The Road to Chicken Dinner: An analysis of PUBG gameplay statistics

Utkarsh Gupta*, Nistha Kumar*
University of California, Santa Cruz

Abstract

PlayerUnknown's Battlegrounds (PUBG) is an online multiplayer battle royale game with millions of players worldwide. This project analyzes a dataset of PUBG match statistics to develop models for predicting the winning placement percentile of players and the type of match. Linear regression is utilized to predict a player's likelihood to win based on features like number of kills, damage dealt, healing items used, etc. Additionally, logistic regression is implemented to classify the matches as solo, duo, or squad matches using all the available predictors. The linear model attained an R-squared score of 0.88 in predicting winning likelihood. Meanwhile, the logistic regression classifier reached 99.67% accuracy in identifying the match type. The results indicate significant predictive capabilities from the models and demonstrate the insights that can be gained about PUBG matches through statistical analysis. Practical applications could help players evaluate their in-game performance or assist gaming companies in matchmaking design.

1. Introduction

Video games that mimic Battle Royale have become extremely popular worldwide. The play zone keeps getting smaller as 100 players are dumped off on an empty island where they must explore, scavenge, and eliminate other players until only one is left standing, all while the play zone continues to shrink. What's the best strategy to win in PUBG? Should you sit in one spot and hide your way into victory, or do you need to be the top shot? Let's let the data do the talking!

The data consists of many anonymized PUBG game stats, formatted so that each row contains one player's post-game stats. The data comes from matches of all types: solos, duos, squads, and custom; there is no guarantee of there being 100 players per match, nor at most 4 players per group. In a PUBG game, up to 100 players start in each match (matchId). Players can be on teams (groupId) that get ranked at the end of the game (winPlacePerc) based on how many other teams are still alive when they are eliminated. In the game, players can pick up different munitions, revive downed-but-not-out (knocked) teammates, drive vehicles, swim, run, shoot, and experience all of the consequences, such as falling too far or running themselves over and eliminating themselves.

The dataset[1] has around 4.4 million rows and 29 columns. Each row in the dataset represents the data for one particular player and there is data for all the players from matches. There is no duplicate row for players and each row is independent.

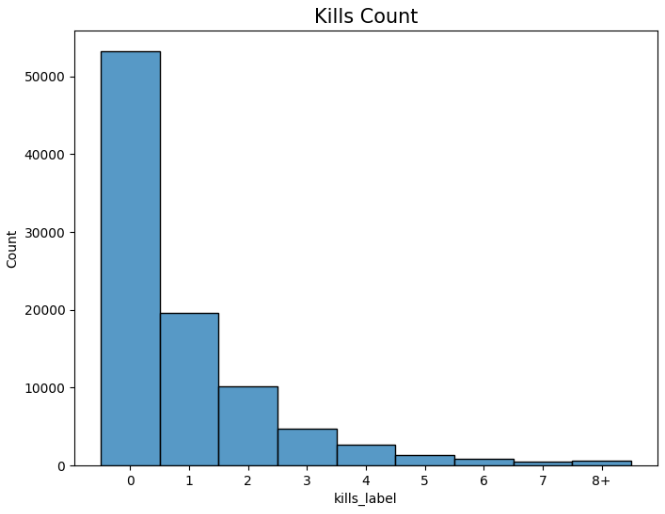

Figure 1: Visualization of the kills count as a histogram to detect the skewness in the dataset.

1.1 Objectives

In this project work, our contributions are as follows:

1. Using the multiple regression techniques, we will predict players' finishing placement (winPlacePerc) and perform model selection using AIC and BIC.

2. Unlike previous works, we will also predict the match type, a qualitative attribute, for a given observation.

3. We will try to uncover the relationship between the attributes revives, team kills, individual kills, match type, etc., with the winPlacePerc response. This helps us answer the impact of team play to win the game.

