Machine Learning in Educational Technology

Laksh Advani
Department of Computer Science
University of Colorado Boulder
laksh.advani@colorado.edu

Erik Holbrook
Department of Computer Science
University of Colorado Boulder
Erik.Holbrook@colorado.edu

Kylee Budai
Department of Applied Mathematics
University of Colorado Boulder
kylee.budai@colorado.edu

Rashmi Shetty
Department of Computer Science
University of Colorado Boulder
rashmi.shetty@colorado.edu

Ankit Srivastava
Department of Computer Science
University of Colorado Boulder
ankit.srivastava@colorado.edu

Varsha Teratipally
Department of Computer Science
University of Colorado Boulder
varsha.teratipally@colorado.edu

Abstract

Much research has been done to improve educational technology. Generally, though, it focuses on cognitive and behavioral sciences and ignores the potential benefits of integrating machine learning techniques. Being able to classify questions as helpful, unhelpful, and misleading using learning techniques could lead to a significant increase in the overall effectiveness of current-day educational technology. Furthermore, an in-depth conceptualization of students' understanding and cognitive models would allow instructors to better assist in the learning process at an individual level.

In this work, we seek to apply machine learning techniques, namely prediction and clustering, as an avenue to model the abilities of students in their educational environments, and to elucidate commonalities among problems at multiple levels of granularity. We found success in both predicting student success on particular problems across domains, as well as revealed descriptive information about the structure and similarity of schools, students, and problems.

1 Introduction

The goal of this project is to investigate trends at the level of individual student, classroom, and across problems. We will apply a variety of unsupervised techniques to discover common errors. Woot Math [1] is an educational website that provides math lessons for students in elementary and middle school. Access to the site is purchased by the school and students participate in a computer-lab setting under the supervision of a teacher. The goal of Woot Math is to provide a fun, educational experience that allows students to gain experience with new concepts in math.

The layout of an example problem is presented in figure 1. The goal of the Woot Math platform is to provide an engaging interactive experience to teach students, with the help of visual aids, complex and traditionally difficult concepts like fractions and irrational numbers. Students manipulate objects

[1] www.wootmath.com

on a visual canvas to represent their solutions. Students are encouraged to draw and work out solutions using the canvas and their responses are judged based on total correctness.

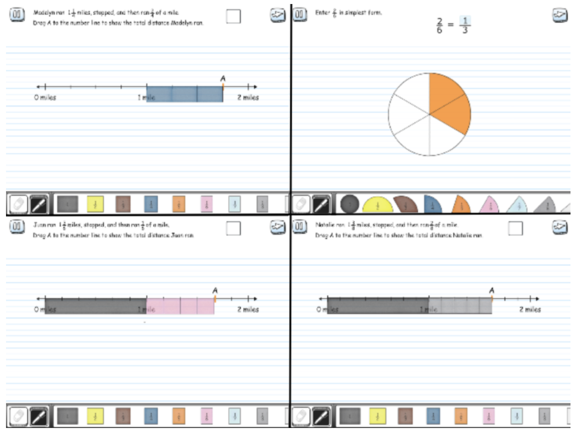


Figure 1: Four examples of canvases where students work on problems within the Woot Math platform

2 Dataset and Preprocessing

Woot Math was able to provide us with a dataset with ∼1 million entries and ∼100 attributes. Those attributes ranged from problem descriptions and types, internal measurements developed by the Woot Math team, and many other useful attributes. Figure 2 shows a small example of what the dataset looked like.

Student ID	Student Grade	School ID	Problem ID	Subject	Time Spent	Time Elapsed	Correct	Difficulty [0,1]
F8W8W9U0R2U1	3	F5asdncA2	t8suuCs7vN.l31_9_parta.b6c1c8dLXx	Fractions	24904	56930	0	.5792
22Q6W2E2A1	6	AFJ3fa022	x9tw0UBBWG.set1.f4F0t0H7hk	Decimals	43222	78220	1	.8346

Figure 2: An example of some of the fields within the Woot Math dataset.

