Surrender at 15 minutes? Shaping Surrender Mechanism in League of Legend

Zheng Zeng

Brown University

Github: https://github.com/itsmax99/-surrender-in-lol.git

Introduction

League of Legends is a team-based strategy game where two teams, blue team and red team, of five players face off to destroy the other's base. To win this game, the team needs to power up themselves, such as leveling up, purchasing equipment, getting some unique buffs and making good strategies. This game usually lasts for 30 to 40 minutes and any player in the team can initiate a surrender when the game is at 15 minutes. Surrendering a game for 15 minutes, when the game is not in its final stages, is a tough choice for the players. Because players are unsure whether this game still has the chance of turning this game over. This uncertainty will make players waste time in situations where they know they are losing, or to miss the chance to recover by giving up too soon. To solve this problem, this study proposes a predict model based on machine learning, which can predict the final outcome of the game according to the core data such as economic gap, kill gap and map control when the game is played for 15 minutes. This model not only helps players to wisely allocate their time on the game, but also improves their overall experience. Except to help players, this model can help designers to reshape the surrender mechanism. For example, in some cases, if the prediction model shows that one side has an extremely low win rate, the conditions for surrender may be relaxed to make it easier for players to surrender.

The dataset comes from Kaggle[1]. Also, the dataset collected from bronze and diamond ranked tier in the League of Legends v12.10 patch. This study uses more than 100k data for training the machine learning model. All the data were collected at 15 minutes which is the time the game allows the player to start to surrender. The study used 47 features and data represent a game in two ways: absolute values and difference values. The absolute values represent the value of a given feature for both teams. The difference values represent how many values for a given feature for the blue team higher than the red team. Those features contain all the most important parts that dominate the direction of the game such as economics differences, experience difference, map control and buff. The Table 1 shows the main difference value features and target variable.

Table 1: The difference value features used and its description

Importance Factors	Features	Description
Target Variable	blue_win	1=blue win, 0=red win
Economics	gold_diff	Total gold different
Experience	xp_diff	experience different
Map control	ward_placed_diff	ward placed different
Map control	ward_destroyed_diff	ward destroyed different
Economics	first_blood	1=blue team get first kill
Map control	turret_diff	turret destroyed difference
Map control	inhibitor_diff	inhibitor difference
Buff	herald_diff	herald difference
Buff	baron_diff	baron difference
Buff	air_diff	air drakes' difference
Buff	hextech_diff	hexetch drakes' difference
Buff	fire_diff	fire drakes' difference
Buff	earth_diff	earth drakes' difference
Buff	water_diff	water drakes' difference

Explore Data Analysis

Since there are many features in my dataset, the study first plots the Pearson Correlation Matrix to find the overall relationship between features and target variable. The study only chooses the different values for plotting because of readability. Figure 1 shows that the gold difference, experience difference and kill champion difference are highly correlated with the target variable. It is worth noting that some features related to the map control such as total ward placed difference and total ward destroyed difference are not highly correlated with the target variable. Also, the features related to the buff are also not highly correlated with the target variable.

In order to explore the reason why buffs related features are not correlated with the target variable. I add up all the features associated with buffs to form a new feature. Figure 2 shows that when the blue team kills one more dragon than the red team, they will win 64% of the games. If they outkill the red team by two dragons, they can win 76% of the games.

In League of Legends, the ward is crucial for strategy decisions. This can allow the team to gain information about enemy movements, objectives and potential ambushes. Ward placed difference measures the number of wards placed by the blue team versus the red team. A positive value indicates that the blue team has stronger vision control. However, the results from the graph are unexpected. The difference in the number of placed wards does not seem to have a significant

effect on the outcome of the game because in most cases the difference in the number of guards is small and the win distribution is close to overlapping.

