Riiid Answer Correctness Prediction

Xinxuan Lu[1] Zhaoxuan Hu[1] Beilei Guo[1] Erqian Xu[1]

***Abstract*— This report presents our analysis on EdNet dataset. We aim to conduct a predictive project which predicts students' performance on next questions based on features such as their historic activities and behaviors, other time related features, etc. We have utilized multiple algorithms which include traditional machine learning methods such as logistic regression (LR), logistic regression CV (LRCV), as well as more advanced methods, such as light gradient boosting (LGBM), recurrent neural network (RNN), and long short-term memory (LSTM). The results have shown that our enhanced methods outperformed the traditional approaches in general, in which LGBM achieved the highest accuracy score among all the models (accuracy=0.78). Limitations and implications on intelligent curriculum design and AI tutoring systems will also be discussed.**

I. Introduction

Knowledge tracing is the task of modeling students' knowledge over time. There has been a growing body of research that demonstrates the importance of "knowledge tracing" to predict students' academic performance on their future interaction [1]. Previous studies have suggested the use of such prediction to promote tailored learning experience for students across countries [1][2]. However, due to the challenges of documenting student activities, such dataset remains scarce. In this report, we have analyzed a subset from a large dataset named EdNet because of its availability. EdNet is a unique dataset which is known as one of "the largest public IES (Interactive Educational Systems) datasets", containing 784,309 students and 131,417,236 interactions over an expansion of more than two years [1]. The subset we used in this study were retrieved from Kaggle.com [3]. It involves 101,230,332 interaction records from 393,656 users, 13,524 unique questions and 419 unique lectures.

We selected this data set for our final project due to several reasons. First, knowledge tracing is an important topic as mentioned above. The findings will shed light on learning path recommendation, intelligent curriculum design and AI tutoring systems. Second, this topic is a great fit for our team with a diverse background. Our team consists with two doctoral students from Warner School of Education and two master candidates from the Department of Data Science. This education related topic is a research focus for our members from Warner school, and in the meanwhile, it is an exciting machine learning prediction task for our members from data science department.

*This work is the final project for DSCC465 Statistical Machine Learning at the University of Rochester.

[1]The authors are graduate students from Goergen Institute for Data Science, University of Rochester, NY, USA

There are four major steps in this report. To begin with, we conducted a literature review on the importance of knowledge tracing and potential approaches for the prediction task. Next, we had a holistic view of the given data set through explanatory data description and then performed preprocessing. Subsequently, to predict whether students would be able to correctly answer their next questions, we applied several different traditional machine learning methods, including logistic regression (LR) and logistic regression CV (LRCV). To improve the performance of our prediction task, we harnessed enhancement methods, namely light gradient boosting (LGBM), and some neural network models including recurrent neural network (RNN), and long short-term memory (LSTM). The results have shown that our enhanced models outperformed the traditional approaches in general. Lastly, the limitations and future improvements were discussed in the paper.

II. Literature Review

Knowledge tracing (KT) is a well-established technique of student modeling, it has been widely adopted to predict test performance [2]. Bayesian knowledge tracing and deep knowledge tracing models are the two mostly used KT techniques in the field of computer-assisted education [4][5]. Literature shows that methods based on Recurrent Neural Networks (RNN) such as Deep Knowledge Tracing (DKT) and Dynamic Key-Value Memory Network (DKVMN) outperformed traditional methods [6].

However, DKT and DKVMN face the issue of not generalizing well while dealing with sparse data [6]. Based on the literature, a variety of methods were developed to achieve better performance in handling data sparsity problem. For instance, Gong and his colleagues [2] found Expectation Maximization (EM)+ KT produce higher predictive accuracy than Brute Force (BF) + KT in Performance Factor Analysis. Pandey and Karypis [6] aslo developed a Self Attentive Knowledge Tracing (SAKT) model that identifies the knowledge concepts from the student's past activities and predicts his/her mastery based on the relatively few knowledge concepts that it picked. This model performs better than methods based on RNN.

Time is an important feature that has been used to extend the knowledge tracing model in the literature. For intsance, Wang and Heffernan [7] conducted an analysis that use discretized first response time data to predict students' correctness of the next question, and leveraged the results in to a knowledge tracing model. In another study, a group of researchers [8] modeled the effect of forgetting in Bayesian

knowledge tracing. They examined knowledge tracing prediction on student responses where a day or more had elapsed since the previous response and found that knowledge tracing over predicted these data in a consistent manner. Literature has confirmed the value of time in modeling student knowledge.

Other than time, item difficulty also has been combined into knowledge tracing model and showed affirmative results in model improvement. In 2011, Pardos and Heffernan [9] conducted a study using datasets from two intelligent tutoring systems with knowledge tracing and found substantial performance gains in KT model when adding item difficulty and keep the modification limited to changing the topology of the model. Another study [10] even combined item response theory (IRT) model and a knowledge model that is based on the deep neural network architecture. It is shown that deep learning based knowledge tracing model outperform traditional knowledge tracing model without the need for human-engineered features. For instance, Yeung conducted a study that use dynamic key-value memory network (DKVMN) to make deep learning based knowledge tracing model explainable, and able to provide an interpretation of both students and items. Specifically, Yeung used the IRT model to assess the probability that a student will answer an item correctly using the item difficulty as well as the students' estimated ability. Results of this study showed a positive effect of IRT model on the performance of deep learning knowledge tracing model.