Many of the attibutes went unusued but the ones which are used are briefly described below.

1. **school_id, student_id, problem_id** - Each school, student, and problem is assigned a unique ID. Student and school were anonymized for discretion.
2. **student grade** - Each student's grade is recorded which helped to classify schools as elementary, middle school, or high school.
3. **subject** - There are four different subjects of problems: fractions, decimals, ratios, and rationals.
4. **time spent, time elapsed** - time spent tracks the amount of time that a student spends on a given problem while time time elapsed tracks the amount of time that a student has spent on a lesson thus far.
5. **difficulty** - Each problem was assigned a difficulty by a board of educators.
6. **mastery mean and standard deviation** - These are internal measures which are assigned to each student based on previous and current performance. They aim to describe the expected probability of success of a given student.

7. **correct** - Indicates whether a student answered a given problem correctly.
8. **bonus** - Indicates whether a student performed well enough to be presented with a bonus question.

3 Methods and Results

We began our investigations by trying to analyze Woot Math's internal difficulty measure, to get a sense of how problems are rated, and compare this with our own results. We then proceed to apply prediction models in several contexts to determine what aspects of student performance might be understood in a timely, adaptive fashion. Finally, we utilize several clustering methods to extract information around school performance, providing insights on the dataset and generalizations of school performance.

3.1 Analysis of Internal Difficulty Measure

The Woot Math team developed an internal difficulty measure which considered the perceived difficulty of a given problem. In order to determine whether we should use the difficulty measure, we began with an analysis of the measurement.

3.1.1 Time Analysis Using SVM (Support Vector Machines:

One of the insights that we wanted to gain was to identify the amount of time it takes for question to be classified as a difficult question. The motivation behind using SVM was to get a clear and distinctive separator. Initially the SVM model took a very long time to run because of its complexity and the size of our data. What we did to improve this was to reduce our dataset sampling size and use a larger cache. This improved our speed to about 1 hour for a 1000 set sample.

Figure 3 represents the separation between the two classifications, we can see when we cross 20000 ms we are in the other class with higher difficulty levels.

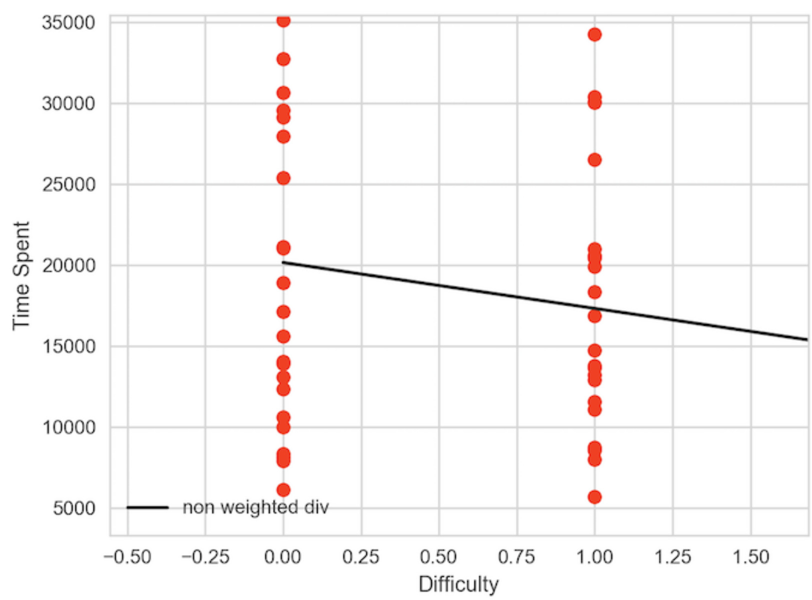


Figure 3: SVM on Time Spent and Difficulty

From our SVM analysis we could gather that if a question takes more than 3 minutes to solve it is generally considered to be a difficult question.

