wrappers and most lime candies have green wrappers but occasionally the candy factory will make a mistake and a cherry candy will have a green wrapper or a lime candy will be have a red wrapper. If the flavor and wrapper color of each candy is known then it is easy to determine the probability that a cherry candy has a red wrapper (P(flavor = cherry | wrapper = red)); however, if the flavor of each candy is not known then determining this probability is much more difficult. This is the challenge facing knowledge tracing, because student knowledge, like the unknown candy flavor, is latent. In the candy example the color of the wrapper can be used to infer the flavor since a red wrapper is a good though not perfect indicator of a cherry candy and a green wrapper is a relatively good indicator of a lime candy. Similarly the student's performance on a problem can be used to infer whether or not the student knows the given skill required to perform correctly on the problem. If the student performs correctly on a problem it is a good though not perfect indicator of the student knowing the skill, and if the student performs incorrectly it is a relatively good indicator of the student not knowing the skill. The probability that a student knows the given skill is updated each time the student encounters a new problem based upon the student's performance on the current problem. Knowledge of a skill is represented by a probability that represents the likelihood the student knows the skill. The model for knowledge tracing can be seen in Figure 7.1.

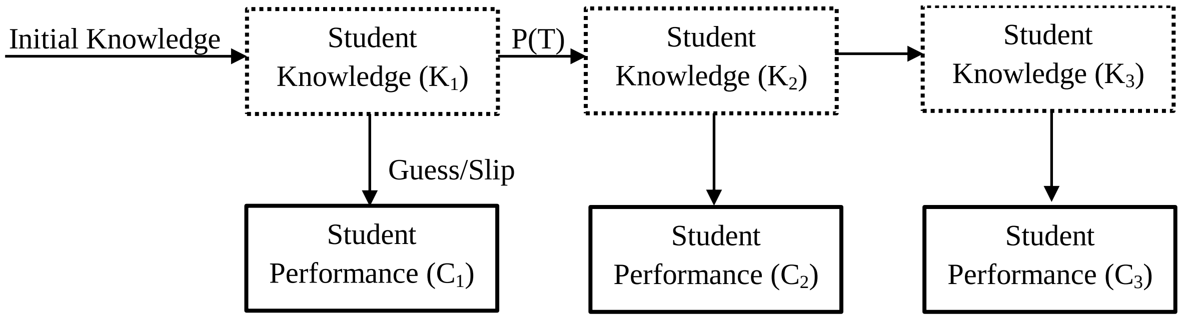

Initial Knowledge
Student Knowledge (K_1)
P(T)
Student Knowledge (K_2)
Student Knowledge (K_3)
Guess/Slip
Student Performance (C_1)
Student Performance (C_2)
Student Performance (C_3)

FIGURE 7.1: KNOWLEDGE TRACING MODEL

7.2.2 FITTING THE MODEL

The knowledge tracing model above is fit to a set of data from numerous students which infers the values of the four knowledge tracing parameters, initial knowledge, probability of transition, guess, and slip. Initial knowledge, $P(K_0)$, is defined as the probability that the student knows a skill before beginning any problems (at time 0). The probability of transition, P(T), is defined as the probability that the student transitions from a not knowing a skill (the unlearned state) to knowing the skill (the learned state) as a result of practice. The guess parameter, P(~knowledge | correct), is defined as the probability that the student answered the problem correctly even though he or she did not know the skill, such as if a student did not know how to solve for the area of a triangle but arbitrarily guessed the correct value or if a student answered a multiple choice question by randomly selecting one of the answers. Finally the slip parameter, P(knowledge | ~correct), is defined as the probability that the student incorrectly answered the problem even though he or she already knew the skill, such as if a student mistyped an answer such as submitting 54 instead of 45 when the student knew the correct answer was 45.

Through training the model to fit the data set, each of these parameters is inferred for each skill in the data set (Beck, 2006). Model training, in essence, is finding a set of model parameters which maximize the fit of the knowledge tracing model to the data, i.e. maximize P(data | model parameters). A common approach used to maximize this value is Expectation Maximization (EM) (Borman, 2009), an iterative procedure which attempts to maximize the log likelihood estimate when there is unknown data, in this case maximizing the P(data | model parameters) when the model parameters are unknown. The EM algorithm initially sets the model parameters to random values. These parameters are updated with each iteration. The EM process consists of two phases the **E**xpectation phase and the **M**aximization phase. In the expectation phase, the conditional expectation for the latent variable, in this case knowledge, given the observed data and the current estimates of the parameters for each of the possible combinations of the parameters. Conditional expectation E is defined as $E_{Z \mid X, \theta_n}(P(X,\ z \mid \theta))$, i.e. the expectation of the occurrence of a hidden variable, Z, in this case knowledge, given both the observed data, X, and the current estimates of the model parameters, θ_n, is equal to the probability of the occurrence of both the observed data, X, and the value of the hidden variable, z, in this case whether or not the student knows a skill, given the model parameters, θ. In the maximization phase EM assumes that the model parameter estimates, θ, are correct and maximizes the conditional expectation based on these estimates, i.e. new model parameters, θ_{n+1}, are selected based on which model parameters, θ, give the highest conditional expectation value. This process of inferring the model parameters is used to estimate the guess, slip, initial knowledge, and transition parameters for each of the skills in the data set.

7.2.3 TRACKING STUDENT KNOWLEDGE

Once the knowledge tracing model parameters are estimated by training the knowledge tracing model to the data, the estimated parameters can be used to track a student's knowledge, based on his or her performance (Corbett & Anderson, 1995). At each time step the student's current knowledge state is updated, in other words, every time a student completes a problem the probability that the student knows the skill that the problem is testing is updated. This is computed using the process described by the pseudo code in Figure 7.2.

```
if $response_{t-1}$ == correct
    $K = K_{t-1}*(1 - slip)/(K_{t-1}*(1 - slip) + guess * (1 - K_{t-1}))$
else
    $K = K_{t-1} * slip / (K_{t-1} * slip + (1 - K_{t-1})*(1 - guess))$
$K_t = K + (1 - K)*P(T)$
```

FIGURE 7.2: PSEUDO CODE TO CALCULATE STUDENT KNOWLEDGE AT TIME t (GONG, BECK, & HEFFERNAN, 2010B)

The probability that the student knows the skill at the current time step (K_t), i.e. when the student completes practice problem t-1 but before the student begins the t^{th} practice problem, is calculated based on the student's prior knowledge, K_{t-1}, and performance, C_{t-1}, on practice problem t-1 and the estimated probability of transition for this skill, P(T). Following the Markov Assumption that each state is only dependent on the state that directly proceeded it, i.e. the state at t-1, and the past states are irrelevant, the model looks only at the prior knowledge estimate, K_{t-1}, and ignores the process by which the student got to that level of knowledge. If the student answered the problem correctly then the best estimate of the student's knowledge, K, is computed by finding the probability that the student knew the skill and did not slip, divided by the probability that he or she answered the problem correctly (either through knowledge or through guessing). If a student answered the problem incorrectly then the best estimate of the student's knowledge, K, is computed by finding the probability that the student knew the skill at time t-1, K_{t-1}, and accidentally submitted an incorrect response or slipped, divided by the probability that the student answered the question incorrectly. The student's knowledge at time t is finally computed by updating K by the probability that the student transitioned from the unlearned to the learned state, P(T), by using the equation $K_t = K + (1-K)*P(T)$, i.e. the student's current knowledge is the sum of the probability that the student already knew the skill and the probability that the student did not know the skill and learned the skill through practice on this problem. The skill that is being tested determines which of the values of the model parameters are used in the equations.

To determine whether a student will respond correctly, the student's current knowledge, K_t, can then be used to predict whether or not the student will answer the current problem correctly by first finding the probability that the student will answer the problem correctly, $P(correct_t)$, using the formula $P(correct_{t+1}) = K_t *(1 - slip) + guess * (1 - knowledge_t)$. The student is predicted to answer the problem correctly if this probability $P(\text{correct}_t) > 0.5$, otherwise the student is predicted to answer the problem incorrectly.

As an example, assume a student answers the first problem in a problem set which tests skill five. The model parameters for skill five are as follows: slip is 0.3, guess is 0.2, probability of transition, P(T), is 0.15, and initial knowledge, K_0 is 0.1. Before the student answers the first problem the student's knowledge for skill five is set to the initial knowledge parameter, the likelihood that the student will answer this problem correctly, $P(\text{correct}_1)$, is computed as follows:

$$\begin{aligned}P(\text{correct}_1) &= K_0 *(1 - \text{slip}) + \text{guess} * (1 - K_0)\\ &= 0.1 *(1 - 0.3) + 0.2 * (1 - 0.1)\\ &= 0.07 + 0.18\\ &= 0.25\end{aligned}$$

This value indicates that the student has a 25% chance of getting the first problem correct. Since this value is less than 0.5 it is predicted that the student will answer problem one incorrectly. Assume the student answers problem one incorrectly. The student's estimated knowledge after finishing the problem at t = 1, K_1, is updated based on the initial knowledge parameter and the student's performance on the problem. Initially the best estimate of the student's knowledge, K, is computed. Since the student answered incorrectly K is calculated following the process in Figure 7.2 using the equation on line four:

$$\begin{aligned}K &= K_0 * \text{slip} / ((K_0 * \text{slip}) + (1 - K_0) * (1 - \text{guess}))\\ &= 0.1 * 0.3 / ((0.1 * 0.3) + (1 - 0.1)*(1 - 0.2))\\ &= 0.03 / 0.75\\ &= 0.04\end{aligned}$$

This best estimate of the student's knowledge, K, is then used to calculate the student's current knowledge, K_1. K_1 is calculated following the process described in Figure 7.2 using the equation on line five:

$K_1 = K + (1 - K) * P(T)$

$= 0.04 + (1 - 0.04) * 0.15$

$= 0.04 + 0.144$

$= 0.184$

This new current knowledge estimate for the student indicates that the student is 8.4% more likely to know the skill after performing problem one than before he or she completed problem one as a result of practice. The student's current knowledge estimate can then be used to calculate whether the student will perform correctly on the next problem, $P(correct_2)$, which is computed as follows:

$P(correct_2) = K_1 *(1 - slip) + guess * (1 – K_1)$

$= 0.184 *(1 – 0.3) + 0.2 * (1 – 0.184)$

$= 0.1288 + 0.1636$

$= 0.2924$

This value indicates that the student has a 29% chance of getting the second problem correct. Since this value is less than 0.5 it is predicted that the student will answer problem two incorrectly. Assume the student then completes the second problem correctly. The student's estimated knowledge after finishing the problem at t = 2, K_2, is updated based on the initial knowledge parameter and the student's performance on the problem. Initially the best estimate of the student's knowledge, K, is computed. Since the student answered incorrectly K is calculated following the process in Figure 7.2 using the equation on line four:

$K = K_1 * (1 - slip) / ((K_1 * (1 - slip) + (1 - K_1) * guess)$

$= 0.184 * (1 - 0.3) / ((0.184 * (1 - 0.3) + (1 - 0.184) * 0.2)$

$= 0.1288 / (0.1288 + 0.1632)$

$= 0.1288 / 0.292$

$= 0.44$

This best estimate of the student's knowledge, K, is then used to calculate the student's current knowledge, K_2. K_2 is calculated following the process described in Figure 7.2 using the equation on line five:

$$\begin{aligned} K_1 &= K + (1 - K) * P(T) \\ &= 0.44 + (1 - 0.44) * 0.15 \\ &= 0.44 + 0.084 \\ &= 0.524 \end{aligned}$$

This new current knowledge estimate for the student indicates that the student is 34% more likely to know the skill after performing problem two than before he or she completed problem two as a result of practice. The student's current knowledge estimate can then be used to calculate whether the student will perform correctly on the next problem, $P(correct_3)$, which is computed as follows:

$$\begin{aligned} &P(correct_3) = K_2 *(1 - slip) + guess * (1 - K_2) \\ &= 0.524 *(1 - 0.3) + 0.2 * (1 - 0.524) \\ &= 0.3668 + 0.0952 \\ &= 0.462 \end{aligned}$$

This value indicates that the student has a 46% chance of getting the third problem correct. Since this value is less than 0.5 it is predicted that the student will answer problem three incorrectly.

7.2.4 APPLICATIONS OF KNOWLEDGE TRACING

The knowledge tracing model can be used in a variety of applications. One such application of the model is in the field of Educational Data Mining (www.educationaldatamining.org) which explores interesting data and trends in educational data and through this exploration attempts better understand students and how they learn. Cognitive tutors also use the knowledge tracing model (described above) to determine which skills the student knows and which need to be practiced, and to assign practice problems tailored specifically to the current knowledge of each student. The knowledge tracing model is also used to facilitate Mastery Learning. Mastery Learning is requiring a student to practice a skill until he or she has mastered it. In Cognitive Tutors, mastery is defined as when the knowledge tracing model has P(Knowledge) >= 0.95, i.e. the probability that the student knows the skill is at least 95%.

However, even though this model is used in so many different settings, there are definite limitations to the model. The model shown, for example, has an R^2 value of only 7% for this test dataset (Gong, Beck, Heffernan, & Forbes-Summers, 2010). The R^2 of a model is defined as $1 - \frac{\sum(predicted\ value - actual\ value)^2}{\sum(mean - actual\ value)^2}$. The predicted value is whether or not the model would predict that the student would get the problem correct. A student is predicted to answer the problem correctly if the probability that the student answered the question correctly, P(correct), is greater than 0.5 where $P(correct) = P(knowledge) * (1 - P(slip)) + (1 - P(know)) * P(guess)$. In other words, P(correct) is the probability that either the student knew the skill and did not answer incorrectly (did not slip) or the student did not know the skill and guessed the correct answer. The mean is the total number of problems students answered correctly for this skill divided by the total number of problems. Having an R-squared gain of only 7% indicates that this model fits the data with only 7% less squared error than a “model” that simply guessed the mean.

Several projects have recently been undertaken in order to try to improve the knowledge tracing model. Instead of using static probabilities for guess and slip, Baker, Corbett, and Aleven instead used contextualized parameters for guess and slip that were able to more accurately predict these behaviors in students (Baker, Corbett, & Aleven, 2008). Using this technique their model greatly outperformed the previous baseline model. Wang and Heffernan decided to use all of the information about the student’s behaviors while completing the problem, such as whether the student asked for a hint or how many times he or she attempted to answer the problem. They used a continuous value for correctness of an answer instead of a boolean value of zero or one. Students would receive a partial credit value ranging between zero and one based on their actions. Students would receive a correctness score of one if they answered the problem correctly on the first attempt and this score was decreased for such events as multiple guesses of the answer or requesting hints (Wang & Heffernan, 2011). Another study by Gong, Beck, and Heffernan used a set of three rules to determine if a student was exhibiting “gaming” behavior while using an online tutor system called ASSISTments. A student was determined to be “gaming” if the student guessed multiple answers to the same question in rapid succession on two consecutive questions, performed any action before the student had enough time to read the problem or the last hint he or she was given, or if the student requested the last hint which displayed the answer in three consecutive problems (Gong, Beck, Heffernan, & Forbes-Summers, 2010). Their study was able to more accurately model the acquisition of knowledge and found that when a student was “gaming” on a problem the student’s learning was decreased, almost zero, compared to problems where the student was not “gaming”. The study also concluded that students were more likely to “game” if they had little knowledge of the skill required to perform correctly on the problem. Lastly, they found that the identity of the student is more effective in predicting if the student will “game” then the given skill that is tested.

8 MOTIVATION

The current knowledge tracing model assumes that students learn at a constant rate as a result of practice (Corbett & Anderson, 1995). The model uses a static probability P(T), the probability the student transitions from a state of not knowing the skill to a state of knowing the skill (i.e. the probability a student learns a skill), to update the probability that a student knows a given skill at each time step using the formula $P(K_t) = P(K_{t-1} \mid evidence) + (1-P(K_{t-1} \mid evidence) * P(t))$. Therefore, the probability a student learns the given skill at this time step is constant across all students and problems and is independent of any other information such as whether the student answered the problem correctly or incorrectly or whether the student was gaming. Using a constant rate of learning not only defies common expectations that different students learn at different rates and these rates change from day to day and problem to problem, but it also defies all prior research on the subject! A student's learning rate is affected by context, the individual student, the emotional state and behavior of the student, and more.

Beck and Mostow performed a study comparing student learning during wide reading and re-reading (Beck & Mostow, How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students, 2010) which showed that the context of the problem affected student learning rates. This study looked at whether students learned more when they read different stories or when they re-read the same story again. Through this study, they concluded that when students read different stories they would actually learn to read better than when a student proceeded to re-read the same story again. The simple model of learning assumes that learning is constant meaning "one unit of learning" occurs every time you complete a task. If this model was accurate in its assumptions that "one unit of learning" occurs every time you complete a task, in this case read a word, section, or book, then not only would every student learn at the same rate but that rate would not change if the student read the same story or different stories, therefore this assumption has to be incorrect.

Another study by Swire, Pardos, and Heffernan on the effects of immediate feedback on K-12 learning (Swire, Pardos, & Heffernan, 2011) demonstrated that different students learn at different rates. Students were given a set of pre-test and post-test questions to solve. Half of the students received immediate feedback on their performance on the post-test while the other half did not. Not only did the students who received this feedback have higher post-test results confirming that the context of problems affects the rate of student learning but different students within the same group performed differently from each other. The study also found that females who were given immediate feedback had a much greater gain in performance from the pre-test to the post-test then males who were given immediate feedback. They concluded, therefore, that immediate feedback aided females more than males. The simple model of learning assumes that learning is constant across students meaning that every student who practices a certain set of problems should gain the same amount of knowledge as a result of this practice. If this model was accurate in its assumptions that all students learn at the same rate, then not only would every student gain the same amount of knowledge and have the exact same gain in performance between the pre- and post-tests but the difference in gender between students would not cause a difference in the amount that immediate feedback aided the student, therefore this assumption has to be incorrect.