III. Data

The data set we retreived from Kaggle Competition is provided by Riiid Lab[1] and collected from a TOEIC studying app Santa. It is a subset of the whole EdNet data set. The dataset[3] involves five components: "train.csv" as the training set and "example_test.csv" as the test set; "questions.csv" and "lectures.csv" as the metadata for the students' previous interaction such as the questions posed to the participants, and lectures they have watched; "example_sample_submission.csv" to obtain the accuracy score from the Kaggle submission. Because of our purpose, we focus on the preceding four sets: the training set, test set, question set and lecture set.

In the training set, there are 101,230,332 observations and 10 columns including one Boolean variable *prior_question_had_explanation* to indicate whether or not the participants have seen correct answer or explanation after completing the prior question; three ordinal variables *row_id*, *user_id*, *content_id* to show the code for the row, user, or user interaction; one binary variable *content_type_id* to see if the event was question posing or lecture watching; four numerical variables *timestamp* (i.e.,time in milliseconds between this user interaction and the first event completion from that user), *user_answer* (i.e., user's answer to the question), *_correctly* (i.e., if the user responded correctly), and *prior_question_elapsed_time* (i.e., average time in milliseconds it took a user to answer each question in the previous question bundle). The user number in train set is 393,656 and the average interactions of each user is 257. The interaction records consists with two types of actions: watching lectures and answering questions. For lectures, the data set only record the watching action, while for question part the data set record the answer of the student. Moreover, the data set provide basic information for all 13,524 unique questions and 419 unique lectures which contains the numerical coded content. Detailed content has not been provided but these numerical information is enough to cluster all these contents. We also used isnull() function to detect missing data. Altogether, there are 2,351,538 missing data in *prior_question_elapsed_time*, and 392,506 in *prior_question_had_explanation*. Model test will be processed on a timestamp based iteration API[3] which has simulated students' cognitive changes. The API will load up to 1 GB of the test set data with 1000 in each batch and 2.5 million questions in all.

The test set has 104 entries and 11 columns in total. The features are similar to training set with an exception of columns named *prior_group_responses* and *prior_group_answers_correct* which are both string data types. The question set contains 5 columns including *question_id*,*bundle_id*, *correct_answer*, *part* and *tags*. The lecture set has 4 columns which involves *lecture_id*, *part*, *tag*, and*type_of*. There is only one missing data in the *tags* feature in question set, and no missing data in lecture set.

According to our coding book, *user_id* is the ID code for the user. Here, we plotted user action distribution (Figure 1) and number of actions for our top 40 users (Figure 2). The minimum of the action number is 1, the maximum is 17,917, and the range is 17,916. The *content_id* is the ID code for the user interaction. We selected the first 30 most frequently used content id. As we can see in Figure 3, the most frequently used one is ID #6116, and the frequency is 213,605; the second one's ID is #6173, the frequency is 202,106. After applying value_counts() function, we also found that one of the least frequency ID number is 10,005, and the frequency is 1. The range of frequency between items are 213,604 (range = 213,605-1).

Fig. 1. User Action Distribution

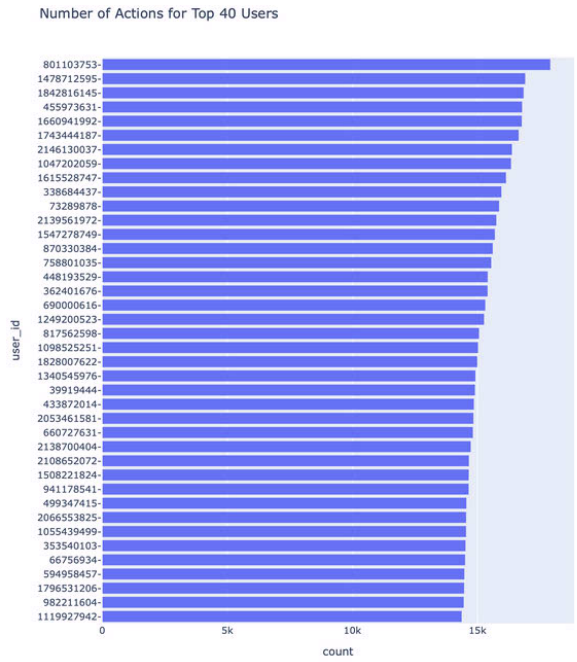

Fig. 2. Number of Actions for Top 40 Users

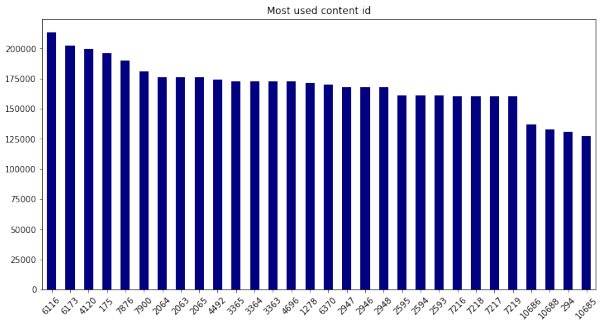

Fig. 3. Mostly Used Content ID

According to instruction, *task_container_id* is the ID code for the batch of lectures or questions. As we can see in Figure 4, the most frequently used *container_id* is #14 with the frequency of 804,285, the second most frequently used *contain_id* is 15 which has the frequency of 798,539. The least frequently used *task_container_id* is #9926 which has been used 170 times. The range of frequency in *task_container_id* is 804,115 (range=804285 - 170). The feature *prior_question_elapsed_time* is to measure the time a user took to answer their previous question bundle. As we can see in the Figure 5, the time each user took to answer their previous question bundle varies among different users.

