Cornell University®

Analysis on the Expenditure of Player in Online Game

Yunhan Liu(yl3287) **Jiawei Gao(jg2352)**

Contribution: Equal Contribution

I. Introduction

A very popular strategy and leadership game called "Brutal Age" has become quite well-known all across the world. Understanding the worth of each player is essential for optimizing advertising tactics and supporting effective operational activities within the game given its regular top rankings in game sales charts across a number of nations. The objective of this project is to predict players' spending behaviors in the well-known mobile online game "Brutal Age" using user data. Our dataset comes from the second Intelligent China Cup (ICC) data competition, Chengdu Nibiru Technology Co., Ltd. The dataset consists of millions of records capturing user behavior during the first 7 days of gameplay. Our goal was to leverage this data to estimate the value of players by predicting their total payment amount within 45 days.

To predict the expenditure of players in "Brutal Age", we have employed several algorithms, including linear regression, random forest, and eXtreme Gradient Boosting(XGBoost). These algorithms have been selected for their ability to capture complex relationships within the data and provide accurate predictions. In this report, we will provide an overview of the dataset and elaborate on the algorithms we used in our analysis . Additionally, we will

provide an overview of the dataset, discuss the algorithms employed in our analysis, present our results obtained, and explore the future implications.

II. Data Description and Exploration

The dataset includes 2288007 rows and 109 columns, in which 107 numeric variables, 1 datetime variable and 1 target variable (numeric). The target variable is “payment_45” which means the

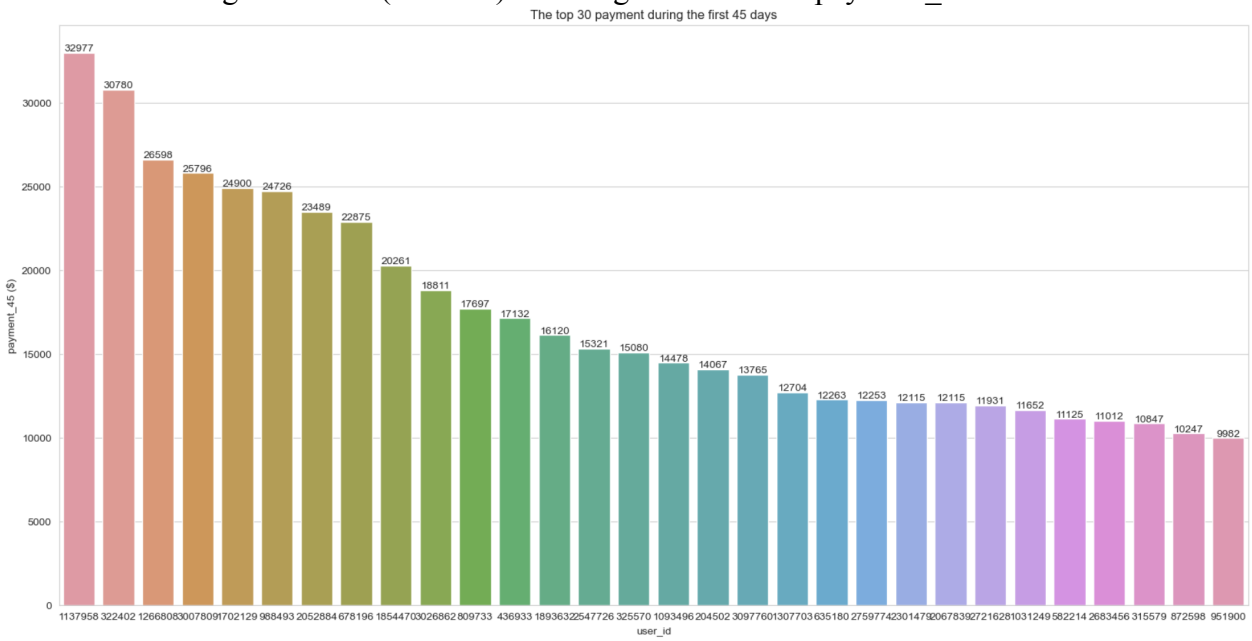

Figure (1) The top 30 expenditures during the first 45 days

expenditure of each game player during the first 45 days from the date they registered the online game account. The 107 features could be classified into three classes, including game props (the number of items they collated or purchased etc.), the record of game performance (PvP or PvE, the win or loss situation of each round of the game, etc), behavior of each player (average of the time the players stay online, the payment they made during the first seven days, etc.)