Gong, Beck, Heffernan, and Forbes-Summers performed a study on the effects of gaming on student learning which also defies the simple model of learning by demonstrating that the student's emotional state and student behavior affect student learning. This study used a set of three rules to determine if a student was exhibiting "gaming" behavior while using the ASSISTments online tutor system. A student was determined to be "gaming" if the student guessed multiple answers to the same question in rapid succession on two consecutive questions, performed any action before the student had enough time to read the problem or the last hint he or she was given, or if the student requested the last hint which displayed the answer in three consecutive problems (Gong, Beck, Heffernan, & Forbes-Summers, 2010). They found that when a student was "gaming" on a problem the student's learning was decreased, almost zero, compared to problems where the student was not "gaming." The study also concluded that students were more likely to "game" if they had little knowledge of the skill required to perform correctly on the problem. Lastly, they found that the identity of the student is more effective in predicting if the student will "game" than the given skill that is tested. The simple model of learning assumes that learning is constant across all time steps and across every problem regardless of the emotional state of the student. If this model were accurate in its assumptions that a student always learns at the same rate, regardless of the student's emotional state, then the rate of learning for a student who is gaming would not be decreased as was shown in the study, therefore this assumption has to be incorrect.

Since the current knowledge tracing model follows the key assumption of the simple model of learning, a constant rate of learning across all students and problems, it therefore does not accurately model student learning. Thus, improvement of how the knowledge tracing model views learning is required in order to create a more accurate model of the student's learning process. Using a contextualized, dynamic learning rate in the knowledge tracing model, where the learning rate is conditioned on certain characteristics which have been shown to influence student learning, such as whether or not a student is gaming, can create a more accurate model of the student learning process. A more accurate model of the student learning process gives better information about how and why students learn providing a better understanding of the student learning process. Utilizing this new information about student learning not only allows adaptation of tutoring systems, such as cognitive tutor systems, in order to nudge students into certain states which have been shown to lead to learning, but gives more accurate information about the student learning process to the cognitive psychology community which can be used to improve classroom teaching methods as well. This new information about student learning can be used for a wide range of tasks, such as pinpointing practice problems that have statistically been seen to produce the greatest gain in learning (Feng, Heffernan, & Beck, 2009) or improving the tailoring of practice problems to students based on the way they learn, among others.

9 METHODOLOGY

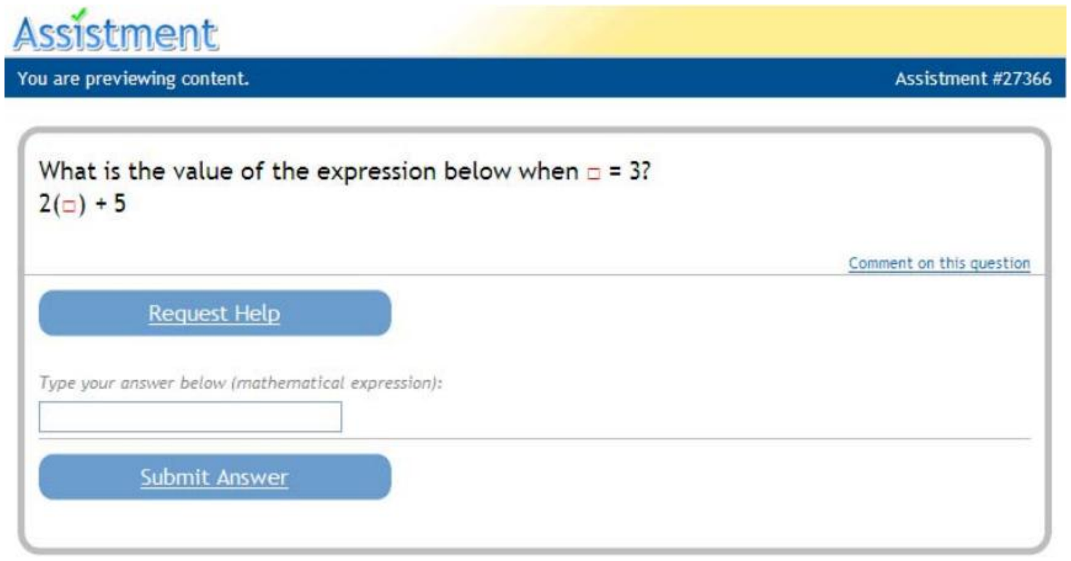
9.1 DATA DESCRIPTION

9.1.1 ORIGINAL DATA SET

For this project, I reused data that had been gathered by Yue Gong for a previous project comparing knowledge tracing and Performance Factor Analysis (Gong, Beck, & Heffernan, Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting, 2010). This data was gathered using ASSISTments, an online tutoring system that aids students in various subjects of study. Students from four eighth-grade classes in urban school districts in the northern United States participated in the study. The actions of three-hundred forty-three students ranging in age from twelve to fourteen were logged as they used the ASSISTments software to help them learn mathematics.

The ASSISTments software allows a student to complete various problem sets assigned to him or her by the student's instructor. ASSISTments problem sets consist of a number of main problems each providing practice on one or more skills (i.e. multiplication, division, area of a circle, etc.). A main problem consists of the primary question and a number of help actions. A main question uses one of two different help strategies: hint or scaffold. If the student answered the problem incorrectly or requested help on a question with a hint strategy then the student would either be able to try to answer the problem again or request a hint. A hint will display a message which shows the student the next step to take when attempting to solve a problem. For example if the student was given the equation 3x + 8 = 38 and asked to solve for x, the first hint would tell the student to subtract 8 from each side and show that the new equation would be 3x = 30. The last hint available to the student shows the student the answer to the problem and was called a bottom out hint.

In the scaffold strategy, if the student answered correctly, then the student would move on to the next problem and skip the scaffold problems. However, if a student incorrectly answered a question or requested help, the student would be given a number of scaffold problems that break down the problem into small steps. For example, assume a student received problem #27366 from the ASSISTments system, shown in Figure 9.1**Error! Reference source not found.**, a sixth-grade algebraic substitution problem that asks the student to solve 2(□) + 5 when □=3. If the student answered this problem incorrectly or clicks “Request Help” then he or she would be prompted to answer a set of scaffolding questions. The first scaffolding problem the student would be required to complete is shown in Figure 9.2. It guides the student in solving the problem by prompting the student to solve the first step, figuring out where the substitution occurs in the problem. Each scaffold problem has a number of available hints which behave in the same manner as the main problem’s hints. If the student is does not know how to answer the scaffold question and clicks “Request Help” or answers the scaffold problem incorrectly, a detailed hint is provided to the student, shown in Figure 9.3. Each hint gives a small clue of the next minor step in the process and the last hint available gives the student the correct answer so the student can move on to the next problem, shown in Figure 9.4. After the student answers the first scaffold problem correctly he or she can move on to the next scaffold problem, seen in Figure 9.5, which will guide the student in solving the next step of the main problem, determining what the next step is to solve the problem. The student will continue to answer scaffold problems in this manner until he or she has correctly answered all of the scaffold problems for this main problem. The last scaffold problem in the scaffold set asks the student to solve the same main problem again, seen in Figure 9.6.


Assistment

You are previewing content. Assistment #27366

What is the value of the expression below when □ = 3?
2(□) + 5

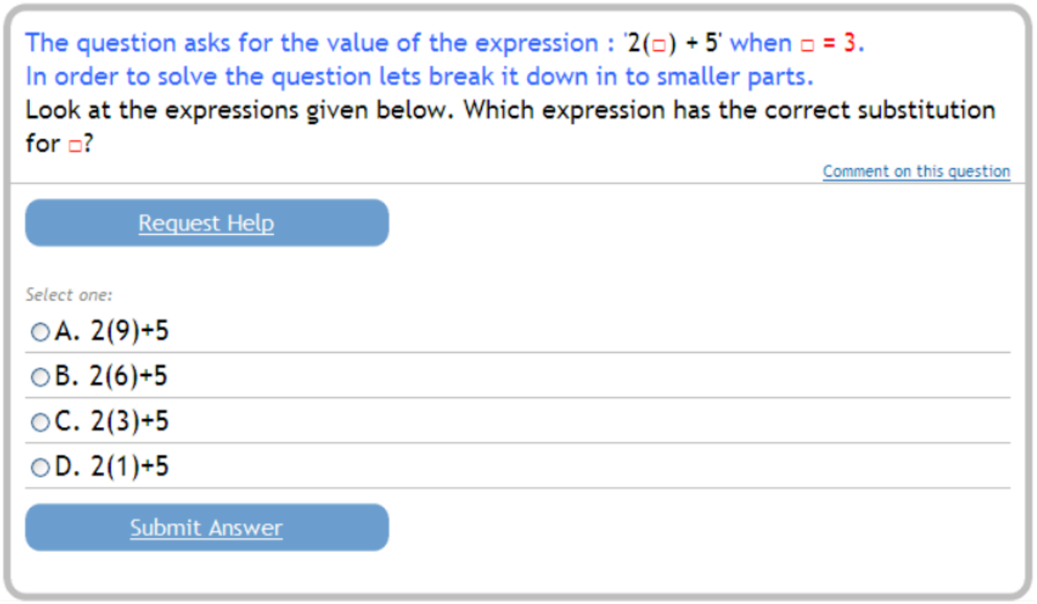
Comment on this question

Request Help

Type your answer below (mathematical expression):

Submit Answer

FIGURE 9.1: ASSISTMENTS PROBLEM #27366


The question asks for the value of the expression : '2(□) + 5' when □ = 3.
In order to solve the question lets break it down in to smaller parts.
Look at the expressions given below. Which expression has the correct substitution for □?

Comment on this question

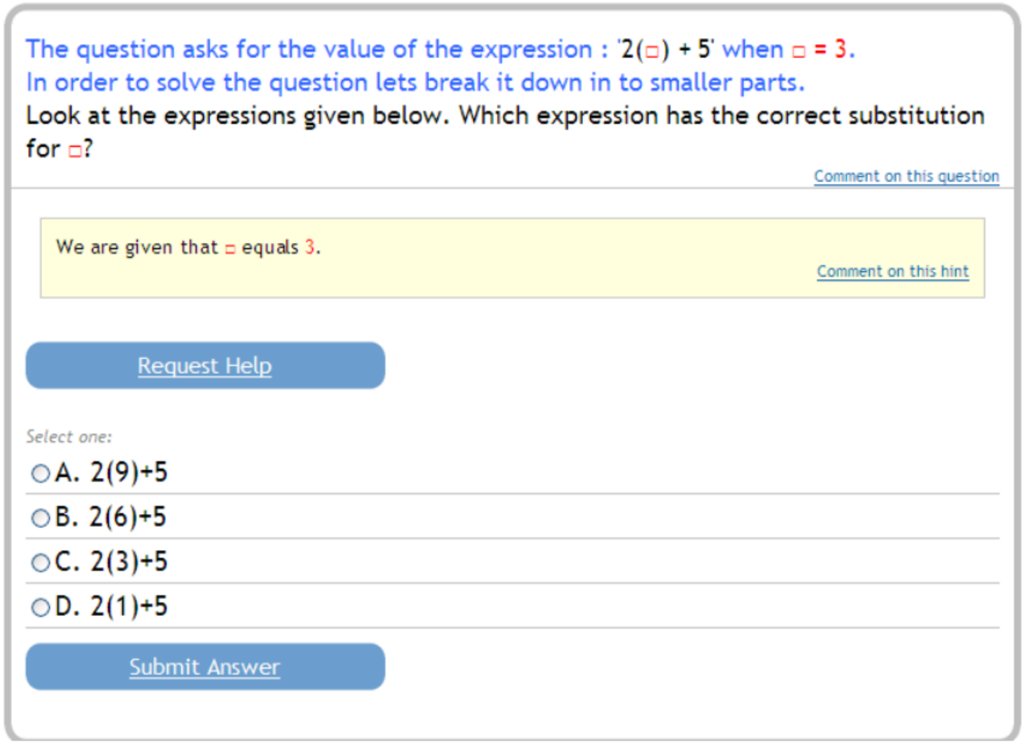
○A. 2(9)+5

○B. 2(6)+5

○C. 2(3)+5

○D. 2(1)+5

FIGURE 9.2: ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM


The question asks for the value of the expression : '2(□) + 5' when □ = 3.
In order to solve the question lets break it down in to smaller parts.
Look at the expressions given below. Which expression has the correct substitution for □?

Comment on this question

We are given that □ equals 3.

Comment on this hint

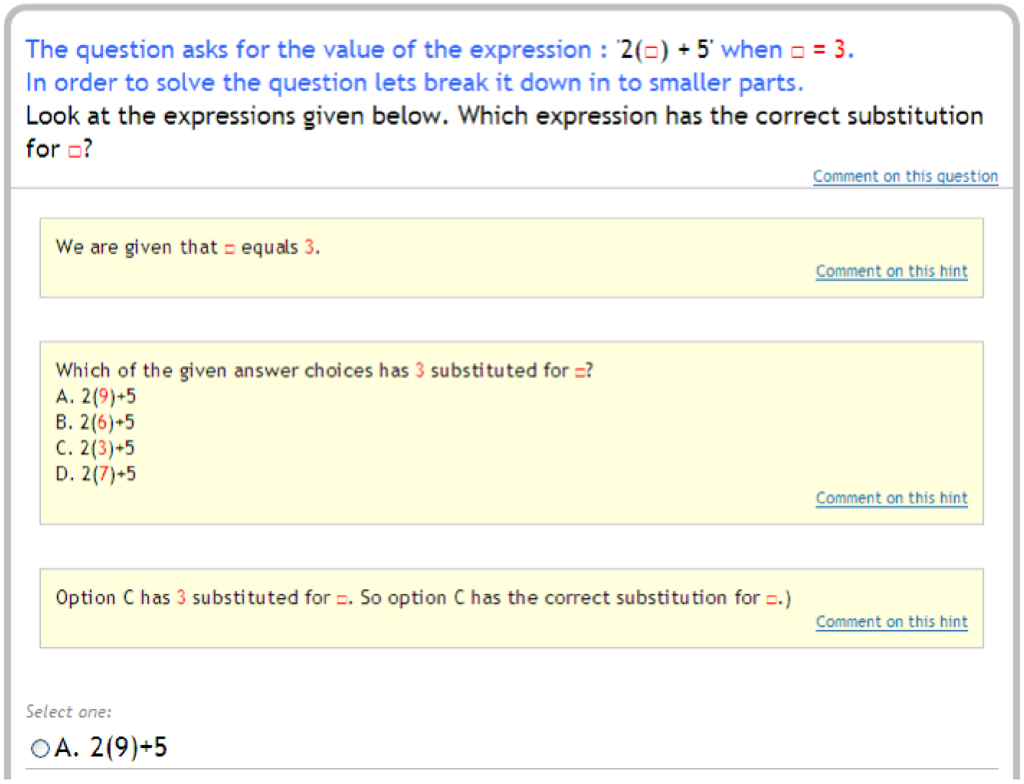
○A. 2(9)+5

○B. 2(6)+5

○C. 2(3)+5

○D. 2(1)+5

FIGURE 9.3: ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM HINT 1



The question asks for the value of the expression : '2(▫) + 5' when ▫ = 3.
In order to solve the question lets break it down in to smaller parts.
Look at the expressions given below. Which expression has the correct substitution for ▫?

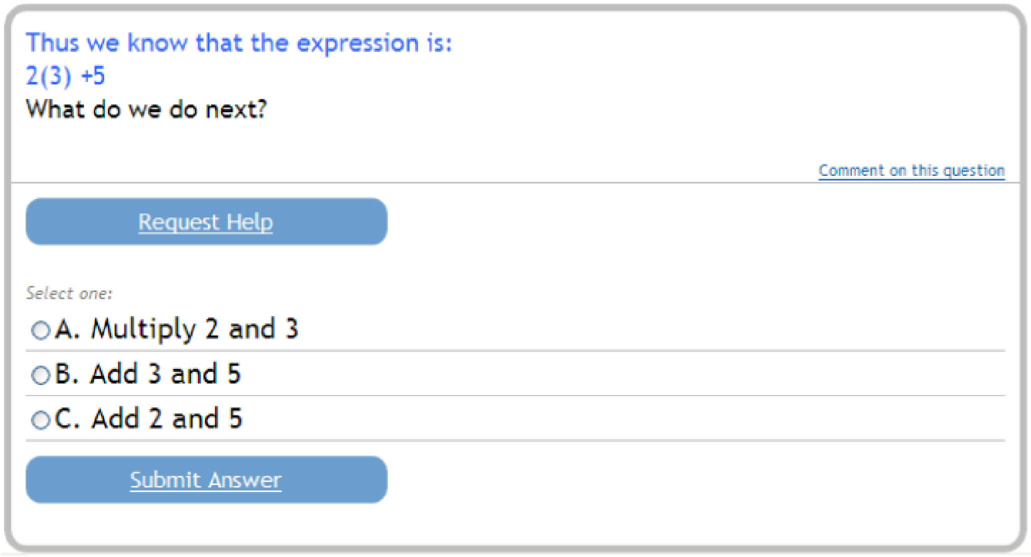
We are given that ▫ equals 3.

Which of the given answer choices has 3 substituted for ▫?
A. 2(9)+5
B. 2(6)+5
C. 2(3)+5
D. 2(7)+5

Option C has 3 substituted for ▫. So option C has the correct substitution for ▫.)

Select one:
○A. 2(9)+5

FIGURE 9.4: AVAILABLE HINTS FOR ASSISTMENTS PROBLEM #27366 FIRST SCAFFOLD PROBLEM



Thus we know that the expression is:
2(3) +5
What do we do next?

Select one:
○A. Multiply 2 and 3
○B. Add 3 and 5
○C. Add 2 and 5

FIGURE 9.5: ASSISTMENTS PROBLEM #27366 SECOND SCAFFOLD PROBLEM

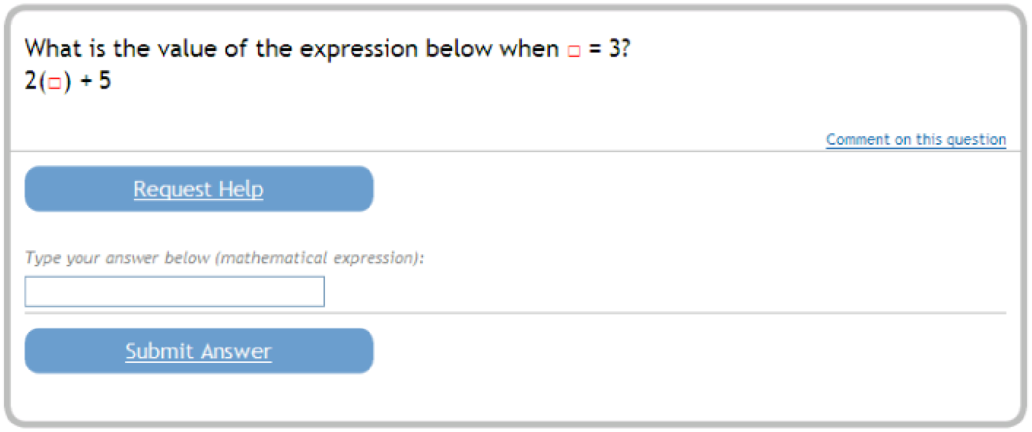

What is the value of the expression below when □ = 3?
2(□) + 5

FIGURE 9.6: ASSISTMENTS PROBLEM #27366 LAST SCAFFOLD PROBLEM

Between November 2008 and February 2009, these students used the ASSISTments program to complete 193,259 main problems on 106 different skills. The actions taken by the students (such as submitting an answer, requesting a hint, starting a scaffold problem, etc.) were logged as was their performance on the problems. There were nine different logged actions listed in Appendix A-1 along with the fields they contain and a description of the action type. As each action performed by each student in this study was logged, while he or she was using the ASSISTments tutoring system, properties of these actions were gathered as well. Appendix A-2 lists all of the data fields gathered for these various actions, the types of actions for which each data field is collected, and a description of each data field. Unfortunately the information collected cannot directly allow detection of whether or not the student is more likely to be learning. Combining and manipulating this information into new derived fields is required in order to detect learning and better model the way in which a student learns.