As we mentioned above, we have two types of events in this context. Event 0 is a question being posted to the user, event 1 is users' lecture watching. Here, timestamp is the time between the first event and the current one. Seen from Figure 6, We have 393,656 users who spent 0 day between the current event and their precious event. According to

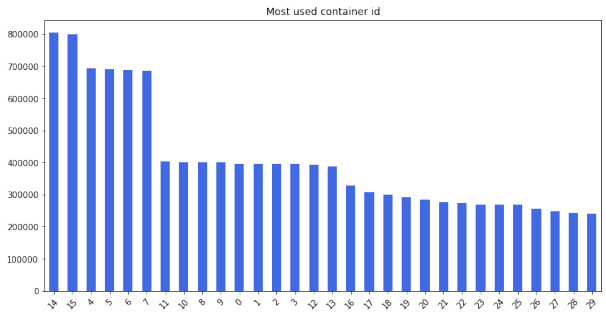

Fig. 4. Mostly Used Container ID

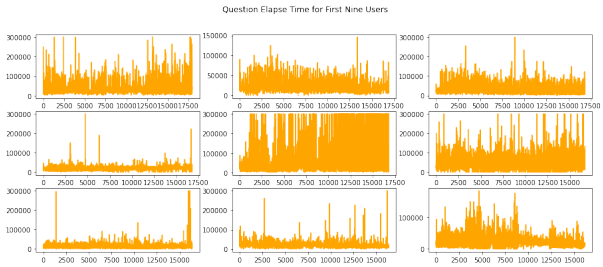

Fig. 5. Question Elapsed Time for First Nine Users

content_type_id, it is equal to 0 when the event is a question being posted to the users, 1 when the event is lectures. As shown in Figure 7, a majority of events are question, and only 1.94% is lectures.

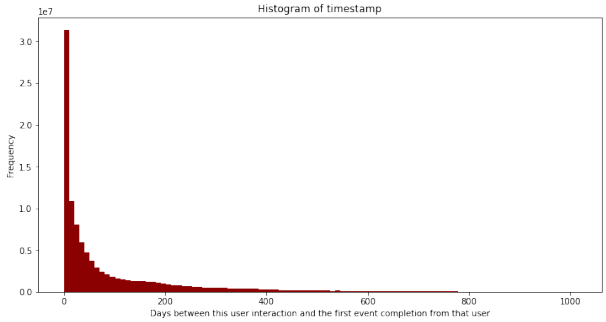

Fig. 6. Histogram of Timestamp

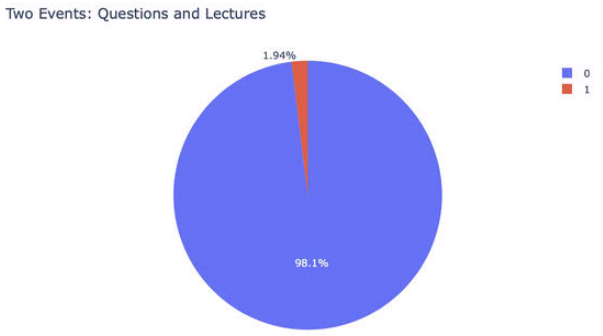

Fig. 7. Percentage of Two Events: Questions and Lectures

The column of *answered_correctly* is a measurement to

see whether the user responded to the questions correctly. As we can see in Figure 8, significantly more questions were answered correctly than those were answered poorly. The number of questions were answered correctly is 65,244,627, while the number of question were answered wrong is 34,026,673. Figure 9 shows us the relationship between content_id and correct rate, when *content_type_id* is equal to 0 which means when the event is a question being posted to the users.

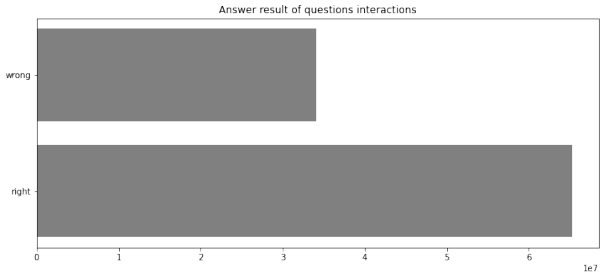

Fig. 8. Answer Result of Questions Interactions

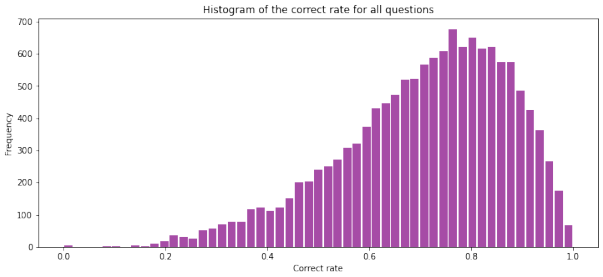

Fig. 9. Histogram of the Correct Rate for All Questions

In question data set, the variable *tags* is the detailed tag codes for the question. Some has one, others have multiple tag codes. As seen in Figure 10, tag #92 is the top one tag with the highest number of questions 2269. Tag #86, on the other hand, only has one question. Based on the correct rate of answering questions, we can see in Figure 11 the top easy questions' correct rate could be up to 99.3% to 100%. On the other hand, the lowest correct rate is only 9.18%.

