# **Final Project Report**

ZHANHAO HUANG

HUAYUE GUO

**The most important factor that determines the final result of PlayerUnknown's Battlegrounds**

## 1. Introduction

Electronic Sports (E-sports), which has officially become a formal athletic competition, is a competition that uses high-tech hardware and software equipment as a sports device to carry out intellectual confrontation (mostly through a game) between players. As E-sports’ influence on both economy and society has continues been growing in the recently year, it will gain more and more attraction, therefore, it is significant to analyze this new area. In this project, we focus on the game PlayerUnknown's Battlegrounds (PUBG), which is one of the most popular online game and also a E-sport competition in the world. We identified the most important factor that determines the final result of PUBG by viewing this problem as a data mining problem and solving it through machine learning algorithms. By the help of the results, both normal players and professional players can improve their game level and can have a better final result more easily and scientifically.

## 2. Objectives

The pursues of this projects are trying to find out the answers of the following questions:

1. Under the same level of personal ability, what is the most important factor that determine the final result in PUBG?
2. If we can predict a player’s actual level in PUBG based on the player’s basic data in the game?
3. If we can predict a player’s final Win Place Percentage shown exactly by a number?

## 3. Methodology

In order to achieve the pursues, we did the following steps in this project. Firstly, we imported and explored the data. Secondly, we did the data pre-processing to clean the data. Thirdly, we analyzed the dataset through histograms, scatter plots, and statistical methods, which are helpful to identify how each independent variables are related to the dependent variables and figure out the most important factor that determines the result of the game through the correlation coefficient matrix. Next, we used the K-Nearest-Neighbor Classification Machine Learning algorithm to create a classifier in order to predict the actual level of a player based on the player's basic data in the game. In addition, we created a Linear Regression Model to predict a player’s final Win Place Percentage shown exactly in a number based on 3 key independent variables. Lastly, we evaluate those model and predict the result by using the testing data.

## 4. Data Source

The dataset in this report was collected from Kaggle competition[1], with the shape of 4446966 observations and 29 features. All the detail information about the features are in the Appendix.

## 5. Data Preprocessing

1. Since there are too many matchType included in the dataset as different strategies need to be used in different match type, we only focus on the solo match type because it is a match type that only rely on the player individual's ability. In addition, we eliminated the ‘matchType’, ‘ DBNO’ and ‘revives’ features as they are meaningless in solo match type.
2. We determined the feature ‘winPlacePerc’ as dependent variable and all other features as independent variables.
3. We check for whether there exist missing values in our DataFrame and refill them if any missing value was exist.
4. We analyzed the outliers by using boxplots and defined some abnormal data as ‘cheater’ (‘kills’>30; ‘heals’>30, ‘weaposnAcquired’>40) and eliminated those data.
5. After data pre-processing, there are 181868 observations and 26 features in the dataset.

## 6. Features Visualization

In the features visualization part, we chose 6 features that we think might be significant to explore, which are walkDistance, rideDistance, kills, damageDealt, heals and weaponsAcquired. (All figures are in the appendix)

#### *6.1 WalkDistance (Fig. 1-2)*

* 99% of the players have a walking distance less than 4152 meters
* Most players' walking distance are between 0 and 3000 meters
* The average walking distance is 985 meters
* The range of player’s walk distance is 0-15370 meters
* WalkDistance seems to have strong, positive and linear relation with winPlacePerc

#### *6.2 RideDistance (Fig. 3-4)*

* 99% of the players have a riding distance less than 7495 meters
* Most players' riding distance are between 0 and 3000 meters.
* The average ride distance is 640 meters
* The range of player’s ride distance is 0-33970 meters
* RideDistance seems to have relatively strong, positive and linear relation with winPlacePerc

#### *6.3 Kills (Fig. 5-6)*

* 99% of the players kill less than 8
* More than 100,000 players have zero kills
* Most players' kills are between 0 and 3
* The average kill is 0.87
* The range of player’s kills is 0-21
* Kills seems to have weak relation with winPlacePerc

#### *6.4 Damage Dealt (Fig. 7-8)*

* 99% of the players' damageDealt are less than 734
* Most players' damageDealt are between 0 and 450
* The average damageDealt is 112
* The range of player’s Damage Dealt is 0-2490
* DamageDealt seems to relatively strong, positive and linear relation with winPlacePerc

#### *6.5 Heals (Fig 9-10)*

* 99% of the players' heals are less than 11
* Most players' heals are between 0 and 5
* The average heals is 1
* The range of player’s heals is 0-30
* Heals seems to have weak relation with winPlacePerc

#### *3.6 WeaponsAcquired (Fig 11-12)*

* 99% of the players' weaponsAcquired are less than 11
* Most players' weaponsAcquired are between 0 and 10
* The average weaponsAcquired is 3.7
* The range of player’s weaponsAcquired is 0-35
* weaponsAcquired seems to have a relatively weak relation with winPlacePerc

## 7. Features Engineering

The main purpose for features engineering is to combined the related features and reduce the irrelevant features. Firstly, we combined the related independent features by adding new feature ‘boosts\_heals ’ which is equal to 'boosts' plus 'heals' and 'totalDistance' which is equal to the sum of 'walkDistance', 'rideDistance' and 'swimDistance'.

Next, we plotted the correlation matrix to explore the relationship between every variables in a heat map (Fig 13). From the correlation matrix, we chose 8 features that have the highest absolute correlation value with the dependent variable 'winPlacePerc' which are walkDistance(0.79), killPlace(-0.67), totalDistance(0.66), boosts(0.62)，weapons acquired(0.59), boosts\_heals(0.57)，kills(0.50) and damageDealt(0.48). The results of this correlation matrix help to answer the question that walk distance is the most important factor that determine the final result of PUBG. However, it is not simply meaning that the more you walk, the better result you get. Actually, it is walk distance together with other significant factors that determine the final result of PUBG.

Lastly, in order to improve the accuracy of our machine learning model, we only remained the independent variables that have a relatively high correlation value with the dependent variable. In this project, we keep the top 8 features that have the highest absolute correlation value with the dependent variable as most of them are higher than 0.5. Therefore, after features engineering, the dataset finally have 181868 observations and 10 features. One thing need to be noticed is that, the feature ‘Id’ is for the purpose of being an unique key and the feature ‘winPlacePerc’ is the independent variable, which means there are 8 independent variables remained. Based on this, we plotted another correlation matrix heatmap that only consider those 8 independent variables (Fig 14).

## 8. Machine Learning

#### *8.1 Labels Assign*

Since our second purpose is to predict the player’s actual level in PUBG. We defined 4 levels to describe a player’s level rely only on the ‘winPlacePerc’, which are ‘Rookies’, ‘Generals’, ‘Elites’ and ‘Experts’. Below are the introduction of them:

* Class 'Rookies' stand for players with 0 <= winPlacePerc > 0.4
* Class 'Generals' stand for players with 0.4 <= winPlacePerc > 0.6
* Class 'Elites' stand for players with 0.6 <= winPlacePerc > 0.8
* Class 'Experts' stand for players with 0.8 <= winPlacePerc >= 1

After labels assigning, we replaced the continuous numerical data into 4 categorical data in the ‘winPlacePerc’ column and found that there are 79096 players (43%) in class Rookies, 33692 players (19%) in class Generals, 32964 players in class Elites (18%) and 36116 players (20%) in class Experts in the dataset.

#### *8.2 Train\_Test\_Split*

For the evaluation process after machine learning, we splitted the dataset into training data (75% of the dataset) and testing data (25% of the dataset).

#### *8.3 Classification: K-Nearest-neighbor classifier and Evaluation*

In order to achieve the second purpose, we used K-Nearest-neighbor machine learning model (with n\_neighbors equal to 17) to create a classifier based on the training data, which can predict the class that a player belongs to by the player’s basic data (independent variables). The confusion matrix (Fig 15) shows that there are 17818 players in the Experts class is truly predicted (TP), 4727 players in the Elites class is truly predicted, 6163 players in the Generals class is truly predicted and 4211 players in the Rookies class is truly predicted. The classifier performs best for class ‘Experts’.

The classification report (Fig 16) shows that the accuracy of this classifier is 0.724 which means it performs good in classification since it has 4 classes. In addition, it works best for Experts class as it has the highest precision, recall, and f1-score compared to other class. The weighted average precision is 0.73 and the weighted average recall is 0.72.

