Kevin Sun

9 Followers About

League of Legends Data Analysis: When is the game really “over”?

Kevin Sun Just now · 12 min read ★

Players are split on surrendering; the game continues

Disagreements about surrendering come up in every ELO. Some players want to surrender as early as 5 minutes while others — especially in ranked matches — are willing to duke it out even if a couple of their teammates disconnect.

Arguments about surrendering can get heated to the point where teammates will flame each other, while others just outright quit or AFK. In certain cases, players will decide

However, are they jumping the gun? When is it really worth the time to keep pushing for a win and when is it time to wave the white flag?

Data Overview

We sourced our main datasets from Kaggle. The first Kaggle dataset compiled League of Legends data over **7,620 professional games** across several regions from 2015–2018. We also wanted to extend our analysis beyond professional games to Ranked Soloqueue, where many more records can be found. This dataset in question has compiled League of Legends data over **100,800 games at the highest level in the Korean region (challenger and master)**. For example, a record (game) contains features like which champions/characters played, gold, minions killed, as well as potential output/response variables such as win/loss.

We also supplemented this dataset with the official Riot API, which allowed us to scrape static data to link player ID/champion ID with the actual character names and other account information. We used the API to add more records to our game dataset, and generate new testing instances to validate our model.

The majority of our analysis will be focused on the first Kaggle professional dataset, but then we will extend the analysis to the larger Kaggle Ranked Soloqueue dataset. The primary difference is that the latter does not have professional features such as team and specific pro player information — plus, the bans and picks aren’t as strategically done as professional matches. The Riot API is primarily used as a fun way to test our models trained on the larger Kaggle Ranked Soloqueue dataset on live test instances, which can later potentially be useful as a web app for all League of Legends players to gain more insight into their games.

Datasets Sources:

League of Legends

League of Legends(LOL) - Ranked Games 2020

challenger,grandmaster,master 108,000 game data (Riot, korea, 2020)

www.kaggle.com

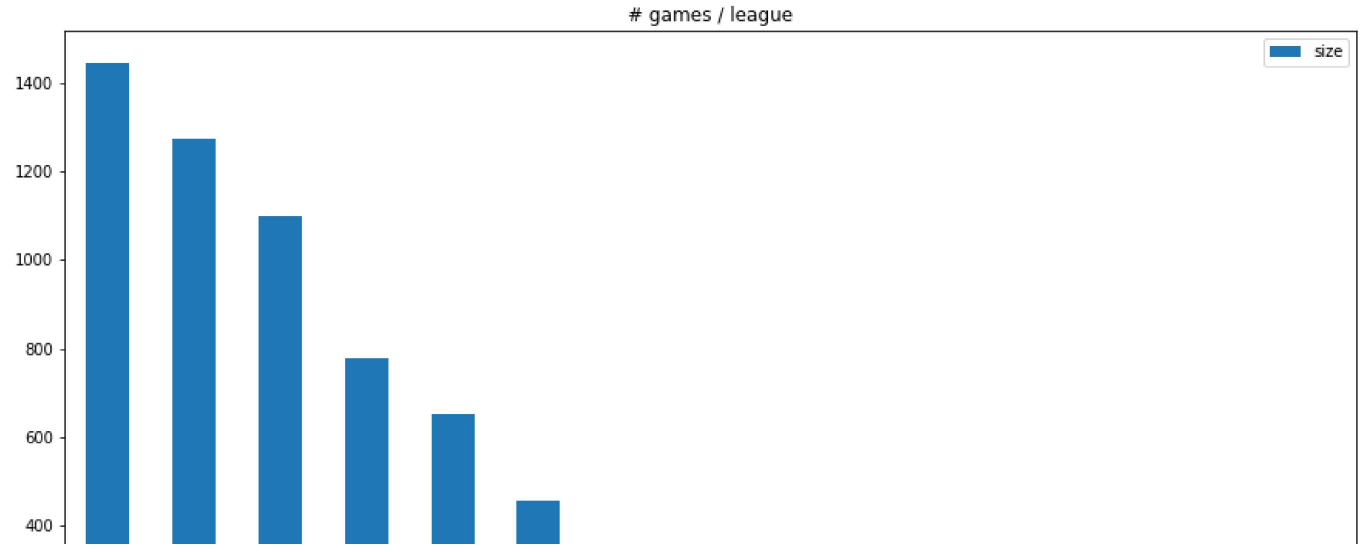
Notebook Link:

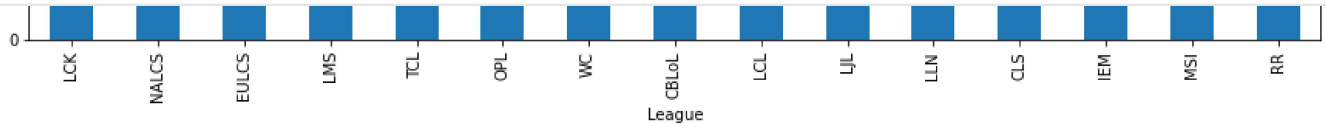
ksun0/league-of-data

Permalink GitHub is home to over 50 million developers working together to host and review code, manage projects, and...

