Prediction and Recommendations for League of Legends Tournament

David Liszewski, Tatdanai Asavamongkolkul

December 2020

1 Problem Motivation

The eSports industry has been growing and continues to grow rapidly. In 2016, viewership had reached 335 million people and is expected to reach an audience of 557 million in 2021 [1].

League of Legends (LoL) is a team-based strategy game where two teams of five powerful characters (champions) challenge each other to destroy the other's base. There are 150 different champion options available for each player.

LoL is widely considered the most popular eSports game, leading all eSports titles in total hours watched and peak viewership in 2019 [2]. As LoL's popularity has grown, so has the monetary prizes for winning the League of Legends World Championship, the ultimate prize for the best LoL team in the world. The 2018 World Championship had a prize pool of $6.4 million; increased by 200% from the last four years, with the grand prize set at $2.4 million [3].

Through the popularity and high stakes of the game, a better understanding of what controls a match's outcomes could give players an advantage over their opponents. In this project, two following objectives were focused on:
1. Understanding the reasons behind each winning game using interpretable predictive methods
2. Improving champion selection strategy using optimal prescriptive trees

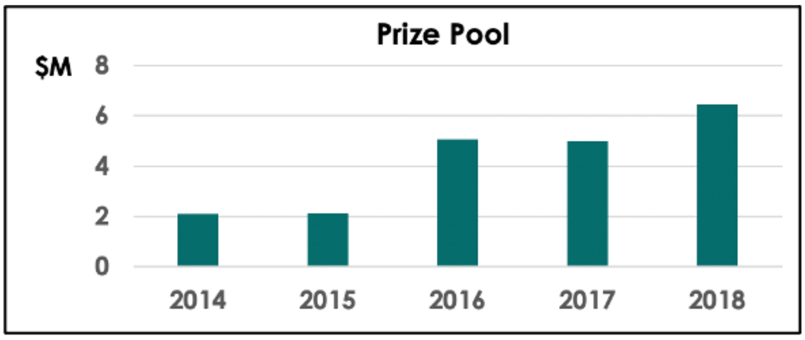


Figure 1: League of Legends World Championship Prize Pool

2 The Data

2.1 Feature Engineering

The initial dataset consisted of match data for professional-level League of Legends matches from January 2017 through August 2020. Past research [4] on a similar problem used limited data that focused on amateur players, which have a wide range of skill sets. In this project, we will focus on the professional level of the game, where the games can be very high stakes.

The data contained information on match metadata (i.e. the date, time, league, etc. of when the match occurred), which players played for which team, player statistics from during the match, the champion they selected, and the outcome of the match (i.e. which team won the match). Extensive data cleaning was necessary before the use of this data. Information on a single match was contained in 12 rows, corresponding to the 10 players and 2 teams. Before assessing features, the data was cleaned so that each match corresponded to one row in the dataset.

We did not want to use the statistics from the current match to retroactively predict the outcome of the match. Instead, we developed features based on past player performance to predict the outcome of the match for Team A against Team B. Past performance statistics for players and champions include important metrics to LoL gameplay such as average KDA (Kills + Assists / Deaths), player win rate, and champion playrate (i.e. how often a champion is used). These metrics were repeated for each player and each champion selected on both teams in the match. We computed the features for both the past 6 and 12 month time windows. Including both the short and long timeframes allows models to account for possible changes in the gameplay over time. The final dataset incorporated around 500 features in total.

It should be noted, that while the dataset contains matches from 2017, we will only focus on matches from 2018 through 2020. Since we are utilizing past performance statistics, the 2017 matches would contain no information on how players performed prior to their matches. Therefore our analysis focused on matches in 2018 through 2020, but utilizes past performance information on the matches conducted from 2017 through 2019.

2.2 Data Splitting

One major drawback to consider when using data accumulated for LoL over the 3 year timeframe is that the game regularly updates features. The updates can directly affect gameplay by changing the speed or strength of an existing champion, or adding new champions. Over time, these small changes can build up and change the current strongest strategy to play, referred to as the meta, for the game.

