IE7275 Fall 2018 Term Project
Sreya Pedamalla

PUBG FINISH PLACEMENT PREDICTIONS

1.INTRODUCTION

Player Unknown's Battlegrounds is a game in which 100 players are thrown into an open world island where their goal is to be the last person standing. Once all the players are ejected from the plane they must explore, scavenge, and eliminate other players until only one is left standing, all this while the play zone continues to shrink. When the playable area shrinks, players will have no choice but to engage. The pulse damage occurs when the player gets out of the safe area. As time passes, the area will soon reduce in size. As the area reduces, so will the playable area. Whether by electrical pulse or bomb drop, players can simply die due to their surroundings if they're not careful.

In the game, players can pick up different munitions, revive downed-but-not-out (knocked) teammates, drive vehicles, swim, run, shoot, and experience all the consequences -- such as falling too far or running themselves over and eliminating themselves.

2. PROBLEM STATEMENT

The rules of PUBG are straightforward. The game can be so unpredictable and even taking a single step or waiting a few seconds too long in the same place can have severe consequences. But with the data acquired one can perform analysis on it to gather meaningful insights on how every movement can affect each player's chances to win the game. Each player in the game has a rank from 1-100. Using the rank two categorical variables are created in such a way that if a player has a rank from 1-50 (class 0) it means that the player did not survive for at least half the game and if the player has a rank from 51-100 (class 1) it means that the player survived for at least half the game and this variable is called "half".

Class	Rank
0	1-50
1	51-100

Similarly, if a player has a rank from 1-99 (class 0) ie., the player lost the game and a player who has a rank of 100 (class 1, win class) ie., the player won the game and this variable is called “win”.

Class	Rank
0	1-99
1	100

The aim of this project is to gather such insights, to build models using machine learning algorithms and to train the models using all the available relevant information to get better predictions on the rank of each player, if a player will survive for at least half the game and who wins the game.
The reason behind having three different predictor variables for the same dataset is to use different subsets of features for different outcomes variables.

3. DATA DESCRIPTION

The Dataset used for this project has 1048575 samples and 31 variables which was provided with a large number of anonymized PUBG game stats, formatted so that each row contains one player's post-game stats. The variables used are as follows:

DBNOs - Number of enemy players knocked.

assists - Number of enemy players this player damaged that were killed by teammates.

boosts - Number of boost items used.

damageDealt - Total damage dealt. Note: Self-inflicted damage is subtracted.

headshotKills - Number of enemy players killed with headshots.

heals - Number of healing items used.

Id - Player’s Id

killPlace - Ranking in match of number of enemy players killed.

killPoints - Kills-based external ranking of player.

killStreaks - Max number of enemy players killed in a short amount of time.

kills - Number of enemy players killed.

longestKill - Longest distance between player and player killed at time of death.

matchDuration - Duration of match in seconds.

matchId - ID to identify match. There are no matches that are in both the training and testing set.

matchType - The standard modes are “solo”, “duo”, “squad”, “solo-fpp”, “duo-fpp”, and “squad-fpp”.

rankPoints - Elo-like ranking of player.

revives - Number of times this player revived teammates.

rideDistance - Total distance traveled in vehicles measured in meters.

roadKills - Number of kills while in a vehicle.

swimDistance - Total distance traveled by swimming measured in meters.

teamKills - Number of times this player killed a teammate.

vehicleDestroys - Number of vehicles destroyed.

walkDistance - Total distance traveled on foot measured in meters.
weaponsAcquired - Number of weapons picked up.
winPoints - Win-based external ranking of player
groupId - ID to identify a group within a match. If the same group of players plays in different matches, they will have a different groupId each time.
numGroups - Number of groups we have data for in the match.
maxPlace - Worst placement we have data for in the match.
winPlacePerc - The target of prediction.

Find a sample of a few rows and variables from the dataset.