github.com

Exploratory Data Analysis on Top Tier/Professional Matches



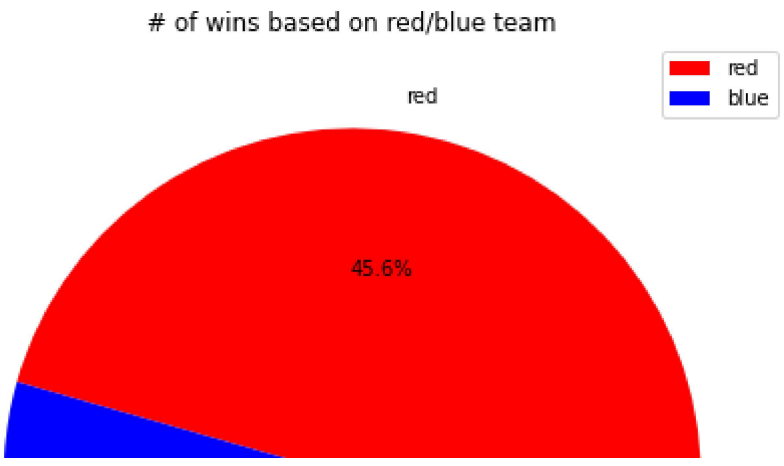


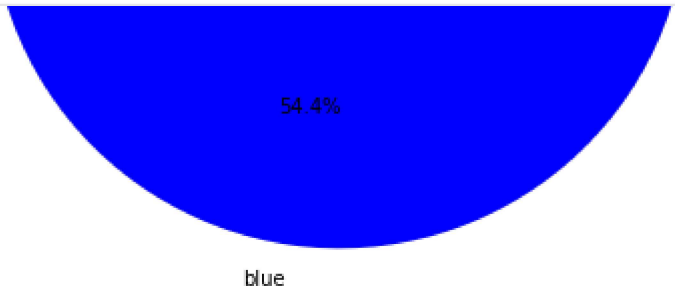
LPL was not included in the data, so it makes sense that the three leagues with the greatest number of matches are the remaining in the Big 4: LCK (Korea), NA LCS (North America), and EU LCS (Europe).

For the sake of this project, we are interested in the aggregate data of all leagues since roughly the same ‘meta’ is shared by all leagues at a given time. Still, to take into account the individual differences between the leagues, we’ll keep it as a feature. We’ll do a bit more analysis of the dataset before cleaning it up for ML.

For some initial data wrangling, we dropped all rows with any missing values — only 38 rows out of 7620 — removing the need to impute or perform any special techniques to deal with these missing values. For now, we stripped temporal data and other non-game related data. The target “y” variable was transformed into a boolean “result” variable, with a value of 0 indicating that Blue side won, and a value of 1 indicating that Red side won.

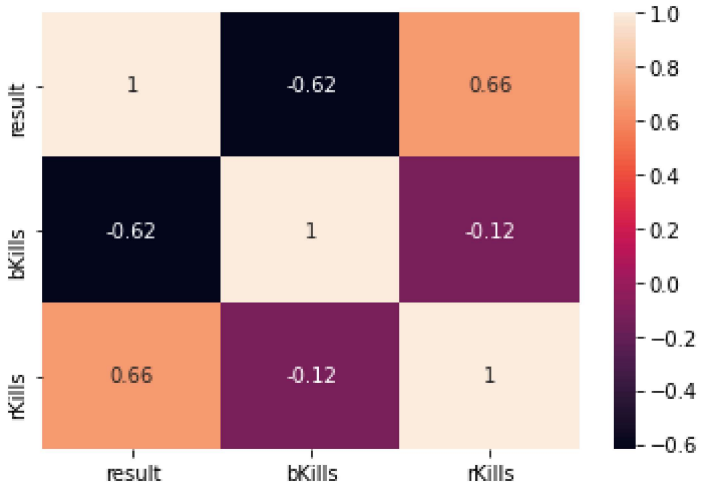
One interesting thing about the professional League of Legends scene is that, depending on the meta, either the red or blue team is more likely to win. This is usually because the blue team gets to pick 1 champion first, and then the red team gets to pick 2 champions — depending on which champions are OP, either team can be more advantageous in terms of its picks.



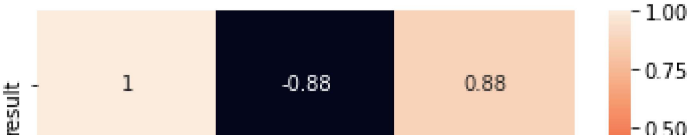


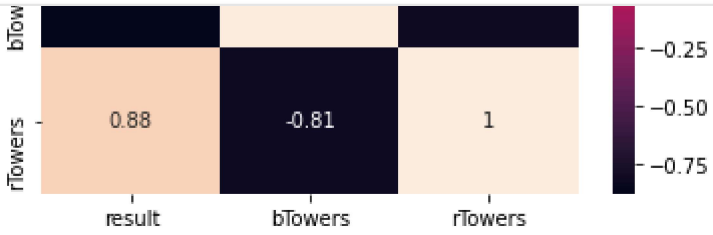
Before any fancy EDA and ML modeling, we notice that the blue side has a tiny advantage of winning: probably due to pick order (regardless, jungle pathing is usually started on bot-side buff so the actual side on the map probably doesn’t matter, though the blue side has to hop over the wall for dragon). But we won’t dive too much into this. We also notice that there’s not much of a class imbalance, so we don’t need to do any techniques here, e.g. resampling (undersampling) or models that incorporate a penalty.