3.1.2 Analysis using Logistic Regression

To further analyze the difficulty parameter, we used it and to predict correctness and compared that to using accuracy to predict correctness. Realistically, a more difficult problem would yield a lower accuracy which would result in a student being more likely to answer that problem incorrectly. Overall, we wanted the following to hold for each problem.

$$\mathbb{P}(\text{correct} \mid \text{diff}) = \mathbb{P}\Big(\text{correct}\Big| \sum_{L}^{i} \frac{\text{nright}[i]}{\text{ntotal}[L]}\Big)$$

Using logistic regression, we were able to see that predicting correctness using difficulty yielded 90% prediction accuracy. This was the same percentage that predicting correctness based on accuracy was. From this, we were able to verify that difficulty is a good measure for accuracy.

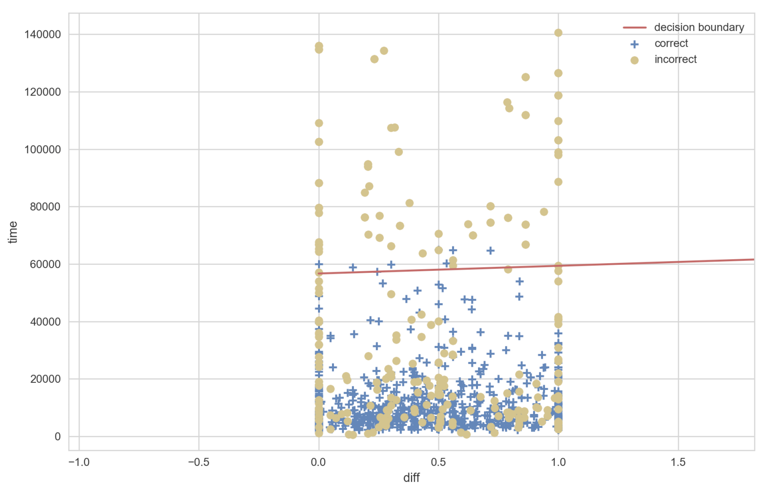


Figure 4: Logistic Regression Boundary on Difficulty and Time

3.2 Prediction

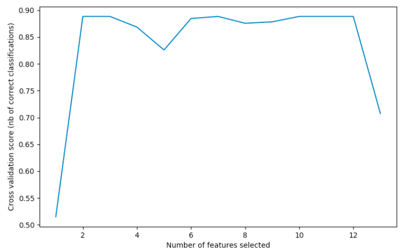
After determining that difficulty was a useful internal measure, we focused on prediction. There were two main areas of prediction in which we focused:

1. Predicting the occurrence of a bonus problem

2. Predicting the grade for a test in particular unit

3.2.1 Recursive Feature Elimination:

We implemented Recursive Feature Elimination with Sigmoid Logistic Regression to determine best features for the prediction of a bonus problem. Of the 39 features in a subset of the dataset, the below 6 were used (sorted by their rank) as the optimal parameters.

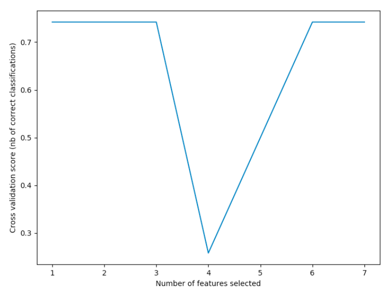


1. Time spent
2. Difficulty Measure
3. Problem ID
4. Number of Correct Answers
5. Number of Unattempted Problems
6. Number of Incomplete Answers

Figure 5: Cross Validation Score vs Number of Parameters and Best 6 Parameters for Predicting a Bonus Question

Figure 5 shows the cross validation score as a function of number of features. Noting that there was a peak around 6 features, we decided to use the highest performing 6 features.

We performed the same analysis to determine the best estimators to predict grade level of the student.



1. Unit Rank
2. Student Grade
3. Time Elapsed
4. Total Problems Answered
5. Total Problems Correct
6. Mastery Mean
7. Difficulty

Figure 6: Cross Validation Score vs Number of Parameters and Best 7 Parameters

Here there was not as obvious of a peak so we chose to use the best 7 parameters.