9.1.2 DETECTION OF LEARNING STATE

The goal of this project is to detect whether or not a student was in the learning state, i.e. when the student was in a state where he or she was more likely to acquire knowledge of a skill. A previous study on gaming by Gong, Beck, Heffernan, and Forbes-Summers (Gong, Beck, Heffernan, & Forbes-Summers, 2010), concluded that students were much less likely to learn if they were gaming then if they were not. Using this data as an initial starting point, this project aimed to generalize this information to find other student behaviors that also affected learning, either positively or negatively and use evidence of these behaviors to detect when a student is more likely to learn a given skill, i.e. when a student is in the learning state.

Because we cannot directly observe students learning, there is no direct information that we can obtain from the ASSISTments tutoring system that measures exactly when a student is in the learning state. Therefore, in order to determine the likelihood that the student is in the learning state the direct information that is given by ASSISTments must be manipulated and combined into new derived fields which provide a better measure of the probability that the student is learning. To decide upon the derived fields to use I looked for characteristics that I believed would either increase or decrease the likelihood that the student is in a state that is likely to lead to learning (i.e. is in the learning state). The derived fields I used were "gameLastX," "gameLastSimilarX," "lastXWrongCurrentRight," "lastXofYWrongCurrentRight," "averageTimeToComplete," "directToAnswerAfterLastHint," and "directToAnswerAfterBottomOutHint". These derived fields are explained in the following sections.

After selecting seven attributes that I felt would aid in determining whether the student was in the learning state I gave each derived field an initial score. I selected the initial score for each derived field to be proportional to the amount that I felt each field would increase or decrease the probability that the student is in the learning state. I used a positive score for fields which would increase the probability the student is in the learning state and a negative score for fields which would decrease the probability the student is in the learning state. Some fields, such as "averageTimeToComplete," I discretized and set different amounts of increase based on what range the value of the derived field fell into.

Each derived field was also given a weight. The weight determined the amount that the score for each field affected the probability that the student is in the learning state. Initially the weights of each field were one meaning that each field equally affected the probability that the student was learning. These weights were adjusted on each time step of the training process, as explained in section 9.1.2.8 below. A weight near zero indicates that the derived field is relatively unimportant in determining whether or not the student is in the learning state. A positive weight indicates that the derived field affects the likelihood that the student is in the learning state as or initial assumptions indicated (the higher the weight the more the derived field affects this likelihood). As an example, if originally it was assumed that a derived field would increase the likelihood the student was learning and the weight was positive, this indicates that the initial assumption was correct and that this derived field indicates that the student is more likely to be in the learning state. If originally it was assumed that a derived field would decrease the likelihood the student was learning and the weight was positive, this indicates that the initial assumption was correct and that this derived field indicates that the student is less likely to be in the learning state. A negative weight indicates that the derived field inversely of our initial assumptions. As an example, if originally it was assumed that a derived field would increase the likelihood the student was learning and the weight was negative, this indicates that the initial assumption was incorrect and that this derived field indicates that the student is actually less likely to be in the learning state. If originally it was assumed that a derived field would decrease the likelihood the student was learning and the weight was negative, this indicates that the initial assumption was incorrect and that this derived field indicates that the student is more likely to be in the learning state. The summation of the scores for each of the fields multiplied by their given weights determined the value of the student's raw learning state score for this problem. The logit transform of this raw learning state score gave the initial estimate for the probability that the student is in the learning state which is used in the knowledge tracing process.

TABLE 9.1: SAMPLE DERIVED FIELD VALUES SCORES AND WEIGHTS

Derived Field	**Value**	**Score**	**Weight**
gameLastX	FALSE	0	0.8
gameLastSimilarX	TRUE	-4	0.9
lastXWrongCurrentRight	FALSE	0	1.5
lastXofYWrongCurrentRight	TRUE	1	1.5
averageTimeToComplete	0	2	1.2
directToAnswerAfterLastHint	1	1	1
directToAnswerAfterBottomOutHint	0	2	1

As an example, assume a student was given an ASSISTments problem. On this problem the values of the derived fields are shown in Table 9.1. The student is given a score for each derived field based on the value of the field as described in sections 9.1.2.1 - 9.1.2.7. The scores corresponding to the values for each of the derived fields are shown in Table 9.1. Each field had a weight which was the same for every student and every problem. Initially this value was one, meaning that each field played an equal role in determining the learning state score for a given student on a given problem. This weight was updated each iteration, as described in section 9.1.2.8 below. The higher the product of the score and weight for a field, the more it affects the probability that the student is in the learning state. The raw learning state score was calculated by taking the summation of the values of each field multiplied by its given weight. For this example, assume that the weights of each of the derived fields are equal to the weights given in Table 9.1. The equation to compute the raw learning state score for this problem is as follows:

$$\begin{aligned}
&Raw\ Learning\ State\ Score \\
&\quad = \big((score\ of\ gameLastX) * (weight\ of\ gameLastX)\big) \\
&\quad + \big((score\ of\ gameLastSimilarX) * (weight\ of\ gameLastSimilarX)\big) \\
&\quad + \big((score\ of\ lastXWrongCurrentRight) * (weight\ of\ lastXWrongCurrentRight)\big) \\
&\quad + \big((score\ of\ lastXofYWrongCurrentRight) \\
&\quad * (weight\ of\ lastXofYWrongCurrentRight)\big) \\
&\quad + \big((score\ of\ averageTimeToComplete) * (weight\ of\ averageTimeToComplete)\big) \\
&\quad + \big((score\ of\ directToAnswerAfterLastHint) \\
&\quad * (weight\ of\ directToAnswerAfterLastHint)\big) \\
&\quad + \big((score\ of\ directToAnswerAfterBottomOutHint) \\
&\quad * (weight\ of\ directToAnswerAfterBottomOutHint)\big)
\end{aligned}$$

Inputting the given scores and weights into this equation outputs the raw learning state score for this student on this problem.

$$\begin{aligned}
&Raw\ Learning\ State\ Score \\
&\quad = (0) * (0.8) + (-4) * (0.9) + (0) * (1.5) + (1) * (1.5) + (2) * (1.2) + (1) * (1) \\
&\quad + (2) * (1) = 3.3
\end{aligned}$$

The logit transform of this raw learning state score is taken to obtain the probabilistic learning state score using the equation $\frac{e^x}{1+e^x}$ where x is the raw learning state score.

$$P(LS) = \frac{e^x}{1 + e^x} = \frac{e^{3.3}}{1 + e^{3.3}} = 0.964428811$$

Therefore, the probabilistic learning state score for this example is 0.964428811 or 96.4%, which indicates there is a 96.4% chance the student is in the learning state and would then be used in the knowledge tracing process as described below in section 9.1.2.8. It is important to note that the values given for each of the features will be modified by the learning procedure, described in section 9.1.2. Thus, these values serve as initial estimates and will be modified over time, so it is not crucial if they are somewhat inaccurate.

9.1.2.1 gameLastX

This derived field signifies whether the student was "gaming" on the past X questions including the current question, where X is the number of questions to look at. The X value I used for this statistic was two, meaning that this feature considered both the current and previous question. Computing this field returned a boolean value. If the student was gaming on both this question and the previous question, then the value would be TRUE. If the student was not gaming on either of the two questions, then the value for this derived field would be FALSE.

In order to determine whether the student was gaming I used the same rules used by Gong, Beck, Heffernan, and Forbes-Summers in their study on the impacts of gaming on learning (Gong, Beck, Heffernan, & Forbes-Summers, 2010). This study used three criteria to detect gaming, "Rapid Guessing," "Rapid Response," and "Repeatedly Bottom-out Hinting". A student was determined to be "Rapid Guessing" if, while answering a question, he or she submitted two different answers to the same question in less than two seconds and he or she did this on two consecutive questions. A student was determined to be giving a "Rapid Response" if, after receiving a hint, he or she performed any action before the amount of time required for the student to read the hint at 400 words per minute or if, initially, the student performs an action before the amount of time required for the student to read the problem (at the same 400 words per minute reading rate— as a reference, the average college student reads at 250 words per minute). A student was determined to be "Repeatedly Bottom-out Hinting" if he or she requested the last hint, which displays the answer, on three successive problems.

Gong, Beck, Heffernan, and Forbes-Summers concluded that gaming reduces the likelihood that the student is learning to near zero (Gong, Beck, Heffernan, & Forbes-Summers, 2010); therefore, gaming is a good detector of a student who is not in the learning state. Because the effect of gaming on learning is so great, a large negative score of -4 is weighted and added to the student's raw score for the current problem if this student is determined to be gaming. If the student is not gaming then the raw score for the current problem remains unchanged.

As an example, assume a student is determined to be gaming on the first two of three problems as seen in Table 9.2. When the first problem is processed the student is determined to be gaming. At this point we have only processed one problem so the student could not have been determined to be gaming on the past two problems. The value of the derived field "gameLastX" for problem one would be FALSE. When the second problem is processed the student is again determined to be gaming. At this point we have processed two problems and the student has been determined to be gaming in both problems. The value of the derived field "gameLastX" for problem two would be TRUE and -4 multiplied by the weight of the derived field would be added to the raw score for problem two. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem two would be decreased by 2. Since this score is negative it decreases the likelihood of the student being in the learning state. When the third problem is processed the student is determined not to be gaming. Because the student was not gaming in this problem the student was only gaming in one of the past two problems. The value of the property "gameLastX" for problem three would be FALSE.

TABLE 9.2: SAMPLE GAMELASTX DATA

Problem Number	Was Student Gaming?	Value of gameLastX	Reason	Score
1	Yes	FALSE	Not enough prior data	0
2	Yes	TRUE	Student was gaming on the current and last problems	-4
3	No	FALSE	Student was not gaming	0

9.1.2.2 gameLastSimilarX

This derived field is similar to the "gameLastX" field as it looks at whether the student was gaming in prior problems but looks only at the last X *similar* problems including this problem. A problem is similar to the current problem if the skills for the problems are the same. I chose to additionally include this derived field because gaming on similar questions in the past likely indicates that the student does not know this skill and is not interested in learning it. For this project, I chose to use an X value of one, and therefore, this field is looking at whether or not the student is gaming on this problem only (for this project, similarity was not important for this feature, but future researchers should consider investigating similarity). Again, since gaming is a good detector of a student who is not in the learning state and the effect of gaming on learning is so great, a large negative score of -4 is weighted and added to the student's raw score for the current problem if the student is determined to be gaming. If the student is not gaming the raw score for the current problem remains unchanged.

Again as an example, assume a student is determined to be gaming on the first two of three similar problems as seen in Table 9.3. Other non-similar problems may be dispersed in between the three problems but since only the similar problems are looked at whether or not the student was gaming on the non-similar problems would not affect the value of the derived field “gameLastSimilarX” for these three problems. The value of the derived field “gameLastSimilarX” for questions one and two would be TRUE because, at each step, the student was determined to be gaming in the previous X questions, in this case the current question since the value of X is one, and -4 multiplied by the weight of the derived field would be added to the raw scores for problems one and two. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw scores for problems one and two would be decreased by 2. Since this score is negative it decreases the likelihood of the student being in the learning state. When the third question is processed the student was determined not to be gaming. Because the student was not gaming on the previous X questions, again in this case the current question, the value of the derived field “gameLastSimilarX” for question three would be False.

TABLE 9.3: SAMPLE GAMELASTSIMILARX DATA

Problem Number	Was Student Gaming?	Value of gameLastSimilarX	Reason	Score
1	Yes	TRUE	Student was gaming	-4
2	Yes	TRUE	Student was gaming	-4
3	No	FALSE	Student was not gaming	0

9.1.2.3 lastXWrongCurrentRight

This derived field signifies whether the student answered the current question correctly but answered the past X questions with the same skill incorrectly, where X is the number of questions to look at. The X value I used for this statistic was two meaning that both the current question and the previous two questions would be considered. Computing this field returned a boolean value. If the value was TRUE, then the student answered the current question correctly and the two previous questions incorrectly. If the student either answered this question incorrect or answered this question correctly and answered at least one of the previous two questions correctly, then the value of this derived field would be FALSE.

I chose to include this derived field because we are not trying to determine whether the student knows the skill but whether he or she is more likely to be learning the skill (i.e. the student is in the learning state) it at the current time step. If the student answers prior questions with the same skill right the student is more likely to already know the skill. If the student answers this question wrong then the student is more likely to not know and not have learned the skill at this time step. If the student, however, answers prior questions with the same skill wrong but the current question right then the student is more likely to have learned the skill at this time step. Because the student could either have guessed the correct answer on the current or previous questions or accidentally answered the previous or current questions wrong, a small score of 1 multiplied by the weight of this derived field was added to the raw score for the current problem if the value of the field was TRUE. If the value of the derived field was FALSE the raw score for the current problem remained unchanged.

As an example, assume a student answered the first two questions in a problem set incorrectly but then proceeded to answer the next two problems correctly as seen in Table 9.4. When the first two problems are processed the student answered both problems incorrectly, and therefore, the value of the derived field “lastXWrongCurrentRight” for each problem would be FALSE. When the third problem is processed the student answered the problem correctly and therefore the prior two problems are viewed. If there were not two prior problems to view, then the value of the derived field “lastXWrongCurrentRight” would be FALSE, however in this example, there is data for two prior problems available. Since the prior two problems were answered incorrectly the value of the derived field “lastXWrongCurrentRight” for problem three would be TRUE and 1 multiplied by the weight of the derived field would be added to the raw score for problem three. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem three would be increased by 0.5. Since this score is positive it increases the likelihood of the student being in the learning state. When the forth problem is processed the student answered the problem correctly. Because the student answered the problem correctly and therefore the prior two questions are viewed. Since the prior problem was answered correctly the value of the derived field "lastXWrongCurrentRight" for problem three would be FALSE.

TABLE 9.4: SAMPLE LASTXWRONGCURRENTRIGHT DATA

Problem Number	Was Answer Correct?	Value of lastXWrongCurrentRight	Reason	Score
1	No	FALSE	Student answered this problem incorrectly	0
2	No	FALSE	Student answered this problem incorrectly	0
3	Yes	TRUE	Student answered this problem correctly and last two questions incorrectly	1
4	Yes	FALSE	Student answered last problem correctly	0

9.1.2.4 lastXofYWrongCurrentRight

This derived field signifies whether the student answered the current question correctly but answered X of the past Y questions with the same skill incorrectly, where Y is the number of previous questions to look at and X is the number of these Y questions that the student answered incorrectly. The Y value I used for this statistic was three and the X value was 1 meaning that both the current question and the previous three questions would be considered. Computing this field returned a boolean value. If the value was TRUE, then the student answered the current question correctly and at least one of the three previous questions incorrectly. If the student either answered this question incorrect or answered this question correctly and answered all of the previous three questions correctly, then the value of this derived field would be FALSE.

Similarly to the previous derived field, I chose to include this field because we are not trying to determine whether the student knows the skill but whether he or she is more likely to be learning the skill (i.e. the student is in the learning state) it at the current time step. If the student answers prior questions with the same skill right the student is more likely to already know the skill. If the student responds incorrectly to this question then the student is more likely to not know and not have learned the skill at this time step. If the student, however, answers prior questions with the same skill wrong but the current question right then the student is more likely to have learned the skill at this time step. However, this derived field, unlike "lastXWrongCurrentRight," still considers situations where the student might have guessed an answer correctly on some of the prior questions. Because the student could either have guessed the correct answer on the current question or accidentally answered some of the previous questions or the current question incorrectly, a small score of 1 multiplied by the weight of this derived field was added to the raw score for the current problem if the value of the field was TRUE. If the value of the derived field was FALSE the raw score for the current problem remained unchanged.

As an example, assume a student answered the first two questions in a problem set incorrectly but then proceeded to answer the next two problems correctly as seen in Table 9.5. When the first two problems are processed the student answered both problems incorrectly, and therefore, the value of the derived field "lastXofYWrongCurrentRight" for each problem would be FALSE. When the third problem is processed the student answered the problem correctly and therefore the prior three problems are viewed. Since there are not thee prior problems to view, the value of the derived field "lastXofYWrongCurrentRight" would be FALSE for this problem. When the forth problem is processed the student answered the problem correctly. Since only one the prior three problems were answered correctly the value of the derived field "lastXofYWrongCurrentRight" for problem four would be TRUE and 1 multiplied by the weight of the derived field would be added to the raw score for problem four. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem four would be increased by 0.5. Since this score is positive it increases the likelihood of the student being in the learning state.

TABLE 9.5: SAMPLE LASTXOFYWRONGCURRENTRIGHT DATA

Problem Number	Was Answer Correct?	Value of lastXofYWrongCurrentRight	Reason	Score
1	No	FALSE	Student answered this problem incorrectly	0

2	No	FALSE	Student answered this problem incorrectly	0
3	Yes	FALSE	Not enough prior data	0
4	Yes	TRUE	Student answered this problem correctly and only one of the past three other problems correctly	1

9.1.2.5 averageTimeToComplete

This derived field signifies within how many time offsets the student took to submit his or her answer from the average time to complete this problem. The offset value I used for this statistic was 5000ms. There was not always enough reference points for each individual question so simply using the average time to complete for the problem was not always a good indicator of how long the problem took to complete on average. Therefore, if the program had encountered this problem at least three times before, it uses the average time to complete for this problem; otherwise it averages the time to complete for all problems with the time to complete for the current problem. Computing this field returned one of five values: -2, -1, 0, 1, and 2. If the student submitted an answer between 5,000ms less than the average time to complete and 5,000ms more than the average time to complete, then the value of this derived field would be 0. If the student submitted an answer between 10,000ms and 5,000ms less than the average time to complete, then the value of this derived field would be -1. If the student submitted an answer more quickly than 10,000ms less than the average time to complete, then the value of this derived field would be -2. If the student submitted an answer between 5,000ms and 10,000ms greater than the average time to complete, then the value of this derived field would be 1. Finally, if the student submitted an answer more slowly than 10,000ms greater than the average time to complete, then the value of this derived field would be 2.

