Machine Learning Engineer Nanodegree

Capstone Proposal

Himanshu Singh

Jan 25th, 2019

Proposal on topic-

PUBG Finish Placement Prediction

Can you predict the battle royale finish of PUBG Players?

Domain Background

The gaming industry is estimated to be worth $ 100 billion globally and is the largest segment within the entertainment industry; standalone, it exceeds the movies or music segments. The Indian gaming industry is currently valued at over $300 million or approximately Rs. 2000 Crores. With a population of 1.3 billion people and two-thirds of them under the age of 35, India has the world's largest youth population. With this distinct advantage, India has the potential to become one of the world's leading markets for gaming.

Gaming is not only one of the fastest growing markets in India, but it is also becoming a career option of choice for a vast number of talented young students from diverse backgrounds. Game Art professionals are in great demand across the globe. The result is that there is huge employment potential for young aspirants within India as it develops as an outsourcing hub for international gaming companies. This is the one reason that motivated me to apply my skills of machine learning to build my career in the gaming industry by knowing how in reality user collected data is used in gaming industries.

PlayerUnknown's BattleGrounds (PUBG) has enjoyed massive popularity. With over 50 million copies sold, it's the fifth best selling game of all time and has millions of active monthly players. The team at PUBG has made official game data available for the public to explore and scavenge outside of "The Blue Circle." This competition is not an official or affiliated PUBG site - Kaggle collected data made possible through the PUBG Developer API.

The interest in finding best winning strategies is another reason that motivated me to work on this project as udacity capstone project.

By knowing winning strategies helps PUBG developers to make winning strategies more complicated, challenging and interesting to attract more and more people to play their game.

Academic Research Citation

Research on operational model of PUBG-

(https://www.matec-conferences.org/articles/matecconf/pdf/2018/32/matecconf_smima2018_03062.pdf) is really helpful in knowing the mathematics behind the converging circle and to know the best time to parachute and also to improve attack hit ratio.

Problem Statement

We are by no means experts on the game or military tactics in general for that matter, so, I decided to take an approach using machine learning that is a more familiar to me: trying to explore trends and patterns using provided data by PUBG developer API to know best strategies and decisions to win the game.

Let's talk on problem statement more- We have provided with a large number of 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. Our aim is to determine the player's finishing placement based on their final stats, on a scale from 1 (first place) to 0 (last place).

A winning player will get finishing placement closes to one and looser will get close to zero.

After that, the winning player, who got finishing placement closes to one, will be analysed by looking at the features that are important to him to win the game. By that only we will come to know the best strategy to win the game.

(https://www.kaggle.com/c/pubg-finish-placement-prediction)

Datasets and Inputs

The datasets for this project is composed of 2 CSV files. It has been obtained from a Kaggle Competition provided by PUBG Developer API.

Both training and testing dataset contains 3 columns which describe the origin of the player-

ID- This will uniquely identify an anonymized PUBG player. There is no common player ID in training and test data.

MatchId- This will identify a match in which the player is playing. There is no common MatchID in training and test data.

GroupId- This will identify a group within a match. If the same group of players plays in different matches, they will have a different groupId each time.

The training dataset is in CSV format containing 4446966 rows and 29 columns with target label as winPlacePerc column.

The testing dataset is in CSV format containing 1934174 rows and 28 columns.

Detail about the Data fields which will be there in Training and testing dataset-

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.

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

headshotKills - Number of enemy players killed with headshots.

heals - Number of healing items used.

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

killPoints - Kills-based external ranking of the player. (Think of this as an Elo ranking where only kills matter.) If there is a value other than -1 in rankPoints, then any 0 in killPoints should be treated as a “None”.

