Process Mining Combined with Item Analysis to Predict Efficient Use of Time on High-Stakes Assessments

Nathan A. Levin
Teachers College, Columbia University
nal2163@tc.columbia.edu

The Big Data for Education Spoke of the NSF Northeast Big Data Innovation Hub and ETS co-sponsored an educational data mining competition in which contestants were asked to predict efficient time use on the NAEP 8th grade mathematics computer-based assessment, based on the log-file of a student's actions on a prior portion of the assessment. In this work, a combined approach of process mining and item analysis was used to build a large set of features which were then trained with an Extreme Gradient Boosting machine learning model to classify students based on whether or not they would use their time efficiently. Predictions were evaluated throughout the competition on half of a hidden data set and then the final results were based on the second half of the hidden data set. The approach used here earned the top score in the competition. The work presented elaborates on the combined technique for analyzing computer-based assessment log-file data with the hope that this approach will offer valuable insight into future predictive model building in educational data mining.

Keywords: Process Mining, Educational Data Mining, Computer-based Assessment, Extreme Gradient Boosting

1. INTRODUCTION

Standardized computer-based testing is rapidly becoming a ubiquitous tool both for summative assessment as well as formative self-evaluation (Pechenizky, 2009). Up to the moment of the test, a teacher may be the one with the most direct influence on a student's performance; however, the moment the test begins the teacher is removed from the equation and relegated to the sideline. At the conclusion of an exam, a teacher can review the results and reverse engineer what might have happened, but retrospective knowledge is imperfect at best. If a student runs out of time on an exam, a teacher can ask the student which questions took the most time, but understanding why those questions took additional time and where the student might have felt anxiety is much more difficult for a teacher to ascertain.

With the advent of computer-based assessments, we now have access to real-time student behavior in the form of clickstream data representing each action taken by a student during an exam (Bannert 2014; Greiff, 2015). The immediacy of this information creates a unique window of opportunity for teachers to provide pertinent support to students, which is a key goal of learning analytics (Macfadyen & Dawson, 2010). Specific to this work, the

granularity of click stream data provides insights into ineffective and effective test-taking behaviors.

To push the field of educational data mining forward, the Big Data for Education Spoke of the NSF Northeast Big Data Innovation Hub and Educational Testing Service organized a competition in 2019 to engage leading researchers worldwide to develop metrics for measuring students' test-taking activities and attempt to predict effective and ineffective test-taking time management behavior. Effective time management increases assessment validity, because students are able to accurately demonstrate the knowledge they possess without running out of time (Ellis & Ryan, 2003). Accurately predicting time management is an invaluable piece of insight for teachers striving to reduce test anxiety which may interfere with optimal test performance (Stenlund, 2017).

The data set published for the competition consists of the log-files for all actions taken by a group of students in two blocks of a math test, Block A and Block B. The goal of the competition was to predict whether students would spend their time efficiently in Block B, based on the actions taken during Block A. Spending time efficiently was defined by two criteria: 1. Not running out of time on Block B, for which they were 30 minutes. 2. Spending a "reasonable amount of time" on each problem, defined as being above the 5th percentile in terms of time taken on each problem.

The input to each predictive model is a set of features derived from the clickstream data log of every action taken by a student in Block A. The output of the model was a binary value for each student indicating whether they would use their time efficiently in Block B (TRUE) or not use their time efficiently in Block B (FALSE).

Different approaches were taken by the top entries in the competition. The third place approach constructed many features (>4,000) using domain knowledge and automated feature engineering methods. The second place team constructed features based on process mining and then applied a genetic algorithm-based feature selection and modeling technique and then collated together the predictions from multiple models into an ensemble model to generate a single prediction.

The model presented in this work took first place in the competition by earning the highest score based on the second half of the hidden data set, although it was not the top scoring model on the first half of the hidden data set, which is an inconsistency that will be addressed in the further research section of this paper. The features were developed using sequential process mining techniques combined with item analysis conducted manually on the student log-files. The XGBoost Regressor model was applied to the final feature set (~330) to generate the final predictions which scored an AUC of 0.658 and a Kappa of 0.228.

Process mining offers a unique lens for uncovering the common navigational patterns across a population of students (Pechenizkiy, 2009; Bogarin, 2014) as well as potentially creating more interpretable machine learning models, because common processes can be extracted from event logs and teachers can readily interpret those processes. Bogarin (2018) describes process mining as "a bridge between data mining (DM) and process modeling and analysis" (p.1). In past work, process mining has been used primarily as a tool to identify

"patterns (strategies) repeated with relatively high frequency … that might correspond to … learning sessions of a set of learners" (Nesbit, p.6, 2007). This work takes the lead from Juhaňák (2019), using process mining to analyze student test taking, but it presents a unique focus on revealing and predicting effective time management techniques in a high-stakes testing environment.