I chose to include this derived field because the time it takes a student to respond can be indicative of certain behaviors. If the student takes a significantly longer time to submit an answer than the average then it is likely an indicator that the student is “goofing off” or not paying attention. A student who is not engaged is unlikely to be in the learning state. If a student take a much shorter time to submit an answer than average it is a possible indicator that the student is cheating. A student who is cheating is unlikely to be in the learning state. If the student took an average amount of time to submit an answer to the problem it is ideal.

I used a different score for each possible value of the derived field. If the student submitted an answer within one offset from the average, value of 0, then the student has answered within the ideal allotment of time and a score of 2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets quicker than the average time to complete, value of -1, then the student is more likely to either be cheating or already knows the material. In either case the student is less likely to be in the learning state, therefore, a score of -1 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer more quickly than two offsets less than the average time to complete, value of -2, then the student is most likely cheating and is not likely to be in the learning state. Therefore, a score of -2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets more slowly than the average time to complete, value of 1, then the student is more likely to be "goofing off" which would decrease the likelihood the student is in the learning state. However, since a slightly longer time may also indicate that the student is simply struggling with a skill which would increase the likelihood the student is in the learning state, therefore, a small score of 1 multiplied by the weight of this derived field was added to the raw score for the current problem. Finally, if the student submitted an answer more slowly than two offsets greater than the average time to complete, value of 2, then the student is most likely "goofing off," but since there is still the possibility that the student is really struggling, the raw score for the current problem remains unchanged.

As an example, assume a student answered three problems in a problem set as seen in Table 9.6. When the first problem is processed, since the problem has only been seen twice the program computes the average time to complete by taking the average of the average time the previous two students took to answer the problem and the average time all students took to answer all previous problems. The average time to complete was determined to be 15,000ms. Since the student answered problem one in 8,000ms which is shorter than the average time to complete minus one offset (10,000ms) but longer than the average time to complete minus two offsets (5,000ms). Therefore, the value of this derived field for problem one would be -1 and a score of -1 multiplied by the weight of the derived field would be added to the raw score for problem one. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem one would be decreased by 0.5. Since this score is negative it decreases the likelihood of the student being in the learning state. When the second problem is processed, since the problem has been previously seen three times, the program computes the average time to complete by taking the average time the previous ten students took to answer this problem. The average time to complete was determined to be 11,000ms. The student answered problem two in 20,000ms which is longer than the average time to complete plus one offset (16,000ms) and shorter than the average time to complete plus two offsets (21,000ms). Therefore, the value of this derived field for problem two would be 1 and a score of 1 multiplied by the weight of the derived field would be added to the raw score for problem two. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem one would be increased by 0.5. Since this score is positive it increases the likelihood of the student being in the learning state. When the third problem is processed, since the problem has been previously seen ten times, the program computes the average time to complete by taking the average time the previous ten students took to answer this problem. The average time to complete was determined to be 12,000ms. The student answered problem three in 26,000ms which is longer than the average time to complete plus two offsets (22,000ms). Therefore, the value of this derived field for problem three would be 2 and the raw score for problem three would remain unchanged.

TABLE 9.6: SAMPLE AVERAGETIMETOCOMPLETE DATA

Problem Number	Times Problem Seen	Average Time (ms)	Student Time (ms)	Value of averageTimeTo Complete	Reason	Score
1	2	15,000	8,000	-1	Student time is less than one but more than two offsets from the average time	-1
2	3	11,000	20,000	1	Student time is more than one but less than two offsets from the average time	1
3	10	12,000	26,000	2	Student time is more than two offsets from the average time	0

9.1.2.6 directToAnswerAfterLastHint

This derived field signifies within how many time offsets from the suggested time the student took to submit his or her answer after receiving his or her last hint. The suggested value I used for this statistic was 10,000ms and offset value I used was 5000ms. Computing this field returned one of four values: -1, 0, 1, and 2. If the student has not received any hints the value for this derived field would be 0. If the student submitted an answer between 5,000ms less than the suggested time and 5,000ms more than the suggested time, then the value of this derived field would be 0. If the student submitted an answer between 10,000ms and 5,000ms less than the suggested time, then the value of this derived field would be -1. If the student submitted an answer between 5,000ms and 10,000ms greater than the suggested time, then the value of this derived field would be 1. Finally, if the student submitted an answer more slowly than 10,000ms greater than the suggested time, then the value of this derived field would be 2.

I chose to include this derived field because the time it takes a student to answer the problem after receiving a hint can be indicative of certain behaviors. If the student takes a significantly longer time to submit an answer after his or her last hint than the suggested time then it is likely an indicator that the student is "goofing off" or not paying attention. An unengaged student is unlikely to be in the learning state. If a student take a much shorter time to submit an answer than the suggested time it is likely an indicator that the student either recognized the answer from the hint right away or is just writing down the answer that the hint told the student to write down. In either case the student is likely not thinking about what the hint is explaining and is not in the learning state. If the student took the suggested amount of time to submit an answer after his or her last hint it is ideal because it is likely that the student is trying to figure out what the hint is explaining.

I used a different score for each possible value of the derived field. If the student did not request any hints, value of 0, a score of 2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer within one offset from the suggested time, value of 0, then the student has answered within the ideal allotment of time and a score of 2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets quicker than the suggested time, value of -1, then the student is more likely to either have recognized the answer immediately from the hint or be simply inputting the answer that was given by the hint. In either case the student is less likely to be in the learning state, therefore, a score of -1 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer more quickly than two offsets less than the suggested time, value of -2, then the student is most likely to either have recognized the answer immediately from the hint or be simply inputting the answer that was given by the hint. In either case the student is unlikely to be in the learning state, therefore, a score of -2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets more slowly than the suggested time, value of 1, then the student is more likely to be "goofing off" which would decrease the likelihood the student is in the learning state. However, since a slightly longer time may also indicate that the student is simply struggling to figure out the hint which would increase the likelihood the student is in the learning state, therefore, a small score of 1 multiplied by the weight of this derived field was added to the raw score for the current problem. Finally, if the student submitted an answer more slowly than two offsets greater than the suggested time, value of 2, then the student is most likely "goofing off," but since there is still the possibility that the student is really struggling to figure out the hint, the raw score for the current problem remains unchanged.

As an example, assume a student answered four problems in a problem set as seen in Table 9.7. When the first problem is processed, the student did not request a hint, therefore, the value of this derived field for problem one would be 0 and a score of 2 multiplied by the weight of the derived field would be added to the raw score for problem one. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem one would be increased by 1. Since this score is positive it increases the likelihood of the student being in the learning state. When the second problem is processed, the student submitted an answer 5,000ms after receiving the hint which is equal to the suggested time minus one offset (5,000ms), therefore, the value of this derived field for problem two would be 0 and a score of 2 multiplied by the weight of the derived field would be added to the raw score for problem two. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem two would be increased by 1. Since this score is positive it increases the likelihood of the student being in the learning state. When the third problem is processed, the student answered 21,000ms after requesting the hint which is longer than the suggested time plus two offsets (20,000ms). Therefore, the value of this derived field for problem three would be 2 and the raw score for problem three would remain unchanged. When the fourth problem is processed, since the student requested two hints only the last hint is considered, student answered problem four 3,000ms after receiving the last hint which is shorter than the suggested time minus one offset (5,000ms) but longer than the suggested time minus two offsets (0ms). Therefore, the value of this derived field for problem four would be -1 and a score of -1 multiplied by the weight of the derived field would be added to the raw score for problem four. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem four would be decreased by 0.5. Since this score is negative it decreases the likelihood of the student being in the learning state.

TABLE 9.7: SAMPLE DIRECTTOANSWERAFTERLASTHINT DATA

Problem Number	No. Requested Hints	Time After Hint (ms)	Value of directToAnswer AfterLastHint	Reason	Score
1	0	N/A	0	Student did not request any hints	2
2	1	5,000	0	Student time is within one offset from the suggested time	2
3	1	21,000	2	Student time is more than two offsets from the suggested time	0
4	2	23,000 (first) / 3,000 (last)	-1	Student time is less than one but more than two offsets from the suggested time	-1

9.1.2.7 directToAnswerAfterBottomOutHint

If a student requested the “bottom out hint,” i.e. the last hint available to the student for the current main or scaffold problem which gives the student the correct answer, this derived field signifies within how many time offsets from the suggested time the student took to submit his or her answer after receiving this “bottom out hint”. The suggested value I used for this statistic was 10,000ms and offset value I used was 5000ms. Computing this field returned one of five values: -2, -1, 0, 1, and 2. If the student did not request the “bottom out hint” the value for this derived field would be 0. If the student submitted an answer between 5,000ms less than the suggested time and 5,000ms more than the suggested time, then the value of this derived field would also be 0. If the student submitted an answer between 10,000ms and 5,000ms less than the suggested time, then the value of this derived field would be -1. If the student submitted an answer more quickly than 10,000ms less than the suggested time, then the value of this derived field would be -2. If the student submitted an answer between 5,000ms and 10,000ms greater than the suggested time, then the value of this derived field would be 1. Finally, if the student submitted an answer more slowly than 10,000ms greater than the suggested time, then the value of this derived field would be 2.

I chose to include this derived field because the time it takes a student to answer the problem after receiving the "bottom out hint" can be indicative of certain behaviors. If the student takes a significantly longer time to submit an answer after his or her last hint than the suggested time then it is likely an indicator that the student is "goofing off" or not paying attention. An unengaged student is unlikely to be in the learning state. If a student take a much shorter time to submit an answer than the suggested time it is likely an indicator that the student is just writing down the answer that the hint told the student to write down and is likely not thinking about what the hint is explaining and is not in the learning state. If the student took the suggested amount of time to submit an answer after his or her last hint, it is ideal because it is likely that the student is trying to figure out what the hint is explaining.

I used a different score for each possible value of the derived field. If the student did not request the "bottom out hint," value of 0, a score of 2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer within one offset from the suggested time, value of 0, then the student has answered within the ideal allotment of time and a score of 2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets quicker than the suggested time, value of -1, then the student is more likely to simply inputting the answer that was given by the hint. In this case the student is less likely to be in the learning state, therefore, a score of -1 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer more quickly than two offsets less than the suggested time, value of -2, then the student is most likely simply inputting the answer that was given by the hint. In this case the student is unlikely to be in the learning state, therefore, a score of -2 multiplied by the weight of this derived field was added to the raw score for the current problem. If the student submitted an answer between one and two offsets more slowly than the suggested time, value of 1, then the student is more likely to be “goofing off” which would decrease the likelihood the student is in the learning state. However, since a slightly longer time may also indicate that the student is simply struggling to figure out the hint which would increase the likelihood the student is in the learning state, therefore, a small score of 1 multiplied by the weight of this derived field was added to the raw score for the current problem. Finally, if the student submitted an answer more slowly than two offsets greater than the suggested time, value of 2, then the student is most likely “goofing off,” but since there is still the possibility that the student is really struggling to figure out the hint, the raw score for the current problem remains unchanged.

As an example, assume a student answered four problems in a problem set as seen in Table 9.8. When the first problem is processed, the student did not request a hint, therefore, the value of this derived field for problem one would be 0 and a score of 2 multiplied by the weight of the derived field would be added to the raw score for problem one. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem one would be increased by 1. Since this score is positive it increases the likelihood of the student being in the learning state. When the second problem is processed, the student requested a hint but did not request the “bottom out hint,” therefore, the value of this derived field for problem two would be 0 and a score of 2 multiplied by the weight of the derived field would be added to the raw score for problem two. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem two would be increased by 1. Since this score is positive it increases the likelihood of the student being in the learning state. When the third problem is processed, the student answered 21,000ms after requesting the “bottom out hint” which is longer than the suggested time plus two offsets (20,000ms). Therefore, the value of this derived field for problem three would be 2 and the raw score for problem three would remain unchanged. When the fourth problem is processed, the student answered problem four 3,000ms after receiving the “bottom out hint” which is shorter than the suggested time minus one offset (5,000ms) but longer than the suggested time minus two offsets (0ms). Therefore, the value of this derived field for problem four would be -1 and a score of -1 multiplied by the weight of the derived field would be added to the raw score for problem four. Initially the weight of the field was one, but for example, if the weight was 0.5 then the raw score for problem four would be decreased by 0.5. Since this score is negative it decreases the likelihood of the student being in the learning state.

TABLE 9.8: SAMPLE DIRECTTOANSWERAFTERBOTTOMOUTHINT DATA

Problem Number	No. Requested Hints	Time After Hint (ms)	Value of directToAnswerAfterBottomOutHint	Reason	Score
1	0	N/A	0	Student did not request any hints	2
2	1 of 3	5,000	0	Student did not request bottom out hint	2
3	5 of 5	21,000	2	Student time is more than two offsets from the suggested time	0
4	2 of 2	3,000	-1	Student time is less than one but more than two offsets from the suggested time	-1

9.1.2.8 Learning State Model

In order to better model the student learning process, the current knowledge tracing model described in section 7.2, was modified to include the value of learning state, LS, i.e. the probability the student was in a state where he or she was more likely to acquire knowledge of a skill, which was calculated using the derived fields as explained in section 9.1.2. Integrating the learning state node into the current knowledge tracing model produced the updated model shown in Figure 9.7.

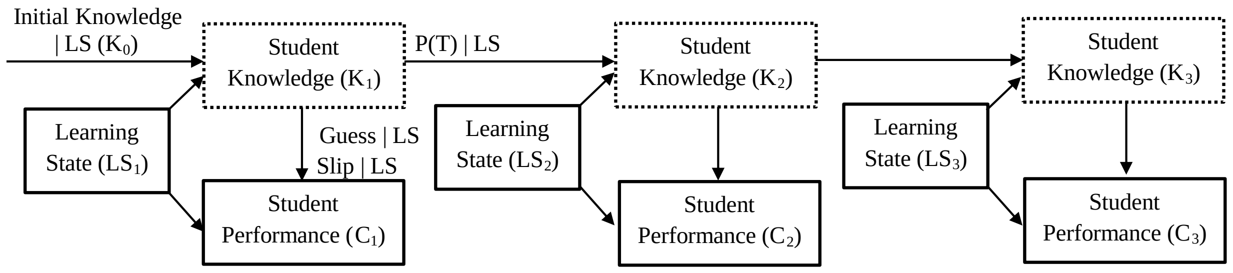

Initial Knowledge | LS (K_0)
Student Knowledge (K_1)
P(T) | LS
Student Knowledge (K_2)
Student Knowledge (K_3)
Learning State (LS_1)
Guess | LS
Slip | LS
Learning State (LS_2)
Learning State (LS_3)
Student Performance (C_1)
Student Performance (C_2)
Student Performance (C_3)

FIGURE 9.7: KNOWLEDGE TRACING MODEL WITH LEARNING STATE

Whether or not the student is in the learning state at a given time step t, LS_t, affects both the student's knowledge and the student's performance through the use of the model parameters. Each of the parameters now has two values, one for when the student is in the learning state and one for when the student is not in the learning state, which are both estimated in the same manner as was described in section 9.1.2. For example, the model parameter slip will actually be one of two parameters, one for the likelihood that the student answers the problem incorrectly given that he or she knows the skill and that he or she is in the learning state, $P(C_t \mid K_t, LS_t =$

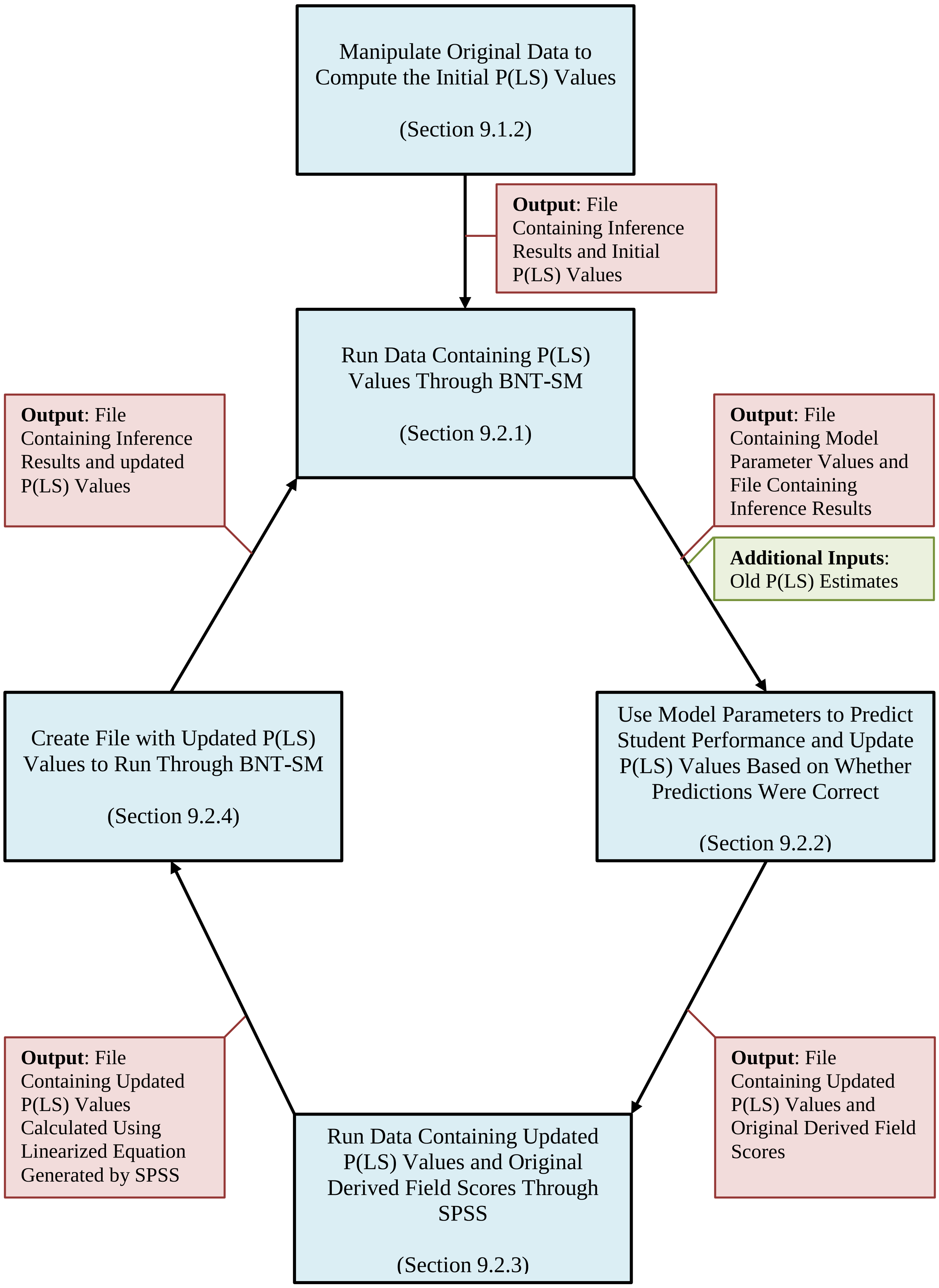
TRUE), and one for the likelihood that the student answers the problem incorrectly given that he or she knows the skill and that he or she is not in the learning state, $P(C_t \mid K_t, LS_t = FALSE)$. Each of the four model parameters, guess, slip, initial knowledge, and probability of transition will similarly have two values.