Additionally, we use Cohen's kappa to evaluate the classifier since it has class labels more than 2 and the Cohen's kappa for this classifier is 0.76, which is good as it is higher than 0.7.

#### *8.5 Linear Regression and Evaluation*

In order to achieve the third purpose, we used Linear Regression Model to summarize a formula, which can predict a player’s final ‘winPlacePerc’ shown by a number. With the results that the coefficients are 0.0229, -0.0033 and 0.0002 and the interception is 0.4436. We summarized the formula with the score of 0.72, Mean squared error 0.03 and Variance score of 0.73 where X1 is boosts, X2 is killPlace, X3 is walkDistance and Y is winPlacePerc:

Y = 0.0229 \* X1 - 0.0033 \* X2 + 0.0002 \* X3 + 0.4436

Based on this formula, it is clear that even though walk distance is the most important factor that determines the final result of PUBG, it does not mean walk distance is the only factor that need to be focused on. The final result of PUBG should be determined based on the combined effect of those independent variables that have the relatively high absolute correlation value with the dependent variable (‘winPlacePerc’).

## 9. Conclusions

After data mining, we solved the questions that we have mentioned previously. Firstly, under the same level of personal ability, ‘walkDistance’ is the most important factor that affects the final result of PUBG and it is the combined effect of ‘walkDistance’, ‘killPlace’, ‘totalDistance’, ‘boosts’, ‘weaponsAcquired’ and so on that truly determines the final result of PUBG. Secondly, a KNN-Classifier is created to predict a player’s actual level (‘Rookies’, ‘Generals’, ‘Elites’ and ‘Experts’) in PUBG based on the player’s basic data in the game. Thirdly, a Linear Regression formula is summarized to predict a player’s final Win Place Percentage shown exactly by a number.

## Appendix: Dataset Features’ detail

\* 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. (Think of this as an Elo ranking where only kills matter.) If there is a value other than -1 in ran\* 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. (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 time of death. This may be misleading, as \* downing a player and driving away may lead to a large longestKill stat.

\* 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 - 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 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 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. (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”.

\* 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.

