**PlayerUnknown's battleground winning prediction**

**Marrapu lavanya1, Vangapandu harika1,Geeta anil kumar1,Chandra mouli1, Peter hrudayraj1, Anupama Angadi2**

1Department of Information Technology, Anil Neerukonda Institute of Technology and

Sciences, Sangivasala, Visakhapatnam, Andhra Pradesh, India.

2Associate professor, Department of Information Technology , Anil Neerukonda Institute of Technology and Sciences, Sangivasala, Visakhapatnam, Andhra Pradesh, India.

**ABSTRACT:**The aim of this project is to predict a player's ranking in the popular video game PUBG,(PlayerUnknown's Battlegrounds) by analysing their position and statistical data. In recent years, PUBG has gained immense popularity, and the last position achieved by a player is considered the most important indicator of their skill level in this game. Our goal is to predict the final rank of the player using a dataset containing multiple variables such as team kills, win points, swim distance, match time, kill streaks, etc. This is essentially a regression problem, and our objective is to predict the player's victory place percentage between 0 and 1. We will be using various machine learning techniques, including LGBM Regressor, after performing exploratory analysis on the data. We will also compare other models like Linear Regression, Random Forest, Decision Trees, XG boost, and Light GBM using the same dataset. To evaluate the accuracy of each model, we will use validation data and identify the algorithm that provides the highest accuracy.

**KEYWORDS**:

PUBG, Random Forest, Linear regression, Light GBM, win place Percentile, XG boost, Decision trees.

**INTRODUCTION**:

The game developed by PUBG Corporation, called Player Unknown BattleGrounds, is a multiplayer online battle game where 100

players participate in combat against each other. Players have the option to fight solo or in a team of two or four, and the team size can be altered using specific features.

The objective of the game is for players to survive until the end, and the winner is the player who successfully achieves this goal. There are various strategies that players can employ to achieve this, including killing opponents using weapons, healing themselves, or hiding. Players can use a range of resources to defeat their foes, including armor, weapons, healing supplies, and vehicles. The dataset used in this study includes 28 features and five million records. The main aim is to predict the winning percentage of players based on factors such as the number of kills, survival rates, and the number of players alive. An algorithm is used to calculate the players' winning percentage, which ranges between 0 and 1. To achieve this, data is analyzed and features are extracted using techniques such as Linear Regression, Random Forest, Decision Tree, XG Boost, and Light GBM. The accuracy of the model is validated, and the results obtained meet the desired requirements. This paper is presented in the following order.

* Related work that is researched on the same data as well as a similar game prediction.
* The methodology used for placement prediction
* Data description with the explanation of their features
* EDA data analysis
* Experimental Setup the process

explanation to perform validation on data

* Results obtained by the applied methods
* The conclusion of the overall research.

**Data Description:**

**Data fields**

* **DBNOs** - The count of opponents knocked down by the player.
* **Assists** - The number of enemy players that were injured or killed by this player due to the actions of their teammates.
* **boosts** - The total amount of boost items used.
* **damageDealt** - The overall damage inflicted, excluding self-inflicted damage.
* **headshotKills** - The total number of enemy players killed by headshots.
* **heals** - The number of items used for healing.
* **Id** - The unique ID of the player.
* **killPlace** - A ranking based on the number of enemies killed by the player.
* **killPoints** - An external ranking of players based only on the number of kills. It can be thought of as an Elo ranking. If rankPoints has a value other than -1, any 0 in killPoints should be treated as "None".
* **killStreaks** - The maximum number of enemy players killed consecutively.
* **kills** - The total number of enemies killed by the player.
* **longestKill** - The greatest distance between the player and the player killed at the time of death. This may be misleading as downing a player and driving away can result in a large longestKill stat.
* **matchDuration** - The duration of the match in seconds.
* **matchId** - Matches are identified by their unique ID called matchId. The testing set and the training set do not contain any matches.
* **matchType** - A string that identifies the game mode the data is from. Standard modes include "solo", "duo", "squad", "solo-fpp", "duo-fpp", and "squad-fpp", while other modes are from events or custom matches.
* **rankPoints** - An Elo-like ranking of players, but it is inconsistent and being deprecated in the API's next version. Therefore, use it with caution. A value of 1 replaces "None".
* **rideDistance** - A metric measurement of the total distance traveled by vehicles.
* **roadKills** - The number of kills made while in a vehicle
* **swimDistance** - a metric measurement of the total distance travelled by swimming.
* **teamKills** - Number of times this player killed a partner
* **vehicleDestroys** - The number of vehicles that have been destroyed
* **walkDistance** - Total distance traveled on foot measured in meters.
* **weaponsAcquired** - The total number of weapons acquired.
* **winPoints** - This is a ranking of players based on their wins in the game. Similar to Elo ranking, only winning matters. In case there is a non-negative value in rankPoints, then any zero in winPoints should be considered as "None".
* **groupId** - groupId is a unique ID used to identify a group of players participating in a match. If the same group plays in multiple matches, each time a new groupId is assigned.
* **numGroups** - This is the number of groups in the match for which data is available.
* **maxPlace** - This indicates the worst placement achieved in the match for which data is available. Since the data may skip some placements, it may not match the number of groups.
* **winPlacePerc** - This is the target for the prediction. It represents the winning percentage of a player, where 1 represents the first place and 0 represents the last place in the match. Since it is based on maxPlace instead of numGroups, there may be missing chunks in a match.