2. METHOD

2.1. DATA

As previously mentioned, this work uses log-file data representing the actions taken by students during the 8th NAEP mathematics assessment in the 2016-2017 academic year. Students completed two blocks of math problems Block A and Block B. Each block contained a set number of problems and students had a 30-minute time limit to complete the problems in each block.

Each row of the data represents a single action taken by a student, most of which consist of the student clicking the mouse or typing on the keyboard. The data set is restricted by a terms of use agreement, therefore the table below is a fictional example of a row in the data set, although for all intents and purposes it would be extraordinarily difficult to differentiate the row below from a row in the actual data.

Table 1: A fictional example row of data from a student log-file

STUDENTID	Block	AccessionNumber	ItemType	Observable	ExtendedInfo	EventTime
12345678	A	VH987654321	MCSS	Click Choice	VH098810_5: checked	2/15/2017 2:11:12 PM

STUDENTID: A unique identifier for each student
Block: The block that the action happened in
AccessionNumber: A unique identification of a problem/item
ItemType: The type of the item
Observable: The type of the action the student took
ExtendedInfo: Additional information on the student action
EventTime: The timestamp of when the action was taken

The goal was to predict efficient use of time on Block B, based on log data provided for the first 10, 20, and 30 minutes of actions. To develop a model attuned to each of these data sets one of the first processing steps was to split the training data into 10, 20, and 30 minutes of actions. This initial clustering was based on the assumption that by separating the data into distinct groups the model would be more valid and less biased because the relative importance of different features fluctuates depending on the cluster being trained upon (Makhlouf, 2020).

These three data sets were used separately for both training and validation of separate models. A total of 1232 students were present in the training data set.

The test data set consisted of hidden data sets for 10, 20, and 30 minutes of actions. The test data sets contained 411, 411, and 410 students respectively. The test data set target labels were withheld throughout the competition, and then subsequently released at the end of the competition, therefore the performance of the models are further validated for having been trained with the test data set completely unseen. Although the questions in Block B will be different than the questions from Block A, we can assume that they cover a similar breadth and depth of material.

The target variable is a binary classifier indicating whether a student used their time efficiently on Block B of the exam. In the training data 744 students had a target value of "TRUE" indicating they used their time efficiently on Block B, while the rest of the students (438) had a target value of FALSE. The imbalance between classes is not enough to justify mitigation through a sampling method, although this may be a course of action for future work to take if carrying this model forward on similar problems.

2.2. Model Selection

XGBoost is an optimized distributed gradient boosting library, chosen for its speed, flexibility, and open source integration with the scikit-learn machine learning package in Python. Like other gradient boosted tree algorithms, XGBoost relies on optimizing a set (an ensemble) of classification/decision trees by iteratively adding trees ("boosting") in a way that decreases the gradient of a specified loss function. The specific implementation of gradient tree boosting within XGBoost is beyond the scope of this work but suffice to say it includes a gradient boosted descent tree regressor called XGBRegressor which was used as the predictor in this model. The data pre-processing and feature engineering was done in Python using the Pandas data manipulation and analysis package. All code and links to the dataset used in this work are freely available on GitHub[1].

2.3. Evaluation Metric

In order to compare all models built for the competition the evaluation criteria was defined by the metric in figure 1 combining AUC and Kappa.

$$AdjustedAUC = \begin{cases} 0 & , if\ AUC < 0.5 \\ 2(AUC - 0.5) & , otherwise \end{cases}$$

$$AdjustedKappa = \begin{cases} 0 & , if\ kappa < 0 \\ kappa & , otherwise \end{cases}$$

$$AggregatedScore = AdjustedAUC + AdjustedKappa$$

[1] https://github.com/nal2163/NAEPDataMining

Figure 1: Evaluation metric combining AUC and Kappa

AUC (Area Under the Curve) compares the false positive rate to the true positive rate of the model. A value of 0.5 however, would indicate a model performing only as well as random chance which motivated the use of Adjusted_AUC for the first component of the evaluation metric. Cohen's Kappa, which compares observed accuracy to expected accuracy of a random model, was the foundation of the second component of the evaluation metric. In the case of Kappa, a value below 0 indicates worse than random chance, hence the adjustment above. By combining the two metrics we obtain a balanced evaluation of whether a model is classifying students correctly while avoiding classifying students incorrectly.

In the training of the XGBoost model, the package allows for the use of a custom scoring function during cross-validation. By writing the evaluation metric above as a function, it could be used to evaluate the hyperparameters of the model, as discussed further in section 2.5 below.