9.2 INFRASTRUCTURE

As stated in section 8, the goal of this process is to improve the current knowledge tracing model, which uses a constant rate of learning across all students and problems and defies both experience and evidence of how students acquire knowledge. In order to create a more accurate model of the student learning process this project utilized a contextualized, dynamic learning rate in the knowledge tracing model, where the learning rate is conditioned on certain characteristics which have been shown to influence student learning, such as whether or not a student is gaming. A more accurate model of the student learning process gives better information about how and why students learn providing a better understanding of the student learning process. This new information about student learning can be used for a wide range of tasks, such as adapting tutoring systems, such as cognitive tutor systems, in order to help direct students into certain states which have been shown to lead to learning, providing more accurate information about the student learning process to the cognitive psychology community which can be used to improve classroom teaching methods, pinpointing practice problems that have statistically been seen to produce the greatest gain in learning (Feng, Heffernan, & Beck, 2009), or improving the tailoring of practice problems to students based on the way they learn, among others.

In order to improve the current knowledge tracing model through the use of contextualized learning rates, I created a new model, described in section 9.1.2, which utilized a secondary input node, other than the student's prior knowledge estimate, called the Learning State (LS) to update the estimate of the student's current knowledge, K_t. In order to fit the generated model to the data, I used the process described by 9.1.2. Initially I manipulated the original data, which is described in section 9.1.1, to create a learning state score P(LS), i.e. the probability that the student is in a state that increases the likelihood of student learning, for each student on every problem he or she completed using the process described in section 9.1.2.

FIGURE 9.8: PROCESS TO INFER MODEL PARAMETERS AND ESTIMATE P(LS) VALUES


Manipulate Original Data to Compute the Initial P(LS) Values
(Section 9.1.2)
Output: File Containing Inference Results and Initial P(LS) Values
Run Data Containing P(LS) Values Through BNT-SM
(Section 9.2.1)
Output: File Containing Inference Results and updated P(LS) Values
Output: File Containing Model Parameter Values and File Containing Inference Results
Additional Inputs: Old P(LS) Estimates
Create File with Updated P(LS) Values to Run Through BNT-SM
(Section 9.2.4)
Use Model Parameters to Predict Student Performance and Update P(LS) Values Based on Whether Predictions Were Correct
(Section 9.2.2)
Output: File Containing Updated P(LS) Values Calculated Using Linearized Equation Generated by SPSS
Output: File Containing Updated P(LS) Values and Original Derived Field Scores
Run Data Containing Updated P(LS) Values and Original Derived Field Scores Through SPSS
(Section 9.2.3)

I then took these initial learning state scores and created an iterative process, which I ran for fifty iterations, in order to infer the values of the model parameters and the learning state scores. This iterative process had four distinct steps. In the first step, described in section 9.2.1, the file which contained the student learning state data was fed into the Bayes Net Toolkit for Student Models (BNT-SM) system, which trained the model and fit the model parameters to the data. Based on the values of the model parameters generated by the BNT-SM process, the second step, described in section 9.2.2, used a program I created to predict whether the student answered the problem correctly and used some heuristics I developed to update the P(LS) estimates to be more consistent with student performance. The original derived field scores, calculated following the process in section 9.1.2, and the updated P(LS) estimates, generated by the prior step, were then given as input to the SPSS Statistics system in the next step, described in section 9.2.3, in order to derive a linearized equation used to generate the P(LS) estimate from the derived field scores. In the final step of the iteration, described in section 9.2.4, these generated P(LS) estimates were used to predict whether or not the student was in the Learning State and the resulting predictions were run back through the BNT-SM system starting the iterative process over again.

9.2.1 BNT-SM

Dynamic Bayesian Networks (DBNs) provide a valuable effective method of representing and evaluating latent variables, such as student knowledge, in time series data and is a common technique for modeling student data. The Bayes Net Toolkit (BNT) (Murphy, 2007) is a general purpose Bayes Net package, implemented using Matlab, which supports a number of inference algorithms including DBNs. BNT is distributed under the GNU Library General Public License. The BNT package was extended by Chang, Beck, Mostow, and Corbett to lower the cost to construct and assess student models resulting in the Bayes Net Toolkit for Student Modeling (BNT-SM) (Chang, Beck, Mostow, & Corbett, 2006). The BNT-SM system provides a powerful and comprehensive method to create and train student models using less code than the BNT. Because of the way it is uniquely tailored to student modeling, I chose to utilize the BNT-SM system to create and evaluate my student model. The BNT-SM program takes an XML specification file, which contains the structure of a student model such as a knowledge tracing model and a set of evidentiary data, and outputs the much longer BNT Matlab code to train and evaluate this model.

In order to use the BNT-SM system a researcher must first create an XML specification file which specifies the data source, the network structure, and the initial values for the model parameters. The data source is a tab delimited evidence file composed of rows that each represent a student's attempt on a practice problem where the columns contain observed data about the student's attempt (such as correctness and when it occurred). Latent variables, such as student knowledge, are represented with "NULL." The data source used for this project contained the user id, the problem id, and the skill id, and a boolean variable signifying whether the student answered the problem correctly or not, as well as all of the derived fields calculated using the method described in Section 9.1.2. I also included the natural logarithm (ln), the squared value, and the inverse of the value for each of these derived field values which would be later used in the SPSS system described in section 9.2.3. Finally, I included the probabilistic learning state score and a boolean representing a prediction of whether the student was in the learning state. For this project, whether or not the student was in the learning state was predicted using a randomly generated continuous variable. If this random variable was lower than the P(LS) value then it was predicted that the student was in the learning state. If the random variable was higher than the P(LS) value then it was predicted that the student was not in the learning state. This non-optimal approach of discretizing a continuous variable was due to an inability to get the BNT-SM to support this node as a continuous variable. This data file was then sorted first by skill, then user id, then problem id in order to prepare it for use by the BNT-SM system.

The next element in the XML specification file for the BNT-SM system is the network structure of the student model. The structure of the learning state model used in this project, described in section 9.1.2.8, has three nodes: knowledge, i.e. whether the student is predicted to know the skill or not; learning state, i.e. whether the student is predicted to be in the learning state; and answer correct, i.e. whether the student answered the problem correctly or incorrectly. The structure of the student model, in this case the learning state model, in the XML specification file contains the individual nodes in the system, there is one "node" XML tag for each node in the system, and the relationships they have with one another, indicated by the "within" and "between" tags. The "within" tag signifies that the relationship to the other node occurs within this time slice, whereas the "between" tag signifies that the relationship to the other node occurs between this time slice and the next time slice. Whether the student is in the learning state or not in the learning state will (presumably) respectively increase and decrease the probability of the student knowing the skill and answering the problem correctly. Therefore, the value of the learning state node influences the values of the knowledge node and the answer correct node. Whether the student knows or does not know the skill on the current problem will affect the student's performance on the problem and will also affect whether the student knows the skill at the next time step. Thus, the value of the knowledge node influences the values of the answer correct node and the knowledge node at the next time step. Since neither the value of the learning state node nor the value of the knowledge node will be affected by the student's performance, the answer correct node does not influence any other nodes. The XML specification of the network structure in Appendix B - describes the structure of the learning state model used for this project.

After the specification of the network structure the initial values for the model parameters must be specified in the XML file. The initial values for the model parameters can be any reasonable probability between zero and one since the model parameters are trained to the data, as detailed in section 7.2.2. As mentioned in section 9.1.2.8, there are now two model parameters for each of the guess, slip, probability of transition, and initial knowledge, one for instances when the student is in the learning state and one for instances when the student is not in the learning state. In order to distinguish between the two similar model parameters, these parameters are labeled with a "_t" for instances when the student is in the learning state and with a "_f" for instances when the student is not in the learning state. There is also an initial parameter that gives the probability of a student being in the learning state. The initial values used in this project are listed in Table 9.9.

TABLE 9.9: INITIAL VALUES OF MODEL PARAMETERS

Model Parameter	Initial Value
learning_state: P(learning_state)	0.893828
know_t: P(knowledge | learning_state)	0.479836
know_f: P(knowledge | ~learning_state)	0.271712
slip_t: P(~correct | knowledge, learning_state)	0.152331
slip_f: P(~correct | knowledge, ~learning_state)	0.266201
guess_t: P(correct | ~knowledge, learning_state)	0.437335

guess_f: P(correct | ~knowledge, ~learning_state)	0.324537
P(T)_t: P(knowledge | ~knowledge, learning_state)	0.198114
P(T)_f: P(knowledge | ~knowledge, ~learning_state)	0.099876

After the XML specification file has been created containing a specification of the data file, the network structure of the model, and the initial parameter values, and the data file has been provided, the BNT-SM system can be run by calling the RunBnet.m script in Matlab. This script will train the model to the data, i.e. use Expectation Maximization to estimate the values of the model parameters that will maximize the data likelihood, and evaluate the latent variable, i.e. estimate the knowledge variable. Both of these processes are described in section 7.2.2.

The BNT-SM system generates two output files which are utilized by this project as explained in section 9.2.2 below. The first output file is similar to the input file except for a column containing the model's estimate of the student's knowledge for each of the rows, i.e. for every problem attempted by every student, which is included in the file. The second output file contains a table where each row contains the skill name, the number of users in the data set who practiced the skill, the number of problems in the data set that tested the given skill, the inferred model parameter values, and the log likelihood values (an approach for measuring model fit) for each skill in the data set. The BNT-SM step in the iterative process took an average of eight hours to complete for each iteration.

9.2.2 PYTHON SCRIPT: RUNAFTERBNT_SM.PY

After the BNT-SM step was complete, I used a python script I wrote, named runAfterBNT-SM.py, in order to nudge the P(LS) values higher or lower depending on how accurately the resulting model and model parameters was able to predict whether the student would answer a problem correctly or incorrectly. The script used the model parameters inferred by the BNT-SM system to predict whether or not the student answered the problem correctly, as described in section 7.2.3, for every student and every problem. If the predicted student performance was accurate, then the P(LS) value would not change. However, if the prediction was not accurate then the script nudged the P(LS) values higher if the prediction was too low, i.e. the student was predicted to answer the problem incorrectly but he or she answered the problem correctly, or lower if the prediction was too high, i.e. the student was predicted to answer the problem correctly but he or she answered the problem incorrectly. The percent delta change in the P(LS) value varied depending on how long the streak of incorrect predictions was, i.e. how many times in a row the prediction was incorrect, and what was the position of this problem in the streak of incorrect predictions. The longer the streak of incorrect predictions, the higher the percent delta value because longer streaks of incorrect predictions likely indicate that the P(LS) value is further from the actual value than it is when the streak is shorter. Similarly, the lower the position of the problem in the streak of incorrect positions, the higher the percent delta change value because it is likely that at the first position there was an incorrect assumption which resulted in a low or high P(LS) score. The script used a lookup table, Table 9.10 contains the percent delta values for iterations 1-14 and Table 9.11 contains the percent delta values for iterations 15-50, to determine which percent delta change value to use depending on the length of the streak and the position within the streak. This approach was inspired by similar work by Baker, Corbett, and Aleven that used future data to adjust predictions (Baker, Corbett, & Aleven, 2008).

TABLE 9.10: PERCENT DELTA VALUE FOR P(LS) LOOKUP TABLE BY POSITION AND STREAK FOR ITERATIONS 1-14

	Streak Length 1	Streak Length 2	Streak Length 3	Streak Length 4+
Position 1 in Streak	0.01	0.15	0.25	0.3
Position 2 in Streak		0.05	0.1	0.15
Position 3 in Streak			0.01	0.05

TABLE 9.11: PERCENT DELTA VALUE FOR P(LS) LOOKUP TABLE BY POSITION AND STREAK FOR ITERATIONS 15-50

	Streak Length 1	Streak Length 2	Streak Length 3	Streak Length 4+
Position 1 in Streak	0.005	0.075	0.0125	0.15
Position 2 in Streak		0.025	0.05	0.075
Position 3 in Streak			0.005	0.025

A P(LS) value which overestimated the learning state, i.e. if it was predicted that the student would answer correctly but he or she answered the problem incorrectly, was decreased based on the percent delta value corresponding to the length of the streak and its position within the streak using the equation $P(LS)_{new} = P(LS)_{old} + (-P(LS)_{old} * percent\ delta\ value)$. A

P(LS) value which underestimated learning state, i.e. if it was predicted that the student would answer the problem incorrectly but he or she answered the problem correctly, was increased based on the percent delta value corresponding to the length of the streak and its position within the streak using the equation

$P(LS)_{new} = P(LS)_{old} + \left((1 - P(LS)_{old}) * percent\ delta\ value\right)$. After nudging the P(LS) value higher or lower depending on the accuracy of the prediction of the student's performance, the P(LS) value was transformed from a probability to a logit in order to model it in SPSS, using the following equation:

$$\text{logit} = \ln\left(\frac{P(LS)_{new}}{(1 - P(LS)_{new})}\right)$$

After this logit value was computed, the script created a new file containing all of the initial derived field scores and the logit value which is utilized by the SPSS system as described in section 9.2.3 below. This step in the iterative process took approximately five minutes to complete for each iteration. Table 9.12 provides an example of how the output file is generated by the script taken from the output of the initial iteration.

TABLE 9.12: PARTIAL RUNAFTERBNT_SM.PY OUTPUT FILE

user	problem ordering	skill	…	raw score	learning state	answer correct	P(LS)old	p(know)	slip value	guess value	p(correct)	predicted correct	change to p(LS)	length of streak	position in streak	delta P(LS)	P(LS)new	New Raw Learning State Score
70306	1649	2		2	2	1	0.8808	0.91478	0.1651	0.3263	0.77429	2	-	3	1	-0.2202	0.660598	0.6659594
70306	1660	2		2	2	1	0.8808	0.77833	0.1651	0.3263	0.77429	2	-	3	2	-0.0881	0.792717	1.3413835
70306	2127	2		2	2	1	0.8808	0.54524	0.1651	0.3263	0.77429	2	-	3	3	-0.0088	0.871989	1.9186616
70307	1022	2		2	2	1	0.8808	0.20386	0.1651	0.3263	0.77429	2	-	1	1	-0.0088	0.871989	1.9186616
70307	1045	2		2	2	2	0.8808	0.62126	0.1651	0.3263	0.77429	2	=	0	0	0	0.880797	2
70307	1649	2		5	2	2	0.99331	0.86242	0.1651	0.3263	0.831521	2	=	0	0	0	0.993307	5
70307	1660	2		2	2	2	0.8808	0.95603	0.1651	0.3263	0.77429	2	=	0	0	0	0.880797	2
70307	2127	2		2	2	1	0.8808	0.87553	0.1651	0.3263	0.77429	2	-	1	1	-0.0088	0.871989	1.9186616
70309	1022	2		2	1	1	0.8808	0.08614	0.1674	0.2245	0.76013	2	-	6	1	-0.2642	0.616558	0.4749638
70309	1045	2		2	2	1	0.8808	0.09523	0.1651	0.3263	0.77429	2	-	6	2	-0.1321	0.748678	1.0915714
70309	1263	2		2	2	1	0.8808	0.0981	0.1651	0.3263	0.77429	2	-	6	3	-0.044	0.836757	1.6342955
70309	1649	2		2	1	1	0.8808	0.06643	0.1674	0.2245	0.76013	2	-	6	4	-0.044	0.836757	1.6342955
70309	1660	2		2	1	1	0.8808	0.05787	0.1674	0.2245	0.76013	2	-	6	5	-0.044	0.836757	1.6342955
70309	2127	2		2	2	1	0.8808	0.08654	0.1651	0.3263	0.77429	2	-	6	6	-0.044	0.836757	1.6342955
70311	1022	2		3	2	2	0.95257	0.72785	0.1651	0.3263	0.810801	2	=	0	0	0	0.952574	3
70311	1045	2		2	2	2	0.8808	0.90677	0.1651	0.3263	0.77429	2	=	0	0	0	0.880797	2
70311	1045	2		3	2	2	0.95257	0.97085	0.1651	0.3263	0.810801	2	=	0	0	0	0.952574	3
70311	1263	2		4	2	1	0.98201	0.91478	0.1651	0.3263	0.825776	2	-	1	1	-0.0098	0.972194	3.5542907
70311	1649	2		2	2	2	0.8808	0.97346	0.1651	0.3263	0.77429	2	=	0	0	0	0.880797	2

As an example, examine the first instance in the table. The probability that the student answered the problem correctly is computed following the process in section 7.2.3.

$$P(Correct) = P(Know) * \left(1 - P(Slip)\right) + \left(1 - P(Know)\right) * P(Guess)$$

$$= 0.91478 * (1 - 0.1651) + (1 - 0.91478) * 0.3263$$

$$= 0.7637 + 0.0278$$
$$= 0.7915$$

Since the probability that the student will answer the problem correctly is greater than 0.5, the student is predicted to answer this problem correctly, i.e. the value of predicted correct is 2 (if the probability was less than 0.5 the student would be predicted to answer incorrectly and the value of predicted correct would be 1). However, the student answered this problem incorrectly, i.e. the value of answer correct is 1, therefore, the prediction was incorrect. Since P(LS) was overestimated, i.e. the prediction was that the student's answer was correct but the student's answer was incorrect, the P(LS) value needs to be lowered. Because this instance is the first incorrect prediction in a streak of length three, the delta P(LS) value is computed as follows.

$$delta\ P(LS) = -P(LS)_{old} * percent\ delta_{length\ 3, position\ 1}$$
$$= -(0.8808) * 0.25$$
$$= -0.2202$$

The new P(LS) value is computed by simply adding this delta P(LS) value to the old P(LS) value.

$$P(LS)_{new} = P(LS)_{old} + delta\ P(LS)$$
$$= 0.8808 + (-0.2202)$$
$$= 0.6606$$

The new P(LS) value was transformed from a probability to a logit in order to model it in SPSS.

$$\text{logit} = \ln\left(\frac{P(LS)_{new}}{(1 - P(LS)_{new})}\right)$$
$$= \ln\left(\frac{0.6606}{(1 - 0.6606)}\right)$$
$$= \ln\left(\frac{0.6606}{0.3393}\right)$$
$$= \ln(1.9469)$$
$$= 0.6662$$

9.2.3 SPSS STATISTICS

Initially created by Nie, Hull, Bent in the 1960s to inspect and interpret large amounts of social science data, the originally named Statistical Package for the Social Sciences (SPSS) (http://www.spss.com) has become successful and widely utilized tool to aid statistical data analysis (Griffith, 2007). In this project the SPSS system is used to derive a linearized equation used to generate the P(LS) estimate from the derived field scores. In order to analyze data in the SPSS Statistics system a user must first provide the SPSS system with a list of *variables* and data for these variables resulting in a number of *cases*. Each variable has given a type, such as *scale* which is a measurement variable such as inches or gallons, or *categorical* for categorized variables such as mood (happy, sad, bored, etc.), pet type (bird, cat, dog, etc.), or weather (sunny, cloudy, rainy, snowy, etc.). All of the variable types in the SPSS system are labeled as holding a specific classification of number, for example a pet type of bird may be given a value of 1, a pet type of cat a value of 2, and pet type of dog a value of 3. The given type of the variable indicates which analysis operations can be performed on the data, for instance you would not be able to generate a mean or median pet type since the specie data cannot be easily sorted. For this project, I used the original derived field scores and the ln(x), x^2, and 1/x values of these scores as the variables for the SPSS system along with the logit values of the updated P(LS) values which were generated by the runAfterBNT-SM.py python script as described in section 9.2.2.