4. Lastly, we will quantify the significance of walking, swimming, and riding strategies to win the game.

1.2 Relevant work

[3] created derived features using the existing attributes to enhance the accuracy of their proposed models. Another work by [4] has concentrated more on feature engineering and feature selection. They only worked on gathering the optimal number of features and training various models to create a leaderboard based on their accuracy

* Equal contribution. Correspondence: {utgupta, nkumar20}@ucsc.edu

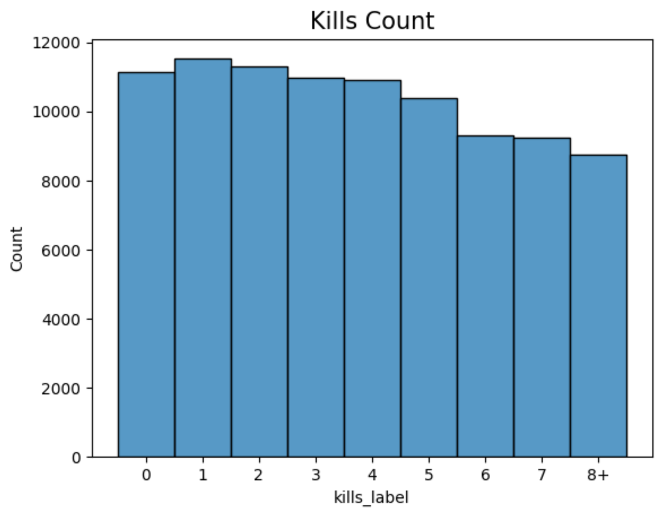

Figure 2: Visualization of the balanced kills count as a histogram. We reduced the number of zero kill rows and traversed extra matches to get the players for different values of kills.

on the test set. While the model selection was done quite well in both works, they could have displayed statistical insights to uncover the uncharted relationships. Therefore, we plan to contribute to the PUBG community by sharing our statistical insights to improve the gameplay. Lastly, [2] has performed a rigorous data analysis that we believe is crucial for us to understand the impact of the research problems that we are trying to solve.

2. Data Manipulation

The dataset[1] has around 4.4 million rows and 29 predictors. We reduced the number of instances from the raw dataset to 93.5K rows to perform a meaningful Exploratory Data Analysis. This led us to select the data of around 1000 matches for this project. While reducing the number of instances, we ensured that our smaller dataset comprised the data representing the whole dataset. Therefore, we carefully designed the code in Python to generate a balanced dataset, which will be explained in subsections §2.1 and §2.2, corresponding to training and testing, respectively. Moreover, attributes like ID, groupId, matchId, numGroups, and rankPoints were dropped as they do not contribute anything to the response variables.

2.1 Training Set

While creating the training set, the attributes *matchType* and *kills* have played a pivotal role. From figure 1, the number of kills suggests the data is right-skewed, which implies that randomly selecting the rows from the raw dataset is a wrong choice. Therefore, we engineered an approach to select fewer rows of zero kills, ensuring equilibrium among kills as shown in 2. Another attribute

Algorithm 1 Building test dataset T for Logistic Regression
M: Match IDs for Test Set
D: Raw Dataset
β: Set of Match Types
δ : Total number of instances in the test dataset
γ: Maximum number of players for a match type

```
$D \leftarrow [d_1, d_2, ..., d_k]$
$M \leftarrow [m_1, m_2, ..., m_n]$
$T \leftarrow \{\phi\}$
$\beta \leftarrow \{solo, duo, squad, solofpp, duofpp, squadfpp\}$
$\gamma \leftarrow 100$
$\delta \leftarrow length(\beta) * \gamma$
for $m \in M$ do                    $\triangleright \{d_i, d_{i+1}, ..., d_j\} \in D[m]$
    $x \leftarrow D[m].matchType$
    if $T.numInstances = \delta$ then
        Break
    else if $T.count(x) = \gamma$ then
        Continue
    end if
    $p \leftarrow Randomly\ sample\ d_p\ from\ D[m]$
    $T \leftarrow T + \{p\}$
end for
```

was *matchType* that needed our attention while creating a balanced dataset. The requirement of the subset being a representative of the complete dataset led us to lay out an approach such that all 6 match types have been sampled across all the parts of the raw dataset to maintain equilibrium.

2.2 Testing Set

Generating a test set for linear and logistic regressions also allowed us to deeply examine the dataset. It was trivial to build a test set for Linear Regression. However, it was a challenging task to create a test set for Logistic Regression. For Linear Regression, we selected one player per match from the total dataset of 44.7K matches. However, for the logistic regression, we had to carefully design an approach as shown in Algorithm 1 such that the test dataset had an equal number of instances from all match types. Also, we only selected one random player from a given match to keep the test set representative of the overall dataset. Therefore, we successfully built a test set with 600 rows, approximately 100 rows per match type, with the cost of our code being able to run in approximately 18-20 minutes.

