Data 1030 Final Project Report

Predict the Outcome of the Ranked Game in League of Legends

Yunxuan Zeng

Brown University

Data Science Initiative

https://github.com/YunxuanZeng/1030Project

Introduction

a. League of Legends

League of Legends (LoL) is a popular multiplayer online battle arena video game for its high competitivity and balance, which are developed and published by Riot Games. It has become one of the most played games around the world with a total of 115 million monthly players. With more than 140 champions, each player is allowed to choose one of them and group with the other four people as a team to battle against the enemy team. To attain victory in the game, the team has to find a way to destroy the enemy's base which is called Nexus.

b. Why Interesting

LoL has a large diversity of champions, players are able to find the perfect match to master one of them using their own unique playstyle. Ranked is LoL's most competitive mode where players keep advancing up the ranked ladder and get ranked reward by continuously winning games. Player are competed with those with the same level of skills. A good team composition and early development in the game play an significant role on winning in the ranked mode.

c. Goals

1. How accurately could I predict the outcome of the ranked game given the composition of two teams in the early stage of the game?
2. How does each feature play a role in the outcome of the game? What features are the most important ones in the first 10-minute ranked games?

d. Data Collection

The dataset has been acquired from Kaggle which contains the first 10mins. Stats of 9,879 ranked games. The skill level of players ranges from DIAMOND I to Master which means that players are roughly from the same level. There are two teams. Each game shares 19 features (38 in total). The target variable is "blueWins": 1 indicates the blue wins and 0 indicates red wins. Therefore, this problem is a type of classification.

e. Public Projects or Publications

From the Kaggle, one author used this dataset to cluster player behavior and learn the optimal team composition for multiplayer online games. By applying the cluster and classification models, he determines wins and losses with around 70% accuracy for their target game. Another author used this dataset to predict a class of blueWins. By applying logistic regression, he makes a prediction about an outcome of a game.

