Introduction

League of Legends (LoL) is the most popular Multiplayer Online Battle Arena (MOBA) at this time, developed by Riot Games. Most of LoL's success is due to its competitive ranked gameplay where two teams of five players fight against each other to destroy the other team's base.

Within each LoL game, it is considered that there are three stages to the game: early, middle, and late game. Many professionals and high-level players consider the early game to be the most important stage to determine the overall outcome of the game. There is no exact time when the early game ends and the middle game begins, but for the purposes of this paper, we consider the early game to be anything occurring within the first 15 minutes of the game (The average total game time is 35 minutes). We will also consider the champions each player selects, where each player is able to select one of over 150+ champions, as this happens before the game begins. Some different compositions of champions are better than others as certain champions may make up for what other champions lack.

In this paper, we use various machine learning models that can predict the outcome of the ranked LoL match. Using only features from the first fifteen minutes of the match, we created a model that can predict the outcome of a match with just over 80% accuracy. Our results indicate that gold, experience, dragon kills, and inhibitors destroyed were the most important features to tell if a team would win the game. The high accuracy of 80% suggests that the early game can decide the fate of the overall game and that performing well in the early game is extremely important to success.

Dataset

Riot Games provides a public API to access several different types of player and match data for LoL. The data was collected over a week period because of API request limitations as well as having to collect vast amounts of data which isn't included in the final data but was necessary to utilize other API endpoints. In the end, a total of just over 25,000 unique ranked matches were collected. From those matches, timelines were obtained for the entirety of the match, where processing had to be done to extract only the first 15 minutes of information. Some of the features that explained the target the best are described below:

- **Gold** - An integer [0, ∞) that tells how much money the given team has to spend on items to get stronger.
- **Experience (XP)** - An integer [0, ∞) signifying the total champion experience a team has collected.
- **Fire, Earth, and Water Dragons** - Integers [0, 2] telling how many of the 3 of 4 dragon types were killed
- **Red Inhibitors Destroyed** - An integer [0, 3] telling how many red team's inhibitors were destroyed. Inhibitors are deep in the base close to where the enemy team would win.

Feature Selection

Using Pearson's R-correlation between all original features, we found that the most important features in telling who will win were gold, XP, dragons killed, and red inhibitors destroyed. Other features were found to have low to no correlation. Blue deaths, assists, and kills were linearly-dependent with their red counterparts and could be used to directly compute the other team's values so only the

blue set was kept. Gold was also found to be highly correlated with the enemy team's deaths as well as CS and Jungle CS.

For each match, we engineered several other features to generalize the killing of an epic monster (dragon or herald) and objectives (turrets and inhibitors). Also, the differences between the teams was used rather than individual values for each team. This reduced the number of features as well as removed the linear dependence.

Predictive Task

As stated before, we will attempt to predict which team will win a League of Legends game based on the champions selected and various statistics from the first 15 minutes of a game. To evaluate the model, we will split the dataset into a training (80% of data) and test (20% of data) set and calculate the model's accuracy on the test set. As a baseline, we will be using a basic logistic regression classifier.

Models

We explored several machine-learning models based on previous works performing similar tasks. We tried a Support Vector Classifier (SVC), Random Forest (RF), k-Nearest Neighbor (kNN), and Gradient Boosting (GBoost). In this section, we will explain why we chose the kNN, the optimizations we made, and any issues we ran into.

After evaluating all of the models with the original dataset and tuning parameters, GBoost performed the best. However, after using the feature importances from the baseline model and only keeping features with importances over 0.09, the best-performing model ended up being kNN. In our model tuning, performed a GridSearch with various parameters to end up with a kNN model with metric='manhattan', n_neighbors=99, and weights='distance'. After feature removals, we ended up reducing the feature complexity from 40 to 11. Also after removing these features, most of the other previously well-performing models dropped in accuracy while the kNN model slightly improved from its previous performance (illustrated in the figure below).

Model training speed also played a role in the kNN model being selected. Other models took over a minute to train on a relatively small (20k rows) dataset compared to the millions of rows of data that could potentially be trained on. The kNN model took less than one second to train, which would help with the scalability of the model.

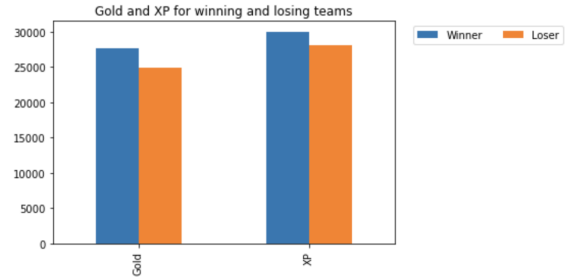
	40 Feature Accuracy	11 Feature Accuracy
Baseline	77.70%	77.90%
SVC	78.01%	77.53%
kNN	80.12%	80.20%
RF	80.26%	79.21%
GBoost	80.34%	79.02%

