Predicting Student Performance in Game-based Learning

Yiwei Peng
University of Michigan

Yonnie Chan
University of Michigan

Chi-Hsiang Yi
University of Michigan

Ping-Lun Lai
University of Michigan

***Abstract*—Jo Wilder is an online educational game developed by PBS Education that allows users to learn through playing. They released one of the largest open datasets of game logs to date. In this project, we aim to help developers and educators by predicting the outcome of gameplay from gaming logs. We implemented several machine-learning classifiers and analyzed their results. After tuning the parameters, our best model achieves an F1 score of 0.808 on the test dataset that the model has not seen. We believe that this will enable game developers to improve this game and support educators in analyzing student performance.**

I. Introduction

Due to the increased availability and use of technology, people have shifted towards playing more video games rather than traditional games. Educational video games offer a fun and interactive way for individuals of all ages to learn new concepts and skills. By combining gameplay with educational content, players are able to engage with the material in a way that is both entertaining and effective. Playing educational video games allows individuals to develop critical thinking, problem-solving, and decision-making skills, all while immersing themselves in an engaging and interactive environment. Moreover, playing video games can also enhance cognitive abilities such as attention, memory, and spatial reasoning. Therefore, learning through playing educational video games is an effective and enjoyable way to acquire knowledge and skills.

Jo Wilder and the Capitol Case [1] is an educational point-and-click adventure game designed for students in grades 3-6. The game allows students to practice historical inquiry skills as they explore the real stories behind mysterious artifacts from two movements in Wisconsin State history. Through engaging in critical thinking and using primary and secondary sources, students will identify and gather historical evidence to make arguments in support of a conclusion. The game also includes vocabulary, save codes, standards supported, and guiding questions, which can be found on the Educator Resources page. By playing Jo Wilder and the Capitol Case, students will develop important skills, such as historical inquiry, critical thinking, and the ability to gather and use evidence from multiple sources.

Training a machine learning model to predict student performance in educational games can provide several benefits. First, it can help teachers and educators identify students who may be struggling with certain concepts or skills, allowing them to intervene early and provide targeted support. Second, it can help personalize the learning experience for each individual student by adapting the difficulty level or content of the game based on their performance. This can help keep students engaged and motivated, leading to better learning outcomes. Third, it can provide valuable insights into how students learn and what factors contribute to their success. This information can be used to develop more effective teaching methods and educational resources. Overall, training a machine learning model to predict outcomes in educational games can be a powerful tool for improving the effectiveness and efficiency of education.

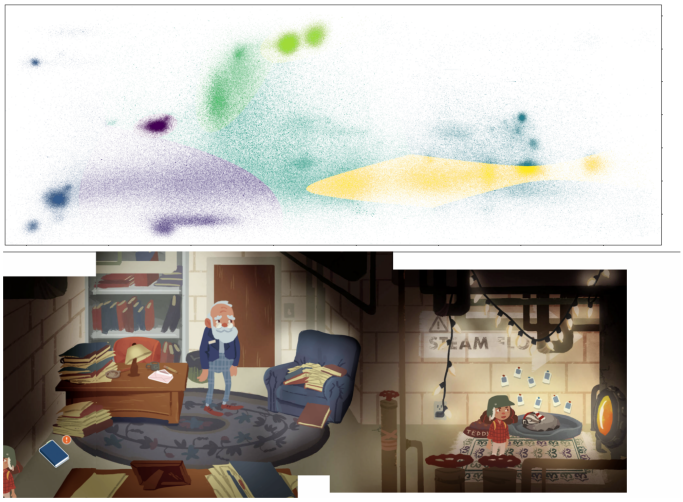

Fig. 1. In order to encode coordinate data, our system uses Gaussian Mixture Model [2] to cluster game click data into categories and then apply a label to each category.

Our project utilized several classification models, including logistic regression [3], random forest [4], XGBoost [5], LightGBM [6], and AdaBoost [7]. These models were employed to achieve a more accurate prediction of the learning outcomes of the educational game Jo Wilder.