## Appendix: All Figures

Box Plots

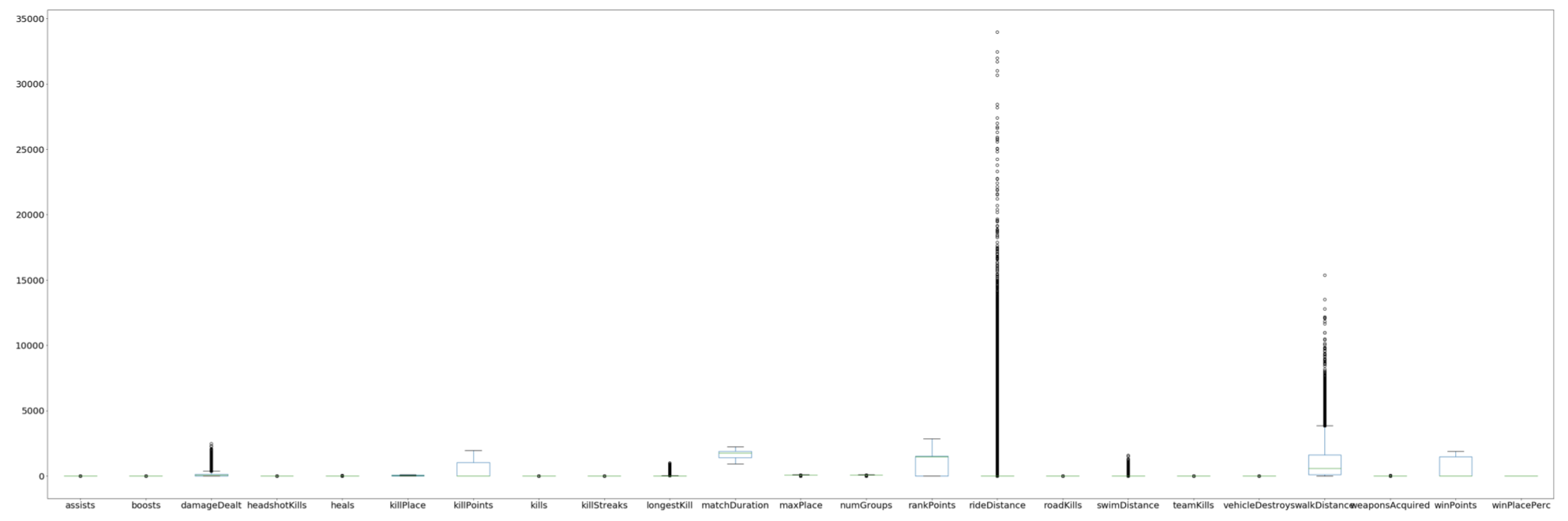


Fig 1 Fig 2

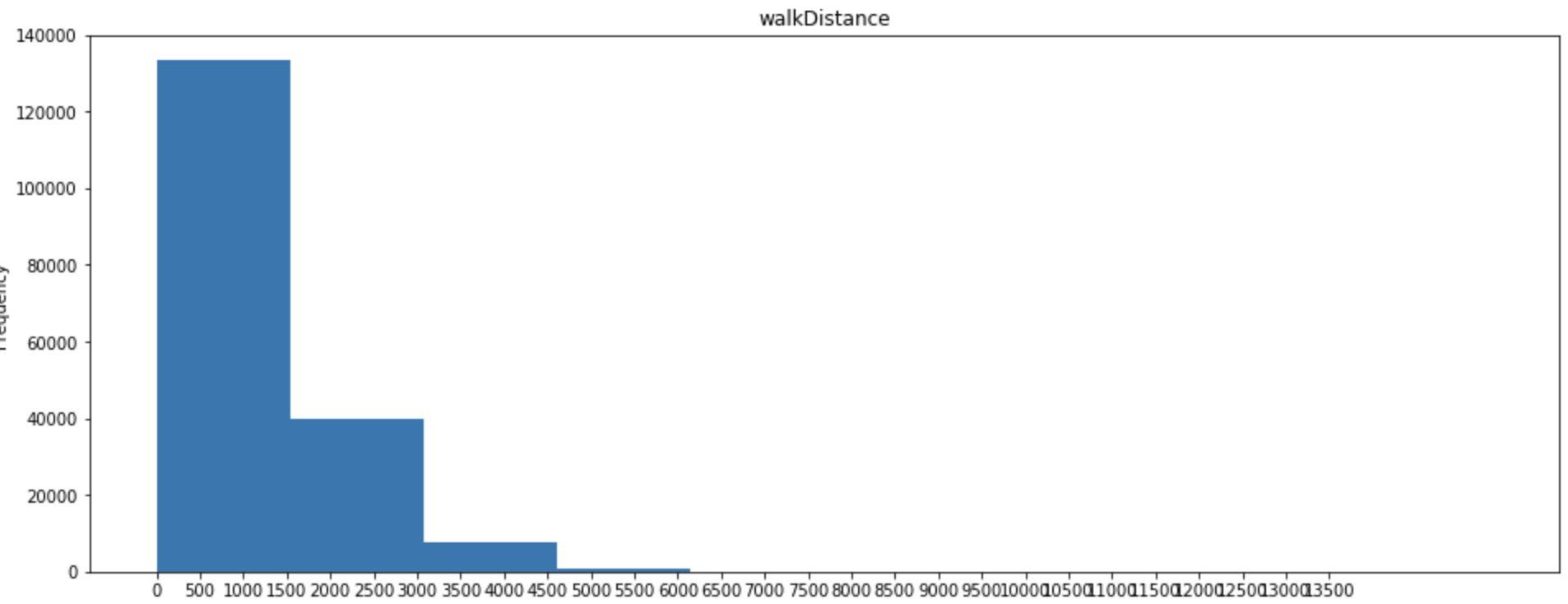
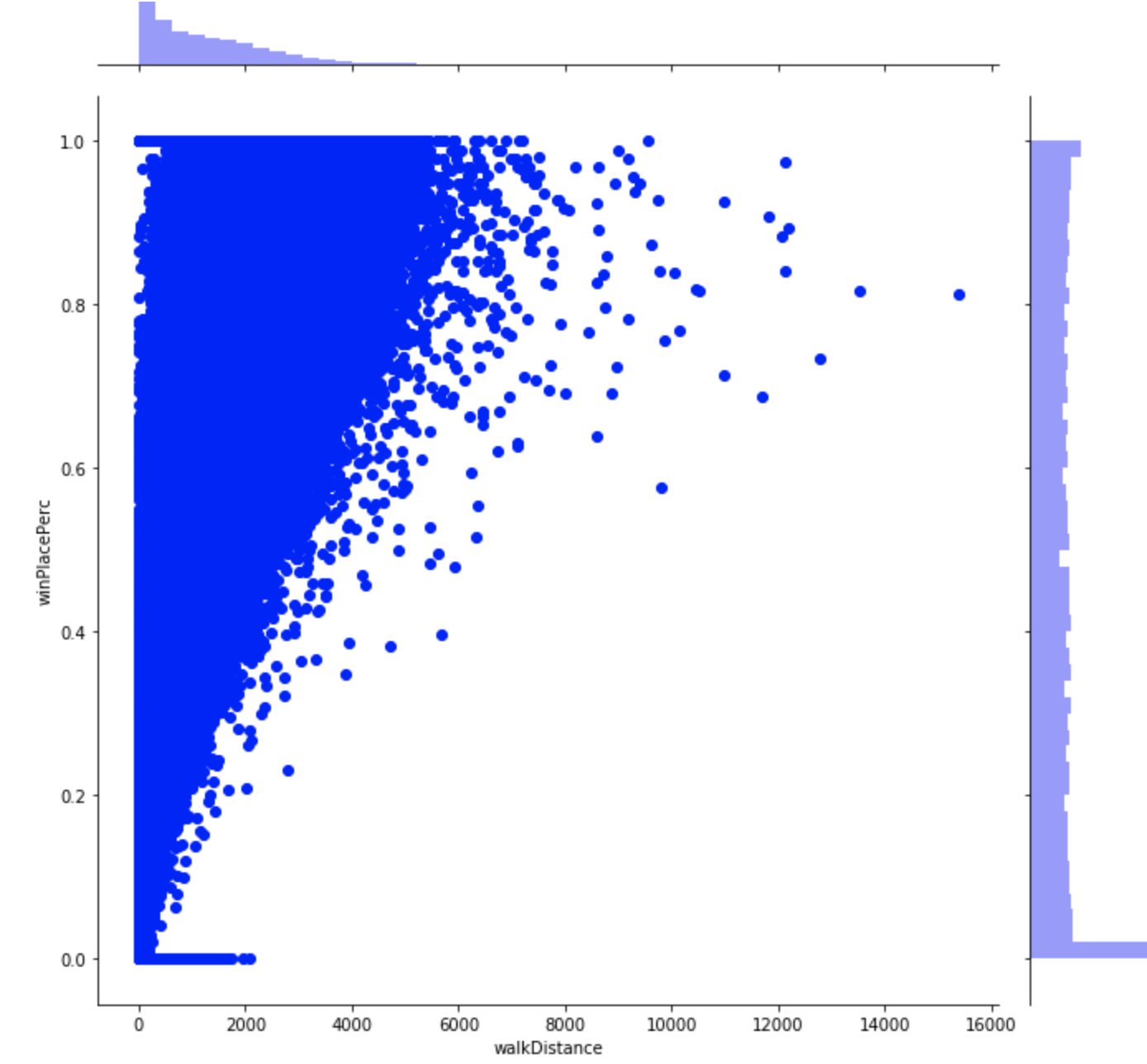


Fig 3 Fig 4

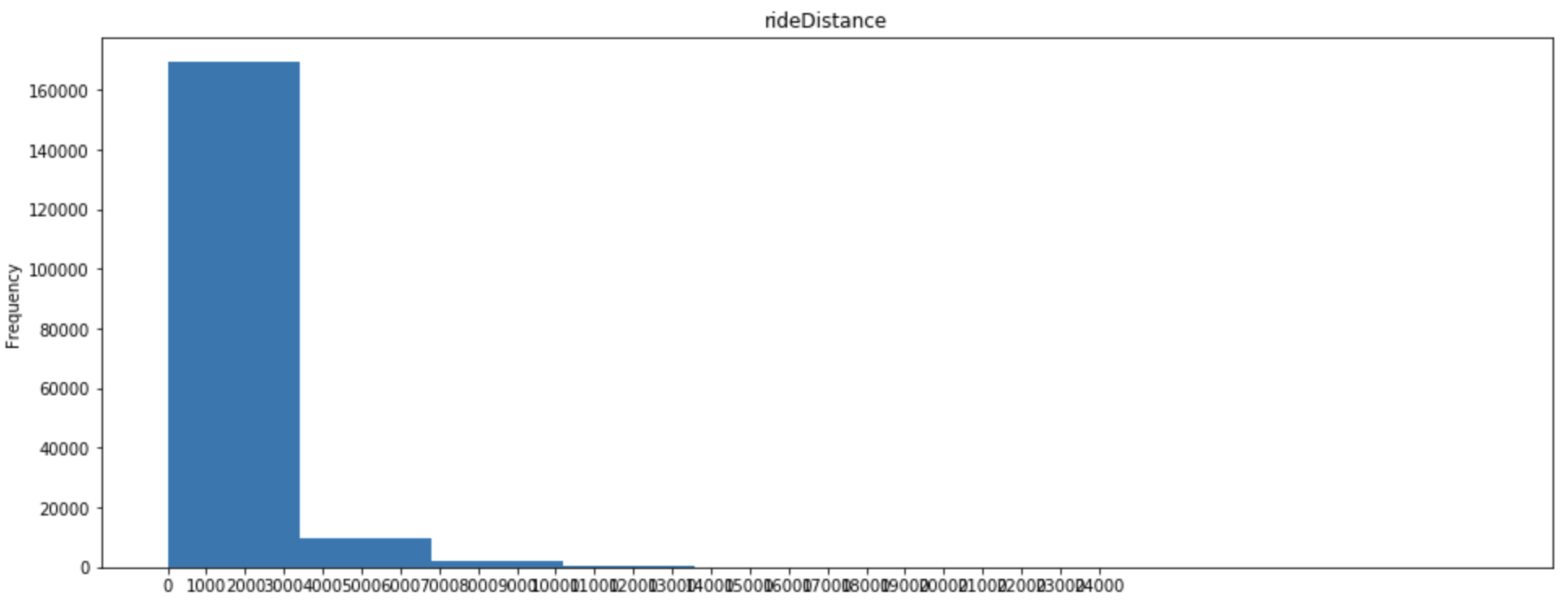
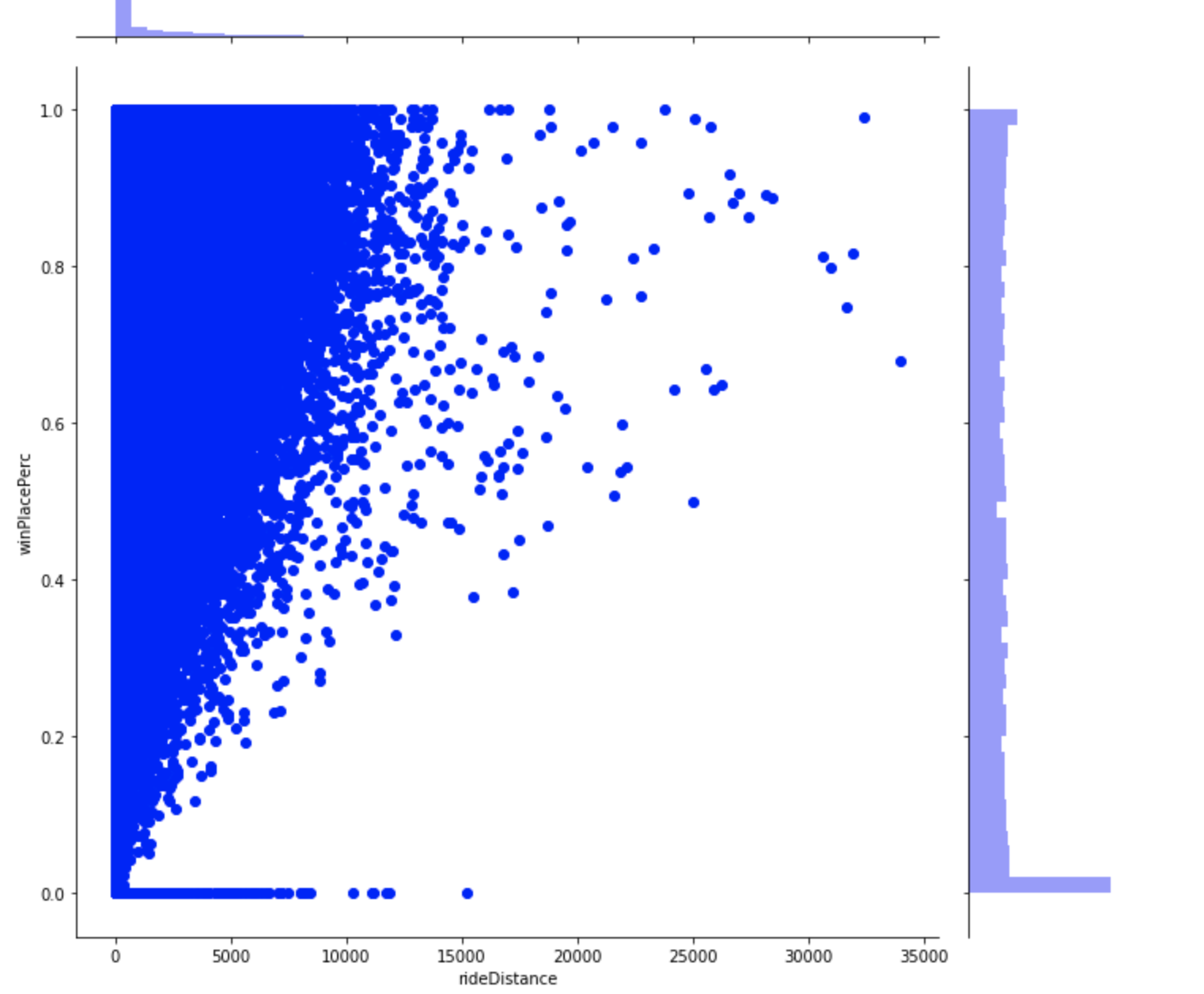