**Project Scope and Direction**: Player Unknown's Battlegrounds, or PUBG for short, is a multiplayer online game. The ranking is important for knowing where each player stands in relation to the game they play. The project's objective is to investigate the system's existing features, add some new ones, and test various models to select the model with the highest performance. The target is a regression problem with continuous values ranging from 0 to 1. The quantile percentage of each player's placement is the variable that needs to be predicted.

**Impact, Significance, and Contribution**: Player Unknown's BattleGrounds (PUBG) is a well-known game where the popular phrase "Winner Winner Chicken Dinner!" is frequently heard. In order to succeed in the game, it's crucial to use strategic techniques based on the current situation. Therefore, collecting and analyzing data from PUBG can be a valuable effort to help players survive. This dataset contains 28 features, including the number of enemies killed and the length of the match, which can be used to predict a player's winning placement. We aim to find the best model for this prediction using the available data. In previous studies, the performance of PUBG players was evaluated using AI and deep learning algorithms, and exploratory data analysis (EDA) was performed to better understand the dataset. Various algorithms, including Direct Regression, Random Forest, and Deep Neural Network, were used in these studies, and the MAE (Mean Absolute Error) was calculated to determine the most suitable algorithm for the large dataset.

**Goals and objectives**: The game PUBG has gained popularity in recent years. In this game, the player's ability is primarily measured by their last position. The goal of this task is to anticipate the final position and find the best ways to play the game. To predict who will win, we use information from PUBG and a few AI techniques, such as the light GBM model and the linear regression model.

Almost 100 individuals, with no gear, hop onto an island toward the beginning of each play. To win the game and eliminate the other players, you must scavenge for weapons and equipment.

Therefore, if you want to watch "Winner Winner Chicken Dinner!," click here. on your showcase, it's exceptionally crucial for make a few sensible techniques in view of genuine condition. For instance, a reasonable area of dropping can assist you with gathering the hardware quicker, the way you escape from the consistently contracting blue circle (out of the circle players will get harm ceaselessly until they bite the dust) can assist you with staying away from a hazardous foes and track down an appropriate area to conceal in the last circle is useful get the 'Chicken Dinner'.

Because of these factors, collecting and analyzing PUBG data is a worthwhile effort to assist the player in finally surviving. Based on the information gathered through the PUBG developer API, we are working on an intriguing project that aims to predict the final ranking percentile.

There are 28 features in this dataset, such as the number of enemy players killed and the duration of the battle. We try to find the best model to predict the winning placement in the final percentile using these data. We need to train a model to predict the player's ranking in the testing set based on their statistical data from each match, which we can do by using the ranking information in the training data and the statistical data from previous players. A higher percentage indicates a higher ranking in that match, and the target label ranking will be a percentage value between 0 and 1.