- Logistic regression [3] is a statistical model that models the probability of an event taking place by having the log-odds for the event be a linear combination of one or more independent variables.
- Random forest [4] is an ensemble learning method that constructs multiple decision trees during training to make

predictions for classification and regression tasks. The output of the model is the class selected by most trees in the forest for classification tasks.
- XGBoost [5] is an open-source software library that provides a regularizing gradient-boosting framework for several programming languages, including C++, Java, Python, R, Julia, Perl, and Scala. This model is designed to be scalable, portable, and distributed.
- LightGBM [6] is a free and open-source distributed gradient-boosting framework that is based on decision tree algorithms and is used for ranking, classification, and other machine learning tasks.
- AdaBoost [7], short for Adaptive Boosting, is an ensemble learning method that combines multiple weak classifiers to form a strong classifier. It assigns weights to training samples and iteratively adjusts these weights based on the misclassified samples from the previous model, thus learning to focus on harder-to-classify samples.

By utilizing these models, we aimed to achieve a more accurate prediction of the learning outcomes of the educational game Jo Wilder.

We utilize several data preprocessing techniques to prepare our data for training. For instance, we employ MobileBERT [8], a lightweight version of the state-of-the-art language model BERT [9], to encode our text data (typically questions or options in text) into vectors. We also apply Principal Component Analysis [10] and the Gaussian Mixture Method clustering technique [2] to reduce the dimensionality of our text features and further categorize them into distinct groups.

For the results of the project, we include an overall F1 score of 0.710 from Logistic Regression as a benchmark. After feature engineering and parameter tuning, the highest F1 score we achieve is 0.808 using LightGBM [6] as the final model.

The upcoming sections of this report will delve into the topic in the following structured manner: Section II will provide an overview and exploratory analysis of the data we use. Section III will present our methodology. In Section IV, we will elaborate on the design of our experiment and analyze its outcomes. Our final thoughts on the matter and future prospects will be discussed in Section V.

II. Data

In this research, the time-series dataset is provided by Kaggle [1]. The dataset contains information on the records of students from October 2020 to November 2022, with the objective of forecasting their performance. Some examples of the columns are:

- **Categorical attributes:**
  session_id, index, event_name, name, level, page, text, fqid, room_fqid, text_fqid, fullscreen, hq, music, and level_group.
- **Numerical attributes:**
  elapsed_time, room_coor_x, room_coor_y, screen_coor_x, screen_coor_y, hover_duration.

A. Raw Data

The dataset consists of four files:

- Training set with game session data (train.csv).
- Labels of correct answers for 18 questions for each session (train_labels.csv).
- Testing set with game session data (test.csv).
- Sample submission file (sample_submission.csv).

In this project, we only include the training and train_labels data for further preprocessing and modeling processes. The training data consists of 23,660,476 rows and 20 features, with 21,204 unique *session_id* entries and 3 unique *level_group* entries. The training labels data contains the correct values for all 18 questions for each session.

The Jo Wilder game is an example of the point-and-click genre. To progress through the game, the player must find hidden objects and/or answer quizzes. Corresponding to the game setup, our data includes:

- Each game is a session defined by *session_id*.
- Each row in the data is a game event, defined by its name, type (*event_name*), unique ID (*fqid*), and game progress index (*index*).
- The player progresses by moving from one game room (*room_fqid*) and level (*level*) to another.
- Remaining columns refer to specific events:
- For click events, the coordinates of the click are defined in *room_coor_x* and *room_coor_y* (in reference to the in-game room) or in *screen_coor_x* and *screen_coor_y* (in reference to the player's screen).
- For notebook-related events, the page identifies the page number.
- For hover events, *hover_duration* shows how long the hover lasted.

B. Exploratory Data Analysis

In order to gain a comprehensive understanding of the dataset, we conducted a thorough investigation of all features included in the training set. While this process was informative, we recognize that not all insights gleaned from the analysis are equally important or relevant to our modeling goals. Therefore, we focused our attention on those features that are most likely to have an impact on our models and warrant further investigation.