count	mean	std	min	50%	75%	90%	99%	max
2288007	1.79	88.46	0.00	0.00	0.00	0.00	3.97	32977

Table (1) The summary of the payment_45

Fortunately, the dataset does not include any missing values or duplicated rows. Every new player who registered the account can obtain the sources in the game. In order to obtain more useful resources and have a better user experience, they can pay for them. Therefore, it is necessary to explore the target variable itself. We check the distribution of the target variable ("payment_45") by the histogram and find that it is left skewed distributed, which means there are a lot of people who did not make any payment in the game. Here is the summary of the payment_45 (Table (1)) and the user with the top 30 expenditures during the first 45 days (Figure (1)). After calculation, we find that the number of people who had paid only 45988, which means only 2.01% of total players paid. The average revenue per player is only $1.79 and the average

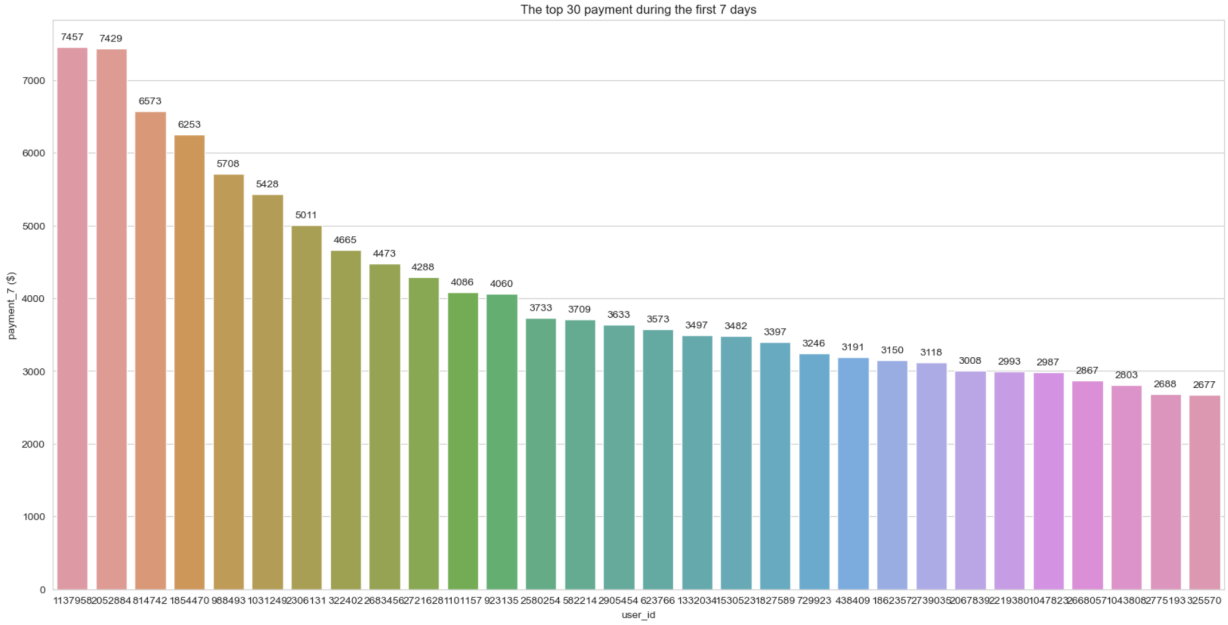

Figure (2) The top 30 expenditures during the first 7 days

revenue per paid player is $89.21. This indicates that there should be a big problem in the conversion stage. In the total of 108 variables, the expenditure in the first 7 days of each player ("payment_7") plays an important role in this case. And we also visualize the top 30 of "payment_7" in Figure (2). In Figure (3), the bar chart refers to the top 30 expenditures during the first 45 days and the line chart refers to their expenditures during the 7 days. Obviously, they have a strong pattern to show the positive relationship between them. 73.52% correlation between them also verifies this point. By calculation, the number of users who didn't pay during the first 7 days is 2246568 while the number of users who didn't pay during the first 7 days but paid after the first 7 days is 4549. We can assume that there is a 99.8% chance that a new player who didn't pay within the first week won't pay either within the month after that. There could be two causes for this. On the one hand, the fact that all players were kept for longer than 45 days shows that the game's conversion techniques are extremely subpar once the new user benefit period has ended and that the conversion incentives are woefully inadequate.