Some features are intuitively more correlated with victory (e.g. number of kills, objectives, etc.), and we use the following correlation matrices to estimate that correlation.

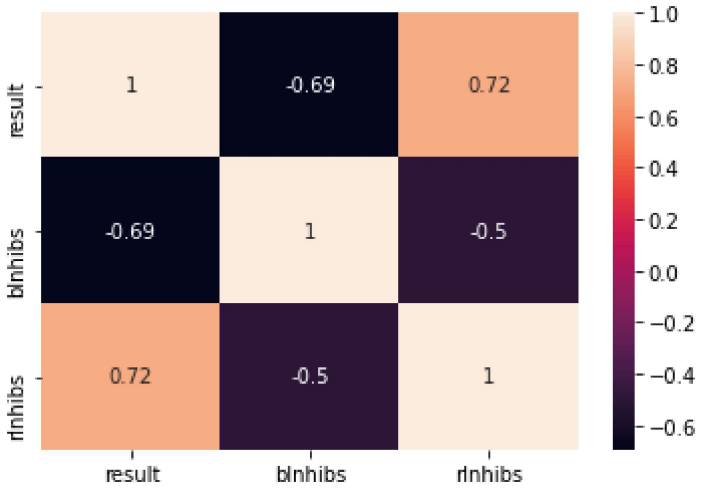


Kills are the main way of getting gold, which is required for items. The more items you have, the stronger (either more damage, more health, etc.) you get, and ultimately the higher the chances of you winning.