3. EDA

The PUBG dataset provided detailed match statistics across over 65,000 games with no missing values, enabling a comprehensive exploratory analysis. The initial investigation focused on the distribution of match types (solo, duo, squad) to understand the composition of the data. As Figure 3 shows, squads in first-person perspec-

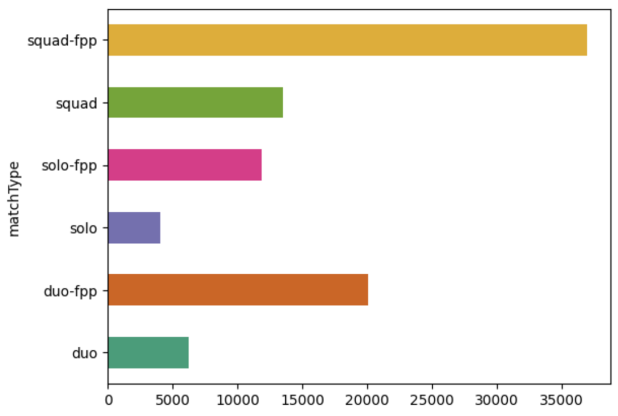

Figure 3: Visualization of *matchType* counts. The diagram implies that balancing match types across the training and testing set must be considered.

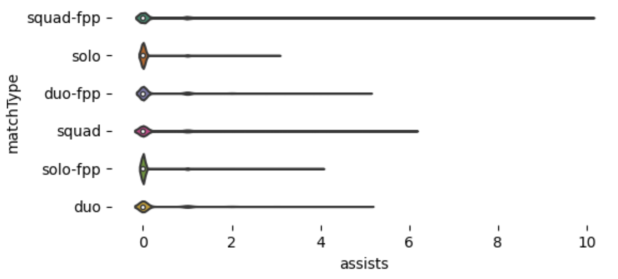

Figure 4: Visualization of *matchType* counts as a Violin plot. The distribution is broad, around 0 assists.

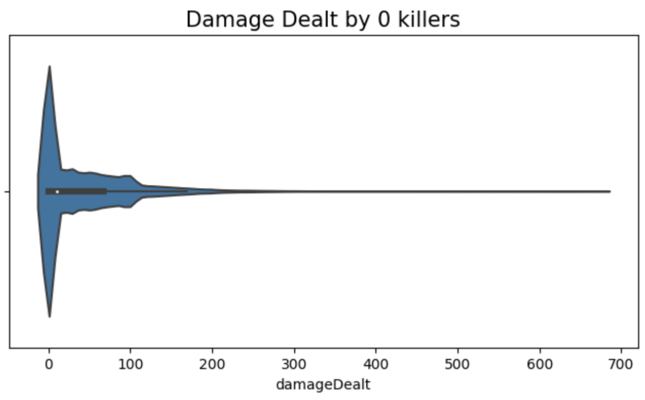

Figure 5: Visualization of *DamageDealt* players having 0 kills.

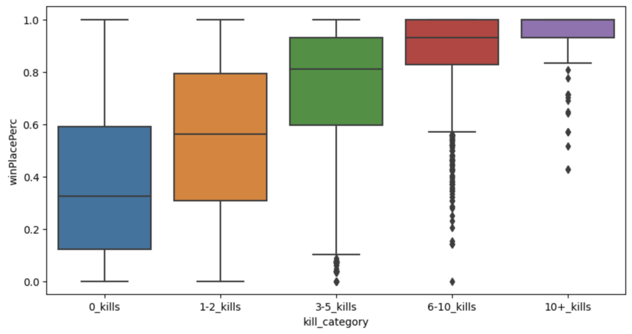

Figure 6: Visualization of the derived attribute *kill_category*. We divided kills into five categories - 0, 1-2, 3-5, 6-10, and 10+ kills.

tive (squad-fpp) comprised the largest portion at 50%, while solo matches accounted for only 10%. This skew towards squad games needed to be considered when constructing the dataset and models.