After all of the variables and data are specified, the user can perform an analysis algorithm on the data by simply selecting it from a drop down menu and selecting the variables on which to perform the analysis. For this project, I used the linear regression analysis algorithm which analyzes the relationships between a dependent variable and a set of independent variables. The set of inferred relationships connecting the dependent variable and the set of independent variables can be combined into a linear equation which can be used to derive a prediction of the logit value using the values of the original derived field scores and the manipulated (ln(x), x^2, and 1/x) derived field scores. The set of independent variables for this project is the set of original and manipulated derived field scores which are used to attempt to generate an equation which will map these independent variables to the derived variable, i.e. the logit value. The SPSS system accomplishes this mapping by modifying the weights of the derived field scores so that when the sum of the individual derived field scores multiplied by their weights (coefficients) is calculated, the resulting sum value is an approximation of the logit value. This step in the iteration took approximately twenty minutes to complete.

9.2.4 PYTHON SCRIPT: RUNAFTERSPSS.PY

Once the SPSS Statistics system generated the new logit values, I used a python script I wrote, named runAfterSPSS.py, in order to transform these logit values back into probabilistic learning state scores, P(LS), which gave the updated estimate of the likelihood that the student is in the learning state. These P(LS) values were generated by taking the logit transform of these new logit values, as explained in section 9.1.2, using the equation $\frac{e^x}{1+ e^x}$ where x is a new logit value.

Using these new P(LS) values, the script predicted whether or not the student was in the learning state at the current time. For this project, whether or not the student was in the learning state was predicted using a randomly generated continuous variable. If this random variable was lower than the P(LS) value then it was predicted that the student was in the learning state. If the random variable was higher than the P(LS) value then it was predicted that the student was not in the learning state.

Once these new P(LS) values were calculated, the script created a new tab delimited evidence file, an updated version of the file described in section 9.2.1 created using the same method. This evidence file is composed of rows that each represent a student's attempt on a practice problem where the columns contain observed data about the student's attempt (such as correctness and when it occurred). Latent variables, such as student knowledge, are represented with "NULL." The data source used for this project contained the user id, the problem id, and the skill id, and a boolean variable signifying whether the student answered the problem correctly or not, as well as all of the derived fields calculated using the method described in Section 9.1.2. I also included the natural logarithm (ln), the squared value, and the inverse of the value for each of these derived field values which would be later used in the SPSS system described in section 9.2.3. Finally, I included the *new* probabilistic learning state score, $P(LS)_{new}$, and a boolean representing the prediction of whether the student was in the learning state. As mentioned in section 9.2.1, for this project, whether or not the student was in the learning state was predicted using a randomly generated continuous variable. If this random variable was lower than the P(LS) value then it was predicted that the student was in the learning state. If the random variable was higher than the P(LS) value then it was predicted that the student was not in the learning state. The newly generated data file did not need to be sorted as described in section 9.2.1 in order to prepare it for use by the BNT-SM system since the data was previously sorted in the initialization step and the order of the data did not change during the iterations. This step of the iterative process took an increasing amount of time as the number of completed iterations grew because the P(LS) values for every step of the iteration were appended to a master file containing all of the P(LS) values for every previous iteration. The larger the number of completed iterations the larger the file size and the more time it takes to append the new data. This step of the iteration took approximately five minutes to complete for the earlier iterations and approximately fifteen minutes to complete for the later iterations. The file containing the resulting predictions was run back through the BNT-SM system initiating the next iteration using the updated P(LS) values.

10 RESULTS

10.1 MODEL FIT

I ran fifty iterations of the process detailed in sections 9.2 in order to estimate the learning state probabilities and infer the model parameters. For each iteration, I computed the R^2 value for the learning state model. The R^2 value indicates how much more accurately the given model, such as the learning state model, can predict whether the student is going to answer a problem correctly, P(correct), than a model which simply uses the average student performance to predict whether the student is going to answer the problem correctly. The R^2 value is calculated by first finding the average value of *answer correct* for the entire data set using the following equation:

$$Average\ Student\ Performance = \left(\frac{1}{n}\right) * \sum_{i=1}^{n} (answer\ correct)_i$$

Where *i* is the instance number, *n* is the number of instances in the data set, and *answer correct* is a boolean value indicating whether the student answered the problem correctly, i.e. *answer correct* = 1, or incorrectly, i.e. *answer correct* = 0. There were 124930 instances in the data set used for this project and average student performance in the data set was calculated to be 0.659862802. Therefore the baseline model would use a P(Correct) value of 0.659862802 in order to predict whether the student would answer the problem correctly, i.e. the baseline model would predict that the probability the student would answer the problem correctly was 66%.

After the average student performance is calculated, the squared error value for each instance for the baseline model, which always predicts the average student performance for P(correct), is calculated using the following equation:

$$Squared\ Baseline\ Error_i = \left((answer\ correct)_i - (average\ student\ performance)\right)^2$$

Where *i* is the instance number and *(answer correct)$_i$*, is a boolean value indicating whether or not the student answered the problem correctly, *(answer correct)$_i$* = 1, or incorrectly, *(answer correct)$_i$* = 0, for this instance. Since the average student performance in the data set used by this project was 0.659862802, if the student answered the problem at the current instance incorrectly, the squared error value for the baseline model for this instance would be calculated as follows:

$$\begin{aligned} Squared\ Baseline\ Error_i &= (0 - 0.659862802)^2 \\ &= (-0.659862802)^2 \\ &= 0.4354 \end{aligned}$$

However, if the student answered the problem at the current instance correctly, the squared error value for the baseline model for this instance would be calculated as follows:

$$\begin{aligned} Squared\ Baseline\ Error_i &= (1 - 0.659862802)^2 \\ &= (0.340137198)^2 \\ &= 0.1157 \end{aligned}$$

In order to calculate the sum-squared error for the baseline model all of the squared baseline error values for each instance are summed using the following equation:

$$Sum\text{-}Squared\ Baseline\ Error = \sum_{i=1}^{n} (Squared\ Baseline\ Error)_i$$

Where *i* is the current instance number and *n* is the number of instances in the data set. The sum-squared baseline error for the 124930 instance data set used in this project was calculated to be 28039.55005. The next step in computing the R^2 value is to calculate the P(correct) value for each instance following the process described in section 7.2.3. After the P(correct) value for each instance in the data set is calculated, the squared error value for each instance for the model is calculated using the following equation:

$$Squared\ Learning\ State\ Model\ Error_i = ((answer\ correct)_i - P(correct)_i)^2$$

Where i is the instance number, *(answer correct)$_i$*, is a boolean value indicating whether or not the student answered the problem correctly, *(answer correct)$_i$* = 1, or incorrectly, *(answer correct)$_i$* = 0, for this instance, and *P(correct)$_i$* is the calculated probability that the student will answer the current problem right for this instance. The following table contains some student data from the fourteenth iteration as an example of how to calculate the squared learning state model error for this student at this instance:

user	Problem ordering	skill	p(know)	slip value	guess value	P(correct)	answer correct
52128	36843	1	0.730571	0.259453	0.381733	0.504849	1

Since the calculated P(correct) value for this instance was 0. 504849, and the student answered the problem correctly, the squared error value for the model for this instance would be calculated as follows:

$$\begin{aligned} Squared\ Learning\ State\ Model\ Error_i &= (1 - 0.504849)^2 \\ &= (0.495151)^2 \\ &= 0.2452 \end{aligned}$$

In order to calculate the sum-squared error for the model all of the squared learning state model error values for each instance are summed using the following equation:

$$\begin{aligned} &Sum\text{-}Squared\ Learning\ State\ Model\ Error \\ &\qquad = \sum_{i=1}^{n} (Squared\ Learning\ State\ Model\ Error)_i \end{aligned}$$

Where i is the current instance number and n is the number of instances in the data set. The sum-squared learning state model error at the fiftieth iteration for the 124930 instance data set used in this project was calculated to be 25740.96139. Finally the R^2 value can be calculated using the following equation:

$$R^2\ Value\ = \left(1 - \left(\frac{Sum\text{-}Squared\ Learning\ State\ Model\ Error}{Sum\text{-}Squared\ Baseline\ Error}\right)\right)$$

This process was completed for each iteration to see the progression of the value throughout the iterations. Figure 10.1 depicts the trend of the R^2 value for the learning state model as it changed over time and eventually converged in the last third of the iterations to an R^2 value of 0.0838 or 8.4%. The computed R^2 value for the learning state model after the fiftieth iteration, i.e. the last iteration completed for this project, was 0.081977 or 8.2% meaning that the learning state model had 8.2% less squared error than the model which only predicted the average student performance for P(correct). The highest value of R^2 that was achieved during this project was generated by the 42nd iteration and had an R^2 value of 0.0848 or 8.5%.

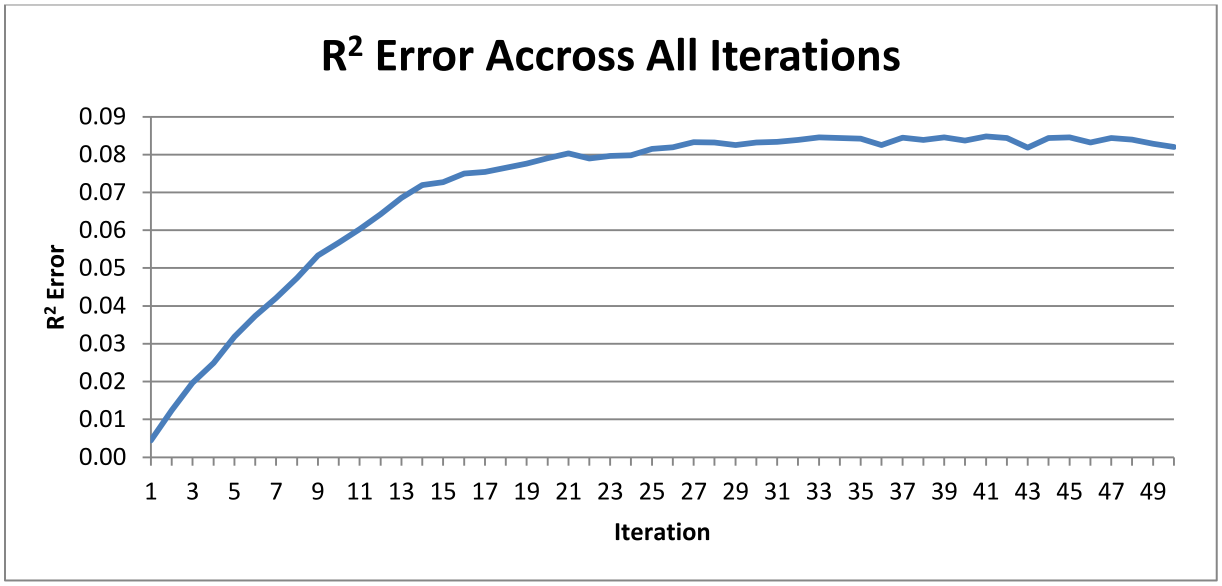


FIGURE 10.1: TREND OF R^2 ERROR ACCROSS ALL ITERATIONS

The simple knowledge tracing model described in section 7.2.1 achieved an R^2 value of about 7% for the same data set (Gong, Beck, Heffernan, & Forbes-Summers, 2010). It is unusual, therefore, for the model to take fifteen iterations to achieve a score of 0.0719 or 7.2% which was the first value of R^2 produced by the learning state model that was greater than the R^2 value of 7% produced by the simple knowledge tracing model. There are a number of different explanations which could have increased number of iterations to achieve the same R^2 value as the simple knowledge tracing model which I was trying to improve. The first possible reasoning for the number of iterations is that the learning state model increased more slowly than the simple knowledge tracing model because the learning state model is simply more complex than the knowledge tracing model resulting in twice the number of model parameters which need to be estimated. As a result of this complexity, the learning state model may not have been able to infer the values of the model parameters as quickly as the simpler knowledge tracing model which had to infer half of the number of model parameters. The increased number of iterations may also be due to a lack of prior research into the best method to use to generate the probability that the student is in the learning state, P(LS), since without any prior information it is unlikely that the first derived fields I tested are the most efficient and accurate indicators of P(LS). This phenomenon could also have simply been the result of updating the P(LS) values too slowly by using a percent delta change value that was too small when updating the P(LS) values following the process described in section 9.2.2.

10.2 UNDERSTANDING STUDENT LEARNING

The goal of this project was to find a set of student behaviors that, when combined in a single derived field would utilize this information to find a set of student behaviors that affect learning, either positively or negatively and use evidence of these behaviors to detect when a student is more likely to learn a given skill, i.e. when a student is in the learning state. Therefore, I created seven derived fields, described in sections 9.1.2.1 to 9.1.2.7, with the goal of detecting whether or not a student was in the learning state. With this project goal in mind, in order to decide which derived fields I would use, I looked for characteristics that I believed would either increase or decrease the probability that the student is in a state that is likely to lead to learning (i.e. is in the learning state or not in the learning state respectively). The derived fields I used were “gameLastX” described in section 9.1.2.1, “gameLastSimilarX” described in section 9.1.2.2, “lastXWrongCurrentRight” described in section 9.1.2.3, “lastXofYWrongCurrentRight” described in section 9.1.2.4, “averageTimeToComplete” described in section 9.1.2.5, “directToAnswerAfterLastHint” described in section 9.1.2.6, and “directToAnswerAfterBottomOutHint” described in section 9.1.2.7. However, since there is currently a lack of prior research into the best method to use to generate the probability that the student is in the learning state, P(LS), and I therefore had insufficient evidence on which to base my derived fields, it is possible that the derived fields I used are not the most efficient or accurate indicators of P(LS). It is also possible that these derived fields could also have little effect on the student learning state or even produce results that counter my hypotheses about how the derived fields should affect the P(LS) values. Therefore, the derived fields I created need to be interpreted and evaluated.

In the SPSS step of the iterative process, described in section 9.2.3, I used the linear regression analysis algorithm which analyzes the relationships between a dependent variable, i.e. the logit value, and a set of independent variables, i.e. the original derived field scores and the manipulated (ln(x), x^2, and 1/x) derived field scores. The set of independent variables is to attempt to generate an equation which will map these independent variables to the derived variable, i.e. the logit value. The SPSS system accomplishes this mapping by modifying the weights (coefficients) of the derived field scores so that when the sum of the individual derived field scores multiplied by their weights (coefficients) is calculated, the resulting sum value is an approximation of the logit value. However these coefficient values generated through the use of the linear regression analysis algorithm in the SPSS system can also be used to interpret and evaluate the derived fields used in this project.

I hypothesized that the derived fields I created would be able to help provide a more accurate indication of when the student was in a state where he or she was more likely to acquire knowledge of a skill. Using the coefficient values generated by the SPSS system, I can evaluate the derived fields I used in order to determine if they follow my hypotheses, if the they contradict my hypotheses, if they seem to have little to no effect on the outcome of the learning state, and to what extent each individual derived field affects the probability that the student is in the learning state. If the derived fields follow my hypothesis or have a strong effect on the probability that the student is in the learning state, this information will help determine which student behaviors are more conducive and which behaviors are less conducive to student learning and will provide a starting platform for future research exploring student learning states. Table 10.1 lists the coefficient values, generated by the SPSS system in the fiftieth iteration, for each of the various original or manipulated derived field values which were included in the generated equation used to approximate the logit value.