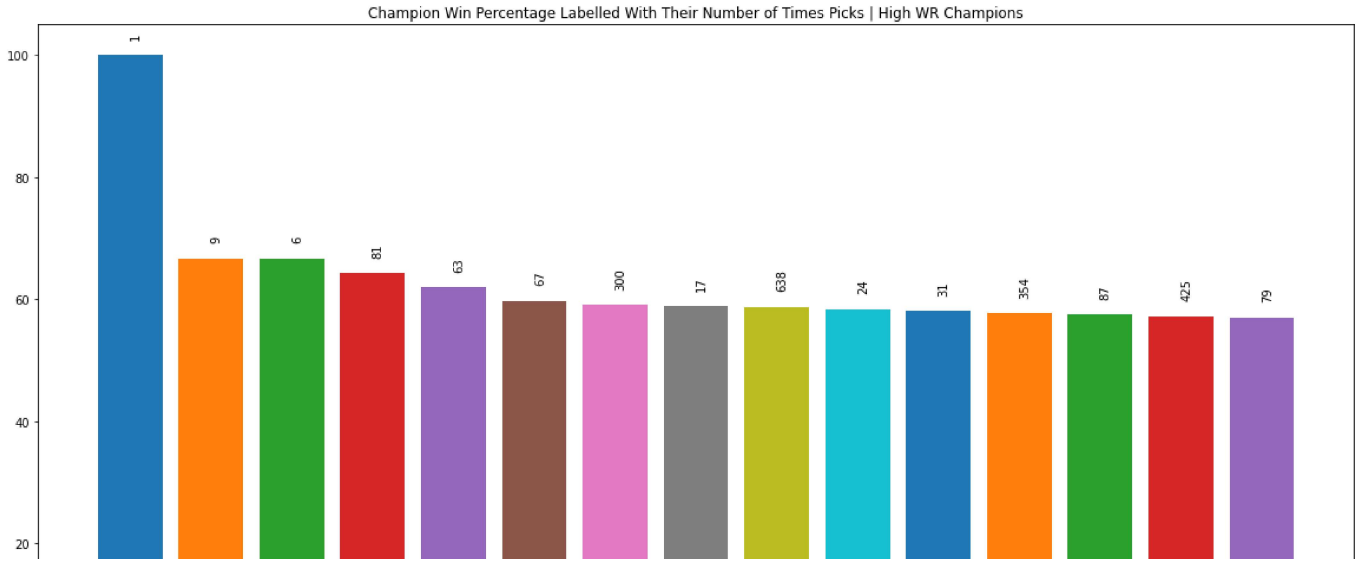


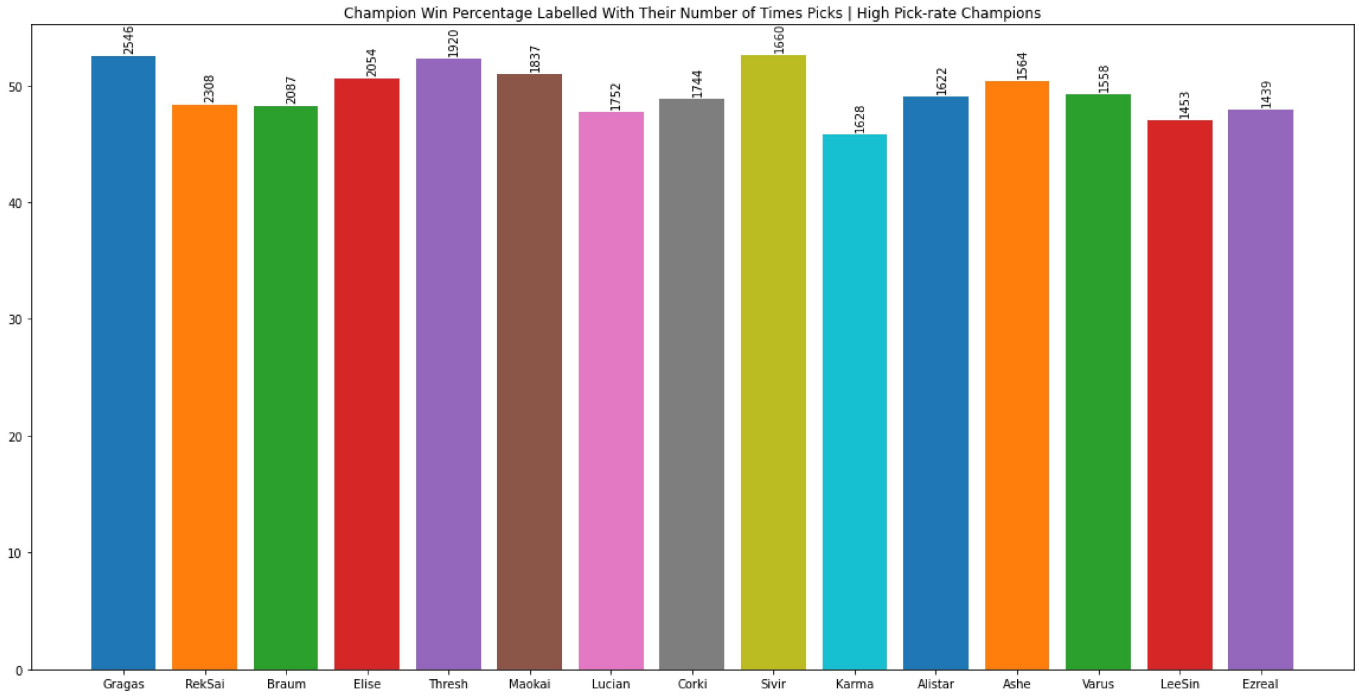
After all, League is a game of destroying structures — you win if you destroy the final structure called the Nexus. Before the Nexus becomes targetable, each team must destroy towers and inhibitors as well. In other words, it’s highly likely that the number of towers destroyed is correlated with victory as shown above.



The correlation is a bit lower for inhibitors, which is counterintuitive as they’re more crucial to victory (inhibitors can only be destroyed after all towers in that lane are destroyed) — but we think this is attributed to the fact that in 2015 and 2016, pro players were able to surrender before any of their inhibitors got destroyed.