Exploratory Data Analysis

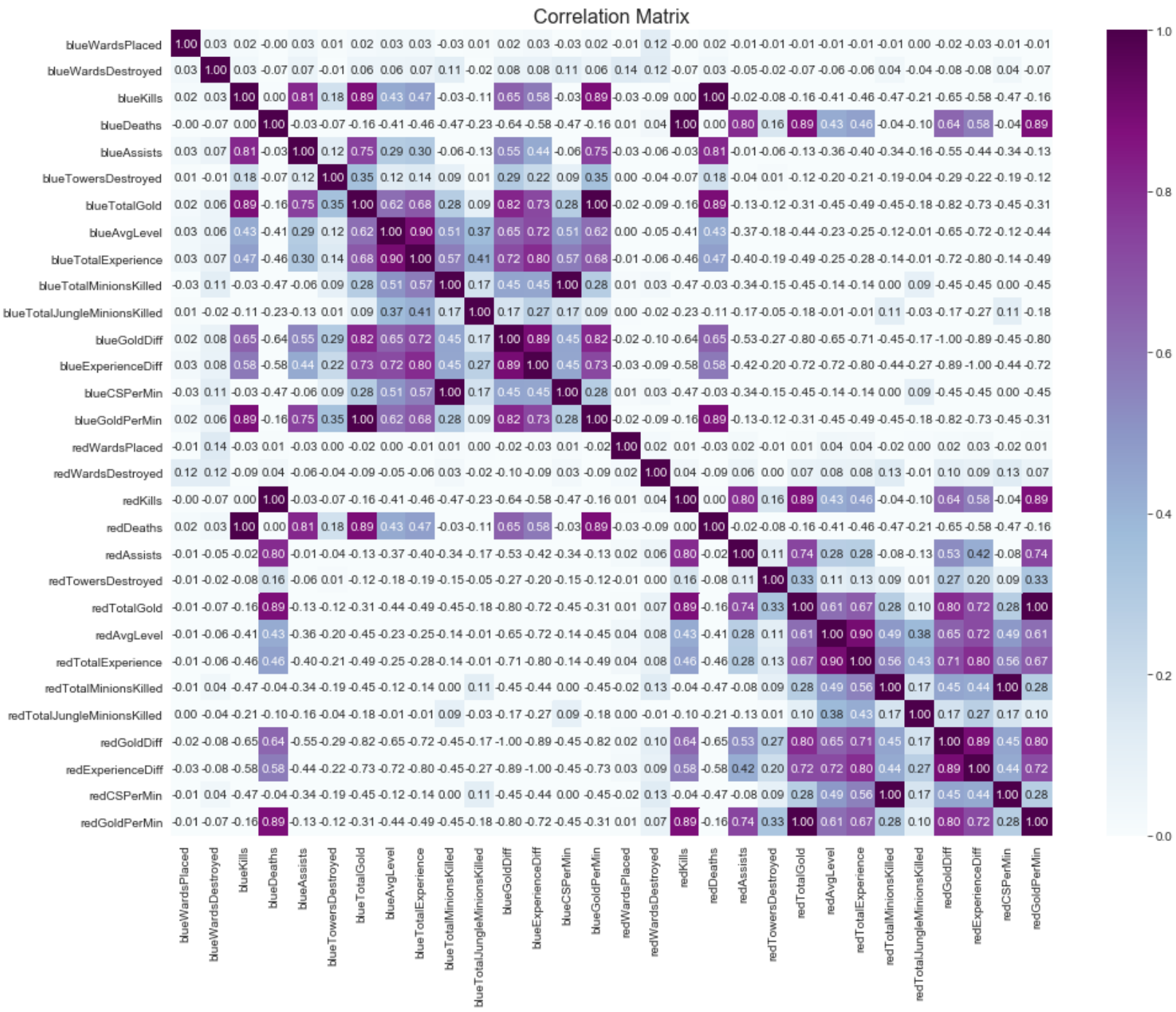

Figure 1: From the figure above, we can see correlations between two features. For instance, "Blue Total Experience" has a strong correlation with "Blue Average Level" with a value of 0.90 but has a weak correlation with "Blue Wards Placed".

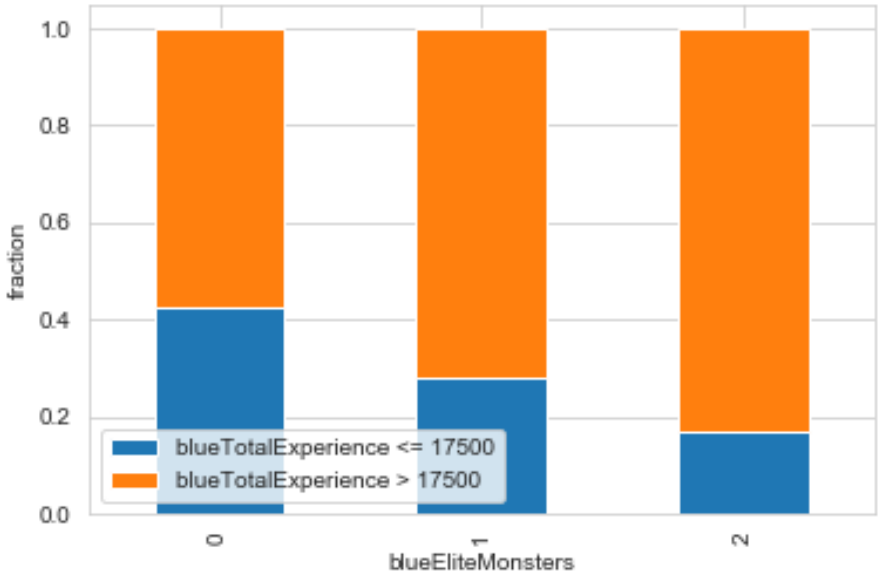

Figure 2: From the figure above, we can see the x-axis is the blue Elite Monsters(categorical) and y-axis is fraction of players in group.
The orange color represents "blue total experience > 17500" while the blue color represents "blue total experience <= 17500".
We can see that as the the number of blue Elite Monsters increases, the fraction of "blue total experience <= 17500" decreases..