Since PUBG matches squad players together, the number of assists where teammates help eliminate opponents was hypothesized to differ by match type. Violin plots visually confirmed this in Figure 4, with solo and solo-fpp matches clustered around 0 assists, while duo and squads showed broader distributions. Solo players cannot assist teammates by definition, nicely validating our assumption.

An interesting finding was players with 0 kills yet serious damage dealt or even match wins, indicating they contributed meaningfully to their team. If we look at the *damageDealt* for players with 0 kills 5, we see that players with 0 kills also did quite a large amount of damage. There were players with 0 kills, and they won the match. The number of such players was 346. Also, there were players with 0 kills and 0 damage, and they won the match. There were 113 such players.

We defined a derived column *kill_category* and, using a pairwise t-test, determined the groups' significance. From figure 6, we can see that the mean for the groups is not the same; thus, the groups could be statistically significant. We got the same result using pairwise t-tests.

An analysis of feature correlations revealed insights that informed model construction. As shown in figure 7, the *killPoints* and *winPoints* predictors exhibited a high correlation (*r=0.89*), indicating potential redundancy. Including both could overfit regression models. Thus, only one was retained, with *winPoints* chosen due to a slightly higher individual correlation to the target *winPlacePerc*.

Additionally, ranking features by their Pearson correlation coefficients with the winning percentage label provided priorities for forward selection. This greedy algorithm incrementally adds the most predictive variable at each step. As visualized in Figure 7, *walkDistance* showed the highest correlation to *winPlacePerc* at 0.63, followed by *weaponsAcquired* and *boosts*. The forward selection process, therefore, built up the model starting with just *walkDistance*, then adding *weaponsAcquired* as the next most informative variable, and so on.

By considering both multicollinearities between features and their individual relationships to the predicted variable, the analysis optimized inputs for model training. Removing redundant variables mitigates overfitting while prioritizing correlated features allows efficiently honing in on the most relevant set to explain the winning percentage. Together, these steps augmented model generaliz-

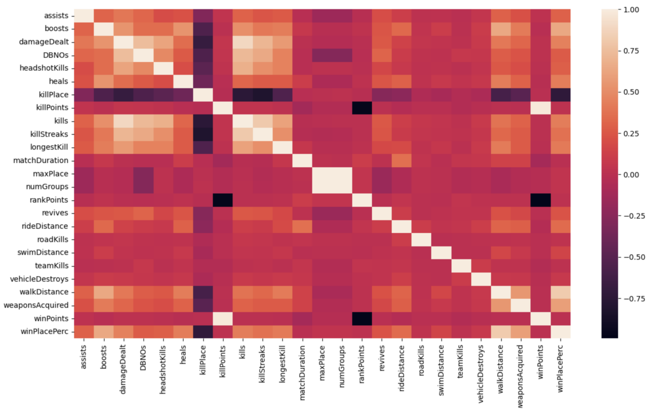

Figure 7: Visualization of the correlation matrix.

ability and performance.

4. Method

Firstly, we propose using Multiple Linear Regression to predict the winPlacePerc using backward elimination and forward selection. The general model formulation is as follows:

$$y = \beta_0 + \beta_{Assist} X_{Assist} + \beta_{Heals} X_{Heals} + ... + \beta_n X_n$$

where $n = 29$ predictors. Using the information criteria like AIC and BIC, we will perform model selection among the regression models. Moreover, we did the variable selection procedure to create subsets of predictors that best describe the impact of team play. We checked the p-values of the coefficients along with the Adjusted R^2 criterion to pick the best predictors. Since we are building models with a large number of predictors, we also performed p-value correction to identify the actual significant predictors. We will train several regression models to achieve the objective of predicting the *winPlacePerc* attribute, quantifying the impact of team play attributes, and identifying the impact of gameplay strategies on the response variable.

Secondly, we trained a Multinomial Logistic Regression classifier to predict the *matchType* response, a nominal variable, with the extracted predictors from the variable selection procedure. The different match types are solo, solo-fpp, duo, duo-fpp, squad and squad-fpp. The model will have the following formulation:

$$\log \frac{P_{C_j}}{P_{C_6}} = \beta_{0j} + \beta_{1j} X_1 + \beta_{2j} X_2 + ... + \beta_{nj} X_n$$

where P_{C_j} is the probability value for the class j and $n = 29$ predictors. To evaluate the model, we will report the confusion matrix and the classification accuracy achieved in the test set. Like p-value adjustment performed for the Multiple Linear Regression models, we also followed the same procedure for identifying the significant variables for the Logistic Regression model.

