League of Legends Champion Predictor

Zhonghan Li and James Xie

League of Legends is one of the most popular games on the planet that even with an incredibly complex gameplay system has managed to captivate millions of players. However, one of the hardest decisions players often face is not during the game, but before- choosing which champion. There are 141 champions as of date, each with their own unique set of "rules" that apply to them in-game, making it tough for a human to know whether or not they'd be good at the specific champion. A champion recommender could have widespread impact on players trying new champions and achieving success within the game. We have created a predictor that allows a player to see what champion they haven't played much but are very likely to play often.

Initial collection of data

Riot documentation was much harder to work with than we anticipated, due to them restricting certain tasks for security reasons that we needed to perform, such as retrieving a list of summoners in a tier. In order to actually begin parsing data into a format that Weka would recognize, we needed to first gather all of our attributes.

The first challenge was, as previously mentioned, retrieving a list of summoners in a tier. Riot does not allow us to get a list of all players in a specific tier (gold, silver, platinum, etc.) or at a specific elo (four digit number indicating level of play), and as a result we had to write a recursive algorithm that, starting from a root player, would find all recently played games based on current tier, and then after parsing players into a list find another and re-run the algorithm on that player. In this fashion, we were able to create a data set of player IDs to parse through that all had the same tier, even though each set of 5000 players took 5 hours to retrieve.

After that, we simply parsed through the data and found the attributes we listed in our previous progress report – the top 20 most mastered champions, the top 20 champions in which the player has the highest winrate on, role played, and tier.

Preliminary results

After running Weka on the dataset for all 5 non-challenger tiers, we retained the following results for running nearest neighbor on the gold tier:

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          13               1.8598 %
Incorrectly Classified Instances       686              98.1402 %
Kappa statistic                         -0.0059
Mean absolute error                      0.014
Root mean squared error                  0.0838
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              699