Fig 5 Fig 6

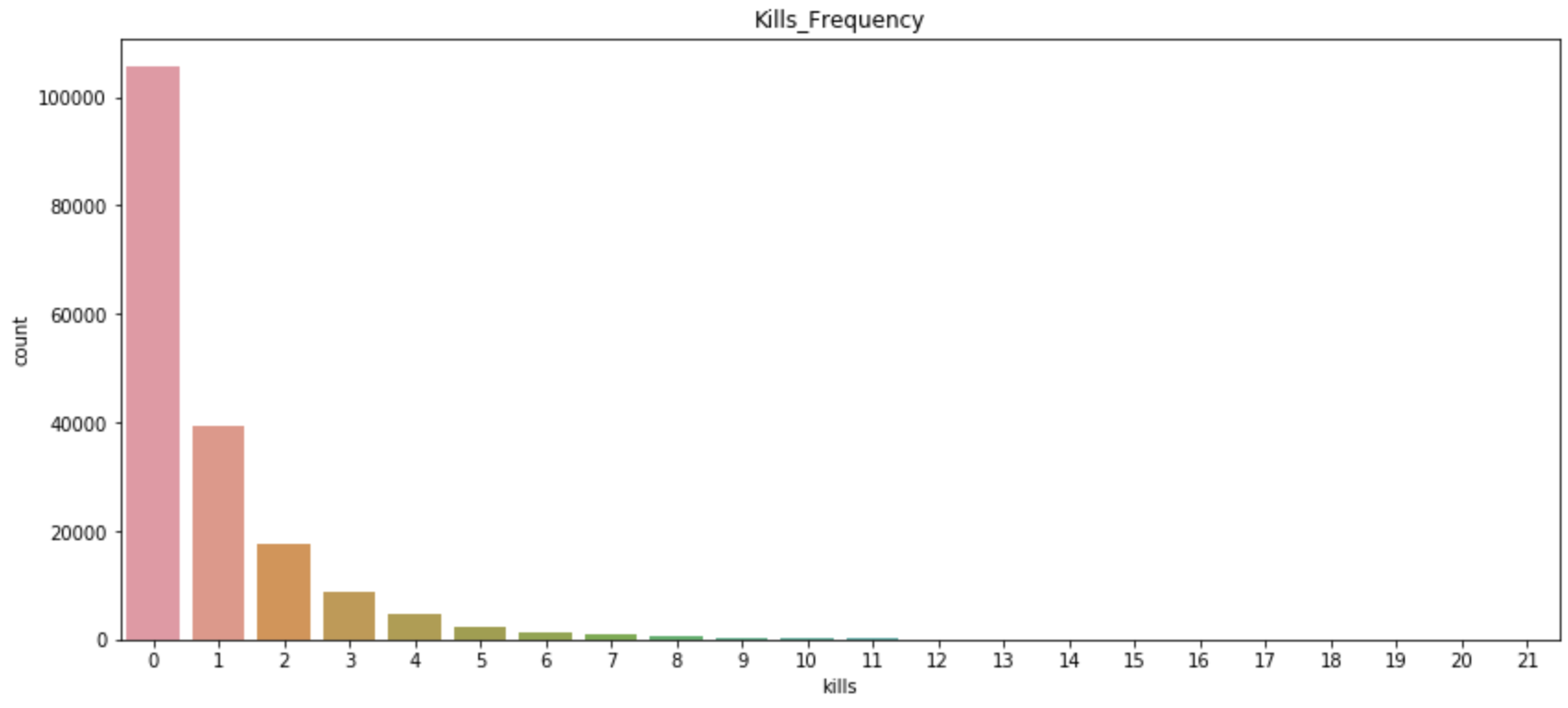
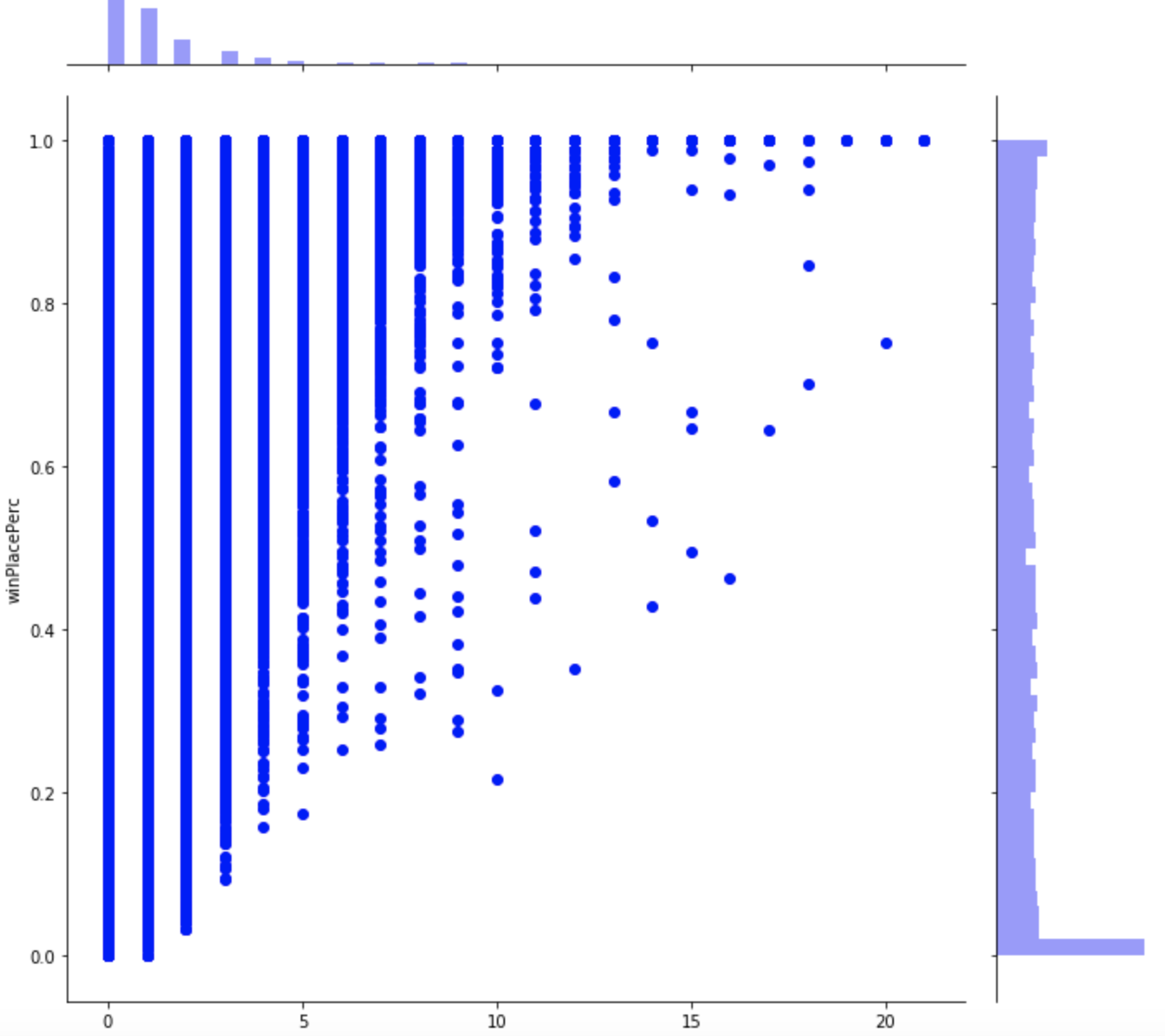