**II. RELATED RESEARCH**

The PUBG ranking system was assessed using machine learning and deep learning algorithms, and exploratory data analysis was performed to better understand the dataset. Direct Regression,

Random Forest, and Deep Neural Network algorithms were used, with Mean Absolute Error (MAE) used to determine which algorithm was best suited for the large dataset. The evaluation revealed that the deep neural network performed well, with MAE values of 0.04012 and 0.07121 for the training and testing datasets, respectively. Basic Random Forest with n\_estimators=42 and max\_features=Sqrt had the highest MAE value for the testing dataset, while Linear Regression had the highest MAE value for the training dataset. The Deep Neural Network had the lowest error value for the testing dataset and was deemed the best algorithm. After exploratory data analysis, the LGBM Regression was used to train the dataset, and four additional models (Decision Tree, Linear Regression, Random Forest Algorithm, and XgBoost) were compared. The validation data was used to identify the best algorithm and report the accuracy of each model. The dataset contained features such as team kills, win points, swim distance, match duration, and kill streaks. The highest number of kills ever recorded in a single match was 72, and 99% of players had fewer than 7.0 killsEach PUBG match starts with up to 100 players (matchId). Players can be on teams (groupId) that are ranked at the conclusion of the game (winPlacePerc) according to the number of other teams that are still alive when they are eliminated. In game, players can get various weapons, resuscitate brought down however not-out (thumped) colleagues, drive vehicles, swim, run, shoot, and experience the results as a whole - - like falling excessively far or running themselves over and disposing of themselves. An abundance of anonymized PUBG game stats are presented to you, each row containing a player's post-game stats. The data come from all kinds of matches: performances, couples, crews, and custom; There will not always be 100 players in each match, and there will only be four players in each group.

Pre-processing: For the most part, we can see that specific elements certainly provide us with a great deal of data while others give no contribution to terms of our objective variable. This not only demonstrates that the data we have is indeed valuable, but it also provides us with a foundation upon which to apply it. This implies the following stage is getting the real worth from that information. Therefore, we will preprocess the data as follows:

1. omit features that are assumed to be irrelevant: groupId
2. id Type of one-hot encode match: solo (one person), duo (one to two people), or squad (team)
3. Eliminate all rows with NaNs
4. Divide into train and test in such a way that neither test nor train contain instances of the same match.

We can use dimensionality reduction algorithms to automatically remove features that do not appear to be providing any information, or we can manually remove those features.

We are confident in the methods we learned this semester to find that value and believe we can solve our problem by confirming the value of our data. We hope to build a prediction engine that can tell us how likely a given tactic is to win, so Model the data and build a prediction engine.

**III. METHODOLOGY:**

**A)Light GBM (Gradient Boost Machine)**

:

The LGBM is a high-speed gradient boosting framework that utilizes a unique algorithm different from traditional decision tree algorithms. It divides the data either widthwise or depthwise instead of leafwise. This results in more efficient and improved performance than the current algorithms. The LGBM algorithm reduces level order while using the same leafwise algorithm. When trained using deep learning, the Adams learning rate of 0.01 is used to optimize data. This model provides several benefits, including faster training and using less memory, as well as better compatibility with large datasets, and improved accuracy through the gradient boosting algorithm. The primary aim is to reduce mean squared error (MSE) or function loss using this model. The gradient boosting model minimizes loss and uses alpha as the learning rate to determine the predicted value. The best parameters for this model are a learning rate of 0.3 N, a 200 leaf node count, and 250 iterations. **B) Linear Regression:**

we can test out our ideas and focus on winning, rather than improving individually at various in-game tactics. We intend to accomplish this by following these steps:

1. Explore the data
2. Pre-process and engineer the features 3.

In the Linear Regression machine learning algorithm, the best fit line is used to make predictions, and the target attributes are always numerical. The plotted data for this type of regression is based on the Dependent variable, which is represented on the Y-axis, and the Independent variable, which is represented on the Xaxis. A best-fit line, plane, or slope that satisfies those values is created for both axes. The dots around the slope represent the actual values, we can test out our ideas and focus on winning, rather than improving individually at various in-game tactics. We intend to accomplish this by following these steps: 1. Explore the data