	Id	groupId	matchId	assists	boosts	damageDealt	DBNOs	headshotKills	heals	killPlace	...	roadKills	swimDistance	t
0	7f96b2f878858a	4d4b580de459be	a10357fd1a4a91	0	0	0.00	0	0	0	60	...	0	0.00	
1	eef90569b9d03c	684d5656442f9e	aeb375fc57110c	0	0	91.47	0	0	0	57	...	0	11.04	
2	1eaf90ac73de72	6a4a42c3245a74	110163d8bb94ae	1	0	68.00	0	0	0	47	...	0	0.00	
3	4616d365dd2853	a930a9c79cd721	f1f1f4ef412d7e	0	0	32.90	0	0	0	75	...	0	0.00	
4	315c96c26c9aac	de04010b3458dd	6dc8ff871e21e6	0	0	100.00	0	0	0	45	...	0	0.00	

5 rows × 31 columns

4. DATA PREPROCESSING

4.1 Checking for missing values

The dataset did not have any missing values.

```
Id                0
groupId           0
matchId           0
assists           0
boosts            0
damageDealt       0
DBNOs             0
headshotKills     0
heals             0
killPlace         0
killPoints        0
kills             0
killStreaks       0
longestKill       0
matchDuration     0
matchType         0
maxPlace          0
numGroups         0
rankPoints        0
revives           0
rideDistance      0
roadKills         0
swimDistance      0
teamKills         0
vehicleDestroys   0
walkDistance      0
weaponsAcquired   0
winPoints         0
winPlacePerc      0
half game         0
win               0
dtype: int64
```

4.2 Label Encoding

The variable matchtype (solo, duo,squad,etc.) had string data and had to be converted into numerical data

4.3 Creation of Dummy variables

The encoded variable now had to be converted into multiple dummy variables. After this the dataset now had 41 variables.

5. DIMENSION REDUCTION

The purpose of transformation and dimensionality reduction was to improve the performance of the models by considering only the variables that are most important for predicting the outcome variables

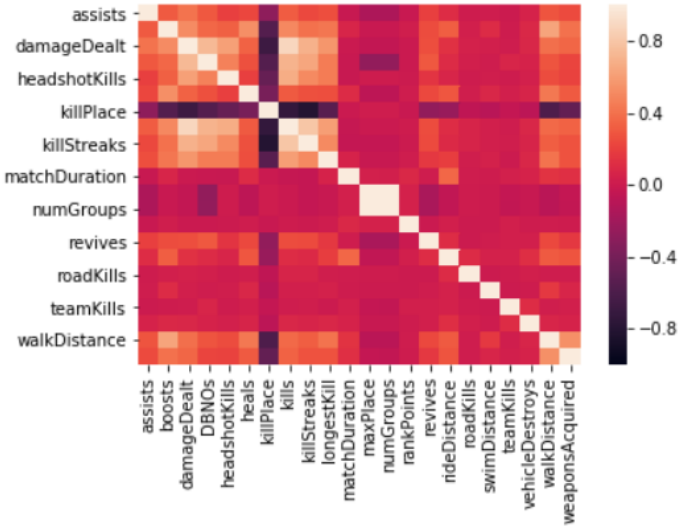
5.1 Domain Knowledge

Using domain knowledge, variables like groupID, MatchID, ID were removed because they clearly do not contribute to the prediction of the outcome variables.

5.2 Correlation Matrix

By plotting the correlation matrix, a correlation of 0.98 and 0.99 was found between killpoints and rankpoints, winpoints and rankpoints respectively. Hence, win points and kill points was removed.

The correlation matrix below is after removing the highly correlated variables and the highest correlation that exists now is -0.82.



5.3 Feature Selection

Feature selection was done to identify and eliminate unneeded, irrelevant and redundant data that does not contribute to the performance of the predictive model. It also helps reduce the complexity of the model. The feature selection method chosen for this project is backward elimination method which starts by considering all the features in the dataset in the model and then eliminate any variable that has a high p-value (>0.05) in an iterative process until there are no more variables to eliminate (all the considered variables are relevant and meet the constraint.)