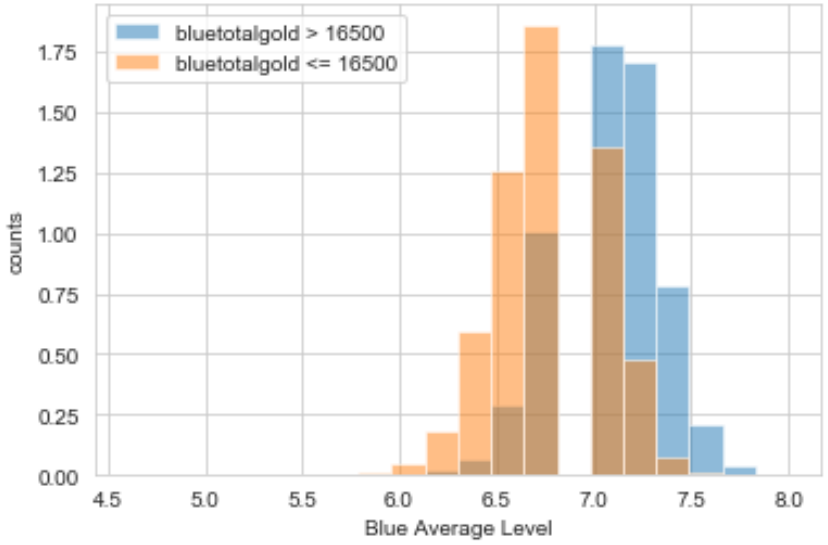

Figure 3: From the figure above, the x-axis is blue average level, and the y-axis is the counts.
The light orange color represents "blue total gold <= 16500" while the light blue color represents "blue total gold > 16500".
We can see that as the blue total gold > 16500, the blue average level is likely higher.

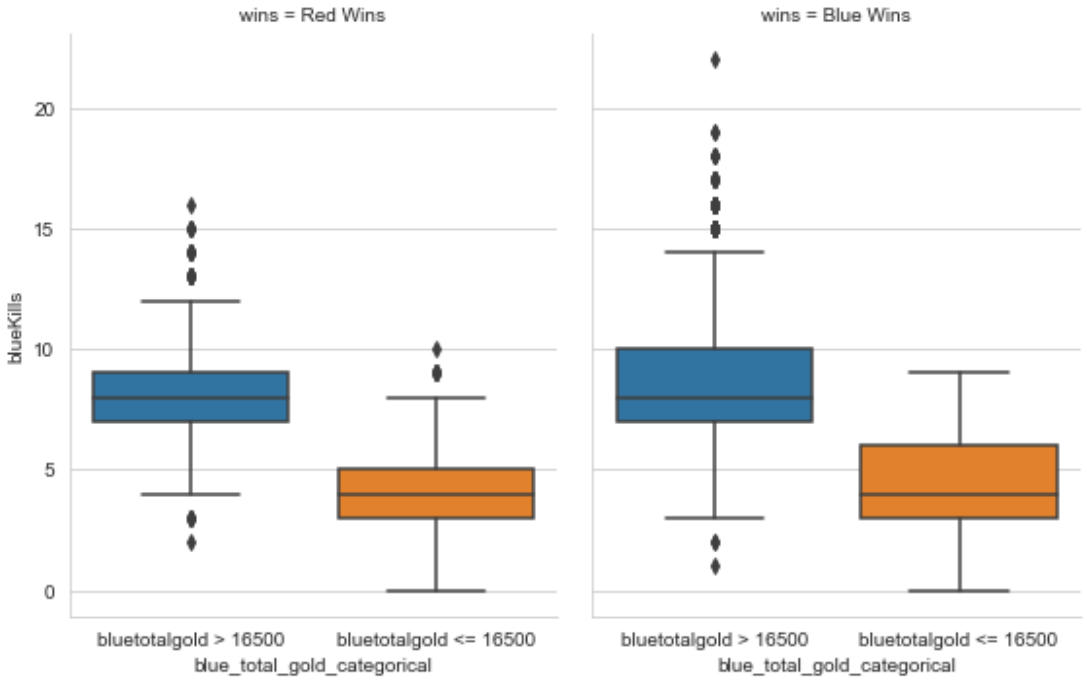

Figure 4: From the figures above, the y axis is "blue kills" and the x axis is the blue total gold by category. The left indicates red wins while the right figure indicates blue wins. We can see that there are more outliers and larger range if blue wins compared to red wins. The interquartile range is smaller in red wins compared to the blue wins. However, their medians are similar.