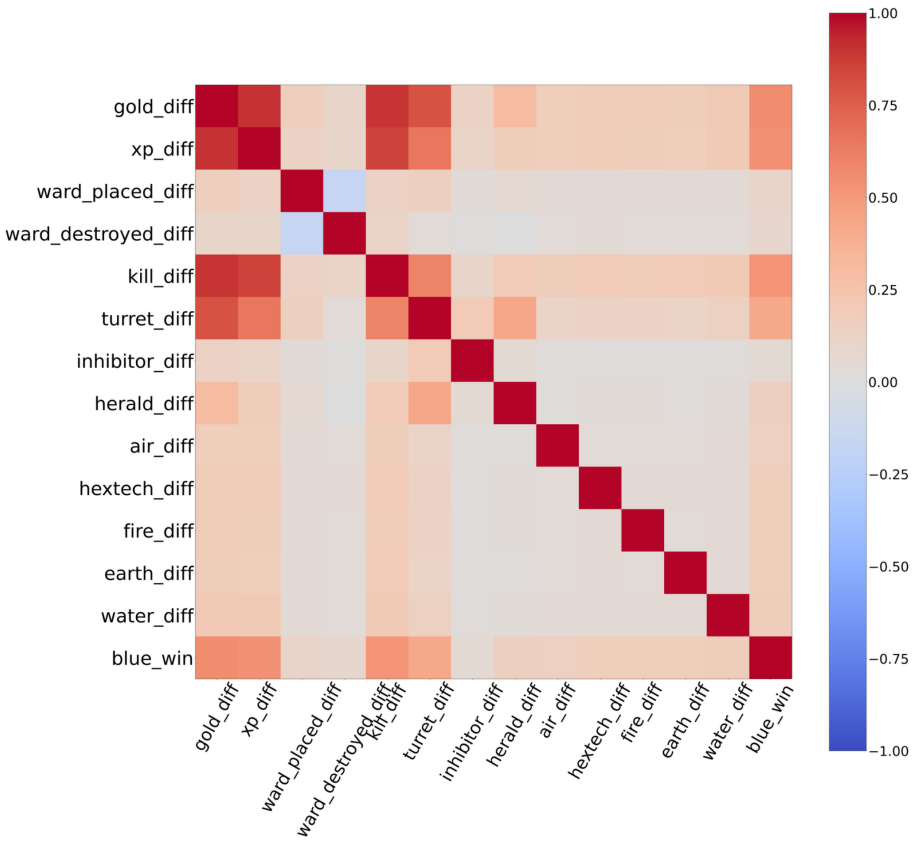

Figure 1: The Pearson correlation heat map for difference value features

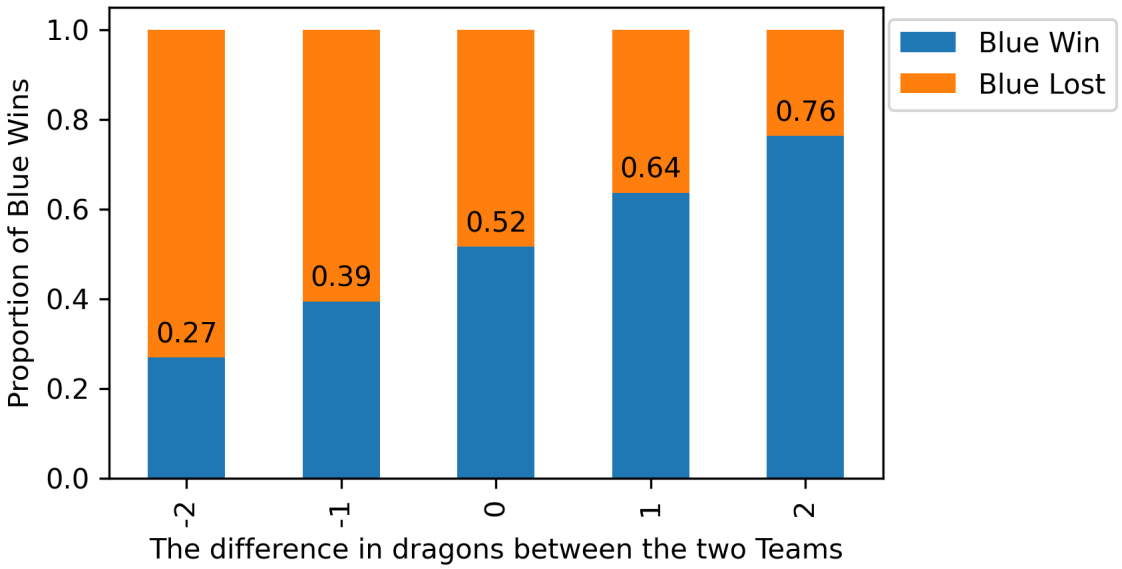

Figure 2: The relationship between the number of dragons and the winning percentage of the blue team

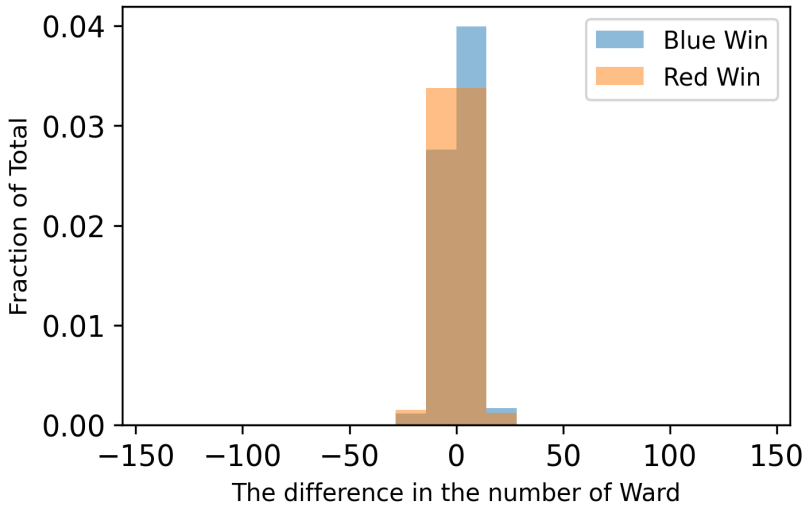

Figure 3: Field control and outcome relationship: distribution of differences in the number of eye insertions

Methods

This study uses the KFold with 4 splits to split my dataset into train set (60%), validation set (20%), and test set (20%). The reason to use this split strategy is because the study wants to comprehensively evaluate model performance and generalization ability. For the XGBoost method,

the study did not use KFold but still split the dataset into train set (60%), validation set (20%), and test set (20%). In the data preprocess stage, as all the features are continuous values and considering some features, such as golf difference, have larger values than others, such as ward placed difference. This study chose to standardize all the features for comparing the features' importance.

The study chooses five machine learning algorithms: Logistic Regression, Random Forest, K-Nearest Neighbors, XGBoost, and Extra Trees. To find the best hyperparameters combination, the study used GridSearchCV for hyperparameter tuning with cross-validation. Also, due to uncertainties from split method and non-deterministic ML methods, the study will use 10 different random states to train and test the model. The Table 2 shows the hyperparameters chosen to be tuned and the tuning value range.

Based on the study purpose, all the ML methods are considered Precision Score as evaluation metrics. The study aims to help the player make the right decision, by accurately predicting negative samples. Using the precision score as evaluation metrics can force the model not to predict the false positive (FP). Because the false positive means that the player needs to put more time on a game that eventually will lose. This situation will increase the player's play time. That's the opposite of what the study was trying to do.