2.4. Feature Engineering

The features for the model were generated through multiple iterations of both item analysis and process mining. Due to the robust nature of the XGBoost model the initial approach was to generate as many features as possible before paring them down based on importance to the model. The complete list of features varied for the 10, 20, and 30-minute models with 272, 334, and 368 features respectively. The increasing number of features was a factor of the additional data in the log-files for 30 minutes of actions, as some of the features sought to capture expanding features such as the number of actions taken each minute. For example, in the 30 minute feature set there is an action rate for each minute resulting in 30 features as opposed to 10 and 20 in the other feature sets.

2.4.1. Item Analysis

Item analysis of the log-files generated features that can be categorized into frequency related and time related features as articulated by Chen and Cui (2020). Frequency related features represent the number of times a student performed specific actions on the exam. In this case the word action is loosely defined to capture a multitude of behaviors from revisiting questions to clicks per minute. The time related features indicate the average and total amount of time a student spent on specific activities during the assessment. The time related features are particularly important because they have been found in the past to help distinguish solution finding behavior from random guessing, which in this case would be classified as the difference between effective and ineffective time management (Juhaňák 2019). A third category of features added to the model were basic statistical descriptors of feature sets, such as standard deviation of time spent on questions and maximum and minimum number of clicks in a 1-minute period.

2.4.2. PROCESS MINING

Process mining was used to generate additional features in the form of common sequences of actions taken by students. Fluxicon's Disco was used to create a process map for students clustered into two groups based on their classification in the training data. Disco facilitates the filtering of process maps based on duration and frequency of edges between specific events. This allowed for rapid generation of the reasonably comprehensible process maps seen in figure 2 and figure 3. To produce the FALSE process map in figure 2, an edge limit was set to only include transitions that had occurred >500 times. For the TRUE process map in figure 3, an edge limit of >1000 was used to account for the larger number of students with the TRUE classification.

A process map was generated for students flagged as false (figure 2) and students flagged as true (figure 3). Common processes were then identified in the two process maps and turned into features. This step was based on work done by van der Aalst (2011) indicating that process models can offer insight into reliable time predictions. The common processes were represented in the form of 2 – 4 successive actions, and then features were generated based on the frequency and duration of those sequences.