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 the time of death. This may be misleading, as downing a player and driving away may lead to a large longestKill stat.

matchDuration - Duration of a match in seconds.

matchType - String identifying the game mode that the data comes from. The standard modes are “solo”, “duo”, “squad”, “solo-fpp”, “duo-fpp”, and “squad-fpp”; other modes are from events or custom matches.

rankPoints - Elo-like ranking of the player. This ranking is inconsistent and is being deprecated in the API’s next version, so use with caution. Value of -1 takes place of “None”.

revives - Number of times this player revived teammates.

rideDistance - Total distance travelled in vehicles measured in meters.

roadKills - Number of kills while in a vehicle.

swimDistance - Total distance travelled by swimming measured in meters.

teamKills - Number of times this player killed a teammate.

vehicleDestroys - Number of vehicles destroyed.

walkDistance - Total distance travelled on foot measured in meters.

weaponsAcquired - Number of weapons picked up.

winPoints - Win-based external ranking of the player. (Think of this as an Elo ranking where only winning matters.) If there is a value other than -1 in rankPoints, then any 0 in winPoints should be treated as a “None”.

numGroups - Number of groups we have data for in the match.

maxPlace - Worst placement we have data for in the match. This may not match with numGroups, as sometimes the data skips over placements.

winPlacePerc - The target of prediction. This is a percentile winning placement, where 1 corresponds to 1st place, and 0 corresponds to the last place in the match. It is calculated off of maxPlace, not numGroups, so it is possible to have missing chunks in a match.

Solution Statement

The solution will largely utilize the fact that the correlation between different features(data fields) to make the correct decision to decide the winning strategy.

I am going to add more and more features to this dataset based on existing data field to get more accurate results.

I will put more efforts in detecting and removing anomalies or outliers such as carrying more than limited weapons as I can see playerid=3f2bcf53b108c4 acquired 236 weapons in one game, killing without moving or kill from more than 1km away etc.

Then using supervised learning models to predicts players' finishing placement based on their final stats, on a scale from 1 (first place) to 0 (last place).

I will use the training dataset to train the model with cross-validation WinPlacePerc as the target.

I will then test various models such as SVM, Decision Trees, Random Forest etc. we have learned in this course along with techniques such Grid-SearchCV to optimize and other models such as XGBoost which are used effectively in competitive environments such Kaggle.

Benchmark Model

Due to a large dataset, a CART(classification and regression Tree) model like Random forest with Default parameters will be used for the benchmark model.

Then I will use parameter tuning on Random forest and Extreme gradient boosting to compare it with the benchmark model.

Evaluation Metrics

Since this is a Kaggle Challenge, we already have an evaluation metric, i.e. "Mean Absolute Error" This will be between your predicted winPlacePerc and the observed winPlacePerc.

Mean Absolute Error (MAE): MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It’s the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|$$

If the absolute value is not taken (the signs of the errors are not removed), the average error becomes the Mean Bias Error (MBE) and is usually intended to measure average model bias. MBE can convey useful information but should be interpreted cautiously because positive and negative errors will cancel out.

Project Design

This workflow is similar to what we follow in udacity's projects.

- ***Preparation***
- ***Exploration the datasets***
- ***Adding new features***
- ***Detecting and removing anomalies or outliers-***
- ***Categorical Variables***
- ***Shuffle and Split Data***
- ***Evaluating Model Performance***
- ***Analyzing the result***

Preparation-

Import the dependencies needed for handling data, visualization and training our model.

Important dependencies are:

Pandas for their data frame structures and easy visualization.

Matplotlib for visualization.

Scikit-learn for machine learning.

Then read the data from training and testing datasets.

Exploration the datasets-

To understand dataset I will explore the dataset from head and tail, by Summary Statistics of the training data and also by knowing Data types, memory usage, shape, etc.

Adding new features-

This step will help to add new features based on existing features. I got this idea after reading the problem 3 of Research on operational model of PUBG by Yong ding(link in academic research citation)

Detecting and removing anomalies or outliers-

such as carrying more than limited weapons as I can see playerid=3f2bcf53b108c4 acquired 236 weapons in one game, killing without moving or kill from more than 1km away etc.

Categorical Variables

I will one hot encode the 'matchType' feature to use it in our supervised machine learning algorithm.

Shuffle and Split Data

As I already have data split into training and testing data. I only do shuffling of data here.

Evaluating Model Performance

I will investigate different algorithms, and determine which is best at modeling the data. Two of these algorithms will be supervised learners like hyper parameter tuning of Random forest and XGboost, then compare it with benchmark model.

Analyzing the result

After predicting player's finishing placement, will analyses using feature importance to know the important features to build the strategy to win the game.