1. Pre-process and engineer the features
2. while the slope represents the predicted values.

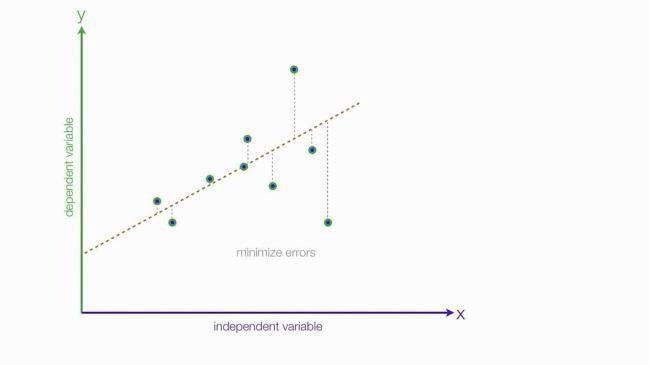
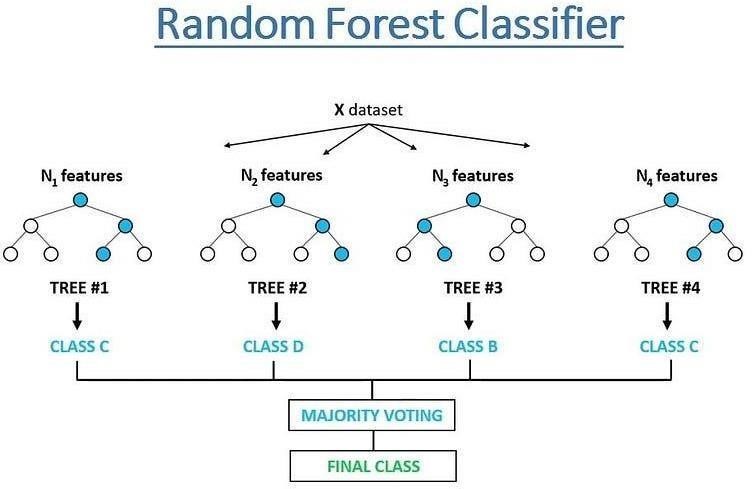


Fig 1. Linear Regression algorithm **C)Random Forest Algorithm :**

The bagging and boosting techniques of ensemble learning allow Random Forest or Random Decision Forest to be utilized for Regression and Classification, depending on the specific use case. The decision tree algorithm yields better results when compared to a single decision tree because it is constructed from multiple trees that form a forest of decisions. The random forest hierarchy comprises various levels, including root, leaf, child, parent, and others. The random forest algorithm relies on majority voting for the final result, where each tree has several results, and the final result is based on the average of the results, producing accurate outcomes. These models have a superior expansion of judgments based on the training data and provide better accuracy compared to other models, as demonstrated in the image below.



**Fig 2. Random Forest Algorithm D) Decision Tree**:

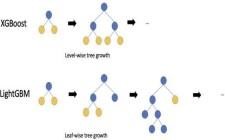
Decision Tree is a type of supervised learning that can handle classification and regression problems, but it is usually used for classification tasks. It employs a treelike structure to classify data, with internal nodes representing features of the dataset, branches representing decision rules, and leaf nodes representing the output. The two types of nodes in a decision tree are decision nodes and leaf nodes. While decision nodes indicate the decisionmaking process based on the dataset's features, leaf nodes provide the output based on thosedecisions.

The Decision Tree is a graphical representation that contains two types of nodes: Decision Nodes and Leaf Nodes. While Leaf Nodes do not have any additional branches and represent the output or final result of the decisions made, Decision Nodes have multiple branches and are used to make decisions based on the features of the dataset. Essentially, it provides all possible solutions to a given problem or decision based on the given conditions.

**E) XgBoost**:

XGBoost, or Extreme Gradient Boosting, becomes the most obvious choice for a superfast machine learning algorithm that works on tree based models and strives for best-in-class accuracy by making optimal use of computational resources.

The XGBoost algorithm, developed by Tianqi Chen, has recently gained a lot of popularity due to its widespread use in the majority of hackathons and Kaggle competitions. straightforward terms, XGBoost might be officially characterized as a choice tree-based gathering learning structure that involves Slope Drop as the hidden goal capability and accompanies a ton of adaptability while conveying the craving results by ideally utilizing computational power.

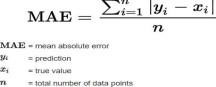


**Fig 3. XGBoost and Light GBM IV. PROPOSED APPROACH:**

After conducting exploratory data analysis, we trained our training dataset with the LGBM (Light Gradient Boosting Machine) Regressor. With the same dataset containing various features like team kills, win points, swim distance, match duration, kill streaks, etc., we compared four additional models: Decision tree, linear regression, random forest algorithm, and XgBoost. Based on the validation data, we identified the best algorithm and reported the accuracy of each model.