Fig 7 Fig 8

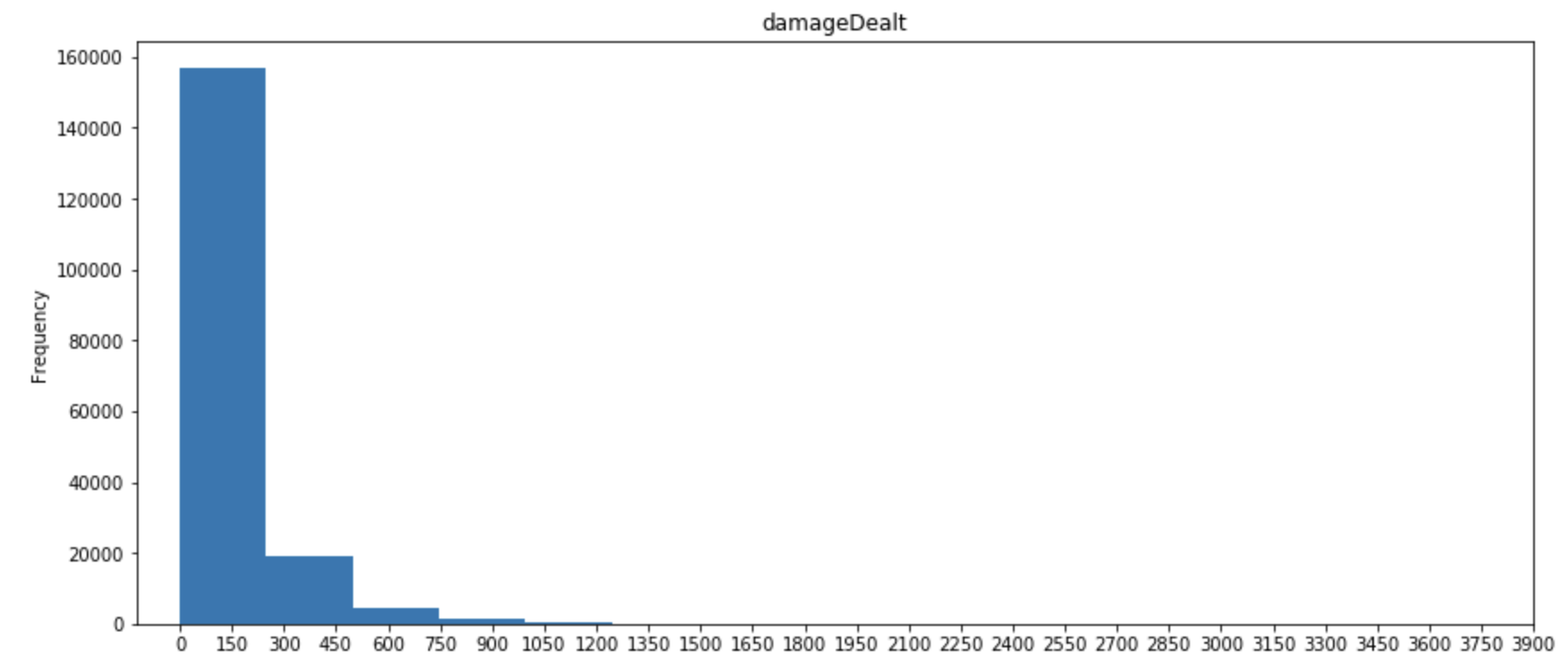
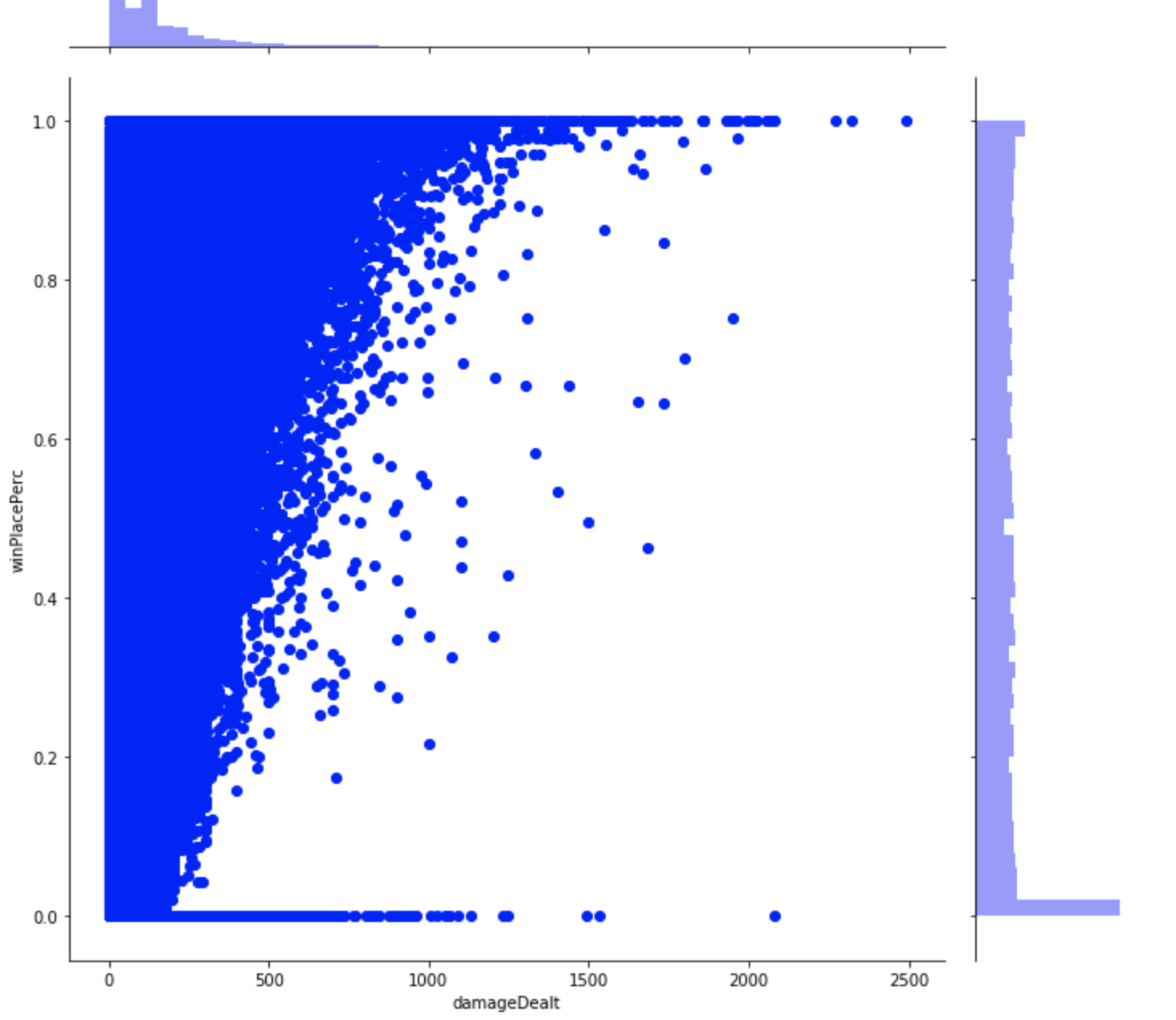