Table 2: Overview of machine learning algorithms and hyperparameter tuning results

Algorithm	Hyperparameters	Parameters Value	Optimal Value
Logistic Regression	penalty	[Elasticnet]	Elasticnet
	C	[0.0001, 0.01, 1, 100, 10000]	0.01
	l1_ratio	[0, 0.4, 0.8, 1]	1
Random Forest	max_depth	[1, 10, 100]	100
	max_features	[0.5, 1.0]	0.5
K-Nearest Neighbors	n_neighbors	[1, 3, 5]	5
	weights	[uniform, distance]	distance
XGBoost	max_depth	[1, 10, 100]	3
	reg_alpha	[1, 10, 100]	10
	reg_lambda	[1, 10, 100]	1
Extra Trees	max_depth	[1, 10, 100]	1
	early_stopping_round	50	50
	max_depth	[1, 10, 100]	100
	max_features	[0.5, 1.0]	0.5

Result

Table 3 contains the mean test score and standard deviation of test score. We can see that all models achieve a precision rate of more than 20% over the baseline model. Logistic regression, extra trees, XGBoost and random forest all reached an accuracy rate of about 76%. The K-Nearest Neighbors only achieve to 72.9% average precision score.

Table 3: Comparison of mean Precision Score and standard deviation of different machine learning algorithms

Algorithm	Mean Precision	Std Precision
Baseline Model	0.509	0.002
Logistic Regression	0.764	0.004
K-Nearest Neighbors	0.729	0.004
Random Forest	0.765	0.004
Extra Trees	0.764	0.004
XGBoost	0.763	0.004

The study chooses the random forest algorithm for further feature importance analysis because this algorithm performs the best among all other ML algorithm, which their mean precision rate reach to 76.5%. The confusion matrix from random forest show that in the 11879 samples predicted to be positive, only 2803 points are incorrectly predicted to be positive, while the remaining 9076 points were correctly predicted to be positive, the precision score is 76.4%。

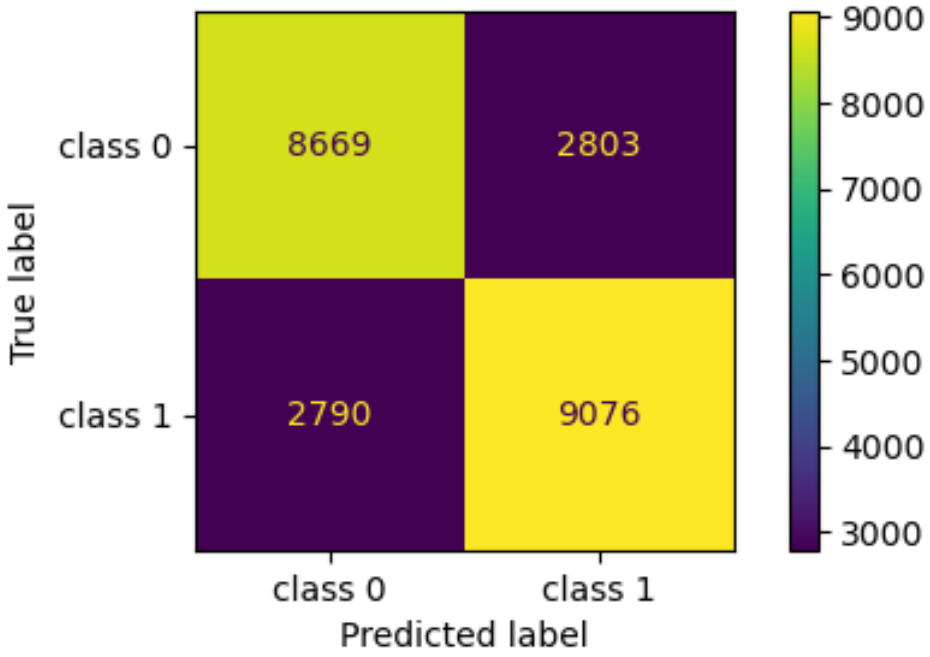

Figure 4: Performance Evaluation: Confusion Matrix

To further explore which features are the most feature for prediction. Permutation important is used for analysis. Figure 5 shows that the gold difference and experience difference are the top 2 important feature. This is because the economic system of MOBA games is crucial, and the game is mainly driven by money. Also, experience is another important dimension, and experience in the game determines the combat effectiveness of the players, which directly affects the direction of the game.