In order to consider the changes to the meta and research their effect on the game, we proposed three separate prediction problems. The three problems vary on three dimensions:
1. Which data is used for training
2. Which data is used for testing
3. How missing data was imputed

A table of the differences between the problems is shown below (Table 1). The main change between the prediction problems arises in which set is used as the testing set. For problem 1, we attempt to predict for matches that will occur in future years. This approach may not be able to foresee the massive game updates that occur at the end of a calendar year. Problems 2 and 3 attempt to predict the second half of the LoL season, based on the 1st half. This approach allows for training on the first half of a new season, so that the model has seen data corresponding to the newer meta updates. The drawback to this prediction problem is that less data is used, and variability in results could increase.

Table 1: Three Prediction Problems and the Corresponding Data Splits

	Problem 1 (3Y3Y)	**Problem 2 (3Y1Y)**	**Problem 3 (1Y1Y)**
Data Imputation	2018, 2019, 2020	2018, 2019, 2020	2019
Train Set	2018, 2019	2019: January - May	2019: January - May
Testing Set	2020	2019: June - December	2019: June - December

Differences in the data imputation used (methods discussed further in Section 2.3) between problems 2 and 3 are also shown. Problem 2 used data imputed that referenced the full dataset with 3 years of data, while problem 3 used only matches from 2019 to impute missing values. This was done to test the tradeoff between using more data points to assist the imputation methods, or how consistent the data points are (i.e. the matches are from the same game update/year). Essentially, the two are a bias and variance trade-off.

2.3 Optimal Imputation

Across the 500 features, 25% of data points contained missing values, specifically for the numerical features. These features were often missing due to the nature of the features. Given that they were past performance features, new players had no past performance metrics. KNN optimal imputation was applied to impute the missing data. However, the initial trial to impute on the whole dataset was unsuccessful. The original assumption was the dataset contained too many categorical variables while having a relatively large number of observations. Also, these categorical variables contained at least 30 levels each, which could be problematic as the model tried to cluster through them.

To verify this assumption, all the categorical variables were removed. As a result, the optimal imputation estimated finishing time was 36 hours. To improve the computation time further, the opponent team's features were dropped (half of all the features). Consequently, the imputation could be done in 2 hours. For this project, this approach (trial 3) was applied to impute the data. After imputation, the enemy's features were generated again as presented in Figure 5.

Table 2: Imputation time

Trial	Dataset	Features	Observations	Categorical Variables	Estimated time
1	**Full - 3 years**	525	25,000	10 (∼30 levels each)	Unsuccessful
2	**Partial - 3 years**	515	25,000	0	1.5 days
3	**Partial - 3 years**	260	25,000	0	2 hours
4	**Full - 1 year**	515	8,000	0	20 minutes

In contrast, a dataset with only one year of observations, even with 515 features, took significantly less time. For this dataset, the enemy's features did not need to be re-generated.

Notably, we did not include the dependent variable during the imputation to avoid bias in our testing set. After the imputation, the categorical variables and dependent variable were appended back to the dataset for subsequent modeling.

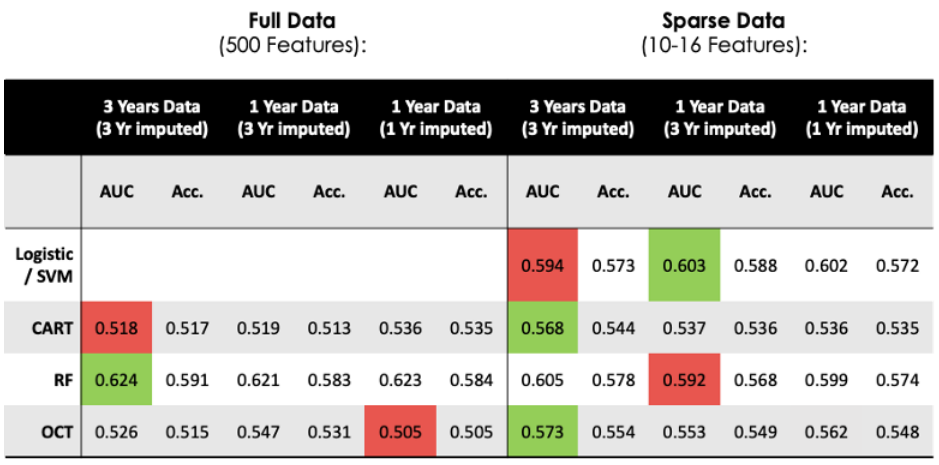
As a result, the imputed data from optimal KNN imputation made total gaming sense, in which players with similar past statistics were imputed with the same set of numbers. In addition, as partially shown in Figure 6, the imputed data were normally distributed.