Table 1: R^2, adjusted R^2, AIC, and BIC values for forward selection in Multiple Linear Regression. While reading the table from top to bottom, the attributes are getting added to the model one by one. AIC and BIC values decrease as the attributes are appended to the model.

Predictor	R^2	*Adjusted*R^2	*AIC*	*BIC*
walkDistance	0.663	0.663	-57.57	-57.54
killPlace	0.752	0.752	-86.37	-86.33
boosts	0.756	0.756	-87.91	-87.86
weapons	0.768	0.768	-92.86	-92.81
kills	0.787	0.787	-100.73	-100.66
killStreaks	0.815	0.815	-113.95	-113.88
rideDistance	0.816	0.816	-114.41	-114.33
assists	0.817	0.817	-114.69	-114.59
DBNOs	0.818	0.818	-115.34	-115.23
revives	0.818	0.818	-115.41	-115.30
swimDistance	0.818	0.818	-115.57	-115.45
numGroups	0.823	0.823	-117.89	-117.76
maxPlace	0.824	0.824	-118.59	-118.45
duration	0.840	0.840	-127.62	-127.47
killCategory	0.868	0.868	-145.92	-145.73
matchType	0.876	0.876	-151.32	-151.08

5. Experiments

5.1 Predicting *winPlacePerc* response

5.1.1 Backward Elimination

Backward elimination is a feature selection technique that starts by fitting a multiple regression model with all candidate predictor variables included. Then, at each step, we remove the least useful predictor variable in the model - the one that is the least statistically significant, in order to improve the model fit. Significance is measured by the p-values on the coefficients as an evaluation criterion. Removing the weakest correlated variable continues iteratively until all remaining independent variables in the model have individual p-values below a 5% significance level. The table 2 shows the predictors that are significant after applying Backward Elimination. The correction of p-values was performed using Bonferroni adjustment, and there was not a significant change that would let us change the model. The MSE of this model is 0.115. Additionally, figures 9 and 10 show the residual-vs-fitted and QQ plots for the trained model. They conclude that the data follows the assumption of normality, and it also doesn't have non-linearity in it. Finally, figure 8 shows the predicted-vs-expected values for the Linear Regression model, which looks good as per our analysis.

5.1.2 Forward Selection

Forward selection starts with no predictor variables, followed by adding the most significant variables one by one based on which candidate input variable has the highest correlation with the target/outcome variable. Sig-

Table 2: This table shows the significant predictors after applying Backward Elimination. Moreover, the Unadjusted and Adjusted columns tell about the p-value corrections done using the Bonferroni adjusted method.

Predictor	*Unadjusted*	*Adjusted*
(Intercept)	0.000000e+00	0.000000e+00
assists	2.179220e-137	3.050908e-136
boosts	0.000000e+00	0.000000e+00
damageDealt	2.149708e-47	1.504796e-46
DBNOs	7.394143e-146	1.109121e-144
heals	4.901532e-08	9.803063e-08
killPlace	0.000000e+00	0.000000e+00
kills	0.000000e+00	0.000000e+00
killStreaks	4.624083e-87	5.548899e-86
longestKill	3.182210e-279	5.091536e-278
matchDuration	0.000000e+00	0.000000e+00
matchTypeduo-fpp	9.842955e-01	9.842955e-01
matchTypesolo	0.000000e+00	0.000000e+00
matchTypesolo-fpp	0.000000e+00	0.000000e+00
matchTypesquad	0.000000e+00	0.000000e+00
matchTypesquad-fpp	0.000000e+00	0.000000e+00
maxPlace	1.589834e-34	7.949168e-34
numGroups	2.921919e-116	3.798495e-115
rankPoints	2.092872e-82	2.302159e-81
revives	3.477876e-66	3.130088e-65
rideDistance	0.000000e+00	0.000000e+00
swimDistance	1.823543e-20	7.294170e-20
teamKills	1.016854e-19	3.050561e-19
walkDistance	0.000000e+00	0.000000e+00
weaponsAcquired	0.000000e+00	0.000000e+00
winPoints	3.824894e-80	3.824894e-79
kill_category1-2_kills	0.000000e+00	0.000000e+00
kill_category10+_kills	7.849980e-62	6.279984e-61
kill_category3-5_kills	0.000000e+00	0.000000e+00
kill_category6-10_kills	6.961080e-42	4.176648e-41