Fig 9 Fig 10

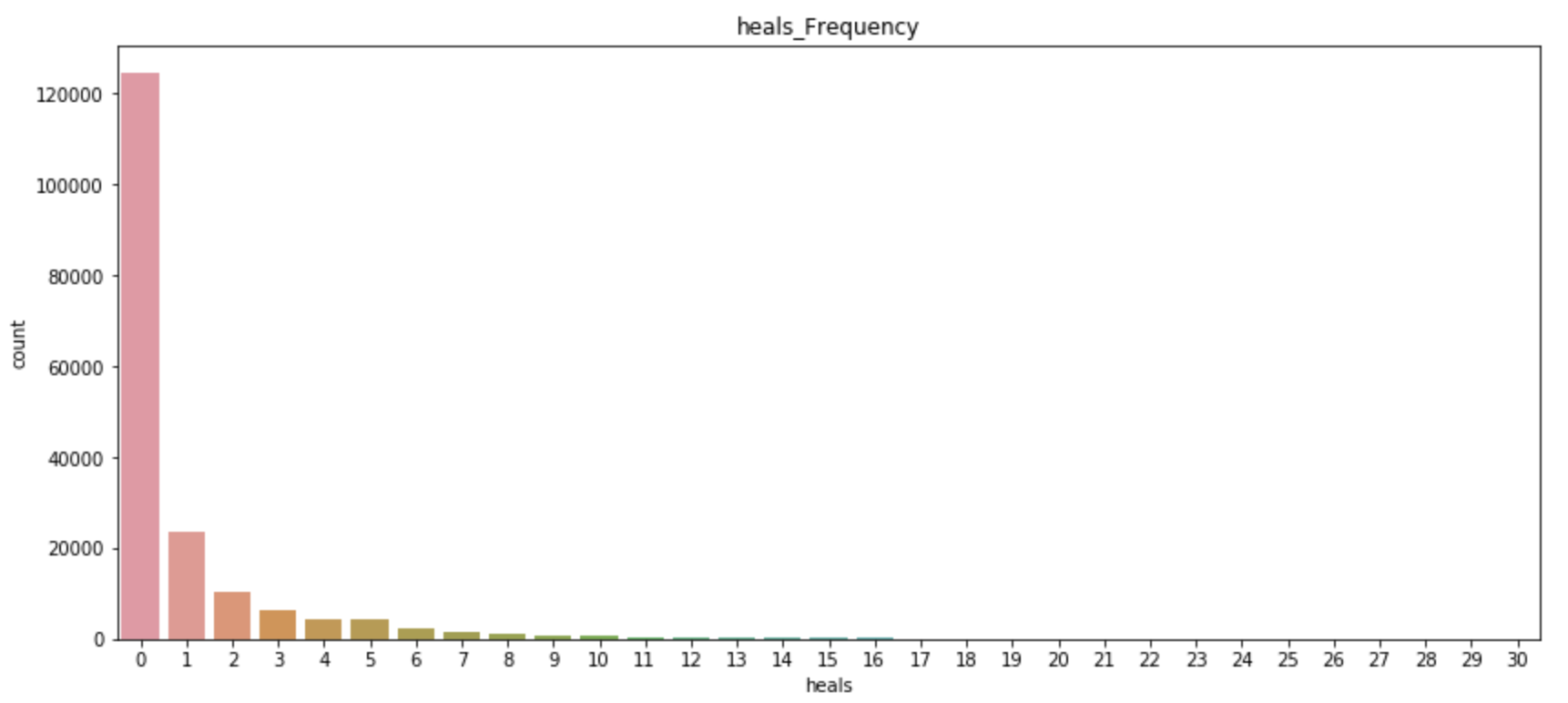
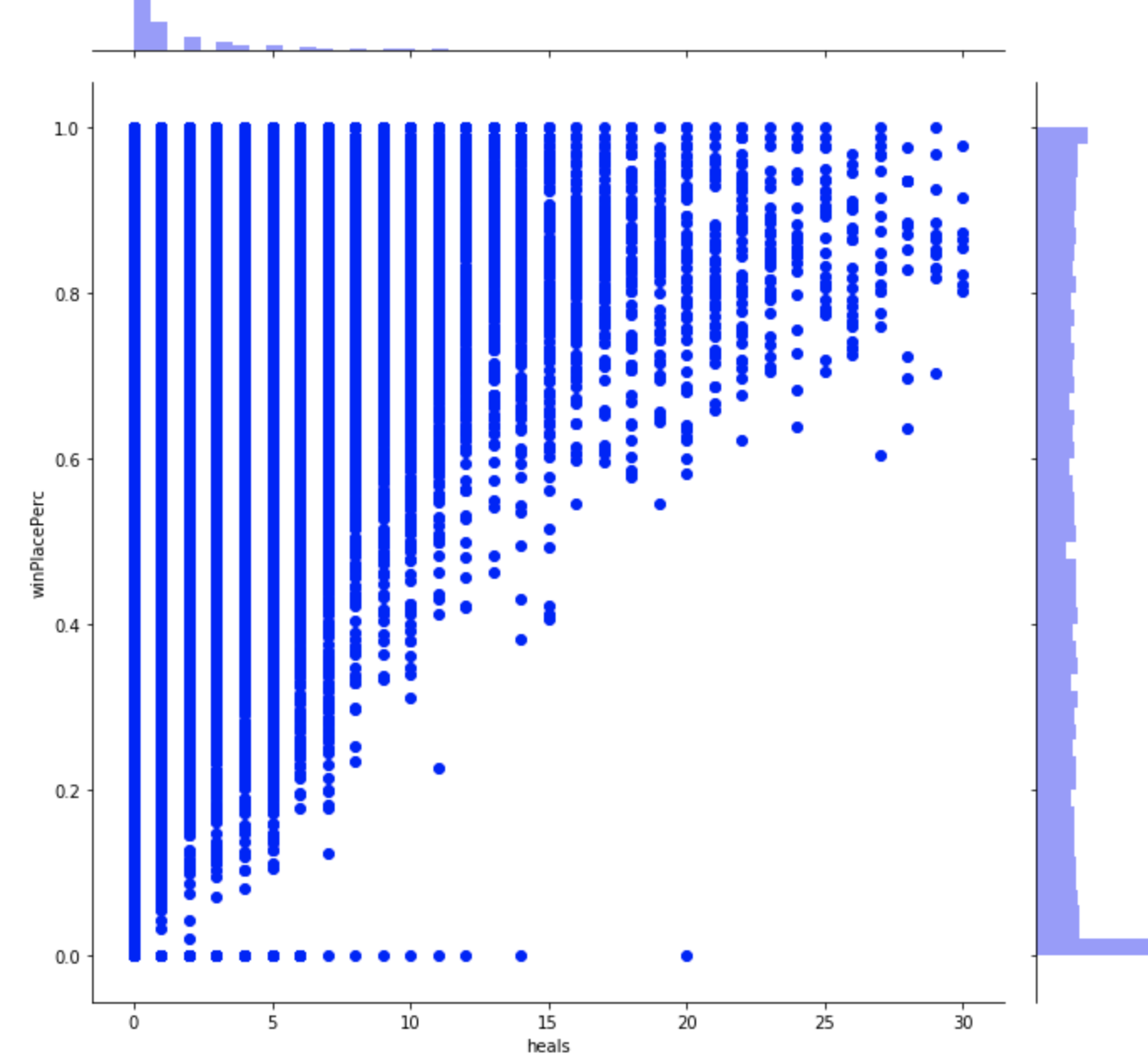