3.2.2 Predicting the occurrence of a bonus problem:

If a student is performing well in a test, they are presented with a bonus problem. For this reason, a bonus problem can serve as an indicator of success on a student's part. We performed regression analysis on the features in the dataset to predict if the student would be given a bonus problem or not. In this project, we have implemented Naive Bayes as the baseline and compared the results with Random Forest, Logistic Regression and Support Vector Machines. SVM, Logistic Regression and Gaussian Naive Bayes were implemented by tuning the parameters using GridSearchCV and cross validation. The accuracies yielded by each of these methods is given in the table below.

Method	Accuracy
Naive Bayes'	78.26%
SVM	73.91%
Logistic Regression	88.3%
Random Forrest	89.2%

3.2.3 Predicting student grade in tests within a unit:

In this task, we formulated a formula to classify students based on their grade level, number of correct answers, stars earned, time spent by high performing students (labelled as 1) and low performing students (labelled as 0). The analysis could help instructors plan their course structure and speed of teaching for better learning experience of students. Thus, the dependent output predicted variable would be high performing (value =1) or low performing(value =0) and the input independent variables would be number of questions, number of correct answers, number of wrong answers, time spent on the level, mastery level, mean difficulty of the questions in the level unit. Regression analysis was performed on these features and was implemented using SVM and Naive Bayes.

Method	Accuracy
Naive Bayes	78.26%
SVM	77.25%
Logistic Regression	74.230%

Figure 7: Prediction of performance level of student

3.3 Clustering

Clustering is a very useful technique that can give rise to many insights about data. We chose to implement clustering in order to separate schools based on performance. From there, we clustered the students within the schools for a higher resolution of the behavior within each cluster.

3.3.1 Clustering Schools - Organizing Data

In order to cluster schools, we had to select important features. We used many of the same features that we used for prediction in order to maintain structure. In order to cluster on schools, we were required to aggregate the data so that we had one feature vector for each school.

We averaged the amount of time spent on every problem within an entire school. Ideally, a school where students generally took longer to answer problems were less successful in answering those problems.

We incorporated many different accuracy measurements in order to consider accuracy in each of the different subjects. There are four terms labeled ratios_acc, decimals_acc, frac_acc, and rationals_acc, which aggregated data in each of the four subjects. For example, if students in a school answered 150 questions related to fractions and successfully answered 100 of them, frac_acc was set to 100/150. Further, if a school did not complete any problems in a given subject, their accuracy for that subject was set to zero. We also tracked an aggregate accuracy which was computed as the total number of problems attempted divided by the total number correct.

We also took advantage of the internal measures mastery_mean, mastery_std, and difficulty. The values assigned to each school was an average of the values across each student in the school.

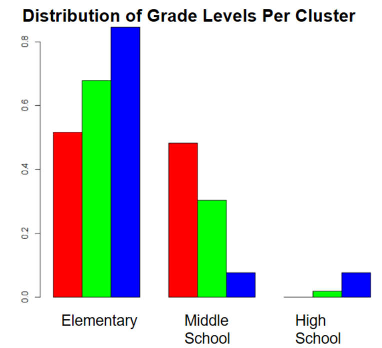
Finally, we were able to classify schools as elementary, middle, and high schools by looking at the average grade of each of the students and making the assumption that

$$\begin{aligned} \text{Average Grade } &< 6 \Rightarrow \text{ Elementary School} \\ 6 \leq \text{ Average Grade } &< 9 \Rightarrow \text{ Middle School} \\ 9 \leq \text{ Average Grade } &\leq 12 \Rightarrow \text{ High School} \end{aligned}$$

Using this assumption, out of the 98 schools, there were 64 elementary schools, 32 middle schools, and 2 high schools.

3.3.2 Clustering Schools - Results

Using three clusters resulted in an obvious separation of schools into highest performing, medium performing, and low performing. We aimed to cluster in a way that resulted in a relatively close number of schools in each cluster. Figure 8 shows the distribution of types of schools in each cluster (elementary, middle school, high school) while the table shows the number of schools in each cluster.