5.4 PCA

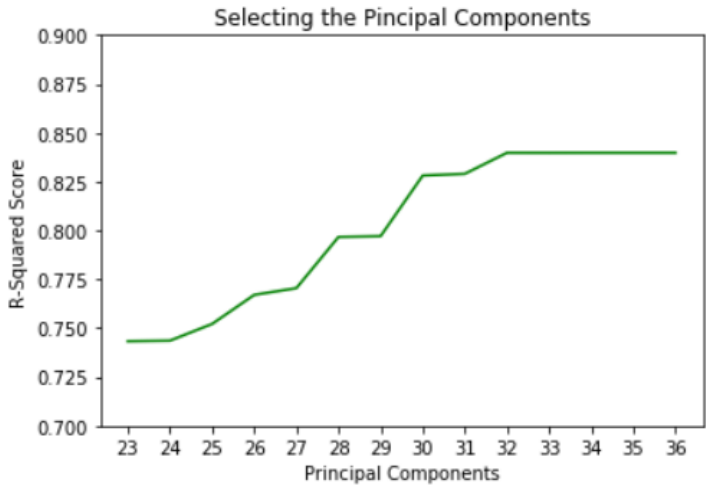
PCA algorithm was applied to further reduce the number of features (because a few variables had -0.82 correlation) to be considered which retain the maximum information of the dataset. This helped in bringing down the number of variables from 38 to 32. We chose the first 32 principal components because they cover over 99% variance in the dataset and including anymore components does not change the performance of the models used.

The Matrix given below shows the cumulative variance covered by the principal components

```
[0.16900786 0.26244598 0.31642458 0.35626667 0.39184333 0.42593298
 0.4589838  0.48999881 0.52059113 0.55101575 0.58018545 0.60899363
 0.63758471 0.66616311 0.69474029 0.72331259 0.75159529 0.77949364
 0.80661057 0.83192894 0.85614128 0.87908492 0.9010395  0.92157157
 0.93770824 0.95347067 0.96703018 0.9785161  0.98715242 0.99321517
 0.99683049 0.99917651 0.99989751 0.99995693 1.         1.        ]
```

Even though the first 24 principal components cover over 90% of the information in the dataset, it still wasn't enough for the models to perform well. So, when the performance of the models was checked by considering different number of principal components, we can see from the line plot below that the performance of the models have been increasing till we consider 32 principal components and after that it does not affect the performance.

The line graph given below shows the change in accuracy with different no of principal components.



6. EXPERIMENTAL RESULTS

The dataset was split into train and test sets for building the model and for validation purposes. 75% of the dataset was used to train the model and the remaining 25% was used to validate the models.

6.1 Data Analysis

Before applying machine learning algorithms to build models for predictions, an analysis was performed to check how changes in different variable will affect the predictions.
Therefore, a few aggregations and grouping functions were used to find the mean of multiple variables for different classes.
The tables below give a clear understanding on how different variables affect the predictions

Half Class	Walking Distance
0	354.663080
1	2110.504629

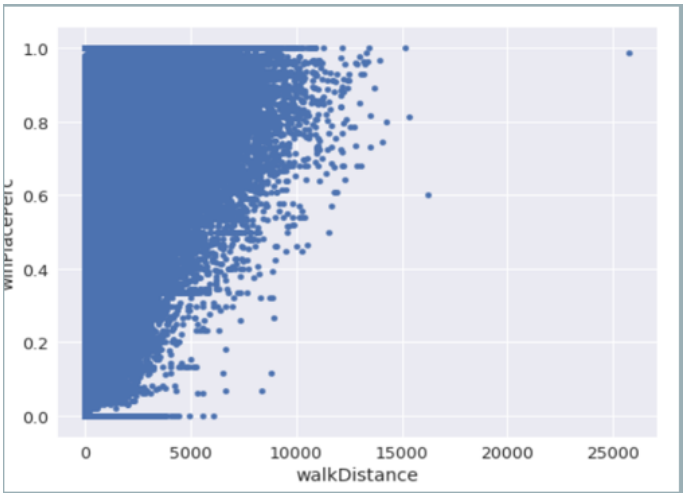
Win Class	Walking Distance
0	1101.739968
1	2966.276763

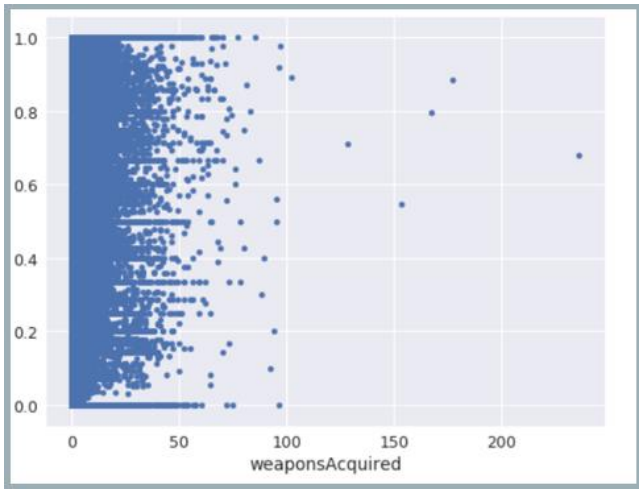
Half Class	Knock Outs	Damage Made to the Enemy	Headshots Kills	Kills
0	0.425453	0.103848	0.104416	0.443153
1	0.936909	0.387502	0.373247	1.501286