TABLE 10.1: DERIVED FIELDS COEFFICIENTS

Derived Field	Coefficient
gameLastX x^2	-0.020
gameLastSimilarX x^2	-0.101
lastXWrongCurrentRight 1/x	-0.006
lastXofYWrongCurrentRight 1/x	-0.027
averageTimeToComplete	-0.223
averageTimeToComplete ln(x)	0.998
averageTimeToComplete x^2	-0.242
averageTimeToComplete 1/x	0.035
directToAnswerAfterLastHint	0.490
directToAnswerAfterLastHint x^2	0.141
directToAnswerAfterLastHint 1/x	0.006
directToAnswerAfterBottomOutHint	0.20
directToAnswerAfterBottomOutHint x^2	0.054
directToAnswerAfterBottomOutHint 1/x	0.010

Every derived field I created had at least one permutation of its derived field score included in the list of derived field values which were used to generate the equation to approximate the logit value. The product of the derived field score and the weight of the derived field (the coefficient) for each of the possible scores generated by the derived field allows for comparison of the effects that each of the score values has on the probability that the student is in the learning state. For instance, the gameLastX derived field has two possible values, TRUE and FALSE, which result in a score of 0 or -4 respectively. Since the permutation of the gameLastX derived field score that was used to generate an approximation of the logit value was x^2, the possible scores for this modified derived field are 0 and 16 for the FALSE and TRUE values respectively. These modified scores are then each multiplied by the coefficient of the derived field, -0.020, and the resulting value is used to generate the estimated logit value. For the gameLastX x^2 derived field, the two possible scores, 0 and 16 are multiplied by the coefficient of the derived field and the resulting value of 0 or -0.32 is used to generate the estimated logit value. Since the score of the gameLastX x^2 value is lower in instances where the student was not determined to be gaming on the last X problems, value of FALSE, than when he or she was determined to be gaming, value of TRUE, the logit value and consequently the new P(LS) value are reduced more in TRUE case. A lower value in the TRUE case would normally indicate that when students gamed on the last X problems they were in a state where they were less likely to be learning, i.e. not in the learning state. However, since the resulting values are small the logit value and therefore the new probability of the student being in the learning state are not greatly affective indicating that this derived field is relatively unimportant to determining whether the student is in the learning state or not.

The gameLastSimilarX derived field similarly has two possible values, TRUE and FALSE, which result in a score of 0 or -4 respectively. Since the permutation of the gameLastSimilarX derived field score used to generate an approximation of the logit value was x^2, the possible scores for this modified derived field are 0 and 16 for the FALSE and TRUE values respectively. These modified scores are then each multiplied by the coefficient of the derived field, -0.101, and the resulting value is used to generate the logit value. For the gameLastX x^2 derived field, the two possible scores, 0 and 16, are multiplied by the coefficient of the derived field and the resulting value of 0 or -1.616 is used to generate the estimated logit value. Since the score of the gameLastSimilarX x^2 value is significantly lower in instances where the student was not determined to be gaming on the current problem, value of FALSE, than when he or she was determined to be gaming, value of TRUE, the logit value and consequently the new P(LS) value are reduced more in TRUE case. This much lower value in the TRUE case indicates that when students gamed on the current problem they were in a state where they were less likely to be learning, i.e. less likely to be in the learning state. Therefore this derived field follows my hypothesis that students who are gaming on the current problem are less likely to be in a state conducive to learning.

The lastXWrongCurrentRight derived field has two possible values, TRUE and FALSE, which result in a score of 1 or 0 respectively. Since the permutation of the lastXWrongCurrentRight derived field score used to generate an approximation of the logit value was 1/x, the possible scores for this modified derived field are 1 and 100 for the FALSE and TRUE values respectively where the possible score for the zero case is computed using an epsilon value of 0.01 for x instead of 0 in order to avoid an undefined result. These modified scores are then each multiplied by the coefficient of the derived field, -0.006, and the resulting value is used to generate the logit value. For the lastXWrongCurrentRight 1/x derived field, the two possible scores, 1 and 100, are multiplied by the coefficient of the derived field and the resulting value of -0.006 or -0.6 is used to generate the estimated logit value. Since the score of the lastXWrongCurrentRight 1/x value is significantly lower in instances where the student did not answer the current problem correctly and the prior two problem incorrectly, value of FALSE, than when he or she answered the current problem correctly and the prior two problems incorrectly, value of TRUE, the logit value and consequently the new P(LS) value are reduced more in FALSE case. This lower value in the FALSE case indicates that when students did not answer the current problem correctly and the prior two problem incorrectly they were in a state where they were less likely to be learning, i.e. less likely to be in the learning state. Therefore this derived field follows my hypothesis that students who either answered the current problem incorrectly or the prior problems correctly are less likely to be in a state conducive to learning.

The lastXofYWrongCurrentRight derived field similarly has two possible values, TRUE and FALSE, which result in a score of 1 or 0 respectively. Since the permutation of the lastXofYWrongCurrentRight derived field score used to generate an approximation of the logit value was 1/x, the possible scores for this modified derived field are 1 and 100 for the FALSE and TRUE values respectively where the possible score for the zero case is computed using an epsilon value of 0.01 for x instead of 0 in order to avoid an undefined result. These modified scores are then each multiplied by the coefficient of the derived field, -0.027, and the resulting value is used to generate the logit value. For the lastXofYWrongCurrentRight 1/x derived field, the two possible scores, 1 and 100, are multiplied by the coefficient of the derived field and the resulting value of -0.027 or -2.7 is used to generate the estimated logit value. Since the score of the lastXofYWrongCurrentRight 1/x value is significantly lower in instances where the student did not answer the current problem correctly and at least one of the prior three problems incorrectly, value of FALSE, than when he or she answered the current problem correctly and at least one of the prior three problem incorrectly, value of TRUE, the logit value and consequently the new P(LS) value are reduced more in FALSE case. This lower value in the FALSE case indicates that when students did not answer the current problem correctly and at least one of the prior three problems incorrectly they were in a state where they were less likely to be learning, i.e. less likely to be in the learning state. Therefore this derived field follows my hypothesis that students who either answered the current problem incorrectly or all of the prior three problems correctly are less likely to be in the learning state.

The evaluation of the final three derived fields, averageTimeToComplete, directToAnswerAfterLastHint, and directToAnswerAfterBottomOutHint, is a little more complicated since there are multiple modified derived fields for each of these three derived fields. The derived field averageTimeToComplete can have one of five different scores, -2, -1, 0, 1 and 2 as described in section 9.1.2.5. All of the possible modified derived fields for averageTimeToComplete and the original averageTimeToComplete derived field are used to generate the logit value. Therefore, the trend for the overarching averageTimeToComplete derived field must be calculated by taking each of the modified and original derived field values for each of the possible scores for averageTimeToComplete and multiplying it by the coefficient corresponding to the given modification or lack thereof and finally summing the values for each of the possible scores as shown in Table 10.2. A graph of the resulting overarching trend for the averageTimeToComplete derived field is shown in Figure 10.2.

TABLE 10.2: POSSIBLE SCORES FOR EACH MODIFICATION OF AVERAGETIMETOCOMPLETE DERIVED FIELD

averageTimeToComplete					
Modification	**none**	**ln(x)**	**x^2**	**1/x**	
Coefficient	**-0.223**	**0.998**	**-0.242**	**0.035**	
Possible Scores					**Sum Score for Each Possible Score**
-2	0.446	-4.596	-0.968	-0.0175	-5.1355
-1	0.223	-4.596	-0.242	-0.035	-4.65
0	0	-4.596	0	3.5	-1.096
1	-0.223	0	-0.242	0.035	-0.43
2	-0.446	0.69176	-0.968	0.0175	-0.7047

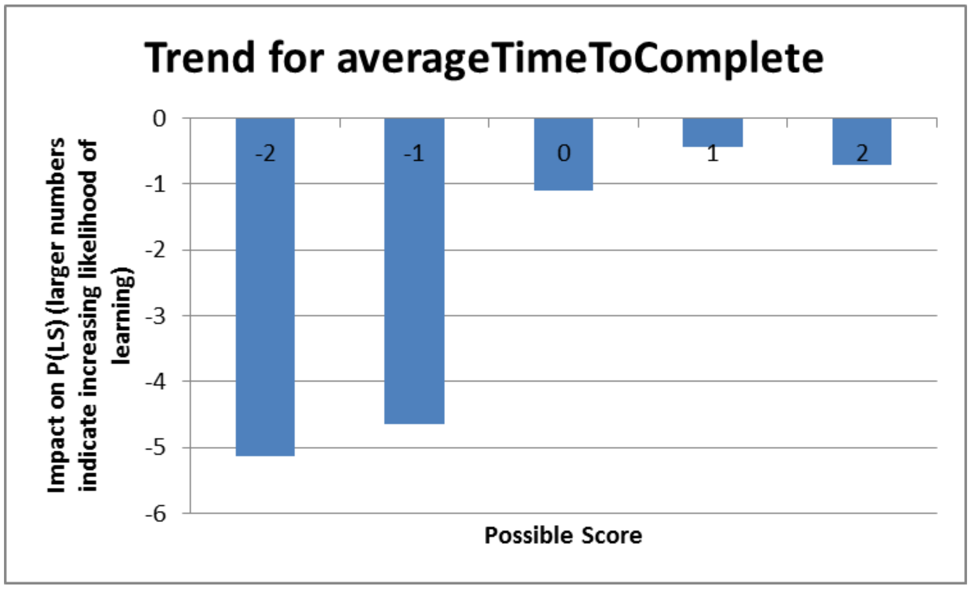


FIGURE 10.2: TREND FOR AVERAGETIMETOCOMPLETE DERIVED FIELD

As can be seen in Figure 10.2, the scores for the averageTimeToComplete derived field have a general upwards trend which follows my hypothesis that students who take a shorter amount of time are less likely to be in the learning state. The students who take a much shorter time than the average to complete the problem, score of -1 to -2, are a lot less likely to be in the learning state than those students who take at least the average amount of time to complete the problem, score of 2 to 0. However, there is little difference between the scores for students who took at least the average amount of time to answer the problem and there is little difference between the scores for the students who took less time than the average amount of time to answer the problem. Therefore, if a student answers a problem in less time than the average how much less time the student took is relatively unimportant and if a student takes at least the average amount of time to answer the problem the amount of additional time that the student took to answer the problem is relatively unimportant as well.

The derived field directToAnswerAfterLastHint can have one of five different scores, -2, -1, 0, 1 and 2 as described in section 9.1.2.6. The x^2 and 1/x modified derived fields for directToAnswerAfterLastHint and the original directToAnswerAfterLastHint derived field are used to generate the logit value. Therefore, the trend for the overarching directToAnswerAfterLastHint derived field must be calculated by taking each of the modified and original derived field values for each of the possible scores for directToAnswerAfterLastHint and multiplying it by the coefficient corresponding to the given modification or lack thereof and finally summing the values for each of the possible scores as shown in Table 10.3. A graph of the resulting overarching trend for the directToAnswerAfterLastHint derived field is shown in Figure 10.3.

Table 10.3: Possible Scores for Each Modification of directToAnswerAfterLastHint Derived Field

directToAnswerAfterLastHint				
Modification	**none**	**x^2**	**1/x**	
Coefficient	0.49	0.141	0.006	
Possible Scores				**Sum Score for Each Possible Score**
-2	-0.98	0.564	-0.003	-0.419
-1	-0.49	0.141	-0.006	-0.355
0	0	0	0.6	0.6
1	0.49	0.141	0.006	0.637
2	0.98	0.564	0.003	1.547

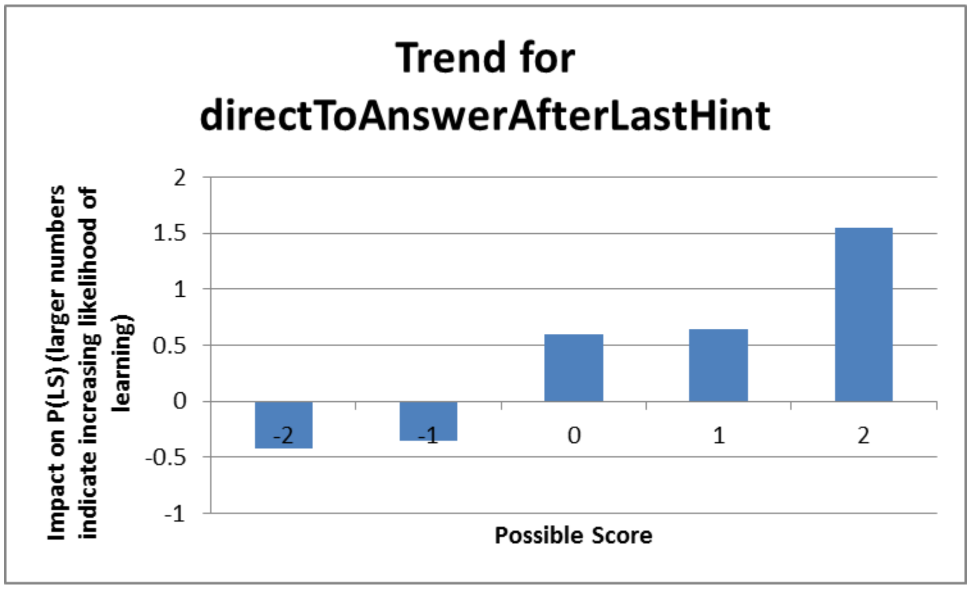


FIGURE 10.3: TREND FOR DIRECTTOANSWERAFTERLASTHINT DERIVED FIELD

As can be seen in Figure 10.3, the scores for the directToAnswerAfterLastHint derived field have a general upwards trend which follows my hypothesis that students who answer the problem in less time than the arbitrarily suggested ten seconds after receiving their last hint, a score of -1 to -2, are less likely to be in the learning state than those who take at least ten seconds to answer the problem after receiving their last hint, a score of 2 to 0. There is little difference between the scores for students who took less than the suggested time to answer the problem after receiving the last hint but there is a definite downward slope for the scores that indicate the student took a lot longer than the suggested time to answer the problem after receiving their last hint. Therefore, if a student answers a problem in less time than the average how much less time the student took is relatively unimportant and if a student takes at least the average amount of time to answer the problem the student is a lot more likely to be in the learning state but this likelihood of being in the learning state rapidly decreases as the student takes more time to answer the problem after receiving their last hint.

The derived field directToAnswerAfterBottomOutHint can have one of five different scores, -2, -1, 0, 1 and 2 as described in section 9.1.2.6. The x^2 and 1/x modified derived fields for directToAnswerAfterBottomOutHint and the original directToAnswerAfterBottomOutHint derived field are used to generate the logit value. Therefore, the trend for the overarching directToAnswerAfterBottomOutHint derived field must be calculated by taking each of the modified and original derived field values for each of the possible scores for directToAnswerAfterBottomOutHint and multiplying it by the coefficient corresponding to the given modification or lack thereof and finally summing the values for each of the possible scores as shown in Table 10.4. A graph of the resulting overarching trend for the directToAnswerAfterBottomOutHint derived field is shown in Figure 10.4.

TABLE 10.4: POSSIBLE SCORES FOR EACH MODIFICATION OF DIRECTTOANSWERAFTERBOTTOMOUTHINT DERIVED FIELD

directToAnswerAfterBottomOutHint				
Modification	**none**	**x^2**	**1/x**	
Coefficient	0.2	0.054	0.01	
Possible Scores				**Sum Score for Each Possible Score**
-2	-0.4	0.216	-0.005	-0.189
-1	-0.2	0.054	-0.01	-0.156
0	0	0	1	1
1	0.2	0.054	0.01	0.264
2	0.4	0.216	0.005	0.621

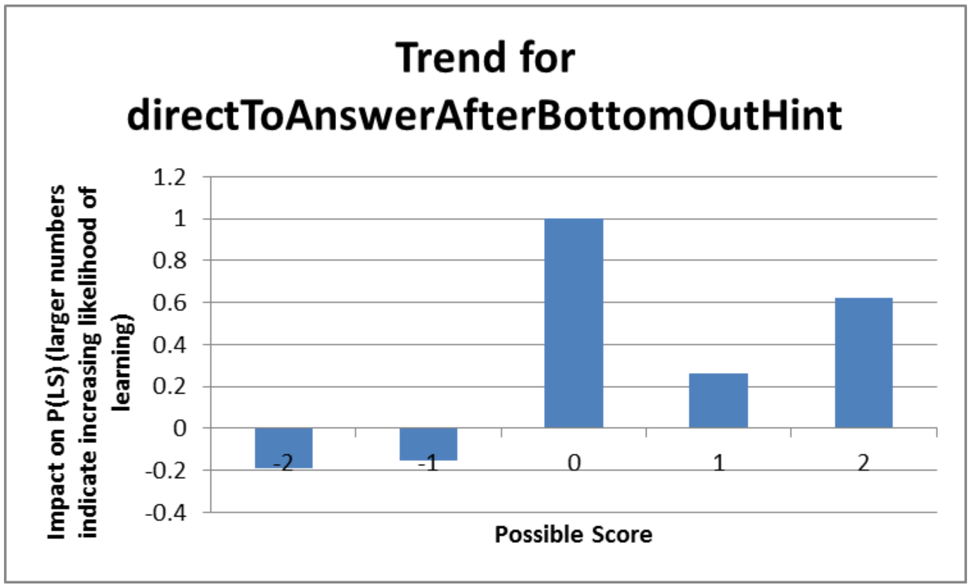


FIGURE 10.4: TREND FOR DIRECTTOANSWERAFTERBOTTOMOUTHINT DERIVED FIELD

As can be seen in Figure 10.4, the scores for the directToAnswerAfterBottomOutHint derived field may have a slight upwards trend but since the summed scores are very small the trend of the directToAnswerAfterBottomOutHint derived field is difficult to interpret.

While using the model-fitting procedure I noticed that there was a glitch in the process which caused the P(LS) values to regress to the mean P(LS) value and systematically decrease from iteration to iteration. A possible reason for the occurrence of this phenomenon is that it is the result of an artifact in the linearization approach but additional investigation must be done in order to prevent this from recurring in future studies.

10.3 IMPACT OF STUDENT LEARNING ON KNOWLEDGE TRACING MODEL

The BNT-SM step in the iterative process inferred the values of the model parameters at each iteration. For the fiftieth iteration, the average model parameters for the skills containing more than five-hundred instances are listed in Table 10.5.