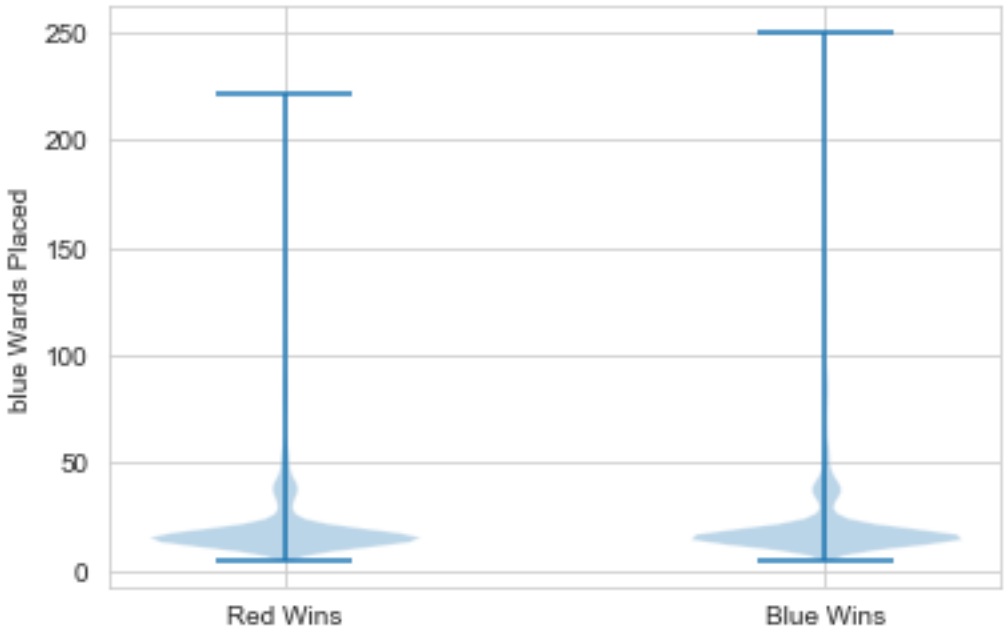

Figure 5: From the figure above, the x-axis is blue wins(target) and y-axis is blue_wards placed. We can see that the range of the target variable for red wins is a little bit smaller than that for blue wins. And the median for both are similar.

Methods

This dataset is iid. since each random variable has the same probability distribution as the others and all are mutually independent. All samples stem from the same generative process and share the same features. And the generative process is assumed to have no memory of past generated samples. And by observation, this dataset does not have a group structure and it is not a time series data.

Splitting Strategy

This dataset only has 9879 samples. It's split into other dataset and test dataset with the ratio of 8:2 and split other dataset into train and validation datasets with the ratio of 4:1. Therefore, 64% of points are in the train dataset, 16% of points are in the validation dataset, and 20% of points are in the testing dataset using the k-fold spitting strategy. The reason to use k-fold is that k-fold is better suited and good to estimate uncertainty due to random splitting of train and validation datasets.

Data Preprocessing

On the other hand, two different preprocessors and 38 features have been used in the preprocessed data. The first one is the OneHotEncoder which converts categorical features into dummy arrays. From the dataset, "firstblood", 'Herald" and "Dragons" in both red and blue teams are applied. The reason is that in the first 10 minutes of the game, there are only 2 categories for each feature which 0 or 1. Therefore, OneHotEncoder is a good way to make conversion.The second one is StandardScaler. From the dataset, all remaining features from both teams are applied to this preprocessor. The reason is that, each feature is continuous and follows a tailed distribution. Therefore, StandardScaler is a good way to be used here. And there is no need to do with the target variable since it look good with 0 standing for red wins and 1 for blue wins.

ML Pipeline

The machine learning pipeline is developed to automating the machine learning workflow by enabling data to be fitted tin to a model to be analyzed. First, the data is split into the train, validation and test datasets using the above splitting strategies. Next, two transformers have been used to preprocess each categorical and continuous feature values. After checking the fraction of missing values in features and the fraction of points with missing values which are both zero, there is no need to handle missing values. Then GridSearchCV which implements a "fit" and a "score" method is applied to put all things together as it's easy to try every possible combination of values of hyper parameter and evaluate different models for each combination. After all combinations are estimated, the method can output the best estimator and parameter combination which gives the highest score. It provides convenience since several algorithms will be applied to make prediction.

Metric

The accuracy score metric is chosen since this dataset is balanced as the 50.1% of points is in classed 0 and 49.9 % points in class 1. It's clear to see TN is not large and the baseline accuracy score is round 0.501. Since this project is aimed to predict the outcome of the ranked game, the

action is not expensive to act and accuracy is appropriate as all the class are equally important. In other words, both true positives and true negative are important.