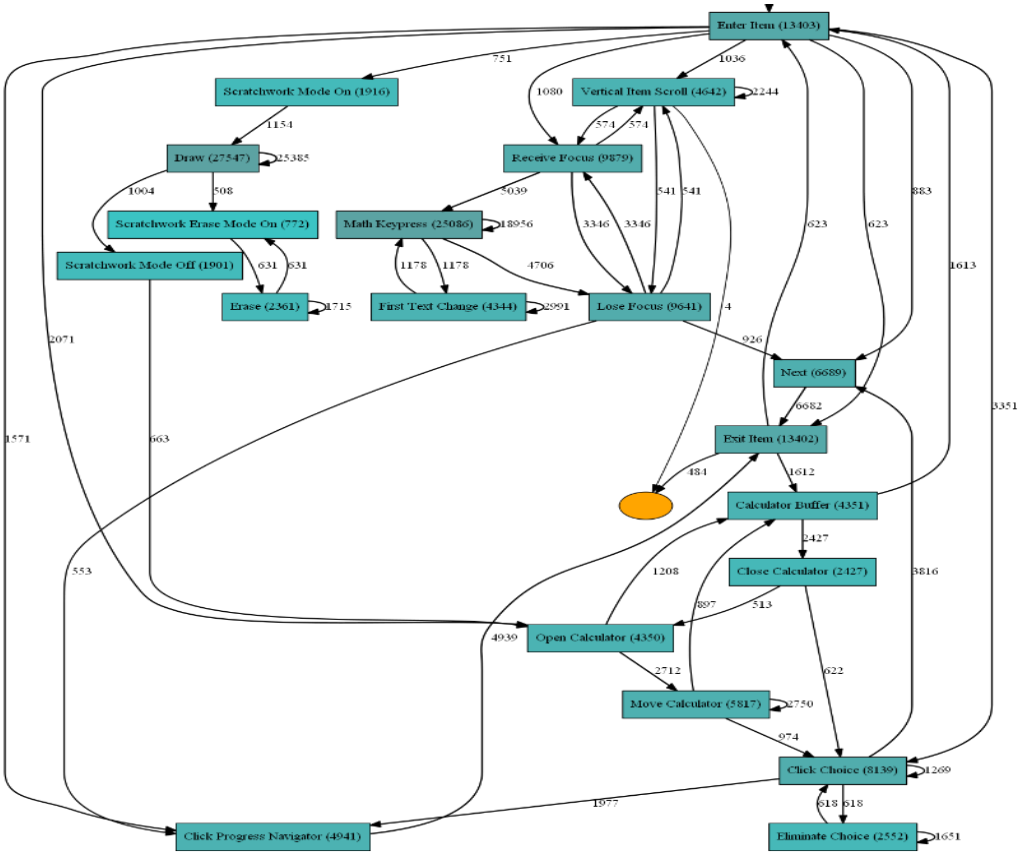

Figure 2: Process map of actions taken by students with FALSE classification

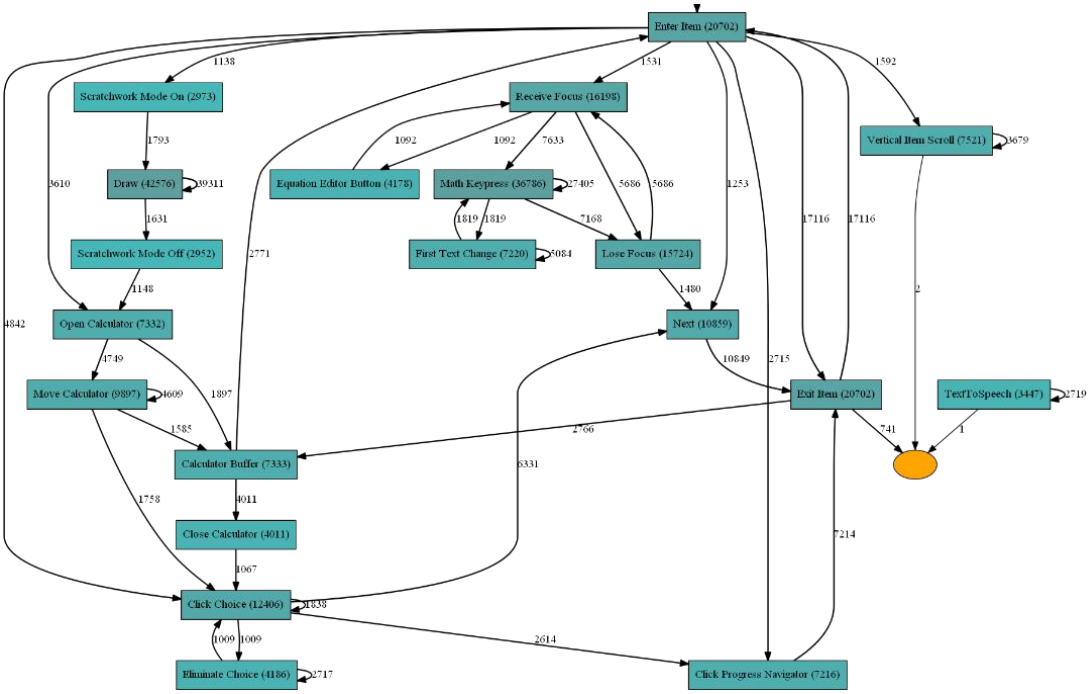

Figure 3: Process map of actions taken by students with TRUE classification

2.4.3 Feature Importance

A model with >300 features has several notable drawbacks. The first being the significant amount of time it takes to train the model and tune the hyperparameters. The second important reason is the low interpretability of a model with too many features. A teacher attempting to help students reduce test anxiety by practicing better time management skills would have a difficult time pulling anything useful from a model with >300 features. Finally, with only 1200 students in the training set, 300 features has the potential to lead to overfitting and would be less likely to retain validity when applied to unseen data.

In this work, a rudimentary iterative feature selection algorithm was used to reduce the number of features. The XGBoost model was trained using a random search for quick hyperparameter tuning, and then the model was evaluated to determine the importance of specific features. In this case feature importance was determined by the weight of the features as measured by the number of decision trees in which they are present. The top 75% of features were retained and then the process was repeated until <30 features remained. This set of features was then used to train the final model for each subset of data.

Each of the three models (10, 20, and 30 minutes) generated a different set of top features indicating that at each duration different indicators were more predictive of efficient time use. The top ten features for each model are graphed in figures 4, 5, and 6 below with their F score which is the same weight value used in the feature selection process, calculated by summing together the number of times a feature occurs in the trees of the model. In tables 2, 3, and 4 following each graph there is a description of the groups of features in each model. These descriptions offer a glimpse at some of the insights into test-taking behavior offered by this

predictive model. A description of every feature in the model would be verbose for the purposes of this paper, instead the top ten features of each model are described in detail here which is sufficient to demonstrate the insights that can be gleaned from examining this model.