Cluster	Number of Schools
Red	29
Green	13
Blue	56

Figure 8: Grade level distribution and size of each cluster

The green cluster has far few students than the other clusters. Figure 9 shows that the green cluster is the highest-performing cluster and therefore it is sensible that the size of this cluster would be significantly smaller.

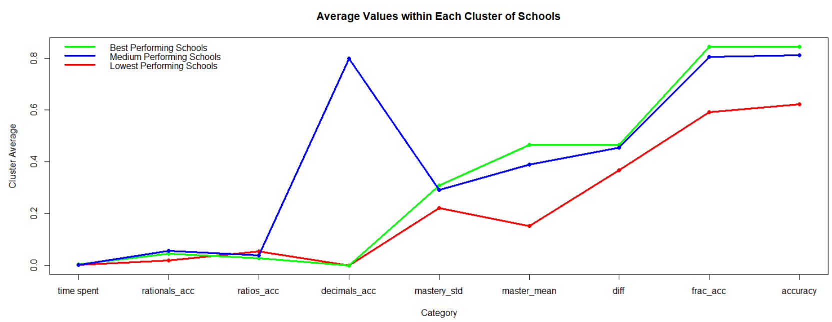


Figure 9: Visualization of cluster centers for each of the three clusters on aggregated school data.

One of the main concerns with this plot is the peak in the medium performing schools in the decimal_acc attribute. Recalling that decimal_acc was set to zero if no students in a school answered questions related to decimals, it is not difficult to understand this peak in performance. The following table shows the number of schools in each cluster that answered problems relating to decimals.

Cluster	Number of Schools that Worked with Decimals
Red	0
Green	0
Blue	13

This shows that all schools that fell within the blue cluster completed problems in decimals while no other schools did so which is a clear indicator as to why the accuracy of this cluster in decimals is far higher than the accuracies of the other clusters.

3.3.3 Clustering Students within School Clusters

A more interesting problem than clustering schools is the clustering of students. In order to address this problem, we clustered students within each of the school clusters. The reason for this was to provide a better understanding of the clustering of the schools. In order to prepare the data to do this, we used a very similar method as for the clustering of schools. The number of students in each cluster of schools is shown in the table below.

Figure 10 shows the averages within each cluster of students within the clusters of schools. The coloring is consistent with the previous section for easier understanding.