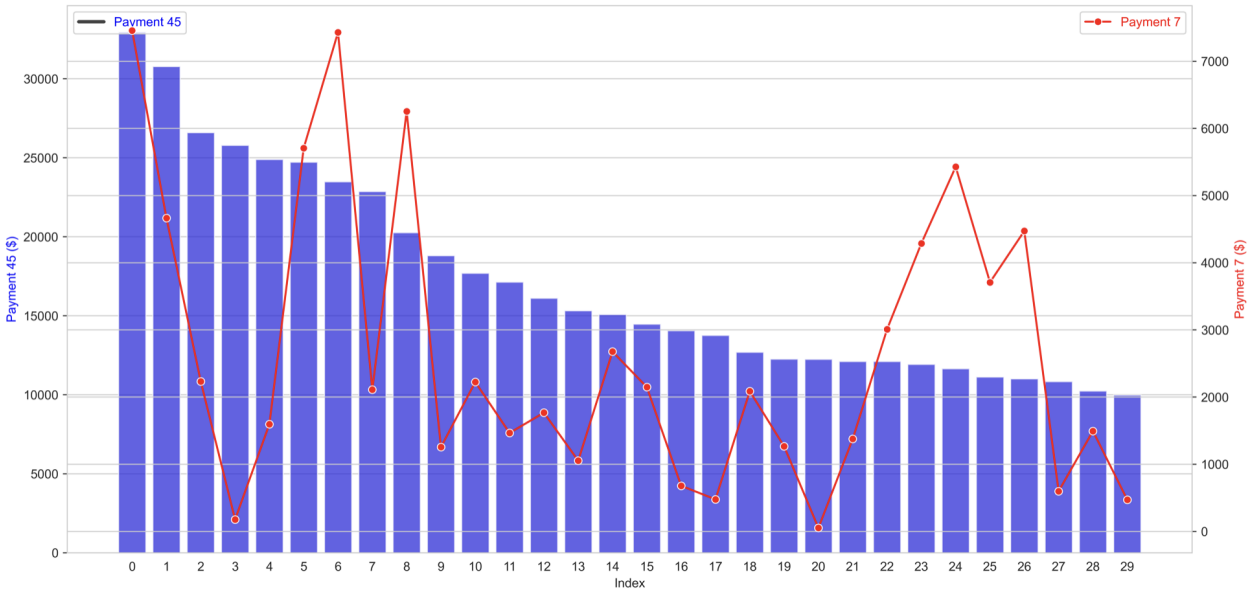

Figure (3) The top 30 expenditures during the first 45 days and 7 days

On the other hand, most consumers had already churned before the 45th day, therefore there was little chance for them to continue to convert to paying users. It is not difficult to find that there are over 30,000 users who paid before the first 7 days and have not paid again after the first 7 days, accounting for 72% of the total number of users who paid during the first 7 days. This indicates that there is still a 70% possibility that a new player who paid within the first week will stop paying within the following month. The majority of users made their first in-game purchase of a super cheap/super affordable beginner's pack, and the advantages and costs of future payment packs are much beyond their expectations. Because the starter pack was excessively