First, the identification and handling of missing data is an important aspect of data preprocessing in machine learning, which is crucial for obtaining accurate and reliable results. In the present study, an analysis of the training and testing datasets was conducted to identify any missing values in the columns. Visual examination of the data was performed through the generation of bar plots in Figure 2, which revealed the presence of a significant number of missing values in some columns.

However, a positive finding of this study is that the distribution of missing values in the training and testing datasets exhibited similar ratios, indicating that the datasets are comparable in terms of missing data. The similar distribution of

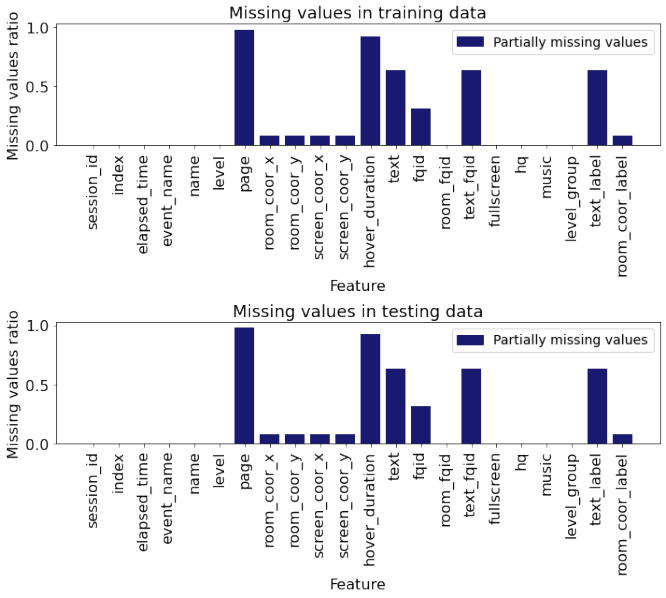

Fig. 2. Missing values on training and testing data

missing values in the two datasets is advantageous as it allows for the use of consistent strategies for handling missing data, ensuring the reliability of the model.

Second, the analysis of the dataset revealed a slight imbalance in the distribution of correct answers across questions, as illustrated in Figure 3.

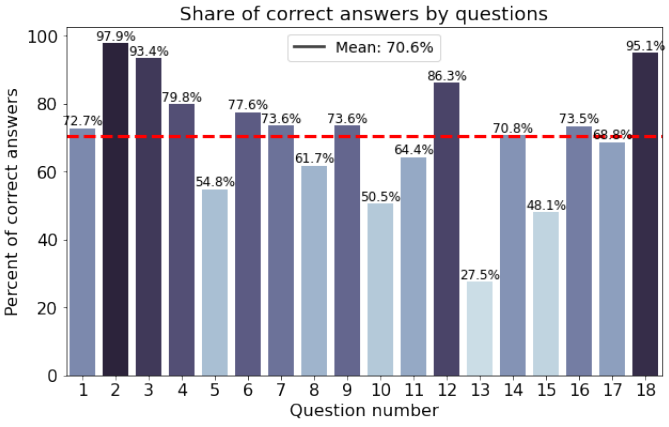

Fig. 3. Distribution of correct answers across questions

To assess the imbalance, the percentage of correct answers for each question was plotted, revealing that the ratio of correct answers varies across questions. The average correct answer ratio for all questions in the dataset is 70.6%. Specifically, question 2 exhibited the highest correct answer ratio, at 97.9%, while question 13 demonstrated the lowest correct answer ratio at 27.5%. The presence of an imbalance in the dataset has important implications for predictive modeling, as it can negatively impact the accuracy of models and the generalization of results. Therefore, careful consideration and appropriate measures would be taken to address this issue in order to improve the performance of models trained on this dataset.

Third, we have observed that the *session_id* feature potentially encodes date and time information, which we plan to leverage during exploratory data analysis and feature engineering stages.