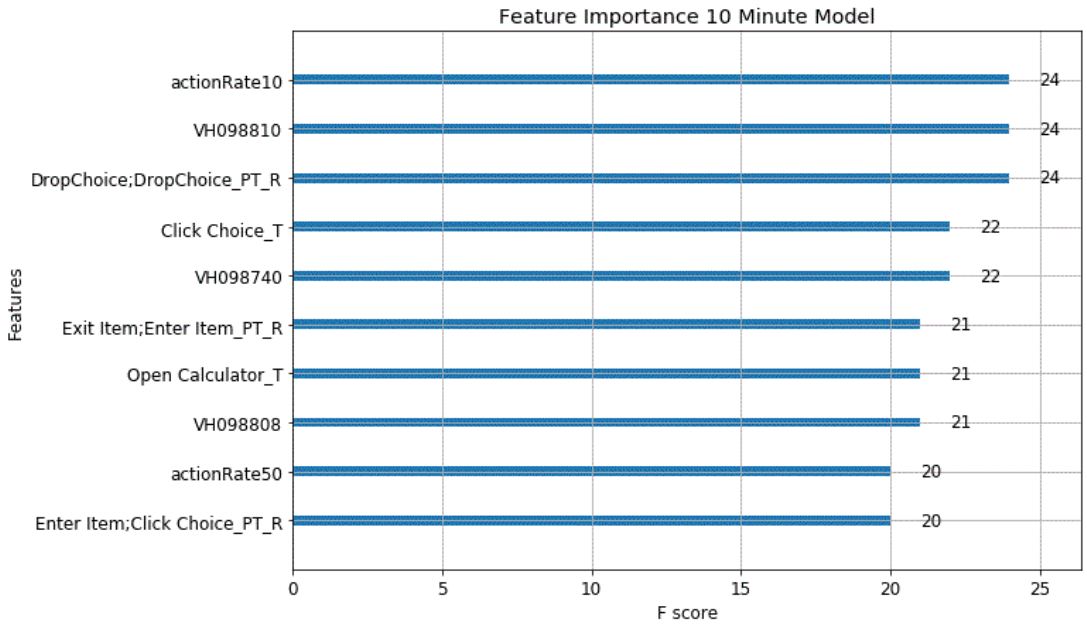

Figure 4: The top 10 features used in the 10-minute model.

Table 2: 10-minute model - feature description

Feature	**Description (for first 10 minutes)**
actionRate10, actionRate50	actionRate is a calculation of the rate at which a student is taking actions. i.e. actionRate40 is the amount of time it took a student to go from 30 total actions to 40 total actions.
VH098810, VH098740, VH098808	The total time spent on these specific questions on the exam.
DropChoice;DropChoice_PT_R, Exit Item; Enter Item_PT_R, Enter Item; Click Choice_PT_R	These features represent average time spent performing the sequence of actions specified. e.g. DropChoice;DropChoice DropChoice is the action of choosing an answer from a dropdown menu twice in a row. PT = Process Time, R = Rate.
Open Calculator_T	The total time spent after taking the "Open Calculator" action and their next action. Intended to capture the lag time between opening the calculator and doing their next action.

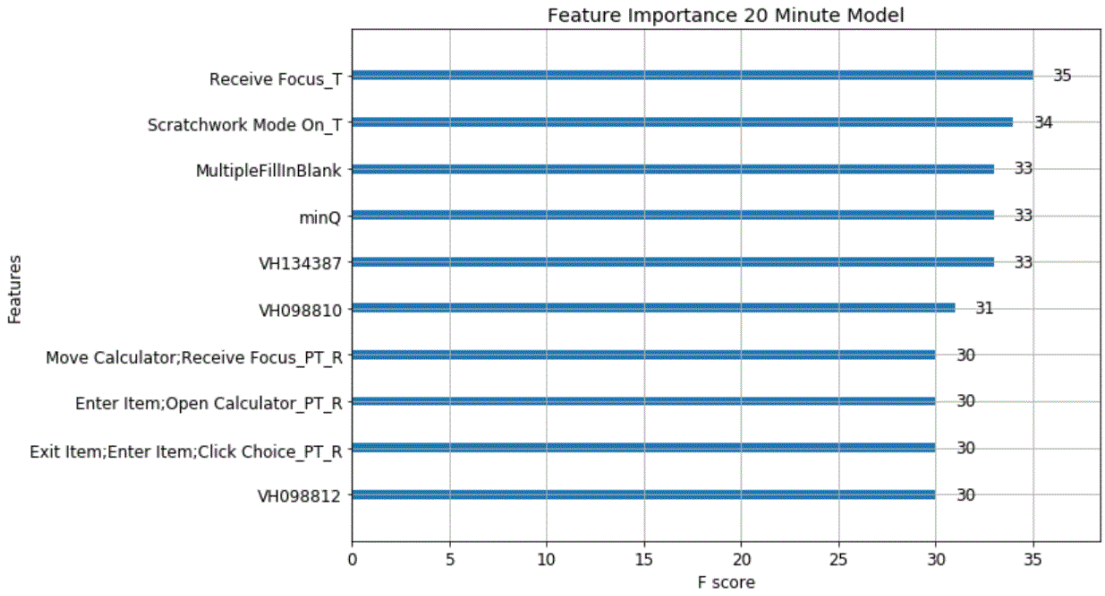

Figure 5: The top 10 features used in the 20-minute model.