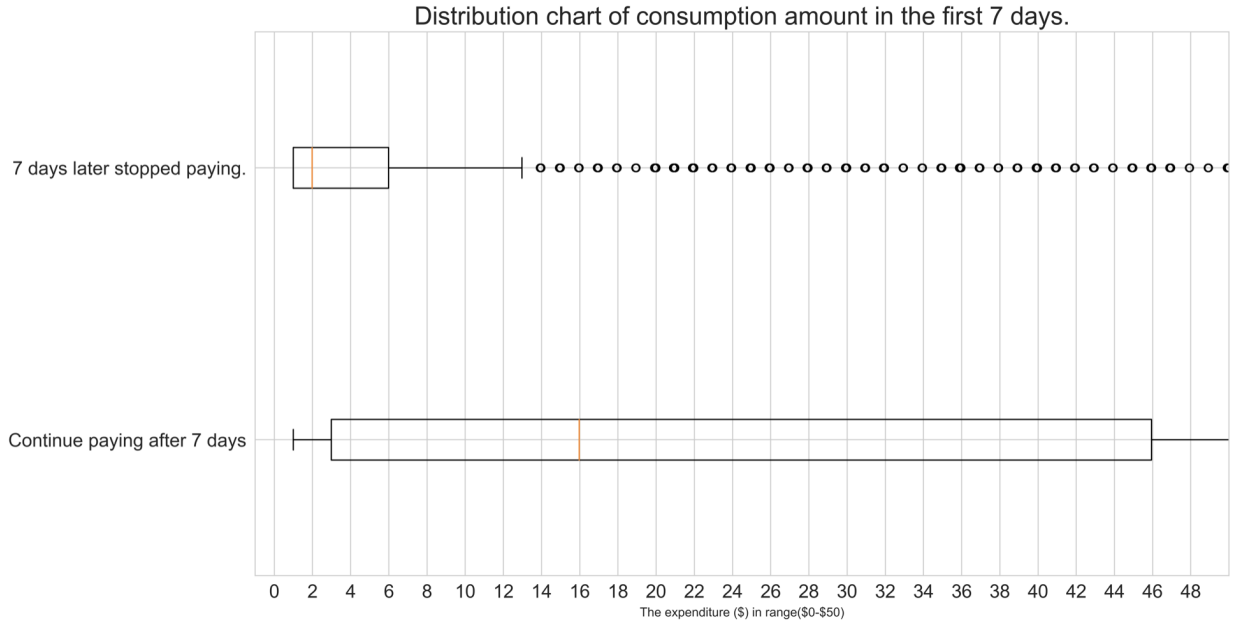

Figure (4) The box plot of the expenditure v.s different class of people

generous, future payment packs appeared to be less cost-effective, and the requirement for users to make extra payments is too high. The game itself isn't very addictive, and most players quit after 7 to 45 days, or even just the first seven. Players initially have an itch to pay to address

issues (itchy feeling), but as they mature, they lose this need and are less likely to pay or immediately exit the game if they cannot mature. The Figure (4) describes the two kinds of people mentioned above: the people who paid during the first 7 days but stopped paying and those who paid during the first 7 days and during the first 7 days but stopped paying and those who paid during the first 7 days and continue paying after 7 days. It is obvious that players who pay after 7 days will spend more money overall than those who don't during the first 7. More than 50% of players who continued to pay after 7 days spent roughly $16, compared to 75% of players who stopped paying after 7 days and only spent less than $6. This overwhelmingly supports our first theory: while the majority of users spend money on the new player package, the pricing of future packages is more than most people anticipate. The final decision to stop playing the game may be the result of natural attrition or the fact that the experience after spending money fell short of some users' expectations. At the same time, many players among those who don't pay after 7 days have spent hundreds or even thousands of dollars. However, based on the amount of money customers who continue to pay after 7 days spend, users like to spend more and more, therefore after spending money, the majority of users should be happy with the experience. Although the range of spending is the same for all groups, it is clear that customers who pay after 7 days spend more within the first 7 days because their distribution is considerably further to the right.

We also explored the average online time of each player. In Figure (5), we do not discover the obvious pattern between the average online time of each player and the expenditure during the 45 days. And the correlation between the two is 17.35%, which means that there is a weak relationship between the two. During the first seven days, 75% of users only engaged with the game for 4.8 minutes or less, which suggests that more than 75% of users gave it a cursory look before quitting. This may be another reason why the expenditure decreases.