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.000    0.000    ?          0.000    ?          ?        0.099     0.003     266
               0.000    0.000    ?          0.000    ?          ?        0.413     0.018     103
               0.000    0.000    ?          0.000    ?          ?        0.197     0.005     84
               0.000    0.000    ?          0.000    ?          ?        0.199     0.006     12
               0.000    0.000    ?          0.000    ?          ?        0.199     0.006     32
               ?        0.000    ?          ?        ?          ?        ?         ?         34
               0.000    0.000    ?          0.000    ?          ?        0.458     0.015     1
               0.529    0.601    0.021      0.529    0.041      -0.023   0.436     0.022     22
               0.000    0.000    ?          0.000    ?          ?        0.099     0.003     136
               0.000    0.000    ?          0.000    ?          ?        0.049     0.001     268
               0.000    0.000    ?          0.000    ?          ?        0.197     0.005     432
               0.176    0.304    0.014      0.176    0.026      -0.043   0.436     0.022     53
               0.000    0.000    ?          0.000    ?          ?        0.148     0.004     63
               0.000    0.000    ?          0.000    ?          ?        0.297     0.008     201
               0.000    0.000    ?          0.000    ?          ?        0.458     0.015     51
               0.000    0.000    ?          0.000    ?          ?        0.049     0.001     164
               0.000    0.000    ?          0.000    ?          ?        0.049     0.001     69
               0.000    0.000    ?          0.000    ?          ?        0.247     0.007     31
               0.000    0.000    ?          0.000    ?          ?        0.149     0.004     42
               0.000    0.000    ?          0.000    ?          ?        0.247     0.007     122
               0.000    0.000    ?          0.000    ?          ?        0.099     0.003     131
               0.000    0.000    ?          0.000    ?          ?        0.249     0.007     119
               0.000    0.000    ?          0.000    ?          ?        0.249     0.007     36
```

Figure 1: Preliminary results; each integer represents the ChampionID of a champion in the game.

Which we considered to be quite poor.

We suspected one of the causes of which to be that players in certain tiers actually have heavily varying elos. A player in gold can have the same elo as the average platinum player (a tier above), and playstyle preferences heavily differ based on elo. The biggest impact factor is that certain champions, which serve as our attributes, will be more powerful in the game based on balancing factors at a certain point in time, but these power inbalances are generally purely statistical. What this means is that as elo rises and the skill level rises, higher elo players are able to overcome these statistical differences and as a result feel more comfortable playing a larger pool of champions.

Moreover, champion winrates, we realized, are extremely nondescript. For example, a certain champion at a tier may have a 30% winrate, but a player may have a 40% winrate on them. In this sense, they have a low overall winrate but a high relative winrate, and should be considered to be good at this champion. Unfortunately, as there is no way to get champion winrates by elo through riot API, we decided to simply remove these as attributes. This is justifiable by understanding that the players do ultimately enjoy winning, and the more they play a champion not only are they more likely to win, but the more they win on a champion the more likely they are to play on it. As a result, mastery and winrate are not completely independent, and running using just mastery is ok as an option.