Table 3: 20-minute model - feature description

Feature	**Description (for first 20 minutes)**
Receive Focus_T	Total time spent between receiving focus and the subsequent action. The receive focus event occurs on MultipleFillInBlank questions and seems to indicate the student clicking on a different component within the question.
Scratchwork Mode On_T	Total time spent between turning on scratchwork mode and taking the subsequent action.
MultipleFillInBlank	Total time spent on the MultipleFillInBlank question type.
minQ	The minimum of the set of the times spent on each question.
VH098812, VH098810, VH124387	The total time spent on these specific questions on the exam.
Move Calculator;ReceiveFocus_PT_R	Move Calculator is the action of moving the calculator. Receive Focus indicates clicking on an element of a MultipleFillInBlankquestion. This process indicates moving the calculator and then clicking on a component of the question. "_PT_R" signifies the average time spent on this process, PT = Process Time, R = Rate.
Exit Item;Enter Item;Click Choice	This process indicates leaving one exam item, opening the next item, and choosing an answer. This feature is the total number of times the student has done this process.

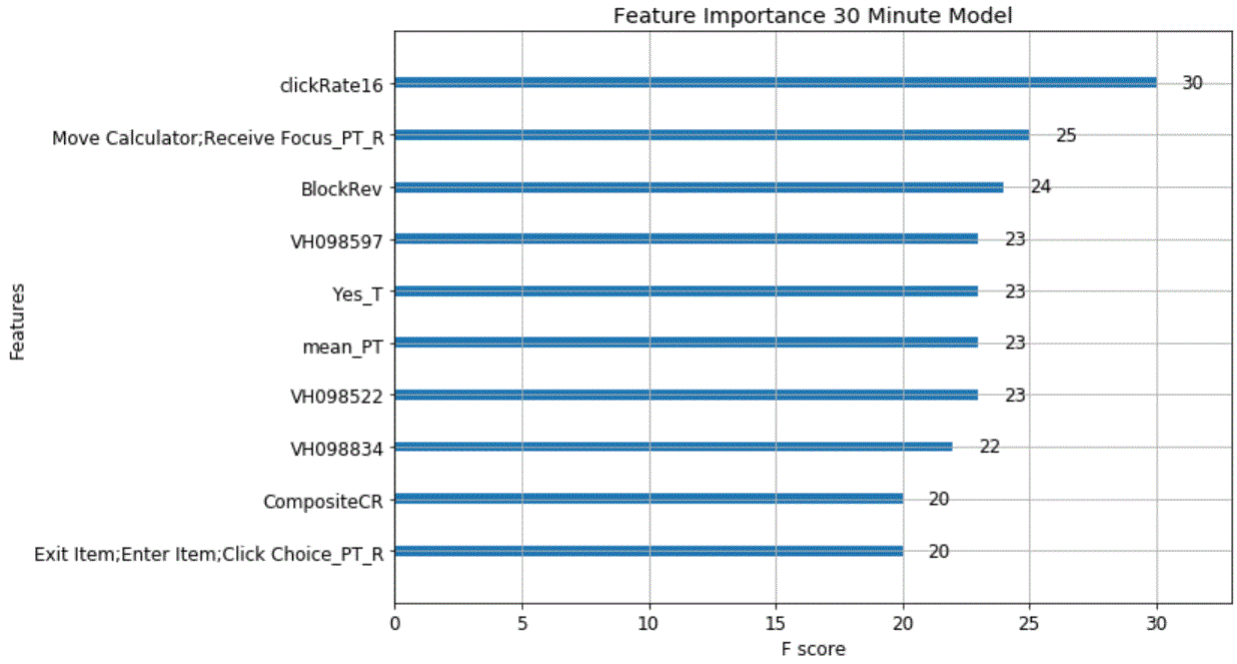

Figure 6: The top 10 features used in the 30-minute model.

Table 4: 30-minute model - feature description

Feature	**Description (for full 30 minutes)**
clickRate16	clickRate# indicates the number of actions in the #th minute. This feature is the number of actions taken between 15 and 16 minutes
Move Calculator;Receive Focus_PT_R	Move Calculator is the action of moving the calculator. Receive Focus indicates clicking on an element of a MultipleFillInBlankquestion. This process indicates moving the calculator and then clicking on a component of the question. "_PT_R" signifies the average time spent on this process, PT = Process Time, R = Rate.
BlockRev	The total time spent on the BlockRev item, which is a screen allowing students to review and navigate to different items on the exam.
VH098597, VH098522, VH098834	The total time spent on these specific questions on the exam.
Yes_T	The amount of time spent between taking the "Yes" action and the subsequent action. "Yes" is only available toward the end of the exam when students receive a "TimeLeftMessage".
Mean_PT	Average time spent on all processes.
CompositeCR	Total time spent on the composite constructed response questions.
Exit Item;Enter Item;Click Choice_PT_R	This process indicates leaving one exam item, opening the next item, and choosing an

	answer. This feature is the total number of times the student has done this process.