nificance is measured by p-values on correlation coefficients or criteria like AIC and BIC. The model with the least AIC and BIC is considered to be the best. Table 1, explains our experiment regarding the same. The addition of predictors starts from *walkDistance*, followed by *killPlace*, *boosts*, *weapons*, *kills*, *killsStreaks*, *rideDistance*, *assists*, *DBNOs*, *revives*, *swinDistance*, *numGroups*, *maxPlace*, *duration*, *killCategory*, and *matchType*. The AIC and BIC values decrease as the predictors are getting added. The R^2 and adjusted R^2 are increasing from top to bottom, as shown in the table. The MSE value of the model is 0.122.

5.1.3 Model Selection

With good accuracy for our prediction and better R-square for our models as our main goal, we want to extract the subset of significant features. We used the approach of Backward elimination and forward selection as explained in §5.1.1 and §5.1.2, respectively. We manually started with the feature selection process: first, we fit the model with all features to get M_0 and selected the features with a p-value smaller than 0.05. Then, we refit the model with only the subset of significant features from step 1 to get M_1. We also dropped killPoints because of its high correlation with winPoints. Finally, we selected M_1 because of its higher adjusted R-square value. The AIC for the final model is smaller. The second approach we use is forward selection. We propose a null model M_{null} to regress winPlacePerc. We add the most correlated feature to the model and check the change in AIC and BIC. At each step, we keep adding the features in decreasing order of correlation and then stop at a point where the criteria stop decreasing. Eventually, we choose the model from the backward elimination M_1 by comparing the adjusted R^2 and the AICs with every other models.

5.2 Predicting *matchType* response

Predicting *matchType* is a classification problem that made us train a Multinomial Logistic Regression model on all the predictors for 6 match types as explained in §4. We call this model as M_{six}. We derived surprising results by testing the logistic regression model on the test set. The confusion matrix table 3 suggests that the logistic regression is confused among match types that differ in gameplay. For instance, almost all the instances of *duo* are classified as *duo-fpp*. *solo* and *solo-fpp*, *squad* and *squad-fpp* are also following the same pattern. The accuracy for the model is only 53.42%. However, the only difference between the categories was the camera angle of a first-person-player (fpp) and a third-person-player game.

Based on the behavior of the previous model, we decided to merge the categories of third-person-player games into first-person-player games. For instance, *solo* into *solo-fpp*, and the same for the duo and squad match types. After retraining the multinomial logistic regression model on 3 classes, and calling the model as M_{three}, the confusion matrix table 4 suggests that the data doesn't have much difference between a third-person-player and a first-person-player games. The prediction results support this, as the accuracy is 99.67%. In §6, we have also performed goodness-of-fit tests to ensure whether these models truly reflect the underlying patterns in the dataset.

5.3 Impact of team play attributes

Based on our game knowledge, we selected important attributes to determine the impact of team play on the *winPlacePerc* response. The attributes are *assists*, *revives*, *teamKills*, *boosts*, *heals*, *killPoints*, *kills*, and *rankPoints*. The table 5 shows the summary of the trained model, and the inference was quite surprising. The *teamKills* attribute is insignificant. Also, the positive coefficient of remaining attributes indicates that the attributes that we have selected are integral to defining the success of a

Table 3: Confusion matrix for the Multinomial logistic regression model trained for 6 *matchType* classes using all the predictors

Prediction	*solo*	*solo-fpp*	*duo*	*duo-fpp*	*squad*	*squad-fpp*
solo	5	0	0	0	0	0
solo-fpp	88	101	0	0	0	0
duo	0	0	13	1	0	0
duo-fpp	1	0	88	100	0	0
squad	0	0	0	0	0	0
squad-fpp	0	0	0	0	101	101

Table 4: Confusion matrix for the Multinomial logistic regression model trained for 3 *matchType* classes using all the predictors

Prediction	*solo-fpp*	*duo-fpp*	*squad-fpp*
solo-fpp	193	0	0
duo-fpp	2	202	0
squad-fpp	0	0	202

Table 5: Multiple Linear Regression model trained to identify the impact of team play attributes.

