Machine Learning Engineer Nanodegree Capstone Proposal: PUBG[1] Finish Placement Prediction

Paco Hobi

January, 2019

Domain Background

The *PUBG Finish Placement Prediction* is a Kaggle competition about the highly successful battle royale game PUBG, which has also become a big esports[2] game. As the esports industry continues to grow its unsurprising that machine learning will be used to train and increase the performance of esports players.

A high profile machine learning project in the game industry is OpenAI Five, which is creating Dota 2[3] bots that already have consistently beaten the top human players. The machine learning bot that the OpenAI team has created has not only learnt advanced game mechanics and strategies, but it also has come up with new strategies that were unknown to the most experienced human players. Projects like this one and others can be used by esports players to train and to discover new strategies.

Problem Statement

In a PUBG game players compete against each other and get ranked at the end of the game based on how many other players are still alive when they are eliminated. In game, players can pick up different munitions, drive vehicles, swim, run, shoot, etc. For this problem we want to create an agent that predicts player ranks based on their final game stats, on a scale of 1 (first place) to 0 (last place).

Datasets and Inputs

For this competition Kaggle provides a dataset including over 65,000 games' worth of player anonymized data. This dataset was built from official publicly available data provided by the PUBG developers through the PUBG Developer API.

The dataset has over four million data points with 29 variables each, one of them being `winPlacePerc`, which is what we are going to predict. The remaining variables contain game statistics like distance traveled, health items used, kills, longest kill streak, etc. This are all important stats of the games which have been precisely captured by the game server, and that should relate somewhat closely with the final ranking of the players in the games.

[1] PlayerUnknown's Battlegrounds is an online multiplayer battle royale game developed and published by PUBG Corporation, a subsidiary of South Korean video game company Bluehole.

[2] Esports is a form of competition using video games. Most commonly, esports takes the form of organized, multiplayer video game competitions.

[3] Dota 2 is a multiplayer online battle arena video game developed and published by Valve Corporation.

Solution Statement

To solve the problem we will build a regressor model that will be able to predict the `winPlacePerc` from the match statistics. To train the model and measure the performance of the predictions we will compare its predictions with the actual `winPlacePerc` of part of the dataset.

Benchmark Model

As a benchmark model I would like to use a AdaBoost regressor with the default scikit parameters and using the reduced dataset without preprocessing. AdaBoost should give us a good benchmark to which to compare our own model.

Evaluation Metrics

We will use the Mean Absolute Error (MAE) as the evaluation metric. MAE is a linear score function and therefore it penalizes huge error less than, for example, the Mean Squared Error. Therefore it is less sensible to outliers. [4]

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|$$

Project Design

The dataset has 4,446,966 data points with 29 variables each. The dataset includes data from different game types, but we will be only working with the data of the game type `solo-fpp` (no teammates and first person), because it is one of my favorite modes and also one of the modes use in competitive mode.

We will remove variables that are no longer needed when only checking solo games: `matchType`, `teamKills`, `revives`, `assists`, `numGroups` and `DBNOs`. We will also remove other variables that are not necessary, like `id`, `matchId`, `groupId`; and also the ranking variables as we know that these are being deprecated from the API because of their unreliability: `killPoints`, `rankPoints`.

By only considering the `solo-fpp` games and removing the unnecessary variables we are left with 536,761 data points with 17 variables each.

Many of the variables are skewed (see Figure 1), so we will experiment with feature scaling using Box-Cox test and feature normalization. We will also check if we benefit from removing outliers.

Once we have the dataset ready we will select a regressor type and do hyperparameter tuning using an exhaustive grid search.

Finally I have some ideas for engineered features, and I will check if we obtain better results by adding this new features.

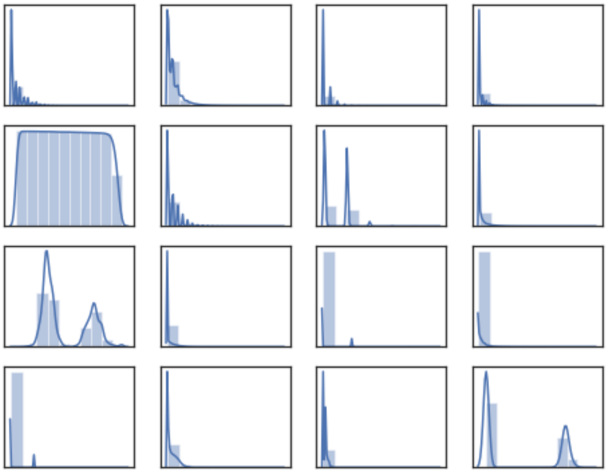

Figure 1. Original distribution of the 16 features in our reduced dataset.

[4] "How to select the Right Evaluation Metric for Machine Learning Models: Part 1 Regression Metrics"