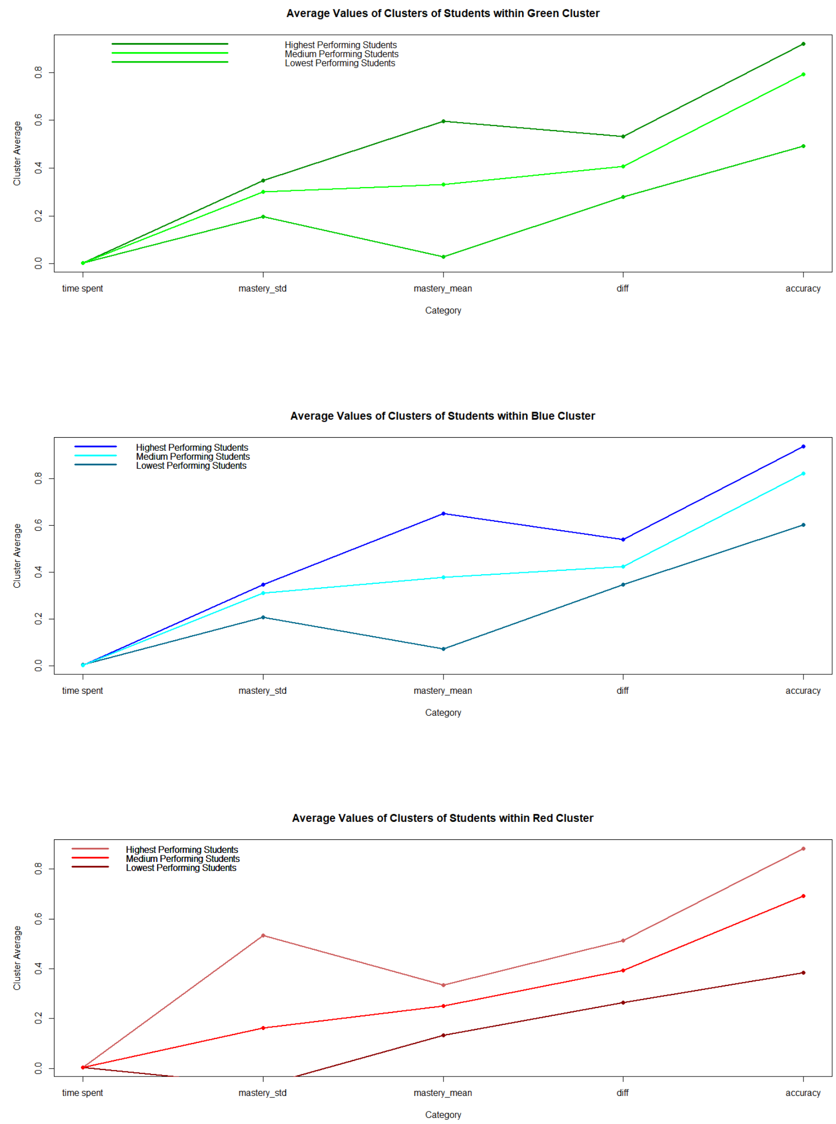


Figure 10: Clustering of Students within Clusters of Schools

There is a much more obvious separation between the clusters of students than the clusters of schools. The following table shows the percentage of students per cluster in each of the clusters of schools.

Student Cluster	Green	Blue	Red
Highest Performing Students	49.6%	38.8%	21.8%
Medium Performing Students	38.6%	39.3%	50.3%
Lowest Performing Students	11.8%	21.9%	27.9%

As expected, a higher percentage of students within the best performing schools were classified as high performing students. This verifies that the clusters performed in the previous section were valid clusters for schools.

3.3.4 Clustering of problems: Difficulty level

Beyond clustering schools and students, we also worked on clustering problems. We were able to identify the problems based on the difficulty level and time spent by the student. We have tried two approaches to identifies the clusters based on difficulty: hierarchical clustering and mini batch K-Means clustering. Both yielded similar results, as can be seen in figure 11.

Figure 11: Clusters of problems based on difficulty

4 Conclusions

In this work, we have analyzed the internal problem rating, investigated the efficacy of prediction methods, and revealed information about school-level performance in the Woot Math platform. We found that the internal proprietary rating given by Woot Math predicts the probability of success fairly reliably. We found that prediction techniques can be applied to problem attempts and reliably indicate the student's response; these results could be incorporated to Woot Math's platform to improve the adaptability and flexibility for both high- and low-achieving students. The differences between schools in clusters reveal how schools differ in explicit terms, and imply areas for improvement both in the classroom and in the Woot Math platform.

5 Future Work

Were we to extend this work, we would like to focus on better developing a cognitive model for the students. As of this writing, we were only able to use high level features and not well develop a deep set of well-engineered features to use as representation of student knowledge.

Each attempt of a problem generates a screenshot of the canvas where the student performed their work. This image set clearly contains a plethora of information; we initially intended on using these in our knowledge set, but this would require at the very least a heavy use of image processing, and probably would further entail using a convolutional machine learning approach to glean any useful information.

Finally, our techniques show useful aspects of the dataset already in existence in Woot Math servers. Our final future task would be to help the team implement our most effective methods into their production environment to effectively utilize our techniques to provide business value.

6 Acknowledgements

We would like to sincerely thank Woot Math for their generous contributions to this project. They have substantially assisted us in the process of data acquisition, cleaning, and modification, as well as providing us with areas research focus. We would also like to thank Professor Chenhao Tan and the CU Boulder Computer Science Department for providing us the opportunity to pursue and present this work.