PUBG Finish Placement Prediction

Oskar Randmer, Gustav Nikopensius

Introduction

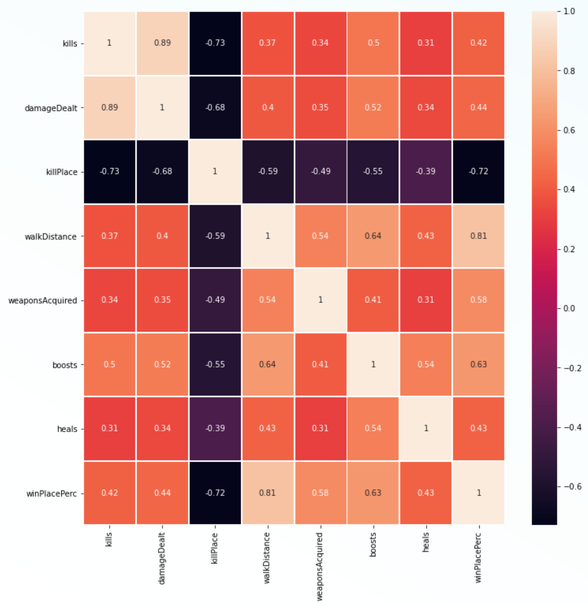
We had player data from 65,000 PUBG games. A row of in-game statistics and match details for each player in each game. Project goal was test 5 different models for predicting player finish placement from 100 other players. Models that we chose were: Simple linear Regression, Ridge, Lasso, Random Forest Regression and Lightgbm.

Data description

The data is from the Kaggle competition PUBG Finish Placement Prediction. The data is composed of two csv files: train_V2.csv and test_V2.csv.

Features description

In total data had 28 different features, There are some strong correlations between the target feature and in-game statistics, in particular the distance walked, weapons acquired, total heals, boost items, damage dealt used and killplace.

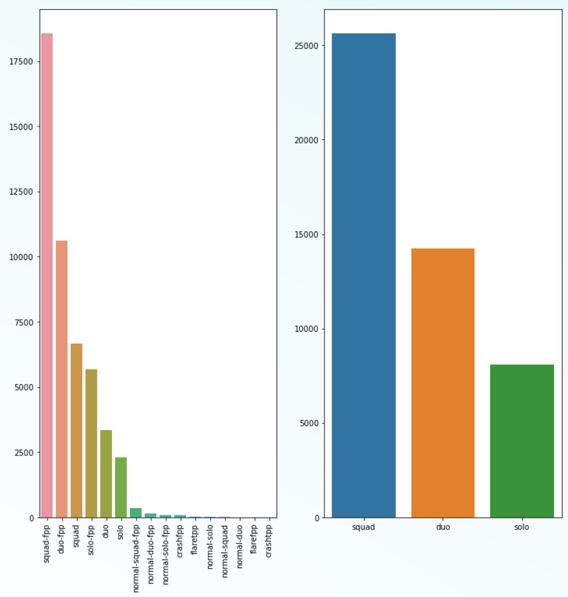


Data Preprocessing

At first, we changed features types in dataframe to reduce memory usage.

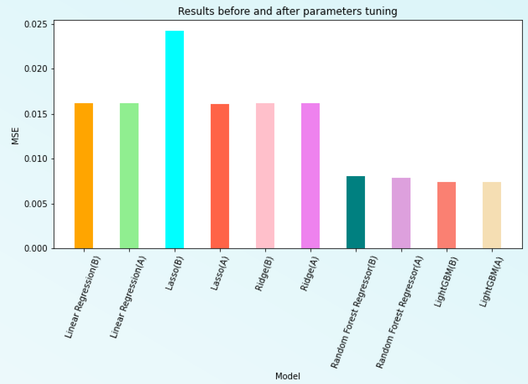
```
dtypes = {
    'Id': 'object',
    'groupId': 'object',
    'matchId': 'object',
    'assists': 'uint8',
    'boosts': 'uint8',
    'damageDealt': 'float16',
    'DBNOs': 'uint8',
    'headshotKills': 'uint8',
    'heals': 'uint8',
    'killPlace': 'uint8',
    'killPoints': 'uint16',
    'kills': 'uint8',
    'killStreaks': 'uint8',
    'longestKill': 'float16',
    'maxPlace': 'uint8',
    'numGroups': 'uint8',
    'revives': 'uint8',
    'rideDistance': 'float16',
    'roadKills': 'uint8',
    'swimDistance': 'float16',
    'teamKills': 'uint8',
    'vehicleDestroys': 'uint8',
    'walkDistance': 'float16',
    'weaponsAcquired': 'uint8',
    'winPoints': 'uint8',
    'winPlacePerc': 'float16'
}
```

Feature „matchType“ had 16 different match types, we decided to separate to the 3 main match types: solo, duo and squad. After we predicted players finish placement according to those three match types separately.



Models Parameters Tuning

For parameters tuning we used GridSearch and RandomizedSearch. GridSearch uses all given parameter settings for testing but with RandomizedSearch parameter settings are selected randomly and not all are used. Tuning was done on given training data and without any preprocessing.



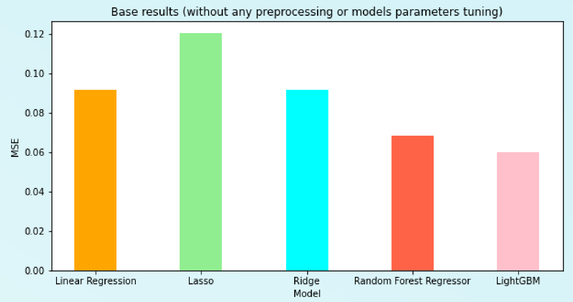
Before Parameters Tuning				
Linear Regression	Lasso	Ridge	Random Forest Regressor	LightGBM
0.016149	0.024231	0.016213	0.008025	0.0074168

After Parameters Tuning				
Linear Regression	Lasso	Ridge	Random Forest Regressor	LightGBM
0.016149	0.016071	0.016212	0.0078874	0.0073966

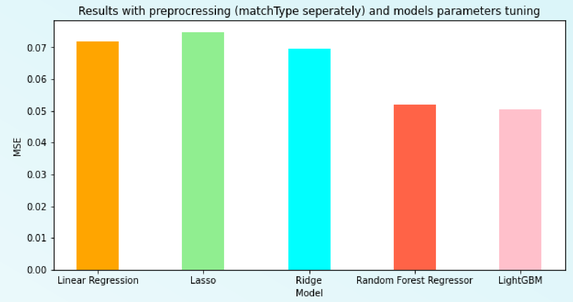
As we can MSE improvement with models' parameters tuning was quite little.

Results

Best base result without any data preprocessing and models' parameters tuning was MSE 0.05970 with Lightgbm and worst 0.09171 with Linear Regression and Ridge.



Best result after preprocessing and models' parameters tuning was MSE 0.05034 still with Lightgbm and worst 0.07473 now with Lasso. Biggest improvement we got with Random Forest Regression MSE went from 0.06841 to 0.05202 and Lasso from 0.12045 to 0.07473.



Conclusion

Our goal was to reach below 0.05 MSE and the best result that we got was 0.05034, what is very close to our goal. As we can see in the results preprocessing was more important than tuning models' parameters,