Fig 11 Fig 12

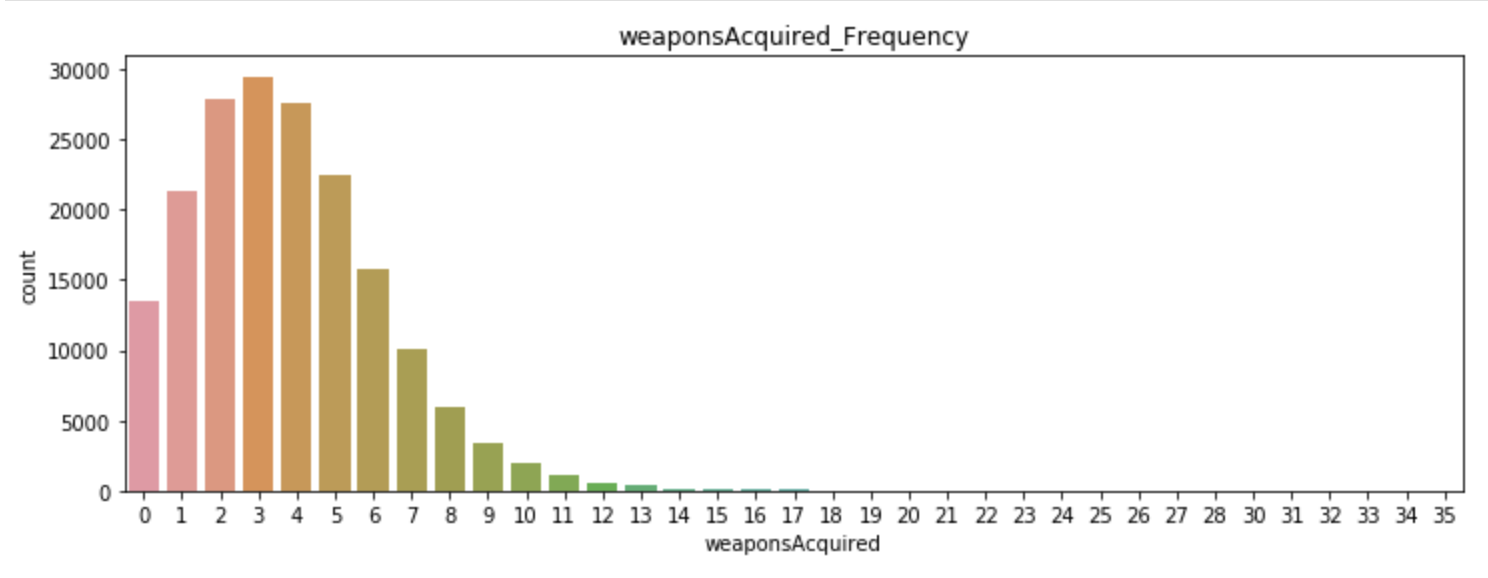
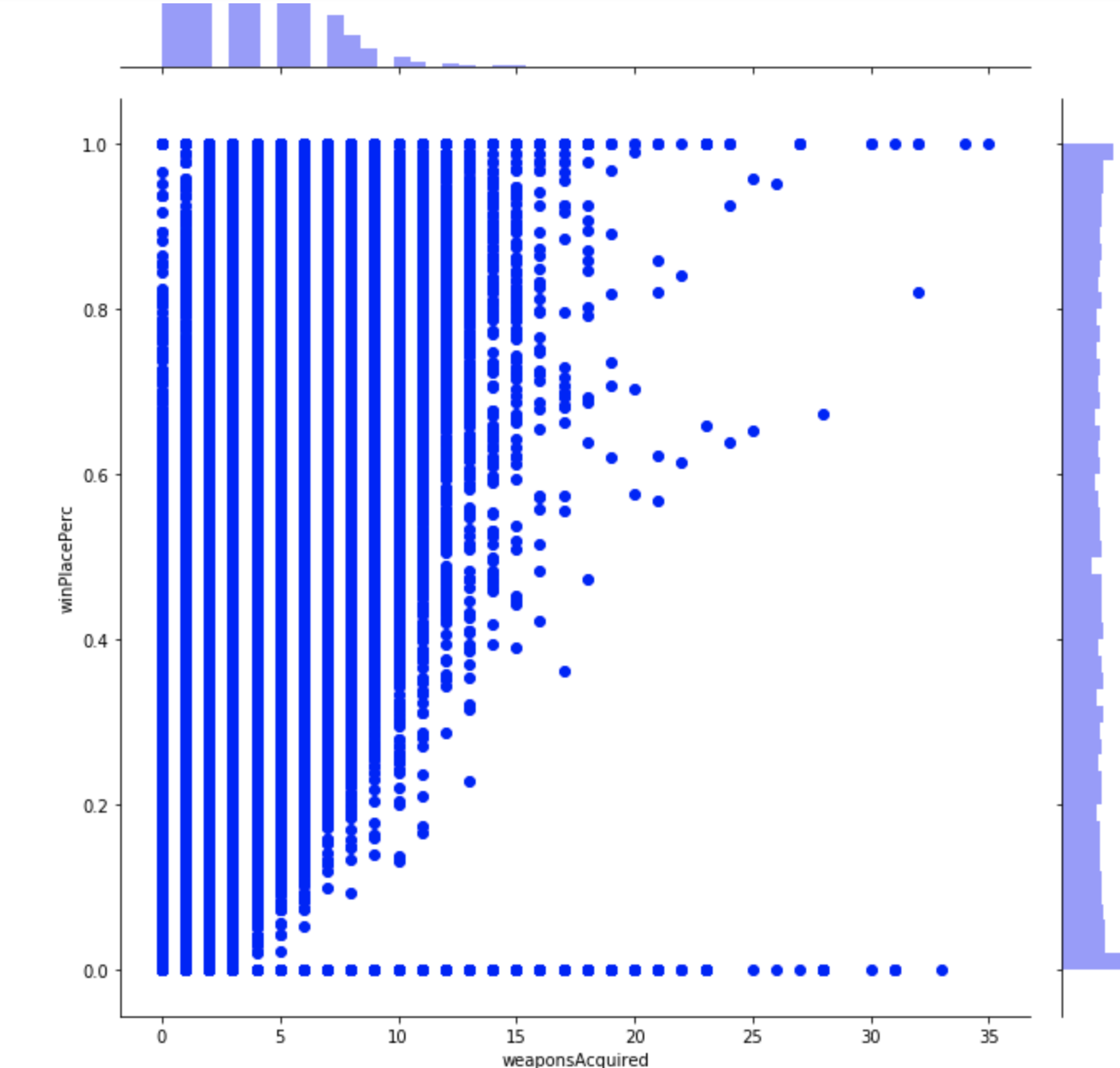