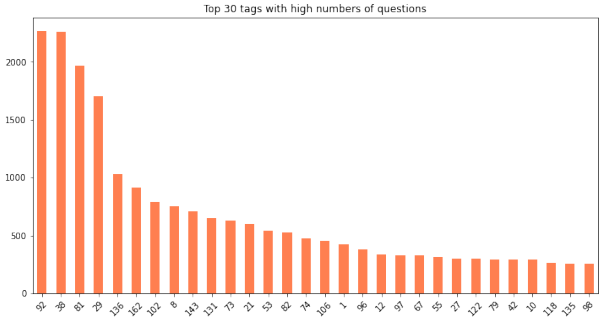

Fig. 10. Top 30 Tags with High Numbers of Questions

The feature *tag* is the tag code for the lecture. The meaning of the tags are not provided here, but we still can see

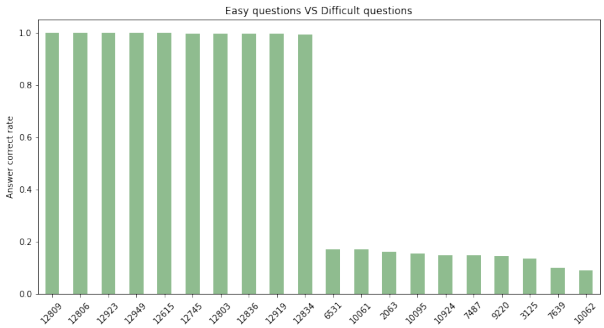

Fig. 11. Easy Questions vs Difficult Questions

some information from Figure 12. Tag #136 has occurred 7 times, while tag #2 only occurred once. Every lecture has a core purpose. The column *Type_of* is a brief description of main purpose of the lecture. Altogether, there are four different types of lecture purposes: concept, solving question, intention and starter. The frequency of concept as purpose is 222, solving question is 186 times, intention is 7, and lecture as starter is 3 times (Figure 13).

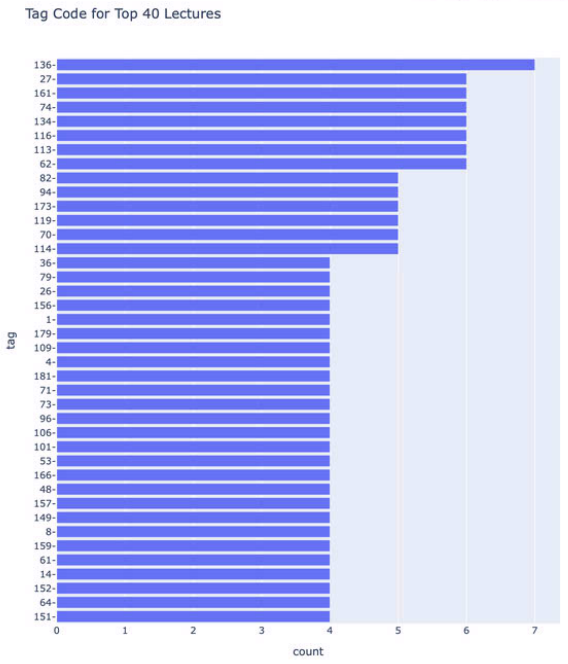

Fig. 12. Tag Code for Top 40 Lectures

IV. Hypotheses and Goals

The panel dataset includes features related to correctness of answers as well as the variation of students' behaviour over time. We intuitively assumed that time period can have influence to students' performance. To verify our assumption, we would sort our dataset by timestamp and take the average correct rate of students as the performance at each time point. By comparing the trend of performance in test data and that predicted by ARIMA model, we can have a general

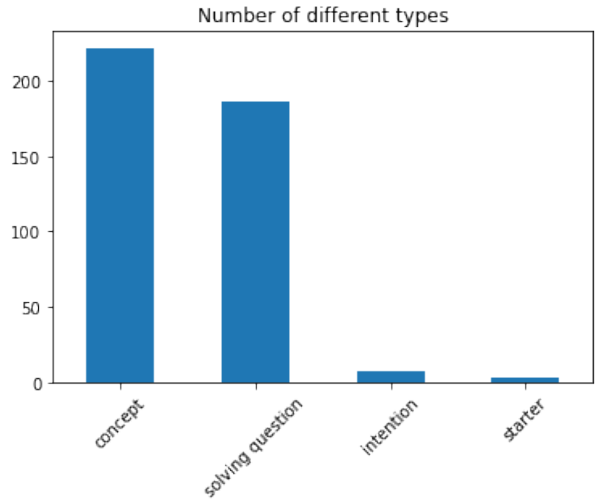

Fig. 13. Number of Different Types in Lectures

view of whether time is an important factor to students' behaviour. We would first harness given features and apply traditional machine learning models such as logistic regression and logistic regression CV to predict students' behaviour. Then, we could utilized more advanced models such as gradient boosting classifier model, vanilla RNN and LSTM to compared the results with traditional models. This enhancement combines the effect of both time and features that can be utilized to do prediction. The goal is to find deterministic factors of students' performance and make accurate prediction. We also hope to help students design interaction plans to reach the best learning efficiency based on our analysis.