Final results

After tweaking our preprocessing and shaving it down to only use the following attributes:

ROLE, MASTERED CHAMP 1, …, MASTERED CHAMP 12

we decided on gathering a dataset based on elo. After running a few methods on the modified dataset, specifically:

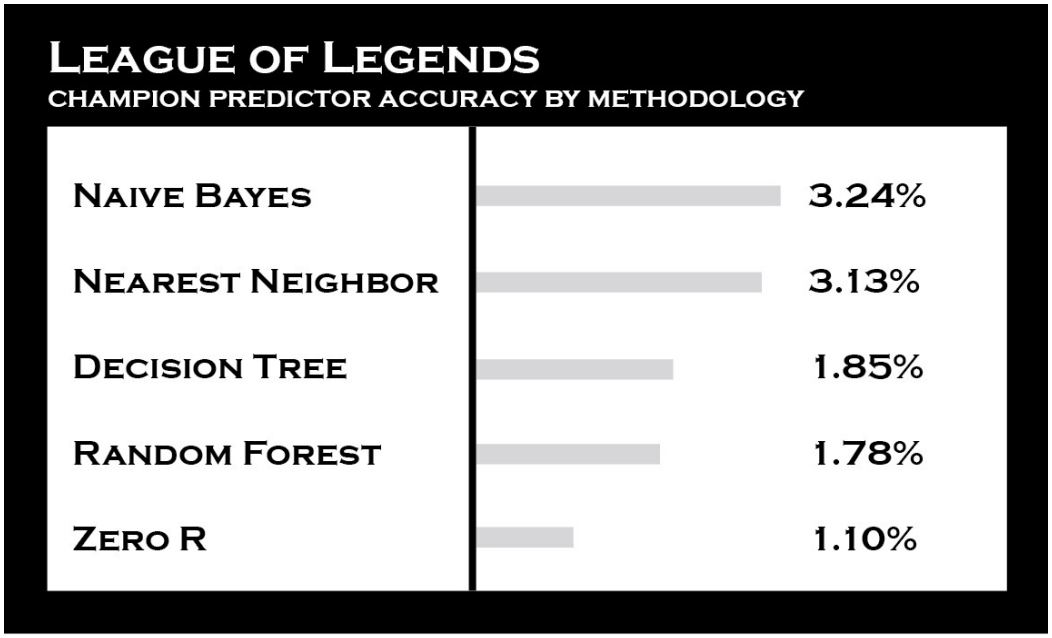

Figure 2: Accuracy by methodology; accuracies are averaged over all tiers of elo

We settled on using Nearest Neighbor, due to it making more sense than Naïve Bayes(should not make naïve assumptions here). Decision Tree and Random Forest were both significantly worse.

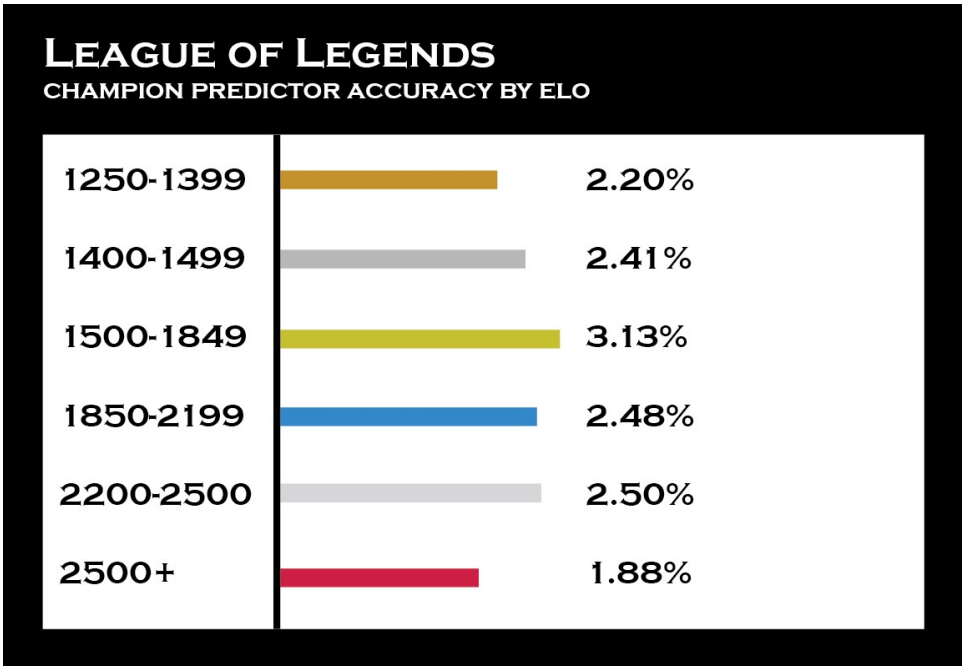

Figure 3: Accuracy by elo; divisions made by estimated tier values

Which are also not terribly impressive at first glance.

However, we must remember that there are 141 champions in total and as such a naïve algorithm where we simply take the most played champion in the game of that the player does not play in their top 20 champions would actually give an accuracy a little higher than .08%, or 1/(141-20). Moreover, we tested accuracy on a single champion class, but if we were to provide a list of say 10 champions to suggest based on a collection of nearest neighbors, the accuracy would be around 0.9687^{10}, or 27.2%. Naïve Bayes actually reached a higher accuracy, but as the method didn't make as much sense when applied to this dataset we dismissed the difference.

Analysis

As expected, the percentage of accuracy peaks at around gold elo and gets skewed as we progress above the bracket. This makes sense because as explained above, less skilled players tend to have a smaller champion pool to select from due to them only looking to play powerful champions. This will naturally raise the accuracy of any ML algorithm for lower elo players.

However, very low ranked players also tend to not play smart, choosing either a random, non-related assortment of champions (and thus wasting time having to master a larger breadth of skills), or play just one champion exclusively in hopes of mastering it (we call these "one-tricks"). In the first scenario, this makes it so that it's hard to find a strong pattern associated with the data presented, and the second is tough for this algorithm to solve, as the data implies that the players themselves don't really want to learn a new champion.

This is also indicative of the fact that League of Legends players need to expand their champion pool; in accordance with the rules of the game, in any given match 10 champions are removed from play. If players had larger champion pools, mastery level would be a better indicator of favor.

Future improvements

We believe that using a larger dataset would actually increase the accuracy of the predictor more. Even though the difference between 1000 and 5000 was marginal (.08% increase), there are 141 possible classifications for each datapoint, giving each classification an average of 36 datapoints, which is terrible. Unfortunately, our algorithm runs too slowly, and with API access limits we couldn't find a way to gather what we'd see as enough data.

Ideally, we'd be able to refine our recursive algorithm and get a permanent API key such that we could compile several million datapoints without interruption (daily API key resets, Wi-fi fluctuations, etc.), unfortunately, this would take weeks.

Moreover, the API-provided statistics are not exactly indicative of playstyle. There are many of hidden factors that go into the game itself. For example, if we could find a way to classify players as aggressive or defensive players, adding just this attribute could divide the recommended champions in half, potentially doubling our accuracy.