3 Prediction

3.1 Initial Approach

Our initial approach involved testing a number of models on the dataset. Our focus would be a balance between interpretability and predictive accuracy, as we hope to gain powerful insights for managers of LoL teams. We trained CART, Random Forest, and Optimal Classification tree (OCT) [5] models on the full dataset of 500 features. We repeated these models for each of the three prediction problems described in Table 1.

The left side of Figure 2 describes the out-of-sample results for the initial approach. The more interpretable models performed poorly in all 3 cases, with AUC and Accuracy values barely higher than baseline. We suspect they struggled to narrow down which feature to use for each split. The random forest model performed similarly to past research, which verified our methods [4].



	Full Data (500 Features):						**Sparse Data** (10-16 Features):					
	3 Years Data (3 Yr imputed)		**1 Year Data (3 Yr imputed)**		**1 Year Data (1 Yr imputed)**		**3 Years Data (3 Yr imputed)**		**1 Year Data (3 Yr imputed)**		**1 Year Data (1 Yr imputed)**	
	AUC	**Acc.**	**AUC**	**Acc.**	**AUC**	**Acc.**	**AUC**	**Acc.**	**AUC**	**Acc.**	**AUC**	**Acc.**
Logistic / SVM							0.594	0.573	0.603	0.588	0.602	0.572
CART	0.518	0.517	0.519	0.513	0.536	0.535	0.568	0.544	0.537	0.536	0.536	0.535
RF	0.624	0.591	0.621	0.583	0.623	0.584	0.605	0.578	0.592	0.568	0.599	0.574
OCT	0.526	0.515	0.547	0.531	0.505	0.505	0.573	0.554	0.553	0.549	0.562	0.548

Figure 2: Out-of-Sample Results for AUC and Accuracy

3.2 Optimal Features Selection

Given the poor performance of the original CART and OCT models on the whole dataset, we sought to decrease the size of the dataset by limiting the number of features available for use. For this, we implemented optimal feature selection from the IAI package [6][7]. The goal of this feature selection is to find a small set of features in the data that best predict the outcome [7]. Below is the formulation for optimal feature selection.

$$\min_{w,b} \sum_{i=1}^{n} [l(y_i, w^T x_i + b)] + \frac{1}{2\gamma}||w||_2^2 \quad s.t. \quad ||w||_0 \leq k$$

The formulation minimizes a loss function, and in our a classification loss function such as the entropy, L1, or L2 hinge loss functions. The formulation includes a regularization term and a sparsity constraint. When implementing this method, we used the relaxation approach to speed up computation. The relaxation approach typically finds the optimal solution but does not prove optimality. We proceeded to implement feature selection with each loss function, while cross validating on the value of k, the number of features to include in the analysis for each prediction problem. The performance of the best model for each problem is denoted in the 1st row of the right side of Figure 2.

We then selected the features that corresponded to the best feature selection models, and passed these features to CART, random forest, and OCT models. Figure 2 shows AUC and Accuracy scores for these models on the sparse datasets. Interestingly, random forest performed worse than before, albeit still better than CART and OCT. We believe that this is because random forest models prefer to have a large number of random trees to account for noise, rather than sparsity. We see major improvements for the interpretable models CART and OCT. The out-of-sample results perform closely enough to the random forest's performance, so the interpretable findings of the OCT models deserve merit. Below is the OCT model for the 1 year prediction, with 1 year imputed data.