V. Data Pre-processing

We mainly used three data files in the data set. The interaction file saves all user action including completing questions and watching lectures. The other two data sets include detailed information about all questions and lectures appeared. So, what we need to do is combine variable information into the data set. The data includes but is not limited to the user's current knowledge level and the difficulty of the question . This task is a supervised learning model and our goal is to predict users' performance in the next question. Specially, at what probability will the user correctly answer the next question, given the historical user record and question information. The original data set has in the processed data set we will include only the question answering records. We use all available data about users and topics as features, train the model, and finally seek a reasonable mapping relationship to predict the user's performance on the next topic. For each records, we will append statistical data based on the user's current performance and the information about questions and knowledge. Specially, we notice that for each questions we have different tags. A question may involve multiple knowledge points (for example, a composition topic may involve both vocabulary and grammar knowledge), and the user's mastery of these knowledge points will affect the user's answer to this question at the same time. This is tough to process as each questions has different numbers of different tags. We use clustering method and separate all questions into 22 groups. This method will be explained in detail in Section V-B. The original interaction data has included timestamp and questions for each interactions, 6 valuable features in all. Obviously, these feature data are not enough to train the model to obtain effective prediction results. In order to complete the task, we need to further expand the number of features. All features and the explanation has been listed in Appendix I.

A. Time Feature Engineering

The original task belongs to time series analysis, and we need to simulate his knowledge level through a series of previous behaviors. The problem brought about by this is that some users have rich behaviors on the platform (up to 17,917 behavior records), while others may be new users, and they have less than 10 behavior data in the database. Such users 30% of the total number of users. If time series analysis is used purely, the prediction results of these users will not be guaranteed to be accurate. One strategy is that we take all the time-dependent variables as input features of the model, and train the model by combining the user's own feature data, so that we can combine the data of all users to make predictions. This strategy is reasonable because users with similar characteristics have similar knowledge levels at the same point in time. So we can use the data of user1 to help predict the performance of user2. The input data will record the current relative time point of the user (that is, the time span from the user's first behavior to this piece of data, in milliseconds), the user's time span from the last behavior, and the mod of these timestamps by days and months.

B. Question Feature Engineering

Most questions can be related to different knowledge. Which means to it can include different numbers of different tags for each question. This will bring inconvenience for us to build features for them. So, here we include all tags and some basic information into one label which we called it tag-group. Our purpose is to classify all the questions, the logic behind it is not important, the point is to group all similar topics into one category. This way we can combine multiple problem variables into a single variable. This can be understood as a clustering problem, in this project, we use kmeans clustering to solve this problem. We take the tag list and part as feature and output the classification result. All 13524 questions are clustered into 22 groups. The PCA plot shows in Figure 14. we can see barely from the PCA plot that there are around 6 groups. However, it performs better in the accuracy in our test model with 0.74 in 100000 sample model. So we choose this tag-group as our feature. Here we append the performance of a typical user in the current question's tag-group as feature such as correct number and correct rate.

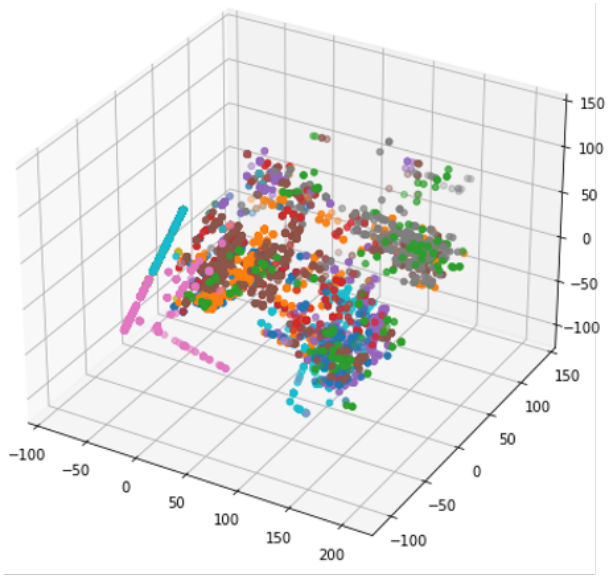

Fig. 14. PCA plot of all questions

C. Lecture Feature Engineering

For the questions that the user has been exposed to, we can make predictions based on the user's past performance on the questions. For questions that the user has not been exposed to, we need to make judgments based on the user's classroom records. For example, the user has not been exposed to topics related to "vocabulary", but has already learned a number of lectures related to vocabulary. In this case, the user's lecture browsing records will also partially reflect the user's mastery of knowledge. Therefore, we will count all the lecture browsing records of users so far, including the number of lectures viewed and the number of lectures related to topics.

D. Other Feature Engineering

We also added the user's short-term quiz records to the dataset. We've all been in a situation where we have a difficult question on a test and get nervous. Even if the questions that follow are simple, we may make the wrong answer. Extracting short-term time series data can help us identify this situation.

We also set some typical characteristics about the TOEIC exam and the SANTA platform, such as the questions completed by the user may reflect whether he or she is a VIP user or whether the test is a diagnostic test. A diagnostic test is a special reminder in many exams that it differs in difficulty from other questions and will not count as a score, but is used only to balance out differences in scores between students.