To make result more robustness, the study use other two global features importance method. Figure 6 based on the Mean Absolute SHAP Value shows that gold difference and experience difference still the top 2 features that most contribute to the model prediction. Figure 7 based on MDI (Mean Decrease in Impurity) shows the same result.

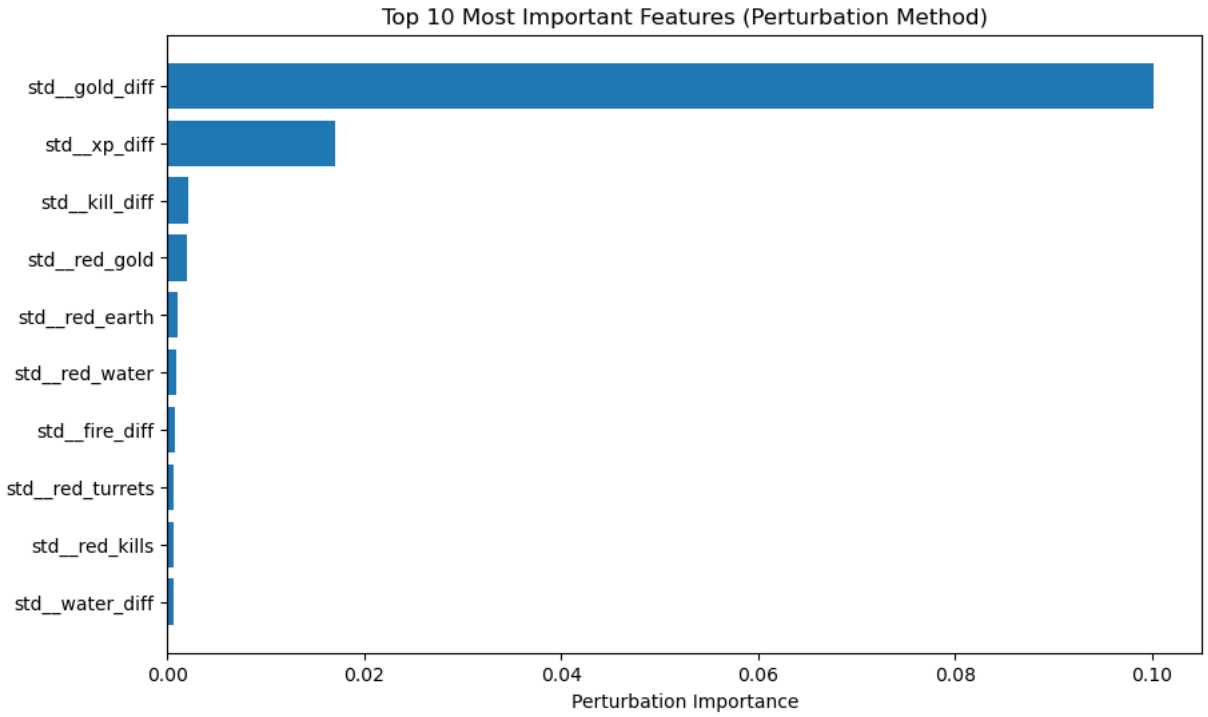

Figure 5: Top 10 Feature Importance in Predicting Game Outcomes (Perturbation Method)