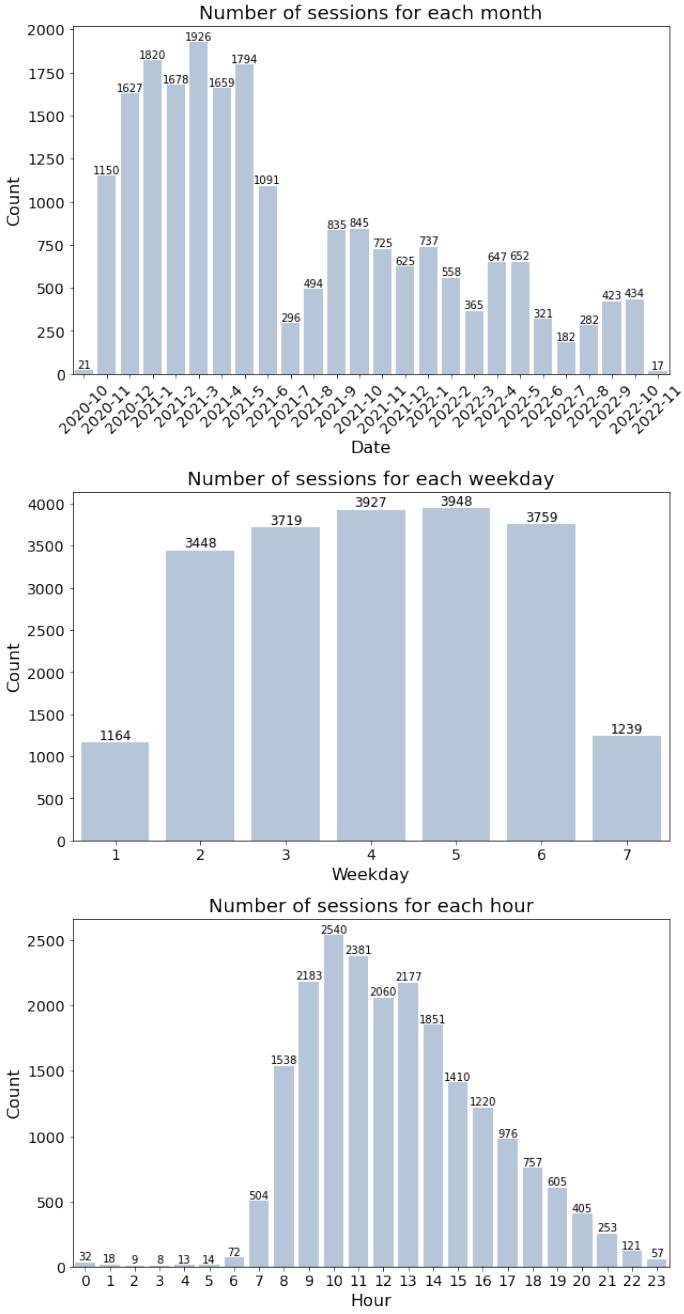

Fig. 4. Number of sessions for each time-related feature in training data

In Figure 4, our analysis of the *session_id* feature revealed that the training data spans from October 2020 to November 2022. As expected, we found that the game is predominantly played during weekdays (Monday to Friday) and working hours (8 A.M. to 6 P.M.). These insights provide valuable

information that can help us to better understand the patterns and behaviors of users interacting with the game.

Fourth, the feature *elapsed_time*, which indicates the time elapsed (in milliseconds) between the start of a session and the recording of an event, is used to predict user behavior. However, our analysis revealed an unexpected behavior of this feature in the dataset. We observed that the *elapsed_time* does not necessarily increase monotonically, contrary to our initial expectations. In fact, we found that 587,640 events had a negative elapsed time change. Moreover, our analysis showed that 21,181 sessions had at least one event with a negative elapsed time change. Despite this unexpected data, we decided to include *elapsed_time* as a feature in our model as it might provide additional information on user behavior that could be useful for prediction purposes.