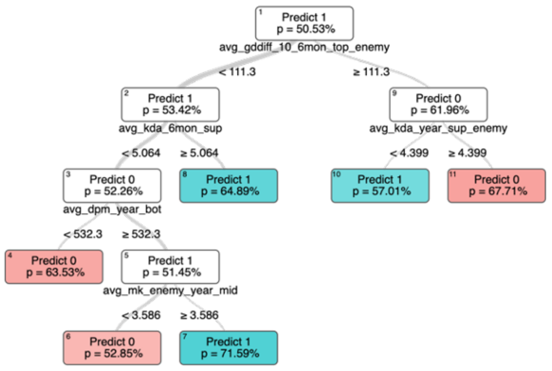


Figure 3: Results from Best Optimal Classification Tree

From this model we can infer takeaways about the factors that lead to winning. We can see that three splits are defined based on enemy data, which a manager may not be able to control. However, the other two factors include the support and bottom lane players for your team. This suggests that improving the performance for the support and bottom lane players, especially in their damage per minute (dpm) and KDA metrics, could lead to further success. These insights allow teams to focus on these areas, rather than the team as a whole.

To change the focus, we want to compare the results of models from the three different prediction problems that we proposed. We find that based on the sparse dataset the 3 year prediction problem, in which we predict on 2020 matches, performs the best. That is contrary to our initial hypothesis that the game changes over time will affect the reasons for winning matches. However, given that the small features used are all player based statistics, we now suspect that players who performed well in the past continue to perform well in the future, regardless of the meta changes.

4 Prescription

Understanding the crucial factors that lead to winning games is vital for team managers to plan at a high-level. However, in this section, we studied the impact of selecting the proper champion for a match, and understanding the factors behind the decision making process using optimal prescriptive trees [5].

For this project, one of the world's best players, Faker, was chosen to try out this method. In other words, the model prescribes whether the mid-lane pro-player (in this case, Faker) should select his main, most utilized champions or any of the other champions before a match.

4.1 Modeling Approach

Two treatments were created for this problem: 1. prescribing main champions 2. prescribing other champions. Main champions are those which a pro-player selects more often due to familiarity, regardless of the games' updates or current meta. The other champions are those that a player uses less often, but they could provide additional situational-specific benefits.

Since the optimal prescriptive tree [8] takes continuous decision variables, our previous binary variable (win/loss) could not be applied. Consequently, another metric that highly relates to the chance of winning is the kill-death-assist ratio (KDA), and this was then used as the decision variable for the prescriptive trees. The goal would be to maximize the player's KDA.

In this project, the optimal prescriptive tree framework from Interpretable AI was applied. Firstly, the data was split into training and testing sest, and prescription maximize function was chosen as the maximum KDA was aimed. Then, grid searching was applied to find the best hyperparameters (i.e. maximum depth). The searching range for maximum depth was 2 - 4 due to the importance of interpretability. In parallel, the prescription factor was manually tested; prescription factor influences the trade-off between mean outcomes and prediction errors. At the end, prescriptive trees with depth 3 and prescription factor of 0.75 were selected.

To evaluate the model, two different approaches were used.
1. Comparison of the mean KDA achieved across all prescriptions on the test set with the mean KDA achieved under the actual treatment assignments observed in the data.
2. The proportion of proper prescribed treatments based on true counterfactuals on a synthetic dataset.

4.2 Results

The mean KDA achieved from the prescriptions was 12.1, while the mean KDA from the original observed data was 7.0. In other words, using the prescribed treatment for champion selection for Faker could increase Faker's KDA rate by 72.9%.

Not only was there a significant improvement in KDA, but the tree was also easily interpretable. This allows managers to understand the reasons and factors behind each treatment. Moreover, only a few crucial factors were selected, leading to an actionable implication. The factors behind choosing treatment included the enemy's top-lane champion, enemy's jungle champion, enemy's mid-lane champion, and our top-lane champion. This made total sense, as mid-lane players typically depend on their top-laner to help them throughout the game while taking into account the enemy's mid-lane matchup and enemy's jungle which normally ganks (or intrudes) the mid-laner.