**V. RESULTS**

The Mean Absolute Error (MAE) is a formula that calculates the absolute difference between the predicted values and the actual values, which is then divided by the number of observations. This formula is used to measure the amount of loss in data during training and validation. The MAE helps to determine the total absolute error that occurred during the prediction of relevant data.



The Mean Absolute Error (MAE) values for various algorithms used in predicting the ranking of PUBG datasets are presented in Table I. The MAE values are calculated for the training datasets using all the algorithms.

|  |  |  |
| --- | --- | --- |
| **S. No** | **Algorithm** | **Training**  **MAE**  **Score** |
| **1** | Linear  Regression | 0.0900 |
| **2** | Decision Tree | 1.646 |
| **3** | Random  Forest | 0.023 |
| **4** | XgBoost | 0.053 |
| **5** | LightGbm | 0.026 |

The second table displays the Mean Absolute Error (MAE) values of the various algorithms used in predicting the rankings of the PUBG datasets, specifically for the validation or testing datasets

|  |  |  |
| --- | --- | --- |
| **S No** | **Algorithms** | **Testing**  **MAE**  **Score** |
| **1** | Linear  Regression | 0.0903 |
| **2** | Decision Tree | 0.087 |
| **3** | Random  Forest | 0.062 |
| **4** | XgBoost | 0.060 |
| **5** | LightGbm | 0.052 |

**Graph** :



0



1



2



Training MAE



Testing MAE

1. **CONCLUSION:**

The study aimed to use machine learning and deep learning algorithms to evaluate the ranking of PUBG players, in addition to performing EDA for improved dataset analysis. Linear Regression, Random Forest, and Light Gbm were the algorithms utilized in this research. The MAE (Mean Absolute Error) was used to determine the algorithm that worked best with the large dataset. The assessment was carried out for both the training and testing datasets, with the profound brain network having an MAE value of 0.04012 and 0.07121 for the training and testing datasets, respectively. The highest MAE value for the training dataset was 0.09521 for Linear Regression, while the highest MAE value for the testing dataset was 0.09521 for Basic Random Forest with n\_estimators equal to 42 and max\_features equal to Sqrt. The best algorithm among all was LightGbm with the lowest error value (MAE worth of 0.02) for the training data and 0.05 for the testing data 4024.

1. **REFERENCES**:
2. Ding, Yong. “Research on operational model of PUBG.” MATEC Web of Conferences. Vol. 173. EDP

Sciences, 2018.

1. Rokad, Brij, et al. “Survival of the Fittest in PlayerUnknown BattleGround.” arXiv preprint arXiv:1905.06052 (2019).
2. D’Souza, Lancy, S. Manish, and S.

Deeksha. “Development and Validation of PUBG

Addiction Test (PAT).” (2019).

1. Mamulpet, Madhurya Manjunath.

“PUBG WINNER PLACEMENT PREDICTION USING ARTIFICIAL

NEURAL NETWORK.”

1. Hodge V, Devlin S, Sephton

N, Block F,

Drachen A, Cowling P. Win Prediction in Esports: Mixed-Rank Match Prediction in Multi player Online Battle Arena Games.

1. Brij Rokad et al., "Survival of the Fittest in PlayerUnknown BattleGround", arXiv preprint arXiv:1905.06052, 2019.
2. David Melhart, Daniele Gravina and

Georgios N. Yannakakis, "Moment-tomoment

Engagement Prediction through the Eyes of

the Observer: PUBG Streaming on Twitch",

International Conference on the

Foundations of Digital Games, 2020.

1. Mr. Parimal Dagdee and Leena

Philip, THE

RISE OF PUBG AND THE MARKETING

STRATEGIES BEHIND ITS SUCCESS,

2019.

1. Andy Liaw and Matthew Wiener,

"Classification and regression by randomForest", R news, vol. 2.3, no. 2002, pp. 18-22.

1. Douglas C. Montgomery, Elizabeth A. Peck and G. Geoffrey Vining, Introduction to linear regression analysis, John Wiley & Sons, vol. 821, 2012.
2. Angshuman Paul et al., "Improved random forest for classification", IEEE Transactions on Image Processing, vol.

27.8, no. 2018, pp. 4012-