Predicting the Winner in CS:GO Matches

(STATS/CSE 780 course project)

Yan Min (Student number: 400546093)

2023-12-11

Abstract

This study explores the application of advanced machine learning techniques, specifically Random Forest and Deep Neural Network(DNN), in predicting the winners of CS:GO matches. We aim to use the past data of matches to forecast future match results. Leveraging the dataset provided by Skybox as part of their CS:GO AI Challenge, we preprocessed the data and then fed the data into the models. The performance of these two models are evaluated by accuracy, precision, and recall metrics. Our results yielded relatively high accuracies for both models, up to about 75%. The two machine learning models we developed improve the prediction and analysis of future match results, which is beneficial for the development of eSports.

Introduction

The dataset("CS:GO Round Winner Classification" 2020) was originally published by Skybox as part of their CS:GO AI Challenge, running from Spring to Fall 2020. The dataset consists of around 700 demos from high level tournament play in 2019 and 2020. Warmup rounds and restarts have been filtered, and for the remaining live rounds a round snapshot have been recorded every 20 seconds until the round is decided. We want to use this dataset to predict the final winner, which can be seen as a classification problem.

There are totally 122410 entries and 97 colums in this dataset. The features have one boolean type, 94 float types and 2 object types. The number of entries is greater than the number of parameters. There are no null values in this dataset. We choose `round_winner` as the output value, which is consisted of the winners `CT` and `T`. There are other different types of features, such as `time_left`, `ct_score` and `ct_health`, which will be used as the input features. Figure 1 shows the distrubutions of selected features after feature slection.

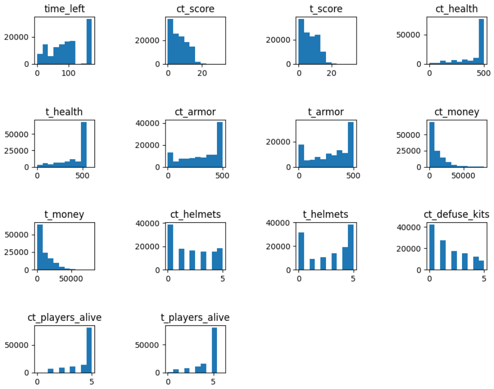

Figure 1: The distribution of selected variables after feature selection

Methods

Firstly, we will do Exploratory Data Analysis(EDA) to find out the relationships among different features. Then, we will drop unnessaray columns and transform the values of some features based on the analysis. Afterwards, we will do scaling.

The reasons for choosing random forest are that it can capture the complex interactions and non-linear relationships among the features, and that it is suitable for classification problems(`CT` or `T`). For the model of random forest, beforing fitting the data, we do the principal component analysis(PCA) for the features. Regarding to the tuning of the parameters, we do grid searching to find the best parameters for the model and the best accuracy score. Then we use the training dataset to check whether the model is overfitting and the test dataset to get the accuracy score.

The reasons for choosing deep neural network(DNN) are that it suits for large datasets, which in this case consists of `122410` entries, and that we can benefit from the trained model for future use. For the model of DNN, we set the number of layers as `4`, the number of nodes of each layer as `300`, and the number of epochs as `50`. Between each layer, we also add batch normalization layers, in order to normalize the activations. We choose the binary cross entropy as the loss function and the `Nadam` as the optimizer. Then we build the DNN model and render the accuracy scores for the training set and test set.

Results

Fistly, we do feature selection. As there are 97 variables in this dataset, we need to decrease the number of features in order to reduce the training time. As we can see, over eighty features are about how much of each weapon is left in the game and we can know that these features are high correlated about `ct_money` and `t_money`. So we drop them off and select 15 of them. Then we draw the correlation matrix of the selected features. From figure 2, we can tell that `ct_players_alive` is high correlated with `ct_health`, and `t_players_alive` is highly correlated with `t_money`. So we drop off the colomns of `ct_players_alive` and `t_players_alive`. Then we can use the left 13 features as inputs and `round_winner` as outputs. To do the prediction, in the colomn of `round_winner`, we convert `CT` to `0`, and `T` to `1`.

For the column of `bomb_planted`, at first, the values are boolean types and we convert `true` to `1` and `false` to `0`. For others features, we do data scaling. For the column of `round_winner`, the number of `CT` and the number of `T` are close to `50%` of the total entries. So we can directly split the dataset. After this, we divide the dataset into training set and test set, according to the test size of `0.3`.

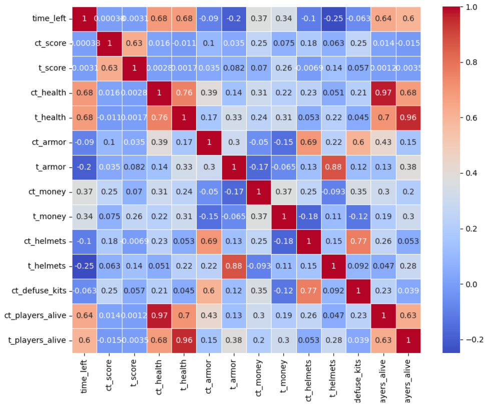

Figure 2: The correlation matrix of selected features

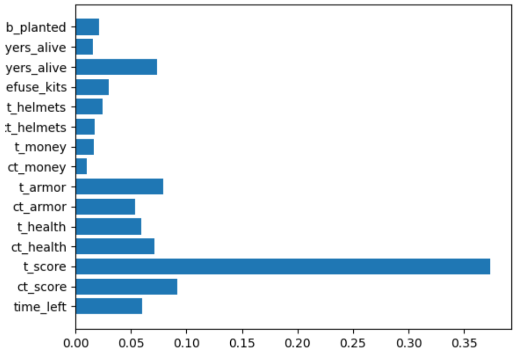

Figure 3: The Feature Importance Derived from the Random Forest Model

From the model of random forest, we can obtain the feature importance, as figure ref{figure3} shows. We can see that `t_score` is the most important among all features, following by `ct_score`. We can see that the scores of both team make the impact on final prediction.

Conclusion

Supplementary material

References

"CS:GO Round Winner Classification." 2020. https://www.kaggle.com/datasets/christianlillelund/csgo-round-winner-classification.