Another evaluation metric was accuracy based on true counterfactuals. The synthetic true counterfactuals were created using random forest. The result shows that the proportion of matching prescribed treatments was 55%.

However, we believe that using true counterfactuals is not the best way to evaluate the model for this specific problem because the data does not contain the true counterfactuals, and using synthetic data may not represent the actual KDA based on both treatments.

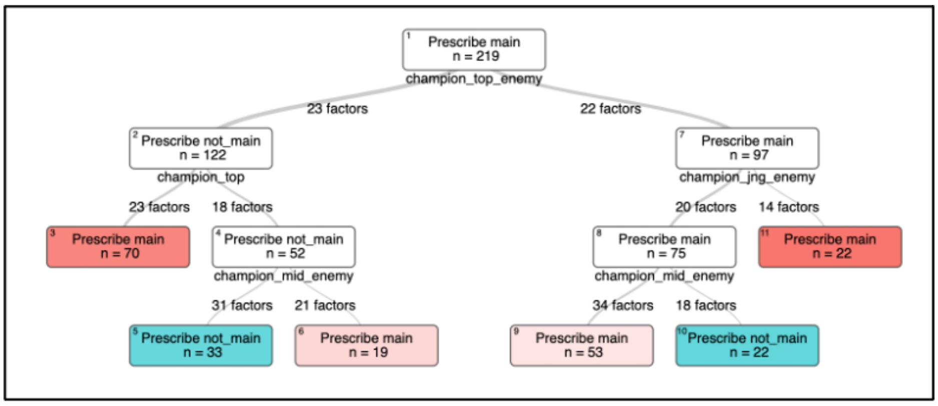


Figure 4: Results from optimal prescriptive tree

5 Implications

Sparsity and robustness can improve the performance of interpretable models in AUC by around 7%. Using interpretable models, we find that we can inform team strategy on how to prepare for matches. We recommend focusing on improving the overall performance of support and bottom-lane players to increase the likelihood of winning matches.

Optimal prescriptive trees effectively maximize the mean outcomes and allows users to interpret the decision process easily. In this project's context, LoL's mid-lane pro-players can effectively select the champions based on a few significant factors, i.e. enemy's top-lane, jungle, mid-lane champions, and our top-lane champion. This potentially increases KDA by 70%, which is the crucial factor leading to wins.

References

[1] Influencer Marketing Hub. *The Incredible Growth of ESports [+ ESports Statistics]*. URL: influencermarketinghub.com/growth-of-esports-stats/ (visited on 01/06/2020).

[2] Adam Fitch. *The Most Watched Esports Events of 2019*. URL: esportsinsider.com/2020/01/esports-viewership-2019/ (visited on 03/31/2020).

[3] Jerome Heath and Sam Nordmark. *The 10 Largest Prize Pools in Esports*. URL: dotesports.com/general/news/biggest-prize-pools-esports-14605. (visited on 09/03/2020).

[4] Jihan Yin. *Predicting League of Legends Ranked Match Outcomes with Machine Learning*. URL: https://hackernoon.com/league-of-legends-predicting-wins-in-champion-select-with-machine-learning-6496523a7ea7 (visited on 2018).