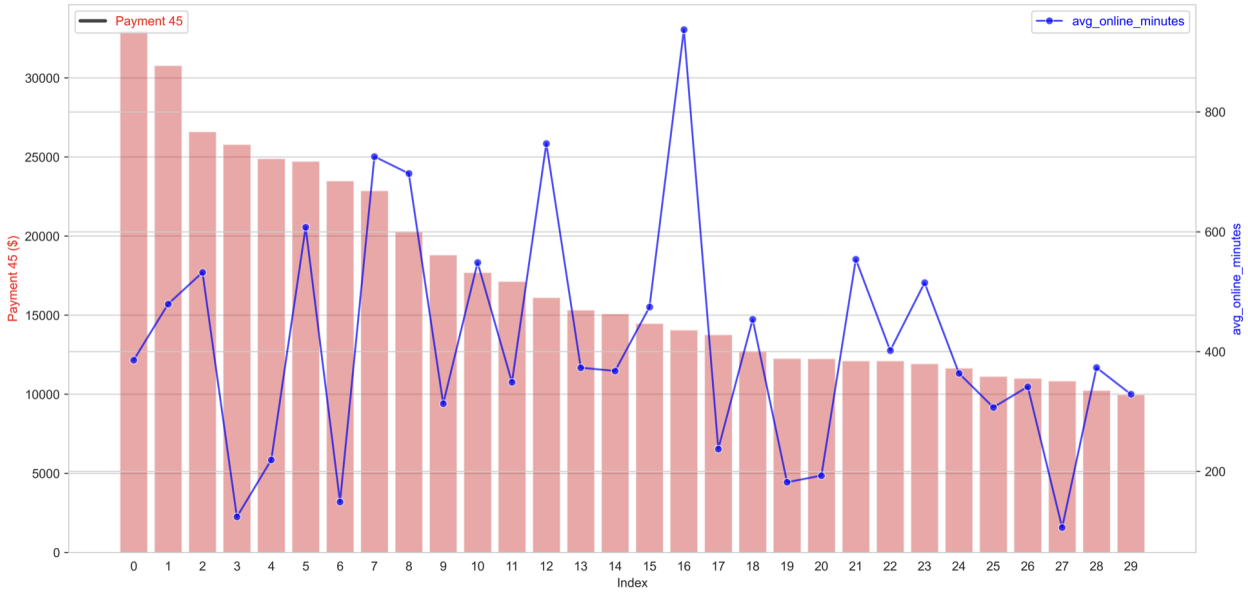

Figure (5) The top expenditure during the 45 days and the average online time

III. Methodology

We firstly used linear regression as a benchmark model to predict the total payment amount within the 45 days("payment_45"). We split the data into training and testing sets, with 80% of the data used for training and 20% for testing. The mean squared error (MSE) of the linear regression model was 3913.

Although linear regression is a straightforward and extensively used method for predicting numerical values, it may not necessarily deliver the highest level of accuracy. More advanced machine learning techniques, such as random forest and XGBoost, can be used to improve our predictions. Both of these algorithms are capable of dealing with non-linear correlations between features and the target variable.

We have chosen to utilize random forest as our first option in this instance to forecast the payment. To provide a final forecast, Random Forest constructs several decision trees and combines their predictions. This enables it to detect intricate patterns in the data that simpler models like linear regression can overlook. On the training data, we utilized the scikit-learn library to train a random forest model with hyperparameter tuning. We first split the data to 80% as training and 20% as testing and build a model with the default parameters. We found the MSE is so big and it is obviously overfitting since the R-squared on the training set is much higher than the R-squared on the testing set. Then, we split the dataset again with 70% as training and 30% as testing. As a result, the MSE decreased a little bit, and the difference between R-squared on training and testing also decreased. After that, we also tuned the parameters "n_estimator" and "max_depth". The MSE and R-squared are shown in table (2). Finally, the model 5 is our best random forest model with lowest MSE and it does not overfit.

Model	Split	n_estimator	max_depth	MSE	R^2 on train	R^2 on test
m1	80%, 20%	100(default)	6(default)	4076.5398	0.8966	0.3149
m2	70%, 30%	10	6(default)	3873.5419	0.8981	0.3733
m3	70%, 30%	10	5	3424.0279	0.7576	0.4461
m4	70%, 30%	10	4	3189.5639	0.7163	0.4840
m5	70%, 30%	10	3	**3176.2768**	0.6514	0.4861
m6	70%, 30%	8	3	3188.3149	0.6475	0.4842

Table (2) The performance of the random forest model

However, XGBoost is also an alternative method. The more sophisticated technique XGBoost employs gradient boosting to incrementally enhance decision tree predictions especially for large datasets and high-dimensional feature spaces. XGBoost, in contrast to Random Forest, is a boosting technique that uses decision trees as basis models. Boosting is a technique that allows the model to learn from its mistakes by training on the residuals of prior models. XGBoost employs a gradient boosting framework, in which each tree is constructed to correct the mistakes of the preceding tree, yielding a very accurate and powerful model.

Another significant distinction is that XGBoost has been improved for both speed and memory efficiency, making it faster and more scalable than Random Forest, particularly for large datasets.