EDA

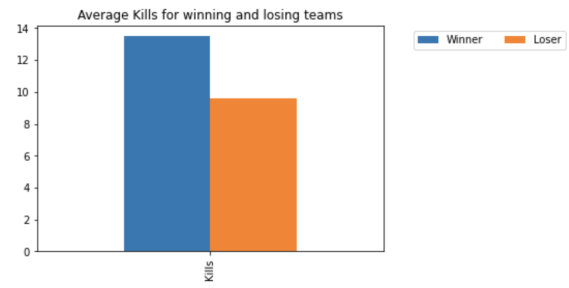
When examining the data in our exploratory data analysis, we first looked at the overall column means to get a sense of what typical values look like. We found that both blue and red teams, as expected, had similar values for each column overall. The average values for each team are as follows: red gold 26,253, blue gold 26,268. Red xp 29,103, blue xp 28,984, red cs 322, blue cs 321, red kills 11.54, blue kills 11.56. In general we see that the values for each column is not statistically different between teams. This makes sense, considering both teams play the same role and have access to the same resources in game, so the only difference between them is nominal. For simplicity's sake, we'll be doing our analysis

on the blue team, and attempting to predict whether they won (although again, the results are the same for the red team). So, our response variable of “winner” will be a binary variable, taking a true value if the winner is blue, and false otherwise. We engineered two new columns, the gold difference (blue gold - red gold) and the xp difference (blue gold - red xp) with this in mind.

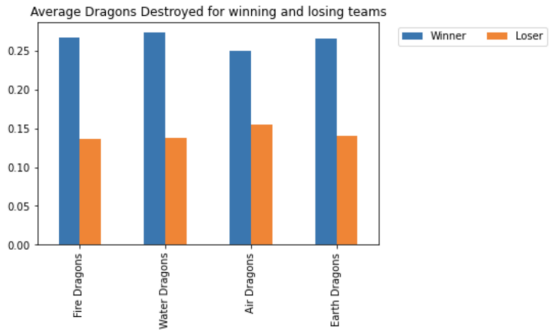
Now, we’d like to take a look at how these values differ depending on which team won. We found in our analysis (as we discussed before) that the values for the respective winning and losing teams were virtually identical whether the blue or red team won, so to conduct this analysis we simply took the average of winning teams. We found that on average, the winning team had 27,634 gold compared to the losing team’s 24,887. The winning team had 30,036 xp compared to the losing team’s 28,052, and the winner had 13.48 kills on average compared to the loser’s 9.62.



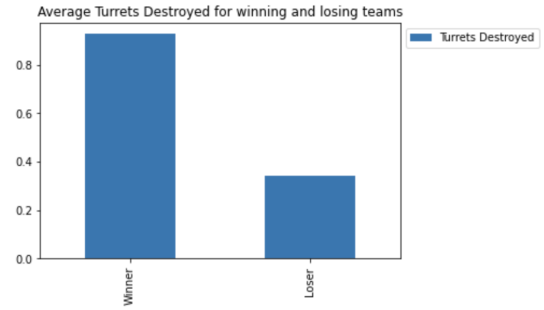
Below is a plot displaying the difference in average kills through 15 minutes:



We also analyzed the differences in the objectives achieved through the first 15 minutes in our dataset. Namely: the average number of dragons destroyed (of each type), along with the average number of inhibitors, turrets, and wards destroyed. Again, we found that there was a significant difference, with the winning team averaging significantly more dragons destroyed of each type (0.27 compared to 0.14 fire, 0.27 compared to 0.14 water, 0.25 vs 0.15 air, and 0.26 vs 0.14 earth).



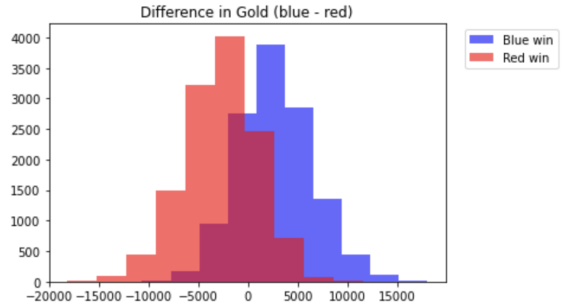
We also saw significant differences in the number of turrets destroyed on average, as can be seen in the plot below.



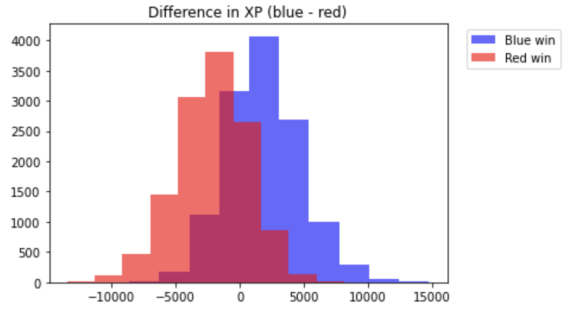
On average the winning team destroyed 0.003 inhibitors, 0.93 turrets, and 5.745 wards compared to the losing team’s 7.96*10^-5 inhibitors, 0.34 turrets and 5.24 wards. We believe that these differences, along with will help inform our predictions, and as such will be useful in our model.