Fifth, we analyzed the 11 unique *event_name* values present in the event properties, namely *navigate_click, observation_click, notification_click, object_click, object_hover, map_hover, map_click, notebook_click, checkpoint*. Our objective was to detect any anomalous events in the dataset. For this purpose, we constructed an interactive boxplot to explore the outliers. Based on the 1.5 IQR rule, we set the upper fence at 1723 and the lower fence at 591. Next, we visualized the distribution of these events using a bar chart and observed that it exhibited a right-skewed distribution.

III. Methods

A. Method Overview

We present the overall workflow of our prediction system in Figure 5. After reading in and cleaning the data, we process the text and room coordinates using Principal Component Analysis [10] and Gaussian Mixture Model [2] to apply labels to each text and coordinate data. Following that, the labels, along with other data, are sent into the feature engineering program to aggregate data based on section id. In the end, we make predictions based on the aggregated data.

B. Encoding Texts and Room Coordinates

The text column includes texts the player sees during the event. To utilize the text data provided, we encode the 591 unique text pieces into vectors of size 512 using MobileBERT [8], a lighter version of BERT [9]. The encoded vectors capture the meaning of the sentences. We further perform a dimension reduction technique using Principal Component Analysis [10]. Then, we employ the Gaussian Mixture Model to categorize the texts into three clusters. Finally, we transform the cluster labels into one-hot encoding before sending them into the model. Figure 6 displays the vectorized 591 text pieces. To visualize them, we use the Principal Component Analysis [10] technique to project the features onto the two most important principal components.

Similarly, to allow the models to utilize the click coordinates more efficiently, we also employ the Gaussian Mixture Model to categorize them into clusters. To elaborate, we first group the candidates by the game room. For each game room, we cluster the clicks in the room into five categories using the Gaussian Mixture Model. Figure 1 illustrates an example in the tunic historical society closet room.

C. Feature Engineering

In order to create meaningful features for our predictive model, we performed feature engineering on the dataset. One important aspect of this process was determining the appropriate aggregation functions to apply to each type of feature after grouping by *session_id* and *level_group*. This was particularly important as our goal was to predict every session at different levels.

For categorical attributes, including *page, fqid, room_fqid, text_fqid, fullscreen, hq, music*, we used the nunique aggregation functions from pandas, which counts the number of unique values for each category within a given group. This allowed us to capture a variety of categorical data within each session.

For numerical attributes, such as *elapsed_time, elapsed_time_change, level, room_coor_x, room_coor_y, screen_coor_x, screen_coor_y, hover_duration*, we used the mean and standard deviation aggregation functions. The mean function provides the average value of the variable across each session, while the standard deviation function gives us a sense of the variability within each session.

For events attributes, such as *navigate_click, person_click, cutscene_click, object_click, map_hover, notification_click, map_click, observation_click, checkpoint*, we extended the value of each event column to nine separate columns using one-hot encoder. After applying one-hot encoding, we used the sum aggregation function for these event features and the *elapsed_time* feature. This allowed us to capture the total number of times each event occurred and the total elapsed time for each session.

The *elapsed_time* feature shows the time elapsed between the start of the session and when the event was recorded. However, it does not necessarily show the actual time that has passed between two events, as it can be affected by various factors such as network latency, device performance, and user behavior. On the other hand, *elapsed_time_change* could reflect the actual time difference between two events, which provides more accurate information about the user's behavior and the flow of the session.

Moreover, using *elapsed_time_change* allows us to detect anomalies and potential issues in the data more effectively. For instance, if the *elapsed_time_change* between two events is significantly longer than usual, it may indicate that the user has encountered an issue or has taken a long break. Such anomalies can provide valuable insights into the user's behavior and can be used to improve the overall user experience.