Fig 13

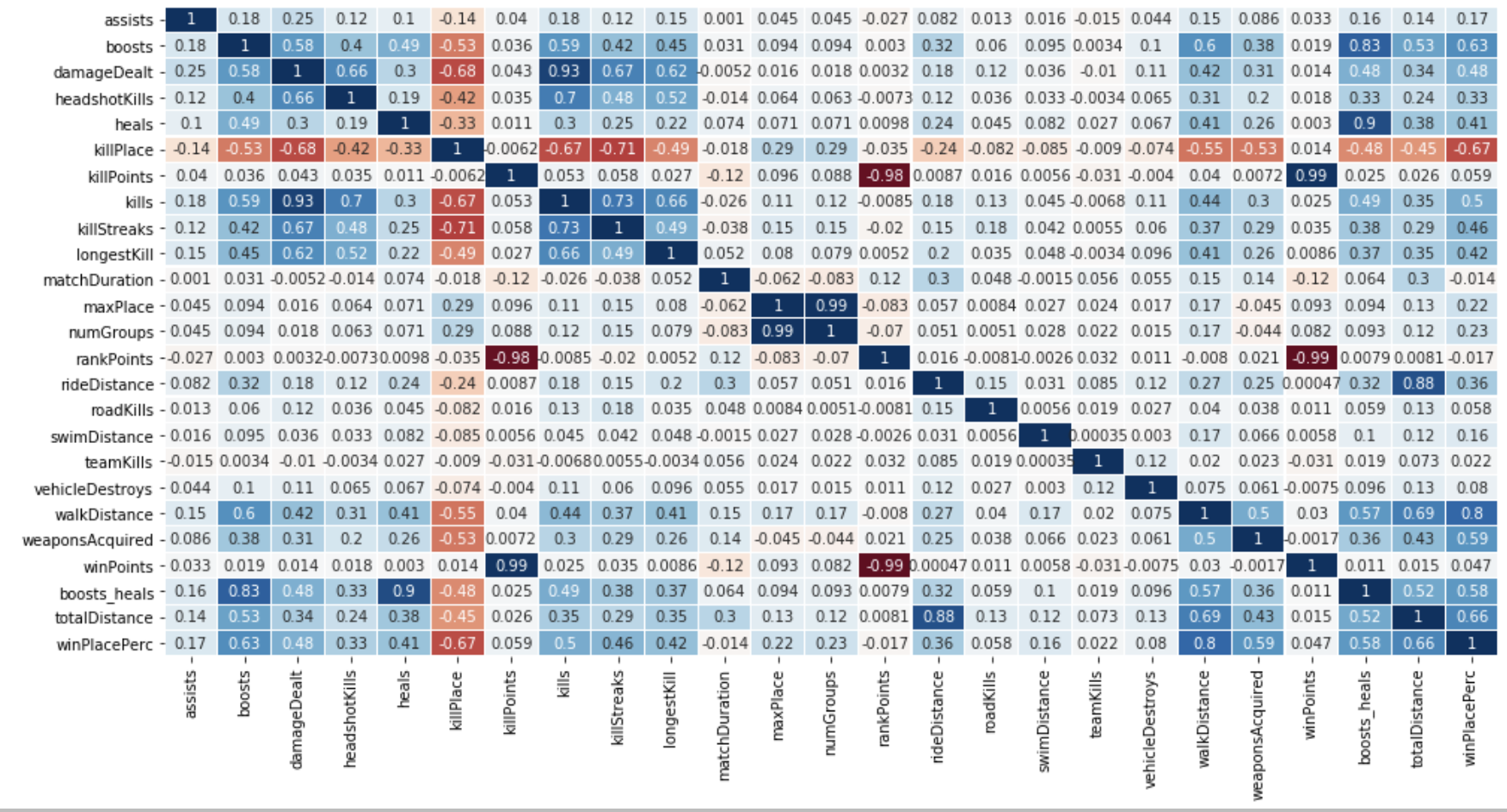
****

Fig 14

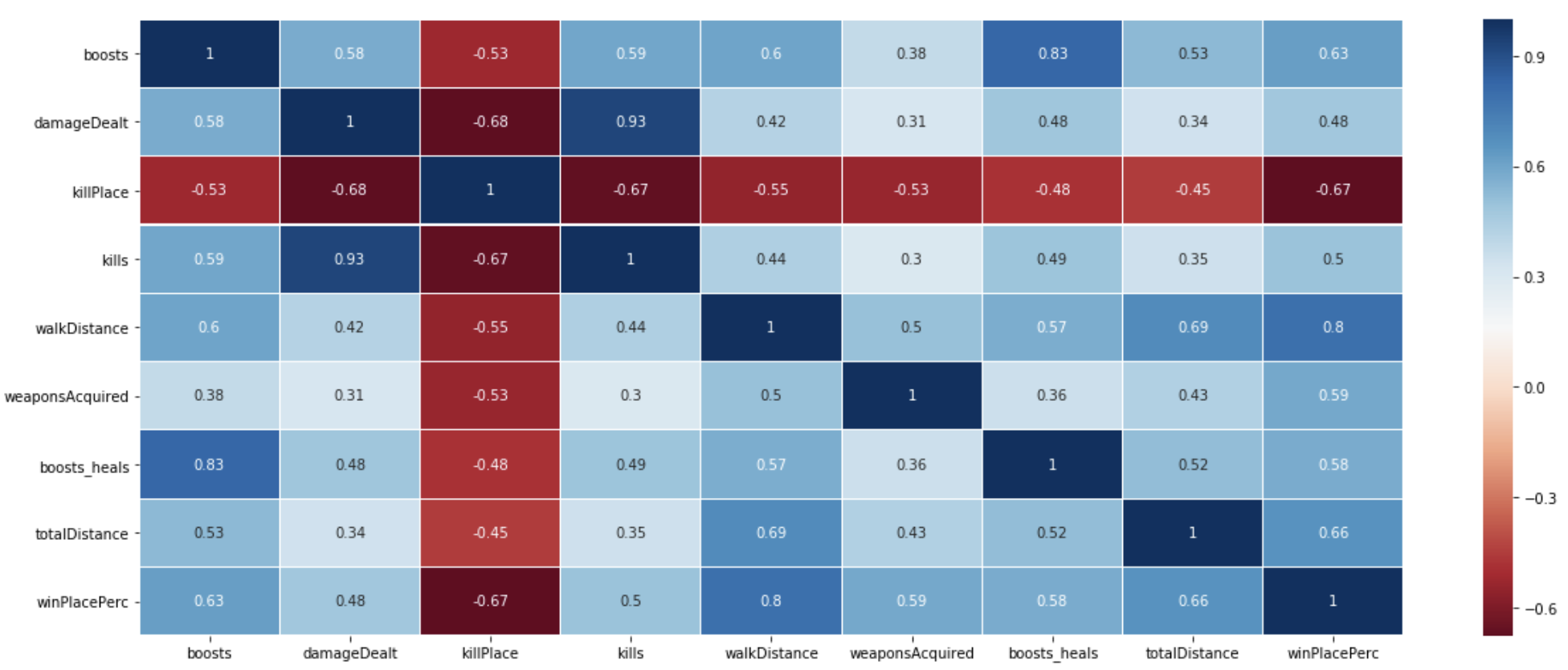


Fig 15

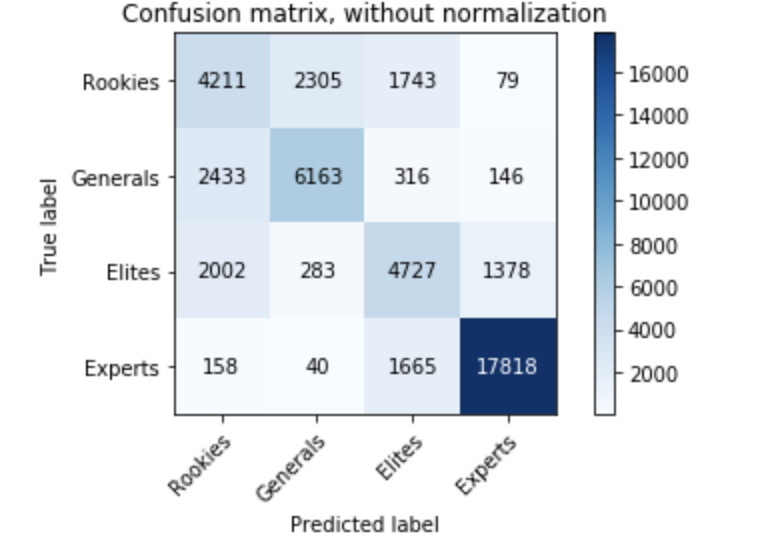
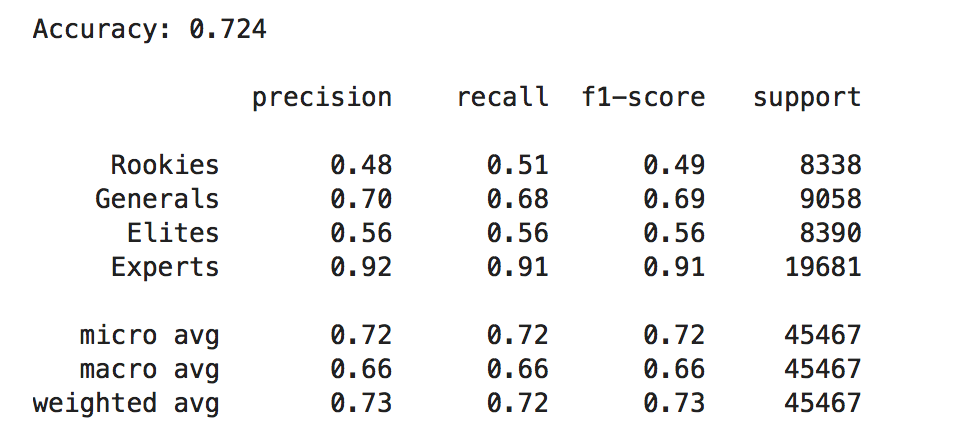


Fig 16



## References:

[1] kaggle, “PUBG Finish Placement Prediction (Kernels Only)”

<https://www.kaggle.com/c/pubg-finish-placement-prediction>