Team Composition and Champion Picks/Bans



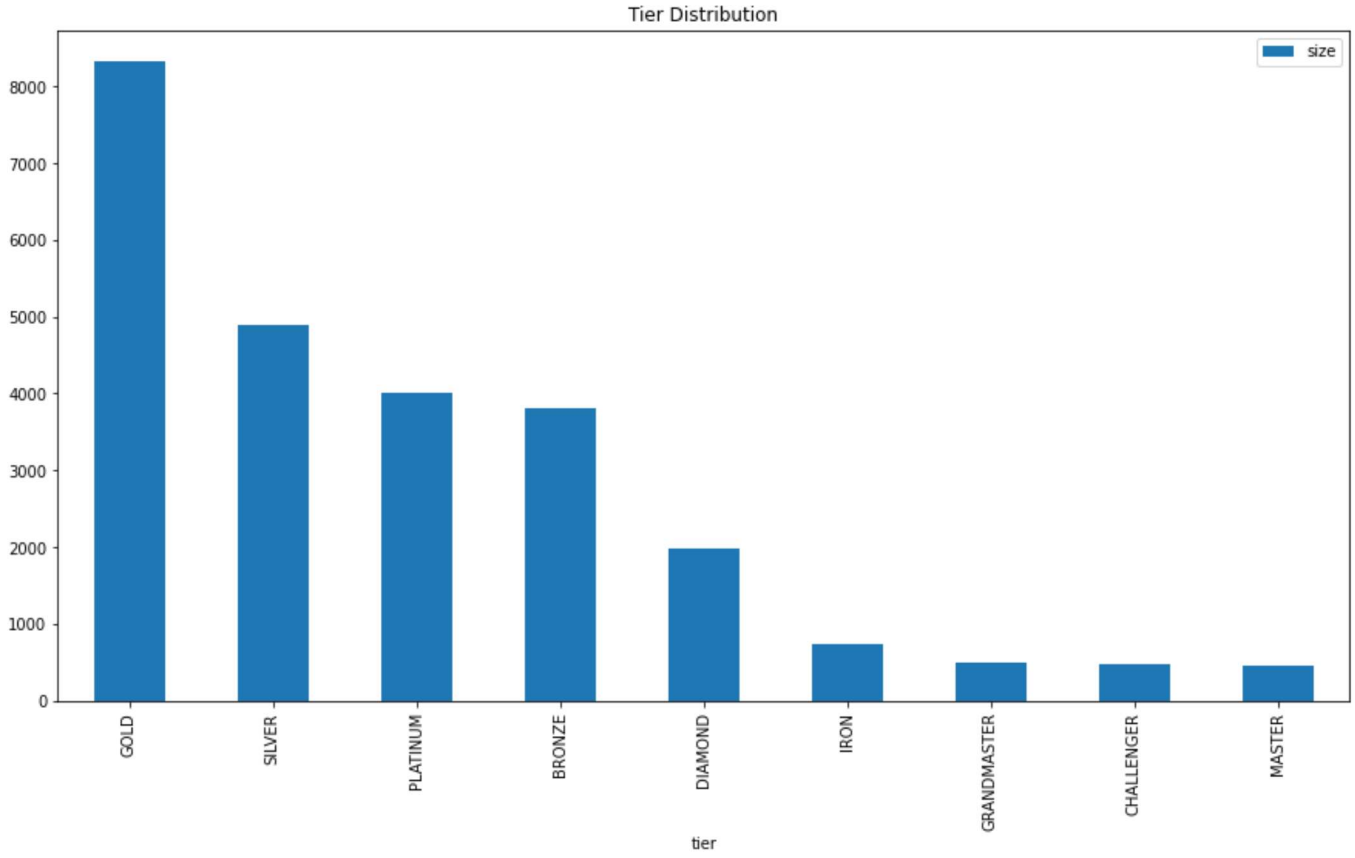


Looking at the two graphs above, we can conclude that high win rate champions (e.g. Tryndamere, XinZhao) do not necessarily have high pick rates, so many pro players who pick these champions likely specialize in them as “pocket-picks” or one-tricks. Picking a specialized champion that the enemy has no experience against can perhaps improve the odds of winning. Furthermore, the more a champion gets picked, the win rate probably stabilizes to a lower, more reasonable percentage around 50–60%. In general, pros tend to prioritize more popular jungle champions such as RekSai — a champion in the important jungle role to impact the rest of the team and provide information with tremor passive. Next, ADC and support champions were prioritized such as Braum, Thresh, and Lucian.

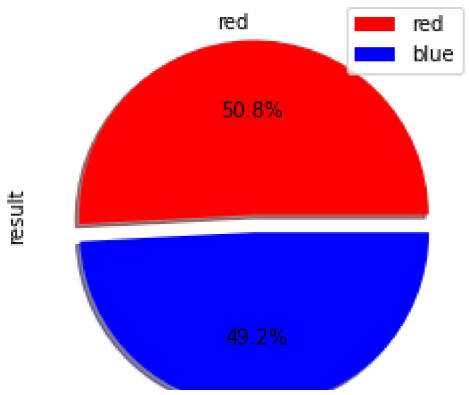
Exploratory Data Analysis on Ranked Soloqueue Matches

Our second Kaggle dataset with many more records from ranked soloqueue matches (at the highest tiers of challenger, grandmaster, and master ranks) was wrangled similarly to the first Kaggle dataset of professional games, The main rationale for using this dataset was to have a lot more datasets and see if features impacting wins are common despite the professional difference.

summoner spells, match outcomes, season, and game IDs. 50 summoners were randomly selected from na.op.gg’s leaderboard feature. Scraping all their ranked solo and ranked flex games resulted in the following:



Gold is the most common tier in the dataset, followed by Silver, Platinum, Bronze, Diamond, Iron, Grandmaster, Challenger, and Master. This distribution is a rough equivalent to what the actual tier distribution is in the game and can therefore provide a model that includes the broad representation of tiers in matches.

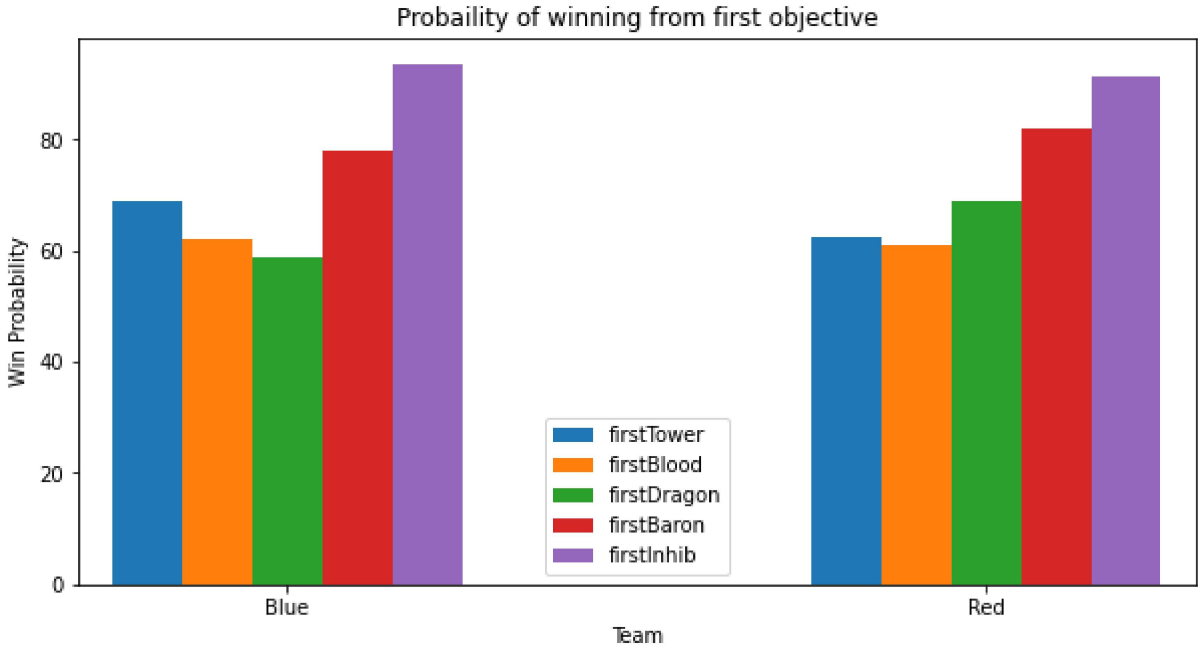


Moreover, in the 2500 ranked games analyzed, 50.8% of won games were from the red team and 49.2% were from the blue team. This alone indicates that sides aren’t necessarily crucial in determining ranked match outcomes.