TABLE 10.5: INITIAL AND FINAL VALUES OF THE INFERRED MODEL PARAMETERS

Model Parameter	Initial Value	Final Average Value
learning_state: P(learning_state)	0.894	0.545
know_t: P(knowledge | learning_state)	0.480	0.466

know_f: P(knowledge | ~learning_state)	0.272	0.455
slip_t: P(~correct | knowledge, learning_state)	0.152	0.180
slip_f: P(~correct | knowledge, ~learning_state)	0.266	0.213
guess_t: P(correct | ~knowledge, learning_state)	0.437	0.531
guess_f: P(correct | ~knowledge, ~learning_state)	0.325	0.296
P(T)_t: P(knowledge | ~knowledge, learning_state)	0.198	0.121
P(T)_f: P(knowledge | ~knowledge, ~learning_state)	0.100	0.081

As noted in section 9.1.2.8, there are two values for each of the model parameters, one for learning_state = TRUE and one for learning_state = FALSE. Whether or not the student is in the learning state affects the various inferred model parameter values in different ways and in different amounts. For the initial knowledge model parameter a student is estimated to have more knowledge when beginning the first problem in for a given skill when he or she is in the learning state than when the student is not in the learning state. This is an expected outcome since a student is likely to be focusing more in the learning state and therefore is more likely to know the skill, however, the learning state does not seem to have a large effect on the initial knowledge parameter. For the slip model parameter a student is estimated to slip less when he or she is in the learning state than when the student is not in the learning state. This is an expected outcome since, again, a student is likely to be focusing more in the learning state and therefore is less likely to make a mistake, however, the learning state does not seem to have a large effect on the slip parameter either. For the guess model parameter a student is estimated to guess more often when he or she is in the learning state than when the student is not in the learning state. The difference in the frequency of guessing when a student is in the learning state vs. when the student is not in the learning state is larger than expected. One possible reason for the guessing frequency to be higher is that students in the learning state are likely trying, so it is not unexpected that the student would answer correctly more often in spite of not knowing skill through the use of partial knowledge and or learning the skill while completing the problem. However, the rate of guessing is much higher than expected which signifies that students who do not know skill are likely to answer the problem correctly. For the P(T) model parameter a student is estimated to learn at a slightly increased rate when he or she is in the learning state than when the student is not in the learning state. The learning state parameter was expected to have a much more significant effect on the rate of learning than was inferred by BNT-SM system.

11 FUTURE WORK AND CONCLUSIONS

Despite the new data that was generated during the course of this project in the field of detecting and modeling student learning, little prior data had been previously generated about the process of modeling student learning and which observed student behaviors could be mapped to certain aspects of the process by which a student acquires knowledge. Because of the lack of prior information it is likely that there are more efficient and accurate ways to detect when student learning occurs and future research is needed in order to further the research that was completed during this project. One such topic of future research which should be addressed is to find the cause of the convergence of the P(LS) values to the mean value from iteration to iteration. Another possible area of interest is to determine why the rate of student learning when the student was in the learning state stayed similar to the rate of student learning when the student was not in the learning state. Further research also needs to be completed to determine whether there is a feasible method of decreasing the time it takes to train the model to the data. Currently it takes about nine hours to complete one iteration and a large number of iterations need to be completed, which takes substantial time. A possible solution to this problem is to update the P(LS) values more rapidly, perhaps by increasing the percent delta change values or to train the model using only a portion of the data set. Furthermore, since I used a number of values which I selected based only on my own estimates of what they should be in order to create the derived fields and update the P(LS) value, these values can likely be improved upon. Therefore, future research can be completed in order to determine how much of an affect these values have on the results and to search for the values which produce the lowest model error. Finally, the most important improvement to this project would be to use a continuous value for the LS value which is given to SPSS instead of randomly generating number and using a value of 1 if the student was not in the learning state and 2 if the student was in the learning state.

Despite the research that still needs to be conducted, this project was able to create a model of student learning that outperformed the simple knowledge tracing model. The results I collected about the derived features I selected to generate the learning state values largely followed my initial hypotheses and therefore it can be concluded that these features do affect student learning. The reason for the guess model parameter value that was inferred cannot be determined, however, one possibility is that the students in the learning state are not really learning faster but they are instead are using partial knowledge more effectively resulting in a performance boost. Another more likely reason is that the knowledge tracing model is assuming that students in the learning state are guessing when they do in fact know the skill since this is a possible consequence of the system inferring more than one accurate set of model parameters and selecting the incorrect accurate set. For example, it is likely that some of the students in the learning state are not "guessing," but instead have actually acquired the skill. In this case, it could negatively affect the P(T) estimates for the learning state, creating the artificial situation where the learning rates look similar. A key bit of future work is untangling exactly what is going on here.

12 Bibliography

Baker, R. S., Corbett, A. T., & Aleven, V. (2008). *More Accurate Student Modeling Through Contextual Estimation of Guess and Slip Probabilities in Bayesian Knowledge Tracing.* Human Computer Interaction Institute.

Barnes, T., & Stamper, J. (2008). *Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data.* Heidelberg, Berlin: Springer-Verlag.

Beck, J. E. (2006). *Difficulties in Inferring Student Knowledge From Observations (and Why You Shouldn't Care).* Carnegie Mellon University, Machine Learning Department, Pittsburgh, PA.

Beck, J. E., & Mostow, J. (2010). *How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students.* Worcester Polytechnic Intstitute, Computer Science Department, Worcester, MA.

Borman, S. (2009, January 9). The Expectation Maximization Algorithm a Short Tutorial.

Cen, H., Koedinger, K., & Junker, B. (2007). Is Overpractice Necessary? - Improving Learning Efficiency with the Cognative Tutor through Educational Data Mining. *The 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work* (pp. 511-518). Amsterdam, The Netherlands: IOS Press.

Chang, K.-m., Beck, J., Mostow, J., & Corbett, A. (2006). A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. *The 8th International Conference on Intelligent Tutoring Systems.* Jhongli, Taiwa.

Corbett, A. T., & Anderson, J. R. (1995). *Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge.* Netherlands: Kluwer Academic Publishers.

Feng, M., Heffernan, N., & Beck, J. E. (2009). Using Learning Decomposition to Analyze Instructional Effectiveness in the ASSISTment System. *The 14th International Conference on Artificial Intelligence in Education (AIED-2009)* (pp. 523-530). Amsterdam, Netherland: IOS Press.

Gong, Y., Beck, J. E., & Heffernan, N. T. (2010). *Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting.* Worcester Polytechnic Institute, Department of Computer Science, Worcester, MA.

Gong, Y., Beck, J. E., & Heffernan, N. T. (2010b). How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis. *International Journal of Artificial Intelligence in Education*.

Gong, Y., Beck, J. E., Heffernan, N. T., & Forbes-Summers, E. (2010). *The Fine-Grained Impact of Gaming (?) on Learning.* Worcester Polytechnic Institute, Department of Computer Science, Worcester, MA.

Griffith, A. (2007). *SPSS for Dummies.* John Wiley & Sons.

Koedinger, K. R., & Anderson, J. R. (1997). Intellegent Tutoring Goes to School in the Big City. *International Journal of Artificial Intelligence in Education, 8,* 30-43.

Murphy, K. (2007, October 19). Retrieved July 3, 2011, from Bayes Net Toolkit for Matlab: http://code.google.com/p/bnt/

Russell, S., & Peter, N. (2010). Learning Probabilistic Models. In S. Russell, & N. Peter, *Artificial Intelligence A Modern Approach* (pp. 802-825). Upper Saddle River, New Jersey: Pearson Education, Inc.

SPSS Inc. (2007). *SPSS Statistics 17.0 Brief Guide.* Retrieved July 3, 2011, from Harvard Kennedy School: http://www.hks.harvard.edu/fs/pnorris/Classes/A%20SPSS%20Manuals/SPSS%20Statistics%20Brief%20Guide%2017.0.pdfhttp://www.hks.harvard.edu/fs/pnorris/Classes/A%20SPSS%20Manuals/SPSS%20Statistics%20Brief%20Guide%2017.0.pdf

Swire, P., Pardos, Z., & Heffernan, N. T. (2011). *Increasing K-12 Homework Effectiveness with Immediate Feedback.* Worcester Polytechnic Institute, Worcester, MA.

Wang, Y., & Heffernan, N. T. (2011). *Extend the Knowledge Tracing Framework using Partial Credit as Performance.* Worcester, MA.

APPENDICES

Appendix A - Detailed Descriptions of Original Data Set

A-1 Action Types

Action Type	Fields	Description
problem summary	problem log id user id assistment id problem id original correct first action hint count bottom hint attempt count time taken(ms) first response time(ms) tutor strategy id skill id item difficulty available hint count tutor strategy title	Lists summary information about the main problem, the student, and any hints or scaffold problems for this main problem. There is a single problem summary per main problem.
main start	time gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes the student has begun a new main problem and provides gaming information about the student at the current time step. There is a single main start action per main problem which occurs directly after the problem summary. The main start and main end actions surround all of the actions performed by the student while attempting to complete the main problem. Any number of hint or answer actions and up to one scaffold set can be contained between these two bookend actions.
main end	time gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes the student has completed the main problem and provides gaming information about the student at the current time step. There is a single main end action per main problem. The main start and main end actions surround all of the actions performed by the student while attempting to complete the main problem. Any number of hint or answer actions and up to one scaffold set can be contained between these two bookend actions.
answer	time answered correctly? gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes that the student submitted an answer to either a main problem or a subsequent scaffold problem. A student can submit any number of answers per main problem or scaffold problem. When a student submits an answer, the value of the attempt count for either the main or scaffold problem, is increased by one. The first answer the student submits to a main or scaffold problem is what is considered when looking at whether the student answered the given main or scaffold problem correctly.
hint	time gaming score is rapid guessing? is rapid response?	Denotes that the student requested a hint either on the main problem or on a subsequent scaffold problem. There can be any number of hint actions, from zero to the number of hint actions available to the student, per main problem or

	is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	scaffold problem. Each time the student requests a hint, the value of the hint count for either the main or scaffold problem, which keeps track of the number of hints the student requested, is increased by one.
scaffold	time gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes that the student has begun a scaffold set either by requesting help on or by answering a main problem where the help strategy was scaffolding. A scaffold action can occur zero or one times per main problem. The actions for the scaffold set and each of the scaffold problems within the set can occur after the scaffold action. Scaffold sets contain one or more scaffold problems.
scaffold summary	problem log id user id assistment id problem id original correct first action hint count bottom hint attempt count time taken(ms) first response time(ms) tutor strategy id skill id item difficulty available hint count tutor strategy title	Lists summary information about the current scaffold problem, the student, and any hints for this scaffold problem. There is one scaffold summary per scaffold problem.
scaffold start	time gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes the student has begun a new scaffold problem and provides gaming information about the student at the current time step. There is a single scaffold start action per scaffold problem which occurs directly after the scaffold summary. The scaffold start and scaffold end actions surround all of the actions performed by the student while attempting to complete the scaffold problem. Any number of hint or answer actions can be contained between these two bookend actions.
scaffold end	time gaming score is rapid guessing? is rapid response? is bottom out hint? times of rapid guessing in a row times of bottom out hint in a row	Denotes the student has completed the scaffold problem and provides gaming information about the student at the current time step. There is a single scaffold end action per scaffold problem. The scaffold start and scaffold end actions surround all of the actions performed by the student while attempting to complete the scaffold problem. Any number of hint or answer actions can be contained between these two bookend actions.

A-2 Fields

Field	Action	Description
problem log id	problem summary scaffold summary	The unique identification number corresponding to the log entry for the current problem (main problem or scaffold problem) which gives a timeline of events (actions). The sooner a student attempts a problem the lower the problem log id value.
user id	problem summary scaffold summary	The unique identification number corresponding to the student completing the problem.

assistment id	problem summary scaffold summary	The unique identification number of the current main problem from ASSISTments which is used to relate scaffolding problems with their corresponding main problem. For problem summary actions (main problem) this assistment id is the same as the problem id, for scaffold summary actions (scaffold problem) this assistment id is the same as the problem id for the main problem that contains this scaffold problem.
problem id	problem summary scaffold summary	For problem summary actions (main problem) this is the id of the main problem, for scaffold summary actions (scaffold problem) this is the id of the current scaffold problem.
original	problem summary scaffold summary	A boolean value, 0 or 1, denoting whether the current problem is a main problem or a scaffold problem. A value of 1 signifies that the current problem is a main problem and a value of 0 signifies that the current problem is a scaffold problem.
correct	problem summary scaffold summary	A boolean value, 0 or 1, denoting whether the student answered the current problem (main problem or scaffold problem) correctly or incorrectly *in his or her first attempt*. A value of 1 signifies that the student answered the current problem correctly on his or her first attempt and a value of 0 signifies that the student answered the current problem incorrectly on his or her first attempt.
first action	problem summary scaffold summary	An enumerated value of 0, 1, or 2 denoting the first action taken by the student. A value of 0 signifies that the first action taken by the student was to submit an answer, a value of 1 signifies that the first action was to request a hint, and a value of 2 signifies that the first action was to request help through a scaffold problem. For a main problem, the value of this field can be any of the enumerated values, 0, 1, or 2, but for a scaffold problem, the value can only be 0 or 1 since a scaffold problem will not contain other scaffold problems.
hint count	problem summary scaffold summary	The number of hints the student requested during this problem (main problem or scaffold problem).
bottom hint	problem summary scaffold summary	A boolean value, 0 or 1, denoting whether or not the student requested the last available hint for the current problem (main problem or scaffold problem) in which the answer to the problem is provided. A value of 1 signifies that the student did request the last available hint and a value of 0 signifies that the student did not request the last available hint.
attempt count	problem summary scaffold summary	The number of times a student submitted an answer on the current problem (main problem or scaffold problem).
time taken(ms)	problem summary scaffold summary	The time the student took to complete this problem (main problem or scaffold problem) in milliseconds. The time taken is calculated by subtracting the start time (the time field from main start or scaffold start action) from the end time.
first response time(ms)	problem summary scaffold summary	The time the student took to initiate the first action (hint, scaffold, or answer) in milliseconds on the current problem (main problem or scaffold problem) The first response time is calculated by subtracting the start time (the time field from the main start or scaffold start action) from the time at which the first action was performed.
tutor strategy id	problem summary scaffold summary	A unique identification number that is used to determine the type of tutorial strategy for the current problem (main problem or scaffold problem). The tutorial strategy for a main problem is labeled as a "Hint," a "Scaffold," or "No Help" and the tutorial strategy for a scaffold problem is labeled as a "Hint" or "No Help".
skill id	problem summary scaffold summary	A delineated list of unique identification numbers for all of the skills that are tested by the current problem (main problem or scaffold problem).

item difficulty	problem summary scaffold summary	The numeric integer score equal to the item difficulty parameter for the current problem (main problem or scaffold problem) from the item response theory (IRT) model. If the value of this field is -1, it means that the difficulty score for this problem was not yet calculated and there is no record of the difficulty parameter in the database.
available hint count	problem summary scaffold summary	The number of hints available to the student for in the current problem (main problem or scaffold problem). A hint count of -1 indicates that there are no available hints for the current problem.
tutor strategy title	problem summary scaffold summary	An enumerated value, “Hint,” “Scaffold,” or “No Help,” which signifies which tutoring strategy is being used by the current problem (main problem or scaffold problem).
time	main start main end answer hint scaffold scaffold start scaffold end	For the main start and scaffold start actions, the value of this field is a timestamp of the day and time in milliseconds when the student started the current problem (main problem or scaffold problem). For the remaining actions, the value of this field indicates the amount of time that has elapsed in milliseconds since the student performed the last action on the current problem.
answered correctly?	answer	A boolean value, TRUE or FALSE, signifying whether or not the student answered the current problem (main problem or scaffold problem) correctly. A value of TRUE indicates that the student answered the problem correctly and a value of FALSE indicates that the student answered the problem incorrectly.
gaming score	main start main end answer hint scaffold scaffold start scaffold end	A probabilistic score between 0 and 1 that signifies how confident the gaming detector is that the student is “gaming” on the current problem (main problem or scaffold problem). The closer this score is to 1 the more confident the gaming detector is that the student is “gaming”. This score is updated every time the student was “rapid guessing” on two consecutive problems, executed a “rapid response,” or requested the “bottom out hint” on three consecutive problems.
is rapid guessing?	main start main end answer hint scaffold scaffold start scaffold end	A boolean value, TRUE or FALSE, which indicates whether the student was “rapid guessing” on the current answer. If the value for this field is TRUE, the student is “rapid guessing” for this field meaning that the student submitted an answer less than 2000ms after submitting his or her previous answer; otherwise the value for this field is FALSE. If the value for this field is TRUE, the field “times of rapid guessing in a row” is incremented by one.
is rapid response?	main start main end answer hint scaffold scaffold start scaffold end	A boolean value, TRUE or FALSE, which indicates whether or not the student executed a “rapid response,” i.e. student performed *any action* before the amount of time required for the student to read the problem (main problem or scaffold problem) or hint at 400 words per minute. If the value of this field is TRUE, the student performed the current action (i.e. hint, answer, scaffold, etc.) before enough time had elapsed for the student to read the current problem or last hint at 400wpm.
is bottom out hint?	main start main end answer hint scaffold scaffold start scaffold end	A boolean value, TRUE or FALSE, which indicates whether or not the student requested the last hint available for the current problem (main problem or scaffold problem) in which the answer is given to the student. If the value of this field is TRUE, the student requested the last hint available and the field “times of bottom out hint in a row” is incremented by one; otherwise the value for this field is FALSE.
times of rapid guessing	main start	The number of consecutive answers in which the student was

in a row	main end answer hint scaffold scaffold start scaffold end	determined to be "rapid guessing". If a student is "rapid guessing," he or she submitted an answer less than 2000ms after submitting the previous answer, in which case the value of the field "is rapid guessing?" will be TRUE and this field will be incremented by one. If the value of this field reaches two, the gaming score for the current problem (main problem or scaffold problem) is updated.
times of bottom out hint in a row	main start main end answer hint scaffold scaffold start scaffold end	The number of consecutive problems on which the student requested the last hint available for the current problem (main problem or scaffold problem) in which the answer is given to the student. If a student requested the "bottom out hint" the value of the field "is bottom out hint?" will be TRUE and this field will be incremented by one. If the value of this field reaches three, the gaming score for the current problem (main problem or scaffold problem) is updated.

Appendix B - XML Specification for Structure of the Learning State Model

```
<nodes>
        <node>
            <id>1</id>
            <name>learning_state</name>
            <type>discrete</type>
            <values>2</values>
            <latent>no</latent>
            <field>learning_state</field>
            <within>
                  <transition>knowledge</transition>
                  <transition>answer_correct</transition>
            </within>
            <between></between>
        </node>

        <node>
            <id>2</id>
            <name>knowledge</name>
            <type>discrete</type>
            <values>2</values>
            <latent>yes</latent>
            <field>knowledge</field>
            <within>
                  <transition>answer_correct</transition>
            </within>
            <between>
                  <transition>knowledge</transition>
            </between>
        </node>

        <node>
            <id>3</id>
            <name>answer_correct</name>
            <type>discrete</type>
            <values>2</values>
            <latent>no</latent>
            <field>answer_correct</field>
            <within></within>
            <between></between>
        </node>
</nodes>
```