Additionally, we found a relatively strong correlation (0.48) between the difference in gold between the two teams and them winning, with a similar correlation

appearing between difference in xp and win rate. Below is a histogram of the gold difference in a blue win overlaid over a histogram of the gold difference in a red win.



As we can see, the distribution for both is approximately normal, with the distribution for blue wins having a much higher average gold difference (i.e., more positive). Interestingly, the mean gold difference in blue wins was 2762, meaning blue had 2762 more gold when they won on average, and in red wins the gold difference was -2730, meaning that red had almost exactly the same amount of gold in games where they won on average. This trend is also present in XP difference, as seen in the plot below.



When blue won, the mean xp difference was 1865, and the mean xp difference was -2103 when blue lost.

Related Literature

There is a fair amount of research concerning models predicting the outcome of League of Legends and other multiplayer online battle arena games (such as DOTA2). Do et al.[1] is one such paper within which the authors outline their methodology for implementing a similar match prediction model for League of Legends. Much like our project, the authors of this paper obtained their data from the official League of Legends public API, which is the most reliable and comprehensive source of data on League of Legends matches. To make their model, the authors used data pertaining to the relationship between each player and the champion they were playing in the match. These features included the proportion of games they played on that champion which resulted in a win, the player's "mastery points" (a measure of a player's skill level with a specific champion), the total number of games they played with the chosen champion, etc. After collecting their data and selecting the most relevant features, they considered a few different algorithms for modeling: support vector machines, k-nearest neighbors, random forest trees, gradient boosting, and deep neural networks. Ultimately, they were able to predict the winner with over 70% accuracy in all of their models. They determined that the deep neural network yielded the best results when considering its relatively high accuracy (75.1%) and low standard error (0.6%). Hanke et al. [2] applied a similar methodology to DOTA 2, another multiplayer online battle arena game. They used the selection of heroes (DOTA 2's equivalent of champions) to train a neural network and predict the outcome of a match achieving 88% accuracy with said model.

Cardoso Silva et al. [3] is another paper which attempts to predict the outcome of League games, with a methodology which is most similar to ours. Their model is based not just on champion selection, but also incorporates in-game

events. Namely, the model considers the amount of gold earned by each champion, the time a champion was killed, and the time an objective was destroyed, along with general information about the match such as the champion chosen. The authors of this paper sourced their data from Kaggle, with the data coming from professional and competitive matches between 2015 and 2017. Their models were based on recurrent neural networks, with their best model utilizing Keras' simpleRNN. The accuracy of this model varied depending on the time interval the data was collected in (naturally so, as data from later in a match will typically offer more insight as to who the winner will be), with the 0-5 minute model having an accuracy of 63.91%, and the 20-25 minute model having an accuracy of 83.54%. For data within the 10-15 minute interval, their simpleRNN model had an accuracy of 80.18%.

Conclusion

In this paper we introduced a k-Nearest Neighbors Classifier that was able to predict the winner of a LoL match with an accuracy of just over 80%. The data was the champion selections from before the game started and statistics from the first 15 minutes of the game. Since we were able to determine the winner of the game with relatively high accuracy using only the first 15 minutes of game data, we can conclude that the early game is one of the most important parts of the game.

The most important features were gold collected, XP gained, inhibitors destroyed, and fire/earth/water dragons killed. Using just these features in prediction kept the accuracies of the best model relatively the same (around 80%) which allowed us to reduce the feature complexity from 40 to 11.

When engineering new features, such as combining the four different dragon types into one feature or turrets and inhibitors destroyed, the models performed worse. The only engineered feature that (extremely marginally) increased model performance was using a difference between the teams gold and xp rather than individual features. The performance increase was from 80.12% to 80.14%. Therefore, this new feature wasn't significantly important to include.

In regards to the k-Nearest Neighbors model's parameters, they were chosen simply because they performed the best. The only caveat is the number of neighbors chosen was always odd to never have a potential tie.

The Support Vector Classifier (SVC) didn't work as well because it required a lot more features that weren't required for the chosen model to perform better. The SVC model also tended to perform worse when there is too high a model complexity.

The Random Forest and Gradient Boosting model didn't work as well because they overfit to the training data. The model's performance dropping after the reduction of features goes to show that they did not explain the variance between the target and important features as well as the chosen model.

As compared to other models mentioned in the related literature, our k-Nearest Neighbors model performed as well as a more complex Recurrent Neural Network which was trained on similar data. To improve our model, we could incorporate the player's skill on their given champion which other literature found to give prediction accuracy of around 75%. This goes to show that the outcome of a game can be explained by data before the game starts and within the first third of the match.

References:

1: https://arxiv.org/pdf/2108.02799.pdf
2: https://aaai.org/ocs/index.php/AIIDE/AIIDE17/paper/view/15902/15164
3: http://www.sbgames.org/sbgames2018/files/papers/ComputacaoShort/188226.pdf