Feature Engineering (For Professional Games)

Before feeding our cleaned data into any models, we one-hot encoded them since we had mostly categorical variables. Additionally, we used StandardScalar() to scale the “X” features, because the distributions of some numerical features such as year and game length were drastically different from the indicator dummy features. Scaling is also important for models that are not invariant to scale, including penalized linear approaches and methods that utilize gradient descent, such as our logistic classifier. We decided not to utilize PCA to keep feature interpretability.

One particular set of features we paid special attention to was the game objectives, including towers, kills, dragons, barons, and inhibitors. The dataset gave them to us as an array of timestamps, so we extracted a custom numerical feature of which team got the first of each objective, and the total count or number of each objective type.



From the graph above, we see that the probability of winning is substantially increased when either team manages to get any major objective first. First inhibitor and first Baron

effect and subsequent win.

result	1.000000
firstTower_red	0.315924
firstDragon_red	0.281493
firstBlood_red	0.230783
redSupport_Wolf	0.077194
redADC_Bang	0.074932
redTeamTag_SKT	0.074932
redMiddle_Faker	0.068686

Furthermore, from plotting the correlation matrix between every pair of features and the response “result” variable, we see that our custom “first objective” features had the highest correlation to the result value of 1, indicating that the red team won. Furthermore, we see the importance of player and team features such as Wolf, Bang, and Faker, who were dominant players on the dominant SKT team during the time period when the data was collected.

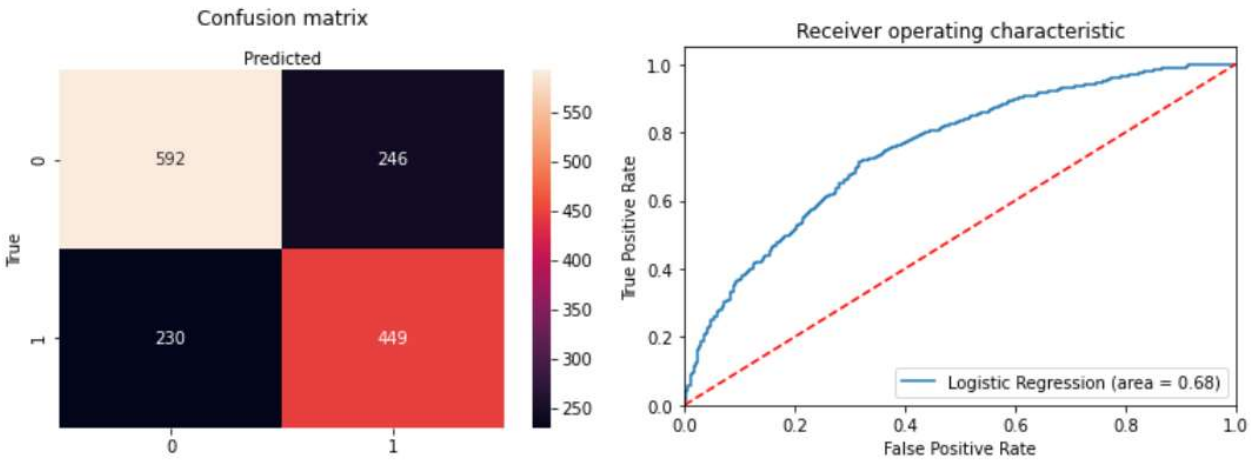
Initially, **our trained model on the features obtained around 95% accuracy**, which seemed too high for any sports/esports prediction model. We realized that it did not make sense to include crucial late-game objectives such as firstBaron and firstInhib to our model, because those indicators pretty much dictate the result of the game anyway. Super minions spawning, global gold and stats, etc. in the late-game are insurmountable leads to overcome, and predicting the winner of a game with these late-game features does not make much sense since throws/comebacks are rare. **Our main hypothesis is that pre-game features (e.g. team composition and bans), as well as early game features (e.g. first tower, blood, and dragon), are enough to accurately predict a team’s chance of winning.**

Our final, full-numeric dataset for machine learning had 5673 columns with **5672 features** due to the large cardinality of categorical variables.

For our modeling, we primarily use the f1 score as a metric to evaluate our models, because it is loosely correlated to a test’s accuracy and considers both precision and recall. Since our professional game dataset had a lot fewer than 100K data points and <10K features, we decided not to utilize any deep learning or Spark/MLlib.