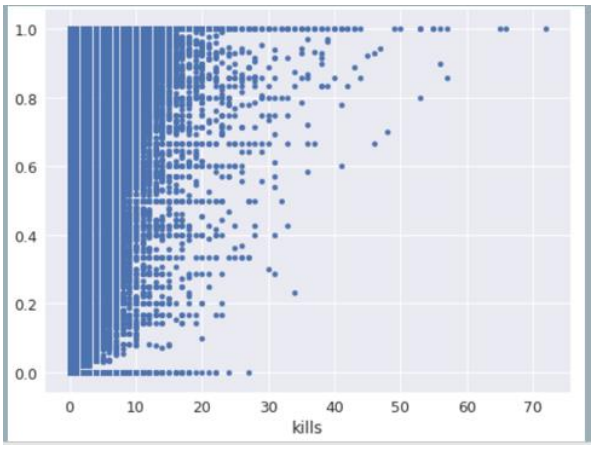
Win Class	Knock Outs	Damage Made to the Enemy	Headshot Kills	Kills
0	0.618183	0.207663	0.205623	0.846017
1	2.030928	1.098584	0.951541	3.624396

From the tables we can see that for the variable walking distance, (the mean distance) covered by players who did not survive for at least half the game (354.6 m) is much less than the distance covered by the players who survived for at least half the game (2110 m). Similarly, for win prediction class 0 (players who lost) cover over 1101 m and class 1 (winners) cover over 2966 m. this can be one of the important variables because if a player's walking distance is a larger number then it directly tells us that they player should have survived long enough (ie, probably he survived for at least half the game)
This observation can be seen for all the other variables considered too. With enough knowledge about the game the variables shown here are a few basic variables that effect the predictions the most.

These scatter plots that show how a few variables help predict the rank of a player





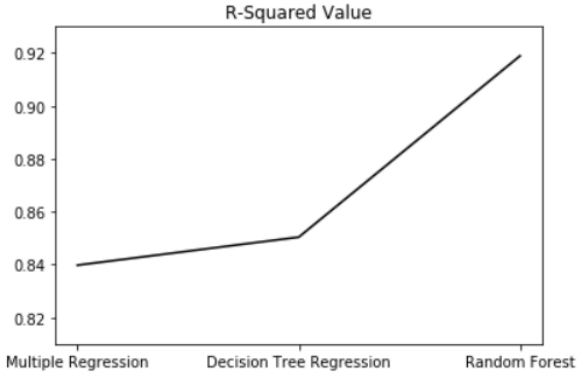


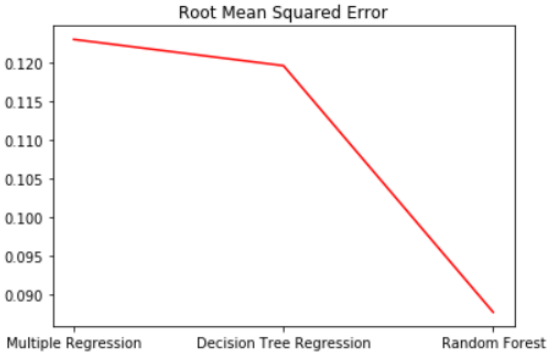
Supporting the tables shown above the scatter pots of the variables also show similar results. We can clearly see from the plots that as the rank (y-axis) increases the variables value also increases (the no of kills increase, the walking distance increases and the weapons acquired also increases.)

6.2 Prediction of the rank outcome

For predicting the Rank outcome variable for each player, 32 variables were considered and we used multiple machine learning algorithms like Multiple Regression, Decision Tree Regressor and Random Forest Regressor. After validating the model, R-squared score and mean squared error for each model was calculated.