Logistic Regression

Logistic regression is the appropriate algorithm to conduct when the target variable is binary. Three different penalties are tried: l1, l2, and elasticnet. The l1 penalty equals the absolute value of the magnitude of coefficients, l2 penalty equals the square of the magnitude of coefficients, and elasticnet is the combination of both two. It's better to try each of them to find the best model. For all, the range of the inverse of regularization strength C is set to [0.01, 0.1, 1, 10, 100]. It because that if C is too large, the model will fail to generalize on new data and if C is too small, the complexity of the hypothesis will be reduced. It's reasonable to crate a range C to find the best fit. Considering the parameter solver, 'liblinear' and 'saga' handle the l1 penalty. Three additional "lbfgs", "sag" and "newton-cg" are added to l2 penalty. "saga" is the one which support the elasticnet penalty.

Decision Trees

Decision tree splits the data with respect to certain features. It's basically if else statements. The parameter max_depth is set to [1,2,3,5,10,50,100]. In general, the deeper the tree grows, the more complex the model will become. Therefore, a wide range of max_depth is a way to avoid overfitting. Another parameter max_features is set to [0.1,0.25,0.5,0.75,1.0] in that the float represents the fraction of features which are considered at each split. Since it's computationally heavy if all feature are chosen which could cause overfitting, selecting a various number of max_features is appropriate to increase the stability of the tree and reduce variances. (Mithrakumar, 2019)

Random Forest

Random forest is an ensemble of random decision trees. What happens is that a bunch of different decision trees are trained so that each tree sees a different random subsample of points and features. Two parameters max_depth and max_features are tuned. The max_depth is set to [1, 2, 3, 5, 10] and the max_feature is set to [0.25,0.5,0.75,1.0]. The reasons for choosing various values are similar to decision tree.

KNN

The k-neighbors classification is the widely used technique, which improves learning based on the k nearest neighbors of each query data point. Two parameters n_neighbors and weights have a strong influence on the predictions. The n_neighbors is set to [1,10,30,50,100] indicating the number of neighbors to use for query points. The weights are set to uniform and distance, where uniform enables all points in each neighborhood to be weighted equally and distance weights points by the inverse of their distance.

Uncertainty

Considering splitting, the parameter random_state used for initializing the internal random number generator contributes to the uncertainty. When the random_state changes, the balance for each dataset respectively changes. Without specifying random_state, different result will be generated. Considering ML algorithms, random_state has a larger impact on random forest and decision tree. The decision tree algorithm is based on the greedy algorithm where it is repeated several times using a random selection of features and samples. This random selection is based on the pseudo-random number generator which takes random_state as input. It's clear to see various random_states outputs different best models in random forest and decision trees. The standard deviation of test scores for the decision tree is larger than other methods. Oppositely, random_state is never used in the logistic regression. The random_state only has impact on the splitting.