During the exploratory data analysis (EDA) phase, we identified that the *session_id* column contains valuable date and time information. In order to make use of this information for time-series analysis, we conducted reverse engineering techniques to extract several time-related features, such as year, month, weekday, hour, minute, second, and millisecond, as well as an unknown part. By leveraging these time-related

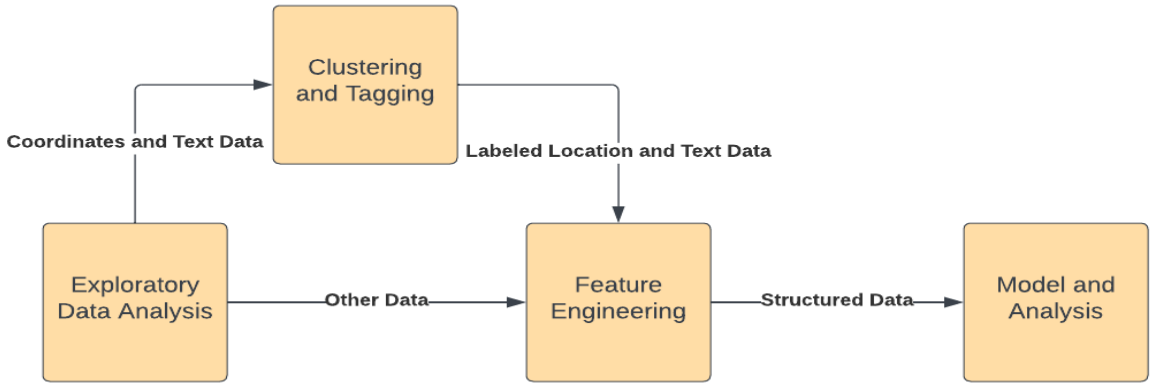

Fig. 5. System overview.

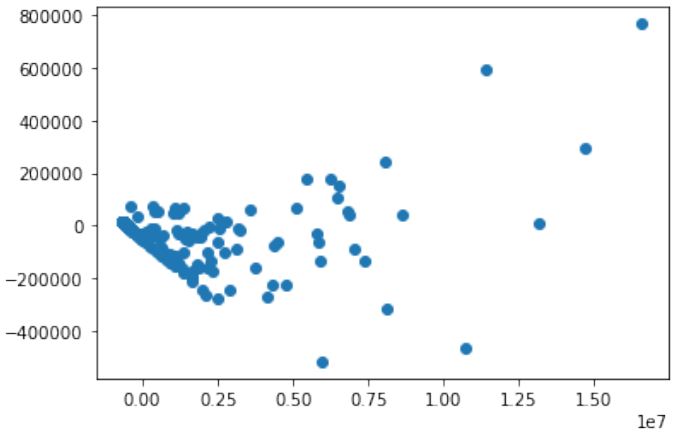

Fig. 6. Visualize encoded text data using PCA [10]. The data was encoded using MobileBERT [8]

features, we are able to transform the dataset into a time-series problem and potentially uncover valuable insights that may be hidden in the data.

D. Classification Models

We surveyed and tried the classic classification methods including Logistic regression [3], Boosting methods [7] [6] [5], and random forest methods [4]. For the Boosting methods, we also exploit different boosting methods under the family. We exploit gradient boosting (XGBoost, LightGBM) [6] [5] and traditional adaptive boosting methods (AdaBoost) [7]. XGBoost is an open-source software library that provides a regularizing gradient-boosting framework for several programming languages, including C++, Java, Python, R, Julia, Perl, and Scala. LightGBM is also a free and open-source distributed gradient-boosting framework that is based on the decision tree.

IV. Experiment and Analysis

A. Training Setup

The models are trained on a dataset containing information from 18,848 users and validated on a separate dataset of 2,356 users. The model training process can be divided into three stages. In the first stage, the models are trained on raw data without any preprocessing or hyperparameter tuning. The second stage involves training the models on processed and selected features. In the third and final stage, a Randomized Cross-Validation (RandomizedCV) method is employed to optimize the hyperparameters of each model.

For each stage, a 5-fold cross-validation strategy is utilized to evaluate the performance of the models, ensuring a reliable assessment of their generalization capabilities. As the primary evaluation metric for the data challenge is the F1 score, it serves as the principal measure used to present and compare the results of the different models throughout the training process.