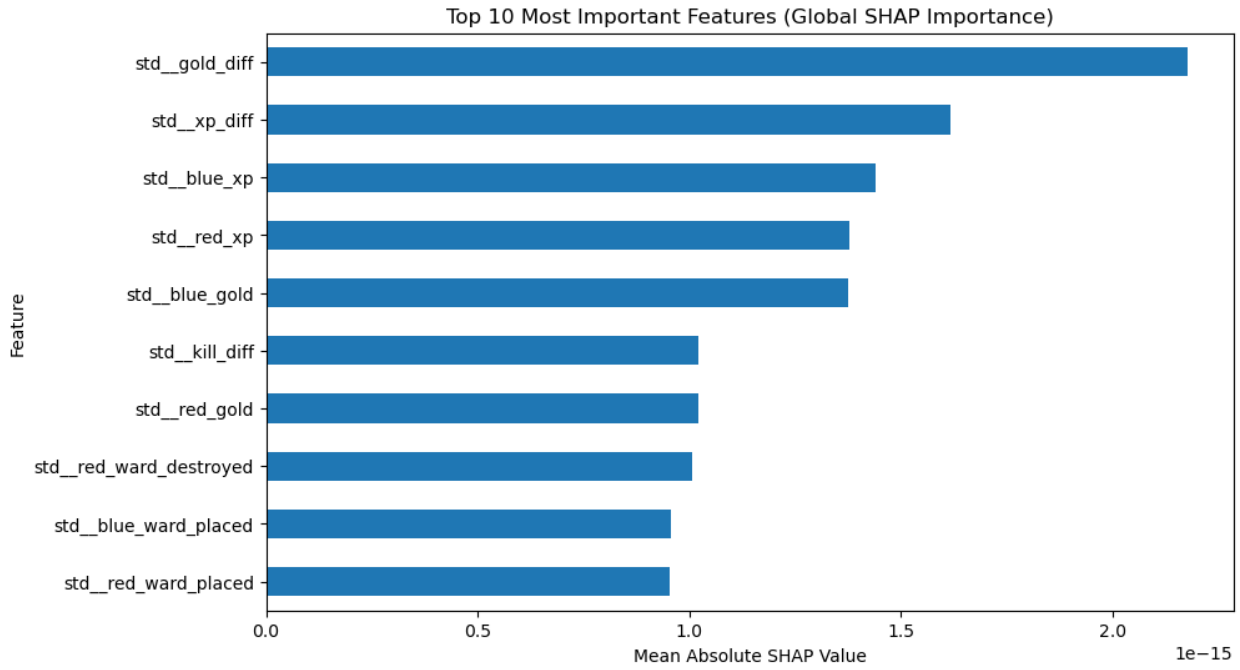

Figure 6: Top 10 Most Important Features for Game Outcome Prediction (SHAP Analysis)