Predictor	*Estimate*	*p-value*
(Intercept)	2.932e-01	$< 2e-16$ ***
assists	4.998e-02	$< 2e-16$ ***
revives	3.777e-02	$< 2e-16$ ***
teamKills	9.230e-03	0.056089 .
boosts	8.829e-02	$< 2e-16$ ***
heals	1.230e-02	$< 2e-16$ ***
killPoints	2.053e-05	0.000675 ***
kills	1.709e-02	$< 2e-16$ ***
rankPoints	1.698e-05	0.001120 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

team winning the game.

5.4 Impact of gaming strategies

This section is about solving the research problem of a player's strategy to win the game. To address this, we trained three separate models for strategies of walking, riding a vehicle, and swimming and determined the optimal way a player should go to win the game. The table 6 describes the three models for the different strategies. Also, the R^2 and Adjusted R^2 are the best for the walking strategy, implying that the best strategy a player could choose is walking to win the game. Moreover, strategies like swimming are not always feasible because a player can be eliminated easily if someone sees them from the high ground. Also, riding a vehicle doesn't promise a win because it causes noise and lets others know about a player's existence. This can make them an easy target from far regions in the game.

Table 6: Three Simple Linear Regression models to quantify the impact of gaming strategies on the *winPlacePerc* attribute.

Model	*Attribute*	R^2	$AdjustedR^2$
M_{walk}	walkDistance	**0.663**	**0.663**
M_{ride}	rideDistance	0.112	0.112
M_{swim}	swimDistance	0.024	0.024

6. Goodness of Fit

The Hosmer-Lemeshow test allows us to evaluate how well our logistic regression model fits the data. This goodness-of-fit test works by ordering the observations by their estimated probability of success based on the model's parameter estimates. These ordered observations are then segmented into deciles - ten groups with equal numbers of observations in each group.

Within each decile, the test tallies the number of 1 and 0 responses from the real data. It also calculates the expected number of 1s and 0s if the model's predicted probabilities were true by summing the estimated success probabilities for all observations where the real response was 1 and summing (1 - estimated probability) where the real response was 0. The statistic calculation is given as:

$$H = \sum_{g=1}^{G=10} \left(\frac{(O_{1g} - E_{1g})^2}{E_{1g}} + \frac{(O_{0g} - E_{0g})^2}{E_{0g}} \right)$$

The distribution of H is a χ^2 with $Q - 2$ degrees of freedom.

In formal terms, the null hypothesis assumes that within each decile, the observed ratio of 1 responses to 0 responses matches the ratio of expected 1s to 0s according to the logistic regression model. More broadly, the essence of the null hypothesis is that our model accurately reflects the true underlying patterns in the data - that it fits the data well.

From table 7, the p-values for the model M_{six} and M_{three} are very small for a 5% significance level. Therefore, we reject the null hypothesis for both types of logistic regression models. In other words, neither model accurately reflects the underlying patterns in the data.

Table 7: Hosmer–Lemeshow test results for logistic regression models trained on six and three categories.

Model	χ^2	*p-value*
M_{six}	386581	$< 2.2e - 16$
M_{three}	35088	$< 2.2e - 16$

7. Conclusion and Future Work

Our analysis determined effective models for predicting a player's probability of winning in PUBG based on their in-game stats and strategies. Our best model for *winPlacePerc* uses backward elimination, selecting key predictors like *walkDistance*, *kills*, and *boosts* while optimizing for high R-squared and low MSE. We also classified match types with 99.67% accuracy after consolidating similar categories that differed only in camera perspective.

Additionally, we assessed how elements of teamwork and gameplay strategies impact one's likelihood of winning. Attributes indicating teamwork, like revives and assists, have positive correlations with *winPlacePerc*, so working together is an integral part of success. Evaluating walking, riding, and swimming strategies showed walking to be optimal, having the best model fit. This suggests players are better off moving quietly on foot rather than risking exposure from vehicles and waterways to win more often. In summary, our models provide actionable insights for PUBG players to improve their probability of being placed first.

