Predicting Wins in League of Legends

Yuchen Cao, Mike Jin, Millie Mao, Louis Raison, Nicole Zou

Project Description

League of Legends is a multiplayer online battle arena video game. Players assume the role of an unseen "summoner" that controls a "champion" with unique abilities and battle against another team of champions. The goal is to destroy the opposing team's "Nexus", a structure that lies at the heart of a base protected by defensive structures. **Each League of Legends match is independent**.

Our goal was to see if at the beginning of a game, when the champions are picked, we were able to predict which of the teams is the most likely to win the game.
The essence of the problem is a binary classification task. We denote 1 as Team 1 winning and 0 otherwise. By outputting a probability of winning for every game, our model could therefore be used to tailor **odds for bets**.

Game structure

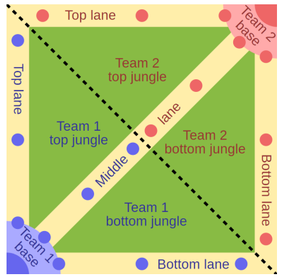

Figure 1: game map.

10 players are divided into 2 teams (blue or red) in the main game. The map is comprised of three lanes — top, middle, and bottom. The green area is the jungle.

- Five positions to choose: Top, Jungle, Mid, ADC, Support, that play different locations on the map
- all champions start off fairly weak but they can increase their strengths by accumulating items and experience over the course of the game
- the goal is to push toward the other team's territory by destroying their turrets and finally the base
- analysis at the beginning of the game: which team has the upper hand? Which players master their champions the best? Which combination of champions is likely to work the best?

Potential Outcomes

The project has a lot of applications, as much for game analysis as for decision-taking on game-related questions. A better understanding of how the game is won could include the following information:

- What are the positions that matter the most?
- Is team play significant?
- Which are the best champions and combinations of champions?
- provide a more objective view on what is important in a team

At the same time, we also have analysis that could directly help decisions in and out of the game:

- Choosing the champions is done in an order (one at a time from each team). The last player to choose could optimize their pick to improve their odds of winning.
- A good predictive model could be used by League of Legends, to make the games more balanced
- A good predictive model for the game that provide a probability could be used to determine odds in *bets*. If the model is relevant for championships, bets could be placed at the beginning of a game, with returns based on the predictions.

Data Engineering

The main part of the work was to work on extracting the data and building features. We included:

- data on the structure of each match (which champions were picked, which players played what roles, etc.),
- the past records of every player, to give statistics (performance with given champion, etc.).

League of Legends provides an online API (Riot Games (2019)) that give access to data on players and games, and we used different of its access points to get players' matches and statistics history.
We took games records for the top 500 players, and took divided the match results in two to prevent "cheating":

- the first part, that we used to compute statistics for every player, on every champion and globally
- the second part, that we used for training and testing purposes

We chose some significant statistics and features that could enable us to better understand what the most important parameters are in a game.

- KDA (kills, deaths, assist) is a measure of the average number of kills & number of assistances (when a player helps a teammate to win a fight), divided by the number of deaths, per game. It is champion-specific
- Win rate for a given champion, for every player
- Champion selected for this game
- Average teamplay scores (vision, turret kills and objectives)

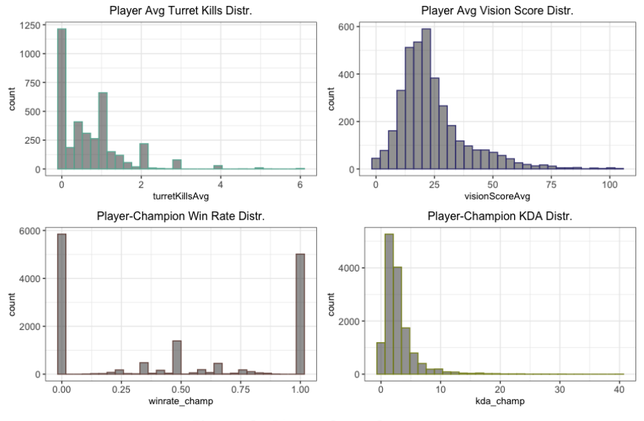

Figure 2: Player Stats Summary

Final structure of the data:

Win	Champ1Team1	Champ2Team1	···	ChampNTeam2	Statistics players
1	0	1		0	···
0	···				
⋮					

Models and Results

Binary classification task, we used *Logistic Regression*, *CART* and *Random Forest* to predict the winning outcome.

We partitioned our dataset with training / validation / test = 0.65 / 0.15 / 0.2. For each of these three model classes, we constructed 3 separated models using:

- ("Naive model") All variables available
- ("Top 20 model") 20 variables via feature selection with best ANOVA F-Values
- ("α model") All variables except for win rates and number of games, whereas engineering a new α variable for each player defined as $\alpha = (winrate - 0.5) * number.of.games.played$

	Logistic Regression	CART	Random Forest
Naïve model	0.7882	0.7304	0.8292
Top 20 model	0.6793	0.7037	0.7420
***α* model**	0.7264	0.6879	0.8066

Table 1: Out-of-sample AUC results

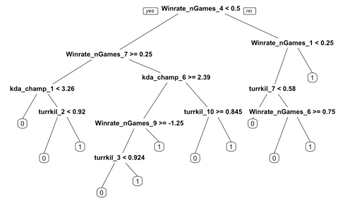

Figure 3. CART Tree Visualization
Teams: 1 to 5 - Team A; 6 to 10 - Team B.
Position: 1/6 - Top, 2/7 - Jungle, 3/8 - Mid, 4/9 - ADC, 5/10 - Support
Leaf: 1 - Team A wins; 0 - Team A loses.

Let's interpret the tree from Team A's perspective (1 - 5). In terms of position, we can see that ADC and Top have critical impact on determining the outcome of the game. It makes sense since under this patch, ADC and top champions are buffed and therefore are more likely to carry the game if they have strong skills (high win rate). When ADC's win rate is low, the Top player needs to have high KDA in order to win. Turret kill is also a very important variable. This can be justified that turret kill not only means the champion is winning the lane but also implies the whole team is roaming.

Conclusions

- games in League of Legends solo queue are easy to predict
- teams should be balanced but they are very often biased towards one team or another
- our models are not using the champions variables much so we cannot really do last-champion-selection; only the ability of a player on a specific champion matters
- for the same reason, championship matches might need their own models

References

Riot Games (2019). Riot Developer Portal, APIs.