XGBoost does this by parallelizing the tree construction process and employing a sparse-aware approach to deal with missing data.
XGBoost has more hyperparameters than Random Forest, which can make tuning for optimal performance challenging. XGBoost's hyperparameters can provide the operator with more control over the model and allow for finer-tuning of the algorithm to improve performance. Another outstanding feature of XGBoost is that it provides feature importance scores, which might help comprehend the significance of each feature in the model.
Similar to how we dealt with random forest models, we tried to tune the parameters "n_estimator" and "max_depth". The evaluation metrics of the performance of the XGBoost models are shown in table (3). Finally, the model 6 with parameters n_estimator = 7 and max_depth = 3 is our best XGboost model. And we chose it as our final model.

Model	Split	n_estimator	max_depth	MSE	R^2 on train	R^2 on test
m1	80%, 20%	10	6(default)	3975.6570	0.8813	0.3319
m2	70%, 30%	10	6(default)	4045.7498	0.8877	0.3454
m3	70%, 30%	10	3	3024.4583	0.7528	0.5107
m4	70%, 30%	10	2	3098.5843	0.6452	0.4987
m5	70%, 30%	8	3	3005.7318	0.7306	0.5137
m6	70%, 30%	7	3	2965.5106	0.7154	0.5202

Table (3) The performance of the XGBoost model

IV. Results and Discussion

After training and testing our XGBoost model , we evaluated the best model's performance on the testing data. We plotted the actual and predicted payment values to assess the accuracy of the model, which can be viewed in figure (6).

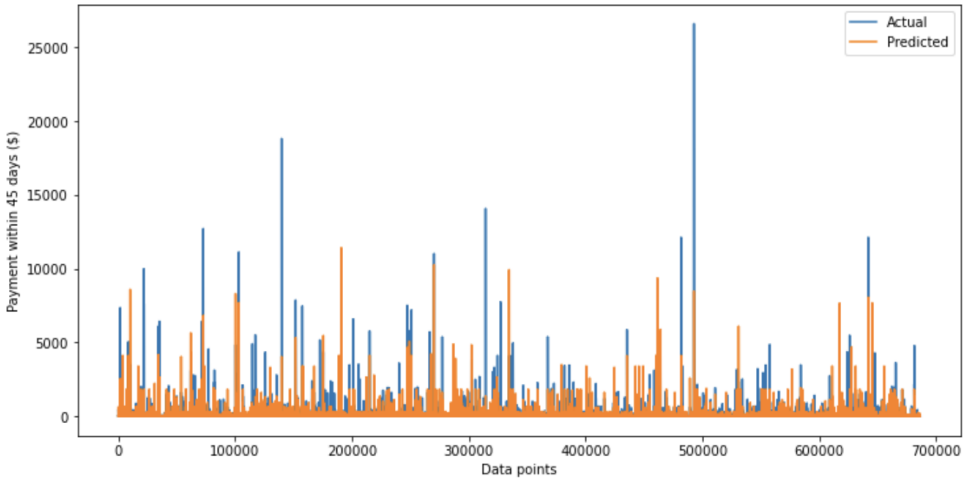

Figure (6) Actual and predicted payment values on the testing data

Regarding prediction accuracy, the XGBoost model performed better than the random forest model. To identify which characteristics were crucial for forecasting the payment amount, we visualized the XGBoost model's feature importance in figure (7). According to the plot, "payment_7" was the feature that was most crucial, followed by "pay_count" and "meat_add_value". The feature selection in future models may benefit from this knowledge. Overall, our models provided accurate predictions of payment amounts and identified the most important features for predicting payment amounts.