E. Important Features

We test the sample data set on a gradient boost model to see whether we have features closely related to the aim label. The result in Table I shows that these 10 feature are closely related to our aim result. cum_crate sum_avgt refers to the correct rate and cost time of all question answering record. Which is obviously reflect the ability of users qc_r and qp refers to the correct rate and appearance probability of the question which refers to the difficulty of questions. Which is also make sense as difficult questions are hard to be answered correctly. tsli refers to the time slapped since last question passed. We clipped the time to 20 minutes that is all tsli over 20 minutes are seemed as 20 minutes. As 20 minutes not doing next question means you have quit the app and do other things.pp ratio refers to the rate of question in part 1,3,6,7 in the history question, these question can only be entered by paid vip users.

feature	importance
cum_crate	11085
qc_r	10946
qp	10042
sum_avgt	9933
tsli3	8593
tsli	8444
clipped_tsli	8389
tsli2	8123
pp_ratio	8046
lp_ratio	7884

TABLE I

TOP 10 IMPORTANT FEATURES IN SAMPLE TEST

VI. MODEL AND ANALYSIS

A. Logistic regression

Logistic Regression is an extensively employed classification algorithm in machine learning that uses n-dimension features to determine an outcome which decide the label of a typical sample. The outcome is measured with a probability variable. In our binary classification problem, if the output probability is bigger than 0.5, we would predict that the user can answer the question correctly. Since it is simple to realize and achieves very good performance with linearly separable classes, we firstly chose this logistic regression as our baseline model and made two attempts on it.

1) Training by complex features: In first attempt, we just used the features generated by data preprocessing step described above to train logistic model. However, the accuracy of model is only 0.66. We further applied logistic regression with cross validation (LRCV) by harnessing these features, the accuracy value is still 0.66.

2) Training by simpler features: In the second attempt, we used a simpler set of features, which including user id, content id, content type and prior question elapsed time. The logistic regression model trained by simpler dataset can surprisingly achieve an accuracy of 0.73.

This implies that the complexity of feature can not determine the performance of logistic regression model for our data. The reason can be that the data is not linearly separable. The linear logistic model is unable to capture information essential for classifying this dataset. Therefore, the application of other advanced model is highly required.

B. Long short-term memory (LSTM)

Recurrent Neural Network (RNN) is a kind of artificial neural network that can take into account historical information learned from previous inputs. However, traditional RNNs are limited to store information for short duration due to the gradient vanishing problem, which means that the gradient may shrinks as it back propagates through time [11]. Such issue can be mitigated by Long short-term memory neural network (LSTM-RNN). It is capable to look back in time for more than 1000 time steps, depending on the complexity of the network [9].

In our problem, whether a user can answer current question correctly or not can be dependent on its activities and performance in previous timestamp. For example, user who have attempted the same question before are more likely to answer correctly. If the correct rate of a user shows an obvious increasing trend in previous time steps, he or she will probability behave well in next question. Since LSTM model can build connection with previous input and memorize information learned from past, it seems that LSTM is a appropriate method to predict users' correctness based on their performance before.

There are several important gates in LSTM neural network structure, which can determine what information in the sequence is important to keep or throw away.
I: input gate, controls how much we want to input into the cell gate.
C: cell gate, decides how much we would write into the input cell.
F: forget gate, represents how much we want to forget from the previous cell state.
O: output gate, means how much we hope to reveal from the cell state into the hidden state.
The computation process can be expressed as follows:

$$\begin{pmatrix} i \\ f \\ o \\ c \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \tag{1}$$

where W is the parameters in neural network, x_t is input variable and $ht-1$ stands for hidden state.

$$c_t = f \bigodot c_{t-1} + i \bigodot c \tag{2}$$

$$h_t = o_t \bigodot tanh(c_t) \tag{3}$$

This is the gate architecture of a single LSTM layer. To improve robustness of our model, we combined two LSTM layers to form a LSTM neural network. Each LSTM layer has 50 neurons. We also randomly dropped out 20 percent of nodes between two adjacent layers. It is an computationally cheap technique to avoid overfitting and widely used in many deep learning models. Then after data preprocessing and feature generation, we sorted our training dataset by user id to form an ordered sequential panel dataset and used such data to train our LSTM neural network.

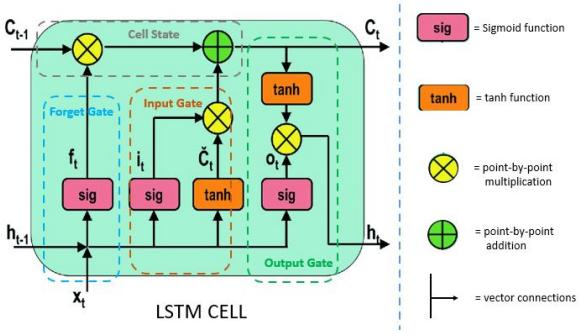

Fig. 15. Gate structure of LSTM layer [12]

C. Gradient boosting decision tree

Gradient boosting decision tree is a kind of ensemble learning. In this model, each decision tree of a fixed size is a base learner. Each base learner continuously learns error from its prior tree and is trained sequentially to minimize the loss function of its last classifier [13]. In this way, many weak learners are combined together and are finally converted to a stronger classifier. Compared with building a single tree with high depth and too many leaves, the ensemble learning method tends to be more flexible and can reduce the risk of overfitting. The algorithm of gradient boosting tree can be explained as below [14]:

Gradient boosting at the m-th step would fit a decision tree h_m. Let J_m be the number of its leaves, so the tree partitions the input space into J_m disjoint regions R_{1m},... ,$R_{J_m m}$ and predicts a value b_{1m},...,$b_{J_m m}$ in each region respectively. The output of tree h_m for input x can be written as the sum:

$$h_m(x) = \sum_{j=1}^{J_m} b_{j_m} I_{R_{j_m}}(x) \tag{4}$$

where $I_{R_{jm}}(x) = 1$ if $x \in R_{jm}$, $I_{R_{jm}}(x) = 0$ if $x \notin R_{jm}$.
Then the coefficients b_{jm} are multiplied by some value γ_m, chosen using the line search so as to minimize the loss function, then the model is updated as the follows[15]:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x) \tag{5}$$

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)) \tag{6}$$

Since the size of our dataset is too large and it is time-consuming to run normal gradient boosting, we use light gradient boosting method (LGBM). It is an efficient and effective implementation of the ordinary gradient boosting algorithm with a type of automatic feature selection as well as focus on boosting examples with larger gradients [16]. Instead of growing a tree row by row, LGBM constructs a tree leaf-wise by selecting the leaf that will yield the largest reduce in loss [17]. Additionally, LGBM does not search the best split point on sorted feature values. It executes a highly optimized histogram-based decision tree learning algorithm, which yields great advantages on both efficiency and memory consumption [17]. These modifications can

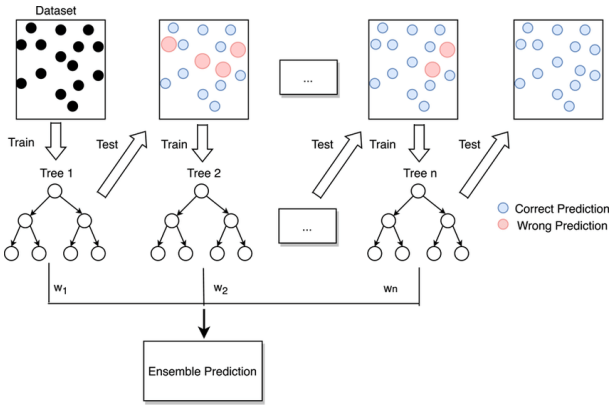

Fig. 16. Main idea of gradient boosting tree

result in a dramatic speedup training and improved model performance.

VII. Expectations and Result

Through the research in this paper, we hope to build a knowledge body tracking model based on machine learning. The model will reasonably predict the current knowledge level of students based on their past problems and assist the online platform to customize the optimal learning plan for them. By comparing the efficiency of different kind of models, we hope to select the most appropriate model for our problem and get some inspiration from feature importance.

Now that's make a comparison of these models. As mentioned before, the accuracy of logistic regression is only 0.66 if we use features mentioned in data processing section. If we change the features to a simpler set, the accuracy can be improved to 0.73.

For more advanced model, in overall, the accuracy of LSTM model is around 0.73 and that of lgbm can reach about 0.78. We discovered that the accuracy of prediction is not stable. For users with much records of interactions in training set, the accuracy of predicting their performance can achieve 0.78. By contrast, for users with few times of interaction in training dataset, the prediction accuracy is just slightly more than 0.7. This implicates that the performance of LSTM network depends on the length of sequential data. More interaction data of a user can pass more information to the model and thus bring us a relatively higher accuracy. However, in our dataset, many users only appear for less than 5 times, which make recurrent neural network model difficult to capture the variation of behaviour of those users over time and further affect the classification efficiency. This is the limitation of LSTM model for this problem.

By contrast, the Light gradient boosting model can be more stable. Even if we changed the size of data used for training, the accuracy of model always fluctuates around 0.78. But this also reflects that the information obtained from current features would not increase with the amount of data. Although LGBM is reliable, it is also difficult for us to further improve the accuracy of this model by enlarging the training dataset. Since we have combined many features which can show how some important factors such as the correct rate of users, the question elapsed time and accumulative number of explanations change over time, it can be also hard to further improve feature complexity. Therefore, light gradient boosting model also has its own limitation.

Model	LR1	LRCV	LR2
Accuracy	0.66	0.66	0.72

TABLE II
Baseline Model and Its Accuracy Score

Model	LSTM	RNN	LGBM
Accuracy	0.73	0.74	0.78

TABLE III
Enhancement Model and Its Accuracy Score

VIII. Conclusion

In summary, our findings have shown that it is feasible to predict student performance on their future instructional interactions based on their past performance and problems. The result urges and provides an opportunity for educational practitioners to give more scaffolding to certain students on specific content knowledge when the prediction result is negative. Also, the prediction could act as part of formative assessment, so that educators can reflect to their curriculum design in a timely manner.