While our current analysis provides valuable insights, there are several areas for future exploration. First, we could incorporate player skill data and rankings over time rather than match-level statistics. This may reveal new trends related to player development. Additionally, studying gameplay videos could offer details on strategies not captured in stats alone. Computer vision techniques could quantify movements and tactics to a greater degree. Finally, as new gameplay modes, maps, and updates are released, retraining our models would provide the most up-to-date recommendations. Expanding the data over time and across various play modes would strengthen our understanding. Overall, there are many promising directions to build on this work and derive even more nuanced strategic guidance tailored for different player archetypes and situations.

References

[1] Playground Code Competition. *PUBG Finish Placement Prediction (Kernels Only)*. Can you predict the battle royale finish of PUBG Players? 2018. URL: `https://www.kaggle.com/competitions/pubg-finish-placement-prediction/data`.

[2] DIMITRIOS EFFROSYNIDIS. *EDA is Fun!* [Online; Nov 5]. 2018. URL: `https://www.kaggle.com/code/deffro/eda-is-fun#The-Swimmers`.

[3] Brij Rokad et al. *Survival of the Fittest in PlayerUnknown BattleGround.* 2019. arXiv: `1905.06052 [cs.LG]`.

[4] Diptakshi Sen et al. *Prediction of the final rank of Players in PUBG with the optimal number of features.* 2021. arXiv: `2107.09016 [cs.LG]`.

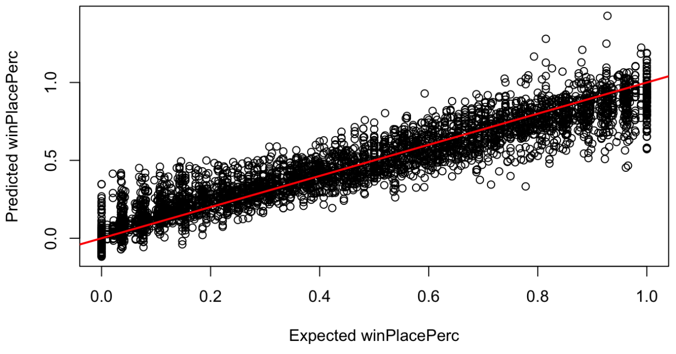

Figure 8: Predicted vs Expected values for Backward Elimination.

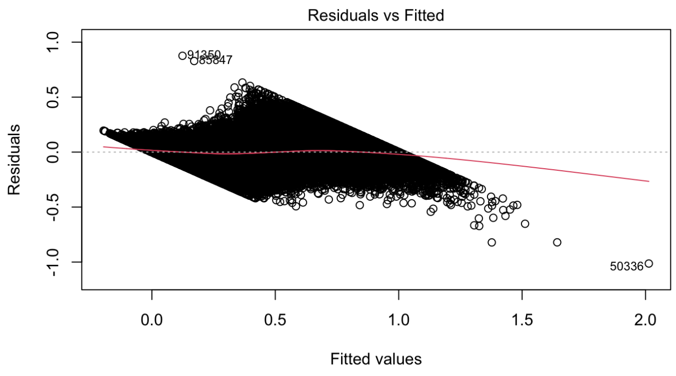

Figure 9: Residual vs Fitted values for Backward Elimination.

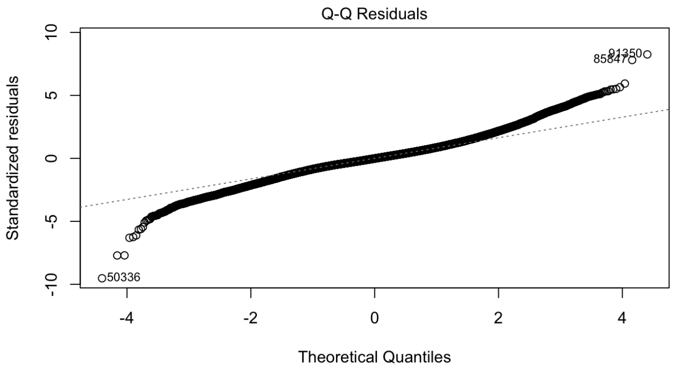

Figure 10: QQPlot for Backward Elimination.