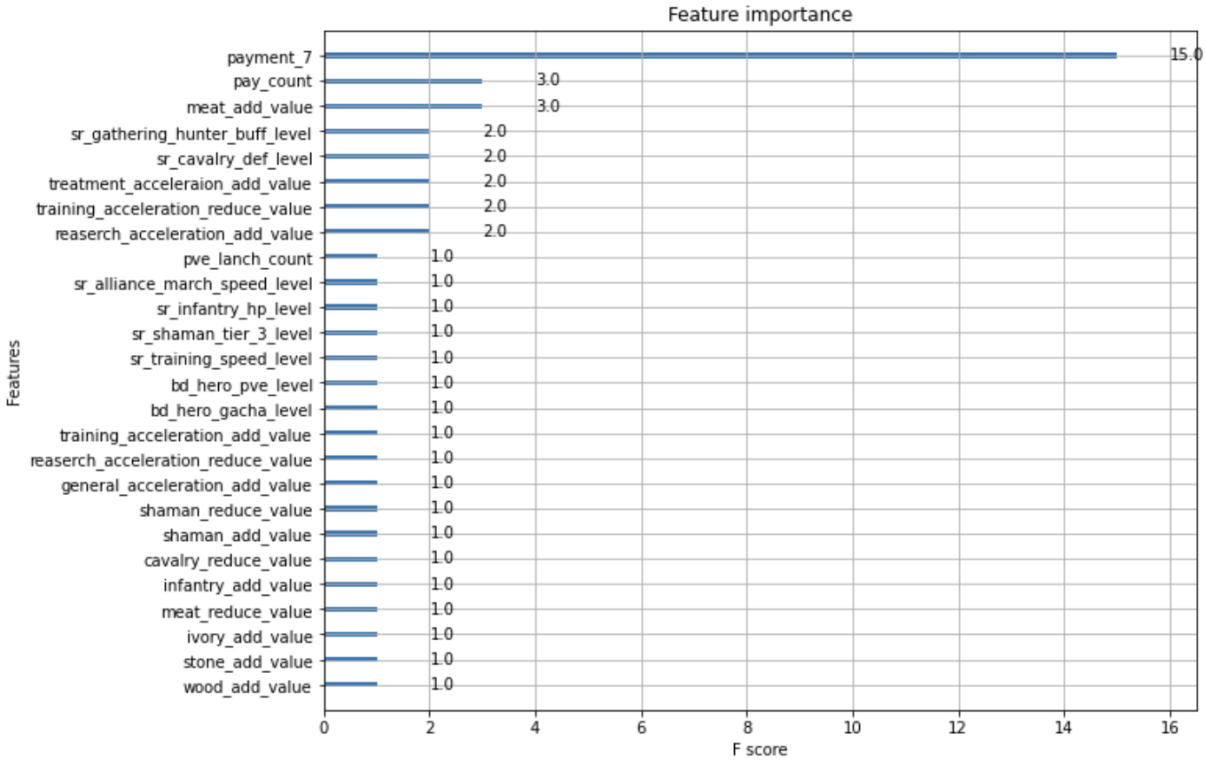

Figure (7)Feature importance of the best XGBoost model

Although our model made accurate predictions, there are several restrictions that must be taken into account. Firstly, we didn't standardize the features, which could have had an impact on how well some of our models performed. Standardization helps guarantee that all characteristics are scaled similarly, which can enhance the models' accuracy and stability.

Furthermore, we employed every feature in our dataset, which could have overfitted the models and made them harder to understand. The performance and interpretability of the model can be enhanced by picking the key features. We may build more nimble models that are simpler to understand and less prone to overfitting by concentrating on the characteristics that have the highest correlation with the target variable.

Our current prediction could be enhanced in a number of ways. Firstly, standardizing features might help some of our models perform better. Second, the performance and interpretability of the models can be strengthened by utilizing feature selection techniques, such as recursive feature elimination or L1 regularization, to choose the most crucial features. Thirdly, utilizing specific knowledge to develop new features may help the models perform better. For instance, adding features that are understood to be connected to each other may provide the models with

additional useful information. As an illustration, defining a high-value player based on the expenditure in the first 7 days of each player and average online time could help identify players who are more likely to pay in the future.

V. Conclusion

In conclusion, we employed machine learning models to forecast players' payment patterns in a mobile game. We developed and tested a number of models, including linear regression, random forest, and XGBoost. We compared the performance of the models and discovered that XGBoost had the lowest mean squared error (MSE) and highest predicted accuracy for the test data.

Our research still points out several shortcomings and possible areas for improvement. The performance and interpretability of the models might be enhanced by standardizing the features and applying feature selection techniques. Additionally, adding new features based on domain expertise may be able to give the models information that is more useful.

Overall, our investigation showed how machine learning models may be utilized to predict payment behavior in the gaming industry. These models might help gaming firms create specific advertising strategies and boost their income with additional optimization and refining.

VI. Reference

Brownlee, J. (2021). How to Develop Your First XGBoost Model in Python. MachineLearningMastery.com.
https://machinelearningmastery.com/develop-first-xgboost-model-python-scikit-learn/