In the meanwhile, we are fully aware of our limitations and future directions. First, We could include detailed information on what questions and lectures are about and the meaning of each tag code for more interpretative content-based analysis. Second, we could balance the percentage of two events (i.e, answering questions, watching lectures), now the percentage of lectures are too low compared to the questions. Third, We could include the whole data set rather than this subset to more accurate results. Lastly, due to restricted computational resources, we were only able to conduct analysis on 80%-90% of the given data. In the future, we could have more powerful computational resources to conduct a more comprehensive analysis. There are also model related limitations. As we mentioned previously, weakness exists in each of our prediction methods. In addition, we haven't applied all possible approaches to improve the accuracy because of our time constraint. Some neural network like self-attentive knowledge tracing (SAKT) and SAINT is also appropriate for this educational prediction problem. The predictions of SAKT are made based on relatively few past activities, it handles the data sparsity problem better than the methods based on RNN [18]. It has outperformed the state-of-the-art models for knowledge tracing. In future work, we intend to implement this model on our data, which may yield a more accurate prediction.

REFERENCES

[1] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo, "Ednet: A large-scale hierarchical dataset in education," in *International Conference on Artificial Intelligence in Education*, pp. 69–73, Springer, 2020.

[2] Y. Gong, J. E. Beck, and N. T. Heffernan, "Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures," in *International conference on intelligent tutoring systems*, pp. 35–44, Springer, 2010.

[3] R. Lab, "Riiid answer correctness prediction."

[4] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon, "Individualized bayesian knowledge tracing models," in *International conference on artificial intelligence in education*, pp. 171–180, Springer, 2013.

[5] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," *Advances in neural information processing systems*, vol. 28, 2015.

[6] S. Pandey and G. Karypis, "A self-attentive model for knowledge tracing," *arXiv preprint arXiv:1907.06837*, 2019.

[7] Y. Wang and N. T. Heffernan, "Leveraging first response time into the knowledge tracing model.," *International Educational Data Mining Society*, 2012.

[8] Y. Qiu, Y. Qi, H. Lu, Z. A. Pardos, and N. T. Heffernan, "Does time matter? modeling the effect of time with bayesian knowledge tracing.," in *EDM*, pp. 139–148, 2011.

[9] Z. A. Pardos and N. T. Heffernan, "Kt-idem: Introducing item difficulty to the knowledge tracing model," in *International conference on user modeling, adaptation, and personalization*, pp. 243–254, Springer, 2011.

[10] C.-K. Yeung, "Deep-irt: Make deep learning based knowledge tracing explainable using item response theory," *arXiv preprint arXiv:1904.11738*, 2019.

[11] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[12] G. Singhal, "Introduction to lstm units in rnn."

[13] N. Ranjan, "Gradient boosted trees.."

[14] Wikipedia, "Gradient boosting."

[15] A. Thennakoon, C. Bhagyani, S. Premadasa, S. Mihiranga, and N. Kuruwitaarachchi, "Real-time credit card fraud detection using machine learning," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 488–493, IEEE, 2019.

[16] J. Fan, X. Ma, L. Wu, F. Zhang, X. Yu, and W. Zeng, "Light gradient boosting machine: An efficient soft computing model for estimating daily reference evapotranspiration with local and external meteorological data," *Agricultural Water Management*, vol. 225, p. 105758, 2019.

[17] A. A. Taha and S. J. Malebary, "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine," *IEEE Access*, vol. 8, pp. 25579–25587, 2020.

[18] X. Zhang, J. Zhang, N. Lin, and X. Yang, "Sequential self-attentive model for knowledge tracing," in *International Conference on Artificial Neural Networks*, pp. 318–330, Springer, 2021.

APPENDIX I
FEATURES OF THE DATA SET

Feature	Explanation
cu	The total number of questions the user has done so far
is_first	Whether the user started doing the question for the first time
ts	The time in milliseconds
ts_mod_day	Timestamp mod day
ts_mod_week	Timestamp mod week
ts_day	Timestamp clipped to day
qid	Question id
container_ord	Question enumber in the task container(0-5)
part	Part id (1-7)
bundle_id	Bundle id
tgg	Tag group
qtag0	Appearance probability of first tag
tag_num	Number of tags
qp	Appearance probability of question
qc_r	Correct rate of question
cum_csum	User accumulative number of correct
cum_crate	User accumulative correct rate
sum_avgt	User average cosume time
a_ans	Whether the user has met this question before
pq	Prior question id
ptag0	Prior question tag
ppart	Prior question part
pbundle	Prior question bundle
ptgg	Prior question tag group
pq_correct	Prior question correct or not
pqet	Prior question const time
pqhe	Whether prior question has explained
tsli	Prior question to current time
tsli2	Prior 2 question to current time
tsli3	Prior 3 question to current time
clipped_tsli	Prior question to current time clipped to 20 minutes
roll_num	Prior question number (0-5)
roll_1	Prior 1 correct or not
roll_2	Prior 2 correct or not
roll_3	Prior 3 correct or not
roll_4	Prior 4 correct or not
roll_5	Prior 5 correct or not
roll_6	Prior 6 correct or not
roll_7	Prior 7 correct or not
roll_8	Prior 8 correct or not
roll_9	Prior 9 correct or not
roll_10	Prior 10 correct or not
tgp	Tag group appear probability
tgcr	Tag group appear correct rate
tgg_num	User question number under this taggroup
tgg_cnum	User correct question number under this taggroup
l_num	Total number of lectures watched by users
l_tag num	Total number of lectures watched under tags
l1_num	Total number of lectures watched under content
l2_num	Total number of lectures watched under question solving
l_part_num	Total number of lectures watched under part
is_d	Is this diagnostic question
pp_num	User number under this part
pp_ration	User ratio under this part

TABLE IV
FEATURE USED