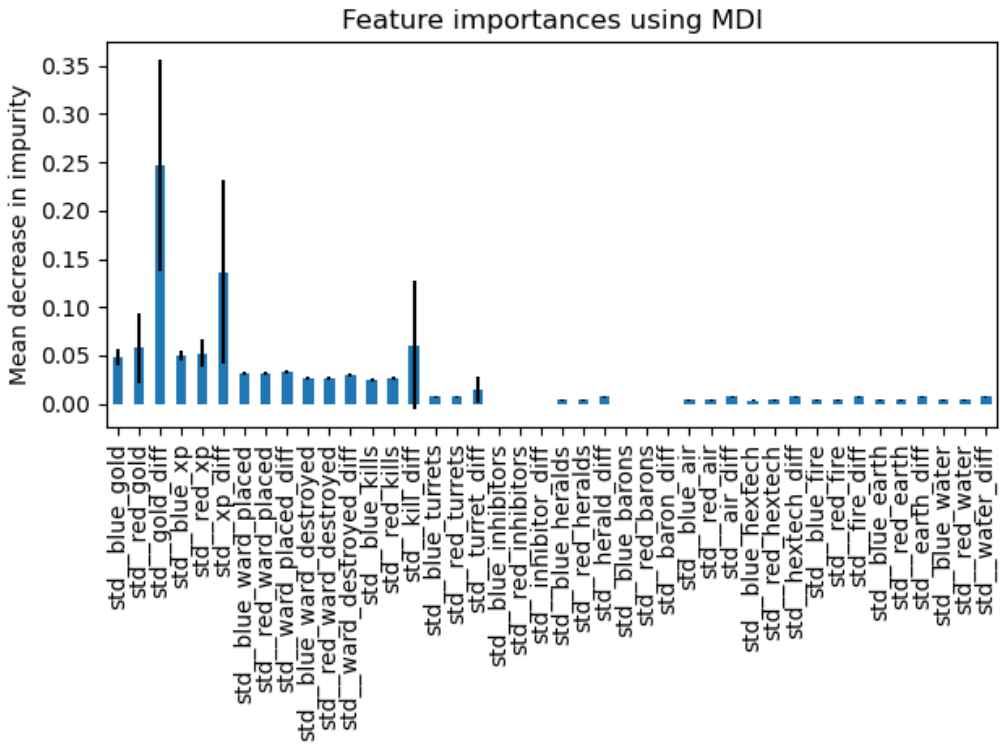

Figure 7: Feature Importance Analysis Using MDI (Mean Decrease in Impurity)

Instead of focus on the global feature importance, the local feature importance can help to understand the features contribution to a certain data point instead of to the whole model performance. Figure 8 is the SHAP force plot for the class of blue team win. This shows that the gold difference, experience difference and the kill champion difference are the three most importance feature for predicting to blue team win, the predict probability is 0.88.

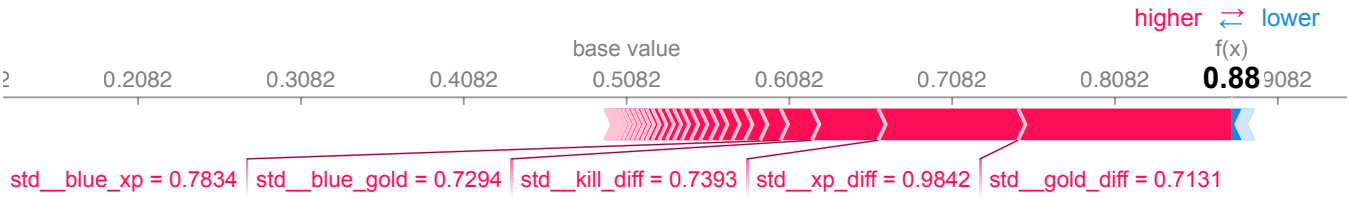

Figure 8: SHAP Force Plot: blue team win

Figure 9 is the SHAP force plot for the class of bule team lose. This shows the same result that the gold difference, experience difference and the kill champion difference are the three most importance feature for predicting to blue team lose, the predict probability is 0.12.

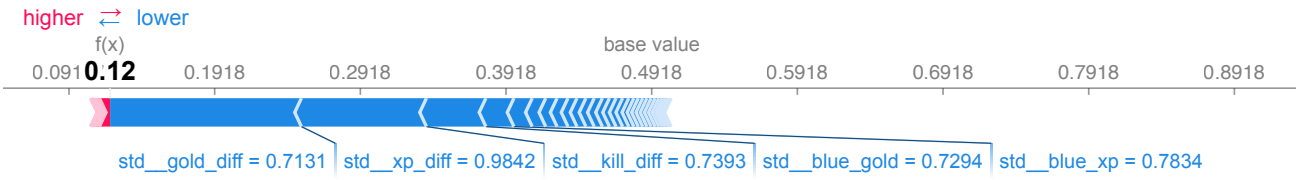

Figure 9: SHAP Force Plot: blue team lose

Outlook

Although this study predict outcome is similar with the previous work[2] . There still a lot of way to improving this model. First, the study should tune more hyperparameters and to try more machine learning algorithms such as support vector machine. Secondly, in terms of data collection, the study should comprehensively collect the data of each rank tier in the game, which can help the model's predictions being more reliable within the different rank tiers. Third, it can be seen from perturbation that only two features have influenced the model, but there are other factors that will affect the perturbation of the model. It is necessary to carry out feature engineering to find new feature that can represent the information that the study miss.

Reference

1. Princic1837592. (2022a, July 18). *League of legends V12.10 diamond games*. Kaggle. https://www.kaggle.com/datasets/princic1837592/league-of-legends-1210-diamond games?select=timeline_DIAMOND_999.csv
2. Junior, J., & Campelo, C. E. (2023). League of Legends: Real-Time Result Prediction. arXiv preprint arXiv:2309.02449.