After performing appropriate train-test splits and employing a variety of validation and hyperparameter tuning techniques, we found that out of Logistic Regression (LR), K-Nearest-Neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF), and XGBOOST (XGB), accuracies ranged between 60–75% on the test/validation set. Our top-performing models performed at the same level on the test/validation dataset, and ensembling the models by averaging the predictions of the four top-performing models also resulted in ~70% accuracy.

	precision	recall	f1-score	support
0	0.72	0.71	0.71	838
1	0.65	0.66	0.65	679
accuracy			0.69	1517
macro avg	0.68	0.68	0.68	1517
weighted avg	0.69	0.69	0.69	1517



Logistic Regression

0	0.70	0.66	0.68	838
1	0.61	0.65	0.63	679
accuracy			0.65	1517
macro avg	0.65	0.65	0.65	1517
weighted avg	0.66	0.65	0.65	1517

XGBoost trained on professional matches

Interestingly, even though it adds boosting on top of the bagging that RF employs over traditional decision tree classifiers, the results were a little worse. Perhaps a larger gridsearch is required to find the optimal parameters for more complex models.

Can a Ranked, Soloqueue Game be Decided in Champion Select?

Training models on the ranked, soloqueue Kaggle dataset, with more training instances, resulted in similar results as the models trained on the professional dataset. Even though there are more training instances, similar accuracies can be attributed to the fact that important features such as the professional players and teams involved in the match do not exist for regular League of Legends games. Without those professional team-specific features, it is harder to determine which team is going to win.

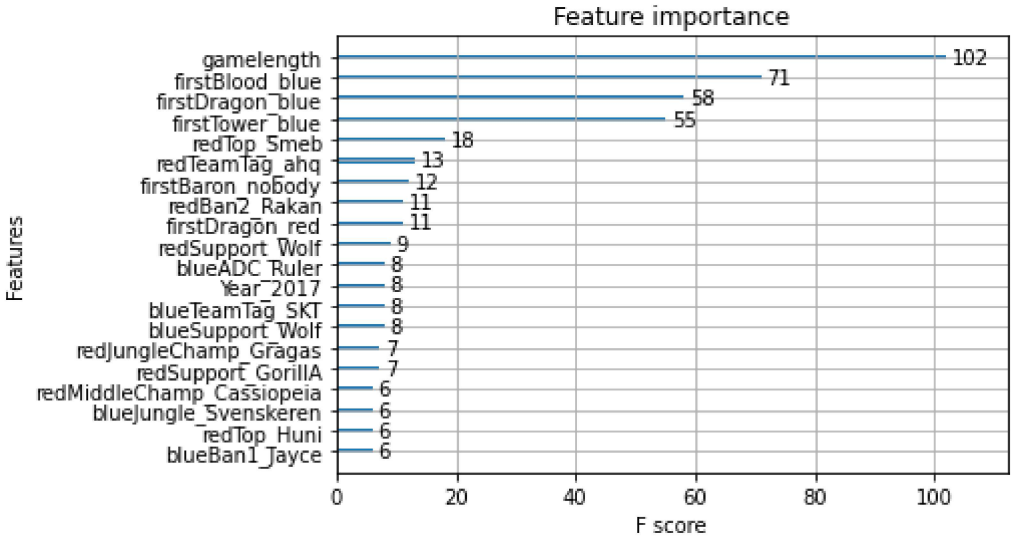
F1 score and accuracy score for training set: 0.7022 , 0.7152.
F1 score and accuracy score for test set: 0.6661 , 0.6774.

	precision	recall	f1-score	support
0	0.69	0.64	0.67	4312
1	0.67	0.71	0.69	4322
accuracy			0.68	8634
macro avg	0.68	0.68	0.68	8634
weighted avg	0.68	0.68	0.68	8634

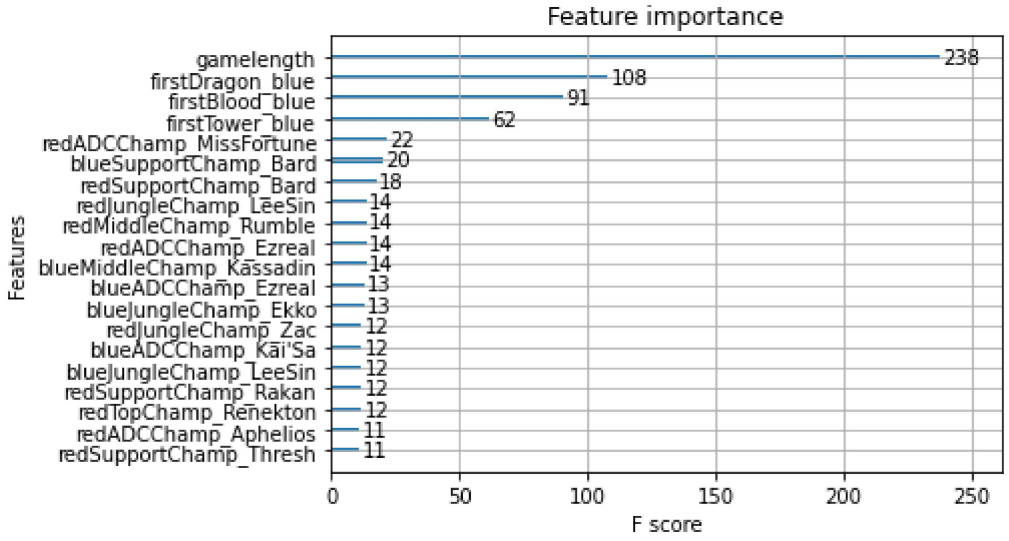
XGBoost trained on our larger soloqueue results

So then when is a game’s result “decisive”?