B. Model Evaluation

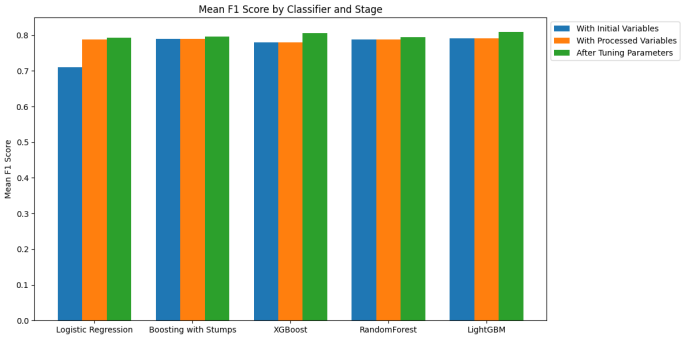

Fig. 7. F1 scores across three different stages

An examination of the results from the three stages reveals the following insights, as illustrated in Figure 7 and Table 1:

1) The plot demonstrates that all the models—Logistic Regression, AdaBoost, XGBoost, RandomForest, and

TABLE I
F1 SCORES BEFORE AND AFTER FEATURE ENGINEERING AND HYPERPARAMETER TUNING

	F1 Scores		
	Before Feature Eng.	After Feature Eng.	After Tuning
Logi. Regression	0.710	0.787	0.792
AdaBoost	0.789	0.789	0.796
XGBoost	0.780	0.780	0.806
RandomForest	0.787	0.787	0.794
LightGBM	0.791	0.791	0.808

LightGBM—display similar F1 scores in the end, after feature engineering and hyperparameter tuning. We found that our feature engineering and encoding methods have the most significant effect on the Logistic Regression model. We believe our new features compensate for the weakness of the Logistic Regression method in modeling non-linear relationships between features. In the other four models, the new features also mildly improve the results.

2) Furthermore, an even more significant improvement in F1 scores is observed across all models after tuning the parameters, enabling models like XGBoost and LightGBM to achieve an overall average F1 score of 0.8. This suggests that hyperparameter tuning considerably influences the performance of the models.

In summary, logistic regression benefits significantly from using features generated by our feature engineering method, boosting its performance from 0.71 to 0.78, while the other models exhibit mild improvement. Hyperparameter tuning plays an even more substantial role in enhancing performance, with all five models demonstrating considerable improvement.

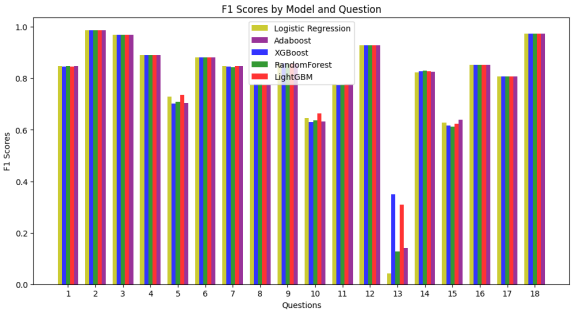

Fig. 8. F1 scores on different questions across models

Next, we examine the results of individual models for each question in the educational game. Looking at Figure 8, we can draw several conclusions:

1) The F1 scores vary across individual questions. The models achieve the highest performance on questions 2 and 18, whereas the lowest performance is observed for question 13. Looking back at Figure 3, which shows the percentage of correct answers for each question, we see a relatively similar pattern to Figure 8. This may suggest that the model's performances depend heavily on the correctness of each question, and there is potential for further improvement by identifying and addressing the factors that contribute to these differences in performance.
2) For most questions, the F1 scores are relatively similar across the three models. This suggests that the choice of the model does not drastically impact performance on individual questions. However, to achieve the best overall F1 score, it may be beneficial to select the model with the highest F1 score for each question. This approach would involve training a separate model for each question and combining their predictions to obtain the final results.

In summary, the model's performance varies across the different questions, with some questions exhibiting better results than others. Additionally, by selecting the model with the highest F1 score for each question, we can potentially maximize the overall performance. However, further investigation and tuning may be necessary to address the factors contributing to the varying performance across questions, such as the imbalance in correct answers, and to achieve optimal results in an academic setting.