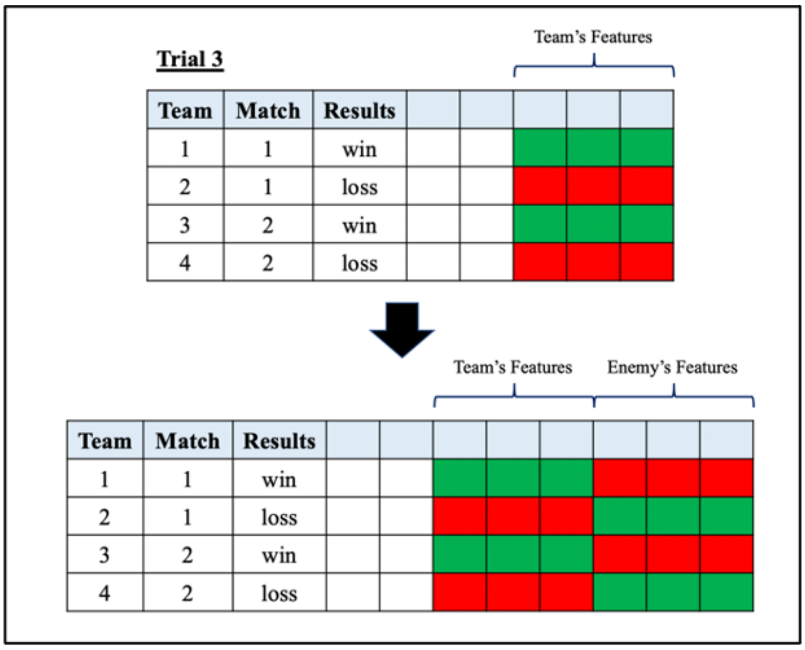
[5] Dimitris Bertsimas and Jack Dunn. "Optimal classification trees". In: *Machine Learning* (2017).

[6] LLC Interpretable AI. *Interpretable AI Documentation*. URL: https://www.interpretable.ai (visited on 2020).

[7] Rahul Mazumder Dimitris Bertsimas Angela King. "Best Subset Selection via a Modern Optimization Lens". In: *stat.ME* (2014).

[8] Nishanth Mundru Dimitris Bertsimas Jack Dunn. "Optimal Prescriptive Trees". In: *INFORMS Journal on Optimization* (2019).

Appendix



Trial 3

Team	Match	Results			Team's Features		
1	1	win			🟩	🟩	🟩
2	1	loss			🟥	🟥	🟥
3	2	win			🟩	🟩	🟩
4	2	loss			🟥	🟥	🟥

Team	Match	Results			Team's Features			Enemy's Features		
1	1	win			🟩	🟩	🟩	🟥	🟥	🟥
2	1	loss			🟥	🟥	🟥	🟩	🟩	🟩
3	2	win			🟩	🟩	🟩	🟥	🟥	🟥
4	2	loss			🟥	🟥	🟥	🟩	🟩	🟩

Figure 5: Feature engineering after imputation of trial 3

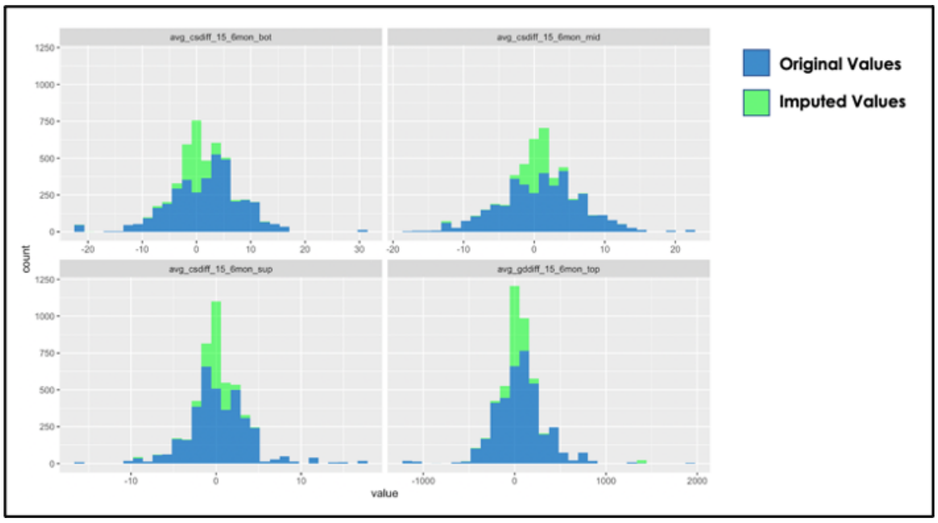


Figure 6: Feature engineering after imputation of trial 3