However, from just the pre-game champion picks and early objectives such as kills and dragons, on both the professional and ranked soloqueue datasets, our models can predict which team is going to win 70% of the time.



Feature Importance of our XGB model trained on professional games



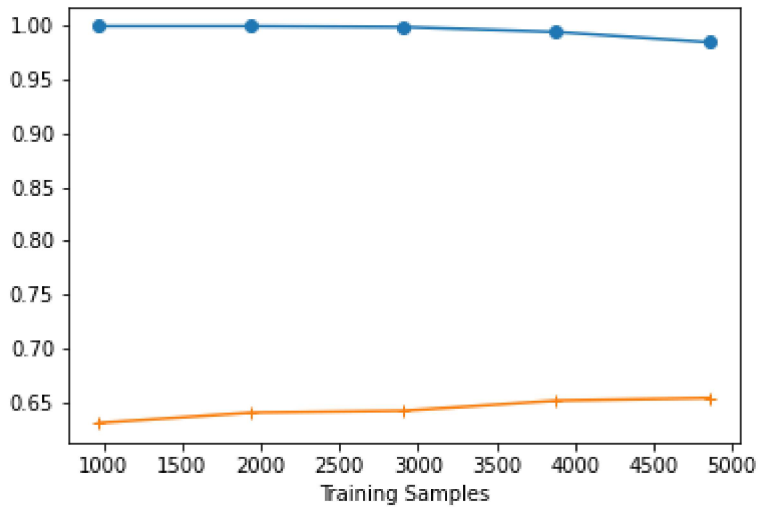
Feature Importance of our XGB model trained on soloqueue games

Using the feature importance plots from our XGBoost model, we see that game length is the most important feature in decision making, likely because longer games capture late-game objectives that can accurately predict which team is going to win. As expected, our custom first objective features are important attributes in decision making. Interestingly,

professional games, we have the additional features corresponding to specific dominant teams and players, including Smeb, Faker, and SKT. These features reflect the meta, overpowered champions, and best performing players when the data was collected, so perhaps a team can decisively win with a high probability if they have these features.

In essence, in both professional and soloqueue matches, the game is not necessarily decided in champion select, and, as such, assuming no outstanding factors, such as toxic teammates or strange, out-of-meta champion picks, players should not dodge games based on draft picks alone. However, certain champion picks or whether you are a top-performing team/player may slightly increase your probability of winning.

Future Considerations



The learning curve from our Logistic Model trained on professional games

From this sample learning curve graph from our logistic regression model on professional matches, we can conclude low bias, and the large gap between the training and validation learning curves mean high variance and an overfitting problem. The validation curves do seem to be slightly increasing as the training accuracy decreases, so more training instances and more data, in general, are very likely to lead to better

Our KNN model can be extended to a recommendation system to help players pick champions surrounding their current team picks and bans. In this case, the model will output the closest training instance in terms of Euclidean distance, so players can see past games where similar team comps have won.

Additional features that are important but difficult to calculate from the data and API provided include in-game movement, jungle proximity, and ward score. Also, features of whether a team won the previous head-to-head matchup, a team’s current win percentage, or even a specific player’s past performance on their specific champion would potentially be important in determining the result of a new game.

Challenges Faced/Closing Thoughts

Utilizing the live API component of our data sources for supplementing the ranked soloqueue dataset and providing more training/testing instances was a challenge due to the API limits (only 100 requests every 2 minutes). As a result, just like our dataset size constraints with professional matches, we were unable to extract a big dataset.

Also, despite domain knowledge about League of Legends, it was difficult to come up with custom features to improve the accuracy of our models. Of course, there are certain factors that the models didn’t take into consideration. For one, in our ranked solo queue dataset, the games analyzed for champion select did not take into account the fact that players with higher ELOs are more likely to beat those with lower ratings. This is because every ranked game is designed to best match a player’s ELO with those that are of similar levels, meaning that ranks analyzed in our models are not as significant as they should be when it comes to determining the outcome of a match. Similarly, in the professional game dataset, similar team comps may have different results depending on the team and players involved. In the professional setting, there is perhaps more variance in team and player skill, as we see certain teams dominate in their respective regions for many seasons in a row. Emotional factors also play a huge role in professional play, especially for games in the championship rounds.

Finally, with new patches, champion meta and gameplay change over time. Since our datasets were drawn from a limited time window, we assumed the gameplay did not

In general, prediction models for sports/ esports are difficult, as with sports betting, it is impossible to achieve 100% accuracy. **Every professional game or even ranked game is unique in its own way and circumstances.** Our **70% accuracy models** for professional gameplay certainly beat the baseline models of random guessing or always guessing that blue side wins (56%), which indicates that game results can be reasonably determined by just the pre-game and early-game features alone, without relying on waiting for the game to unfold late-game. Nevertheless, our models identified important features, and had the ability to predict live games through the Riot API, which will be a useful tool for players to see data-driven insights on their games and how to win future games without feeling the need to surrender.

Data Machine Learning League of Legends Data Analytics E Sports