Result

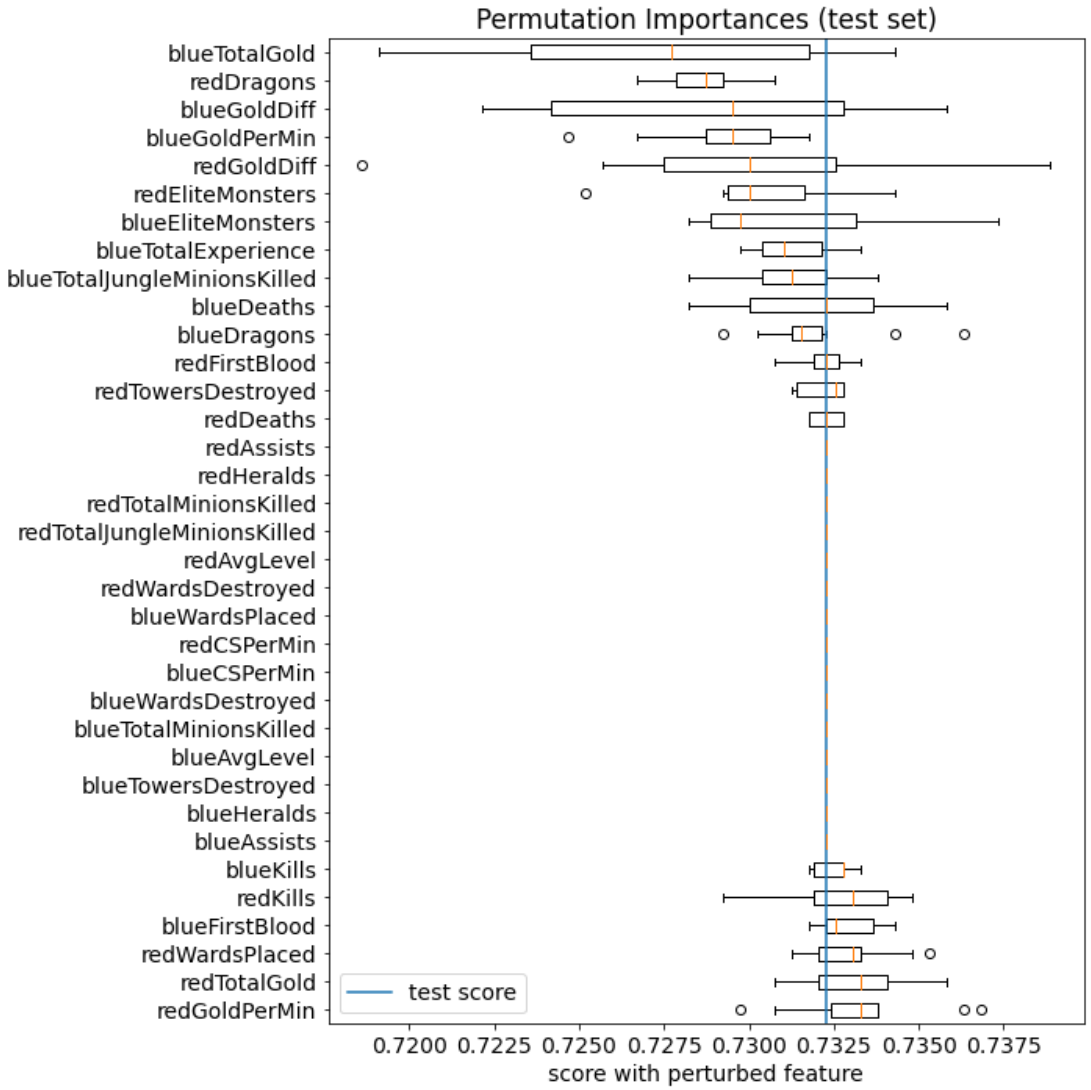

Figure 1: Permuatiaon importances for the top 25 features, blueTotalGold is the most important feature.

- Logistic Regression with l2 and elasticnet yield the highest score of 0.728 which is around 43 standard deviation above the baseline score which is 0.500.
- The bluetotalGold is the most important feature and deaths and assists are least important features.
- Four of the top five feature are related to gold so farming gold in the first 10-min game is a key factor to win the game. In the early game, the first 10 min is safest phase for farming and gold equips champions for fights and gaining powers.
- Dragons and Elite Monsters are second important features since they grant massive bonus experience to killer and nearby allies which are crucial for early development.
- It's surprising to see farming golds and experience are more important than kills and deaths. And the placing and destroying wards has little impact on the outcome the game. To win the game, players should put first priority on the farming and experience rather than killing and do not spend too much on buying and destroying wards. (Brezak, 2020)

Outlook

The accuracy of the best model is not very high. It's clear to see that the accuracy scores for all models applied to this project is around 0.72. There is no big difference in accuracy score. It's possible that there may have some better model with comparatively higher accuracy score. The addition technique can be used to this project could be XGBoost which is another popular tree-based method. It's more adavanced than random forest in that it has L1 and L2 regularization while random forest does not.

Another consideration for this project is that this dataset does not include any information about the compositions of champions in each team. Some team composition could put players at a disadvantage from the beginning of the game. And it would be better if the information about internal relations between champions could be included. Furthermore, players' experiences on different champions could also have an impact on the outcome of the game. It may have higher accuracy score is these data can be collected. (Juras, 2019)

Reference

1. Fanboi, Michel's. "League of Legends Diamond Ranked Games (10 Min)," April 13, 2020. https://www.kaggle.com/bobbyscience/league-of-legends-diamond-ranked-games-10-min.
2. Michalbrezk. "How to Win League of Legends?" Kaggle. Kaggle, November 9, 2020. https://www.kaggle.com/michalbrezk/how-to-win-league-of-legends.
3. Juras, Marta. "League of Legends' Ranking System Explained: How It Works." *Dot Esports*, 16 Nov. 2019, dotesports.com/league-of-legends/news/league-of-legends-ranking-system-explained-17171
4. Mithrakumar, Mukesh. "How to tune a Decision Tree." *Towards Data Science*, 12 Nov. 2019, https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680.

Github Link: https://github.com/YunxuanZeng/1030Project