The important features identified above were not always the same depending on the tuning of the hyperparameters. However, they were chosen based on being the top features for the competition winning versions of the models and for their inclusion in the top performing models across multiple rounds of hyperparameter tuning. In the results section an exploration of the meaning of the important features, especially those appearing in multiple models is warranted.

2.5. Hyperparameter Tuning and Model Evaluation

10-fold cross-validation was used to tune the hyperparameters instead of holding out a set of data for multiple reasons. Firstly, the data set was only 1233 students, and with such a small data it is difficult to withhold a meaningful number of students for validation without also significantly detracting from the data available for training. Secondly, the final test data would act as the hold out set and would serve as a more robust evaluation of the model in the end.

To tune the large range of hyperparameters used in XGBoost the RandomizedSearchCV from the scikit-learn python package was used to conduct a random search of the hyperparameter space. A grid search could be done of the space to conduct a more thorough survey of the permutations of hyperparameters, but a randomized search is capable of finding models that are almost as good as the best in much less time (Bergstra & Bengio, 2012).

3. Results

3.1. Model Prediction

As expected, the predictive power of the model increased as more information became available to the model. This can be seen in the increased AUC from the 10 to the 20 to the 30-minute model (table 5). That said, the increase was small, and we can see that the Kappa value remained relatively unchanged, indicating that the longer duration models were slightly better at avoiding false positives.

The results below are for the combination of the two halves of the hidden data set. For the competition, the first half was used for a public leaderboard during the competition, and the second half was used for the final results of the competition. On the public leaderboard this model earned a 0.643 AUC while on the final leaderboard this model earned a 0.672 AUC. This difference was enough to position this model in the top 10 on the public leaderboard, but moved it to the top position on the final leaderboard. The discrepancy in performance between the two hidden data sets indicates that this model does not demonstrate robust reliability. A larger test data set would be needed to confirm the predictive power.

Table 5: Performance data for the model predictions

	Hidden Test Set				
	AUC	**Adjusted AUC**	**Kappa**	**Adjusted Kappa**	**Evaluation Metric (Adjusted AUC + Adjusted Kappa)**
10 minute model	0.643	0.286	0.227	0.227	0.513
20 minute model	0.658	0.315	0.225	0.225	0.540
30 minute model	0.676	0.352	0.215	0.215	0.567
Combined	0.658	0.315	0.223	0.223	0.538

3.2. FEATURE IMPORTANCE

In all three models time spent on specific questions was one of the key predictive features. This leads to the potential conclusion that a few key questions may require the time management skills that demonstrate students' overall ability to manage their time. Without knowing the student performance, or any of the details of the questions themselves, further conjecture is impossible. At the very least, however, it is worth considering that students may be revealing their time management techniques during a few key questions on an assessment.

One of the most exciting commonalities was the presence of at least one process feature in the top ten for each model. In all three cases we see that a feature based on the average amount of time spent on a process was important to the model. This finding validates the use of process mining to identify time-management behavior in the log-files generated by students in a high-stakes testing environment.

4. DISCUSSION

The utility of a test is diminished by the multitude of extraneous factors beyond the students' knowledge and skills which can impact their final score (Stenlund, 2017). To this end, the goal of this study is to use educational data mining to open a small window into the test-taking, by investigating the actions a student is taking during an exam and how those actions can predict the student's efficient use of time.

The results of this work are encouraging. They present the possibility that we can predict with greater than chance accuracy whether a student will use their time efficiently on an exam. Practically this model could be adapted to the classroom testing environment and enable teachers to further understand and support the test-taking strategies being used by their students.

In terms of generalizability, the model showed strong performance in both validation as well as on the unseen test set. The performance was improved through the process of feature selection, which follows the work done by Chen and Cui (2020) where they found that fewer features led to less variance, and a sacrifice in training performance actually resulted in an increase in testing performance. This work demonstrates that finding the most accurate predictive models can be a filtering process during which the initial attempt of the 'kitchen sink'

approach of throwing in all the features offers a promising starting point from which to winnow down to the best features.

The most influential features in the machine learning model used in this work indicate that significant insight can be gained from delving deeper into the way students are spending their time on specific exam items. The processes that students repeat while taking an online assessment are manifestations of cognitive processes that students are undertaking and likely influenced by their affective state.