A comparison of the calculated performance metric values can be seen below

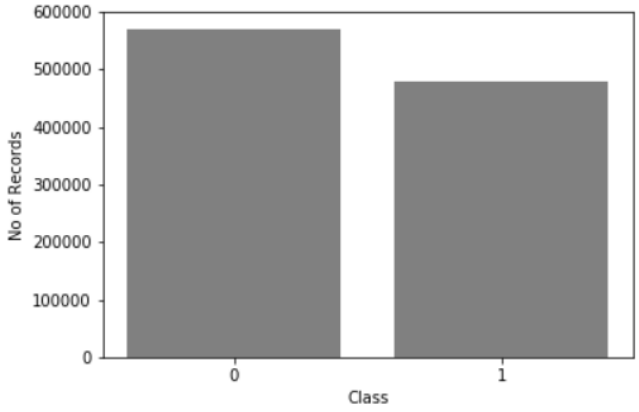




6.3 Predicting if a player will survive for at least half the game

For predicting if a player survives for at least half the game, 36 variables are selected after backward elimination. Machine learning algorithms like Decision tree classifier, Random forest classifier and Naïve Bayes are used.

Below is a bar graph that shows the sampling of class 0 and class 1 records for players who survived for at least half the game and who did not.



For Decision Tree Classifier the classification report is as follows:

Class	Precision	Sensitivity	F1-score	Support
0	0.95	0.91	0.93	142544
1	0.90	0.94	0.91	119600
	0.92	0.92	0.92	262144

For Random Forest Classifier the classification report is as follows:

Class	Precision	Sensitivity	F1-score	Support
0	0.93	0.95	0.94	142544
1	0.94	0.92	0.93	119600
	0.93	0.93	0.94	262144

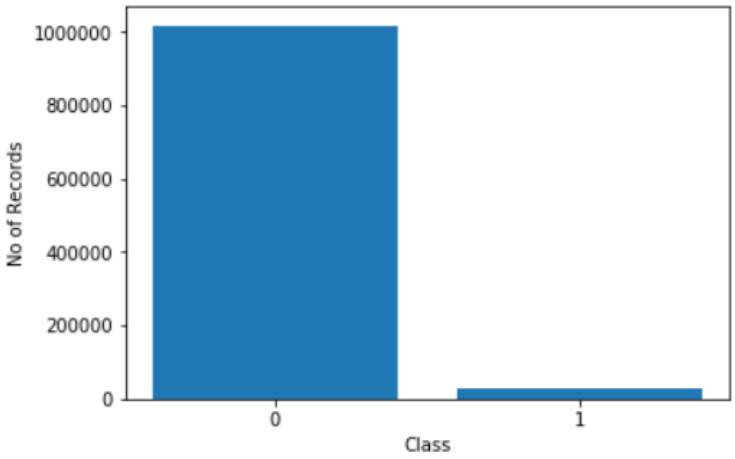
For Naïve Bayes model the classification report is as follows:

Class	Precision	Sensitivity	F1-score	Support
0	0.81	0.91	0.86	142544
1	0.87	0.74	0.80	119600
	0.84	0.83	0.83	262144

6.3 Predicting the winner of the game

For predicting if a player wins a game or not, 31 variables were used after backward elimination. Neural Network algorithms were used to better predict winners. As the game has a no strict rules (like a player could do certain things to win) neural networks are used to better train the model. To train the model rectifier activation function is used to create layers, and soft max function is used in the output layer.

A bar graph showing the number of samples in class 0 and class 1 for win prediction are as follows:



The classification report for Neural Networks is as follows:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	254661
1	0.47	0.35	0.40	7483
micro avg	0.97	0.97	0.97	262144
macro avg	0.73	0.67	0.69	262144
weighted avg	0.97	0.97	0.97	262144

7. CONCLUSION

Looking at the accuracy metrics, classification reports on validation performance, we can see that Random Forest Regressor performs the best for predicting the rank of a player. For predicting if a player survives for at least half the game, both random forest and decision tree classifier performs the best.
For win predictions, Neural networks performs the best.

REFERENCES

https://www.kaggle.com/c/pubg-finish-placement-prediction
https://scikit-learn.org/stable/
www.udemy.com/
https://matplotlib.org/users/index.html