C. Feature Importance

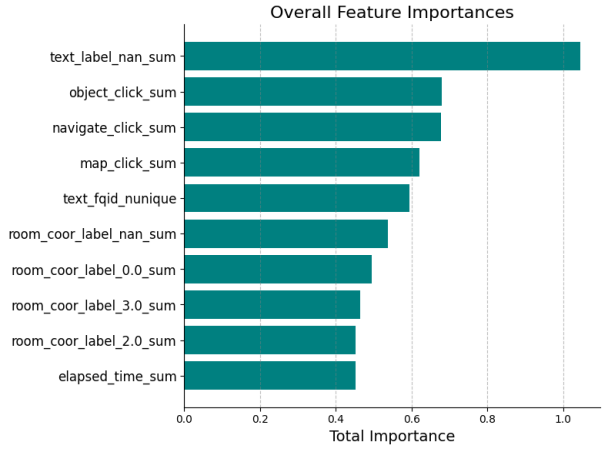

Fig. 9. Feature Importance: Top 10 Features

After building and evaluating our models, we conducted a feature importance analysis to identify the most influential attributes in predicting the correct answers to the questions. The top features were related to the following aspects: missing text labels, user interaction with objects, user navigation within the environment, interactions with the in-game map, unique text identifiers accessed by the user, user location within the environment, and total time spent on the task.

- The missing text labels feature signifies instances where certain labels were unavailable to users, which could be associated with how users interact with the environment and interpret the information provided.

- The user interaction with objects and navigational elements features represent the extent of user engagement and their ability to explore the environment effectively.
- The interactions with the in-game map feature indicate the user's spatial awareness and understanding of the environment.
- The unique text identifiers feature reflects the diversity of information accessed by the user throughout their session, potentially providing insights into their learning process.
- User location features, such as instances of specific room coordinates or missing coordinates, offer information about the user's spatial distribution within the environment, highlighting the significance of their location in relation to their performance.
- Lastly, the total time spent on the task serves as an indicator of the user's overall performance and efficiency in completing the task, providing valuable insights into their learning progress.

In conclusion, feature importance analysis gave us an insight into which features have a higher influence on correctly predicting the correctness of answers.

V. Conclusion and Discussion

This study proposes several data processing methods and finds that the general performance of using LightGBM is the best. The performance highlights of these processing methods include the following:

1) With the help of encoding, we transform texts into vectors and categorize them into three clusters using the Gaussian Mixture Model. Similarly, the coordinates are transformed into clusters. One potential drawback of this model is that it allows outliers to be separated into distinct clusters, which leads to lower accuracy.
2) By employing aggregation, we can abstract essential information within each session. However, the cost is that only the specified aggregation functions are used, which might limit the model's predictability.

An additional factor to consider is the computational efficiency of the models, particularly in the context of this data challenge, which aims to develop small and lightweight models. A noteworthy observation is that Logistic Regression and LightGBM exhibit significantly lower computation time compared to other models. Nevertheless, LightGBM's results are slightly better than those of Logistic Regression. In this case, the choice of the most suitable model would likely favor LightGBM due to its reduced computational demands and superior performance results.

Future work includes incorporating further information about the player in each session. For instance, the performance can significantly differ between 6-year-old and 10-year-old players. More features of certain variables can be explored to enhance model performance. For imbalanced data in each question, effective methods such as downsampling can be implemented.

References

[1] "Predict student performance from game play," https://www.kaggle.com.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215–232, 1958.

[4] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, vol. 1. IEEE, 1995, pp. 278–282.

[5] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785

[6] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, pp. 3146–3154, 2017.

[7] R. E. Schapire, "Explaining adaboost," in *Empirical inference*. Springer, 2013, pp. 37–52.

[8] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "MobileBERT: a compact task-agnostic BERT for resource-limited devices," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 2158–2170. [Online]. Available: https://aclanthology.org/2020.acl-main.195

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[10] K. P. F.R.S., "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.