For the field of Educational Data Mining, this work puts forth a combination of process mining and item analysis for generating insights from computer-based assessment log data. Process mining was developed for business operations, but it can yield unique features when applied to educational log data. This work demonstrates that at the micro-level process mining can offer additional insights into the test-taking behaviors students are demonstrating.

5. Further Research

For the ends to justify the means, the results of any educational data mining must be in service of teachers and even more importantly students who can make use of the research. As Bogarin (2014) offers, "we want to improve both the performance and comprehensibility of the models obtained" (p.1). Admittedly this is a mandate somewhat beyond the scope of this work, but one that hopefully will be mitigated in the extension of these findings which offer an entry point into further analysis of and applications for computer-based assessment log-file data.

The work performed here demonstrates a small example of combining process mining with item analysis, but the application to predicting efficient use of time is just one possible application. In the future this same feature extraction technique could be applied to predicting student performance in terms of both learning and test-taking strategies. By understanding the minutia of the processes successful test takers use, we may be able to gain additional insight into how to design tests with better validity while coaching students to achieve their highest potential.

This data set also offers the potential to investigate the efficacy and appropriateness of extended time accommodations. By predicting efficient use of time, we may be able to better understand how and why some students benefit from extra time on assessments and how those accommodations can be made effectively.

Finally, I hope that this work will be just a starting point for refining this model of predicting efficient time use. During the hyperparameter tuning and training processes, the stochastic nature of machine learning algorithms resulted in varied results at times, and depending on the day this model could have been outperformed by the other models in the top swath of the competition. None of the top 3 models were able to consistently achieve an AUC score above 0.7. In future work, through collaboration, breaking the 0.7 barrier is all but certain.

Acknowledgements

The author would like to thank the Teachers College Data Science in Education Association, Karen Zhou, Alexander Audet, Benjamin Finkelstein, and Dr. Charles Lang for their invaluable conversations and support.

REFERENCES

Amrieh, E.A., Hamtini, T., and Aljarah, I. 2016. Mining Educational Data to Predict Student's academic Performance using Ensemble Methods. *International Journal of Database Theory and Application 9*, 8, 119–136.

Bannert, M., Reimann, P., and Sonnenberg, C. 2014. Process mining techniques for analysing patterns and strategies in students' self-regulated learning. *Metacognition and Learning 9*, 2, 161–185.

Bergstra, J. and Bengio, Y., 2012. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, *13*(1), pp.281-305.

Bogarín, A., Cerezo, R., and Romero, C. 2018. A survey on educational process mining: Survey on educational process mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8*, 1, e1230.

Bogarín, A., Romero, C., Cerezo, R., and Sánchez-Santillán, M. 2014. Clustering for improving Educational Process Mining. 5.

Chen, F. and Cui, Y. 2020. Utilizing Student Time Series Behaviour in Learning Management Systems for Early Prediction of Course Performance. *Journal of Learning Analytics 7*, 2, 1–17.

Greiff, S., Wüstenberg, S., and Avvisati, F. 2015. Computer-generated log-file analyses as a window into students' minds? A showcase study based on the PISA 2012 assessment of problem solving. *Computers & Education 91*, 92–105.

Juhaňák, L., Zounek, J., and Rohlíková, L. 2019. Using process mining to analyze students' quiz-taking behavior patterns in a learning management system. *Computers in Human Behavior 92*, 496–506.

Macfadyen, L.P. and Dawson, S. 2010. Mining LMS data to develop an "early warning system" for educators: A proof of concept. *Computers & Education 54*, 2, 588–599.

Makany, T., Engelbrecht, P.C., Meadmore, K., Dudley, R., Redhead, E.S., and Dror, I.E. GIVING THE LEARNERS CONTROL OF NAVIGATION: COGNITIVE GAINS AND LOSSES. 7.

Makhlouf, J. and Mine, T. 2020. Analysis of Click-Stream Data to Predict STEM Careers from Student Usage of an Intelligent Tutoring System. *12*, 2, 18.

Nesbit, J.C., Zhou, M., Xu, Y., and Winne, P.H. Advancing Log Analysis of Student Interactions with Cognitive Tools. 20.

Pechenizkiy, M. and Tr, N. 2009. Process Mining Online Assessment Data. *Educational Data Mining*, 10.

Sedrakyan, G., De Weerdt, J., and Snoeck, M. 2016. Process-mining enabled feedback: "Tell me what I did wrong" vs. "tell me how to do it right." *Computers in Human Behavior 57*,

352–376.

Stenlund, T., Eklöf, H., and Lyrén, P.-E. 2017. Group differences in test-taking behaviour: an example from a high-stakes testing program. *Assessment in Education: Principles, Policy & Practice 24*, 1, 4–20.

van der Aalst, W.M.P., Schonenberg, M.H., and Song, M. 2011. Time prediction based on process mining. *Information Systems 36*, 2, 450–475.