Using Machine Learning to Optimize Champion Draft for High Performance League of Legends Matches

<https://github.com/Christian-Martens-UNCC/ECGR-4105/tree/main/Final_Project>

Christian Martens   
Electrical and Computer Engineering  
University of North Carolina at CharlotteCharlotte, NC  
cmarten1@uncc.edu

Edward Pascual-Bautista  
Electrical and Computer Engineering  
University of North Carolina at CharlotteCharlotte, NC  
epascua2@uncc.edu

***Riot Game’s “League of Legends” video game is the largest Esport in the world. Garnering hundreds of millions of unique viewers every year, the growing entertainment medium is becoming more enticing for advertisers to support. Professional gaming clubs have a financial interest in winning these events to become more appealing to new sponsors. Using data collected from high performance League of Legends games, several machine learning models were developed to assist in the “draft phase” of a League of Legends match. Using predictive machine learning models, the outcomes of League of Legends matches could be accurately predicted 67.7% of the time just from the limited information of the draft phase.***

***Keywords—Esports, advertising, gaming, machine learning, match***

# Introduction and Motivation

“League of Legends” is the world’s largest Esport, garnering hundreds of millions of viewers every year on a variety of major television and streaming services. In League of Legends, two teams of five players battle in player-versus-player combat, each team occupying and defending their half of the map. Each of the ten players controls a character, known as a "champion", with unique abilities and differing styles of play. Professional gaming clubs, organizations that manage and pay pros to play in major tournaments, primarily make their profit off of sponsorship deals. As the Esport world grows, so too does its appeal for advertisers. Thus, owners of professional gaming clubs have a vested commercial interest in winning games to attract more advertisers to their cooperation.

Before a game of League of Legends can begin, each team must select five champions to play, one for each player on the team. This is called the “draft” or “champion select” phase of the game, and the impact this phase has on the rest of the game is hard to understate. Having a good draft phase can give a team a huge advantage during the match phase, whereas having a poor draft phase can doom a team’s chances of winning before the match phase begins. Additionally, unlike the other phase of the game, the draft phase is highly data driven, although the data is highly non-linear. Currently, the coaching staff and players assess what champions are the best to pick in each situation using the talents and instincts they’ve trained over thousands of games, but this does not mean that they make optimal decisions all the time. Optimizing this portion of the game would lead to a noticeable increase in the winning percentage of high-performance clubs which would boost the profits from apparel revenue and sponsorship deals.

# Approach

## Artificial Neural Network Model

Due to the highly non-linear nature of the input variables and their effect on the outcome of the game, the team originally assessed that constructing a neural network would provide the most consistent and accurate answers compared to other machine learning models due to the property of neural networks to be extremely efficient in predicting non-linear outcomes.

## Logistic Regression Model

## Logistic method was used as a cross comparison to the neural network for the data validation accuracy. As the data is non-linear, the logistic approach would be a good model to test the training.

## Naïve-Bayes Model

The Naïve-Bayes approach is another method assessed for comparison to the neural network results. The model is a straightforward probabilistic prediction model that would provide another insight to the accuracy rate of the previous two. The setup would be simple to implement and get quick results in the classical machine approach.

# Dataset and Training Setup

The dataset for this project was obtained from the OpenML dataset library. The dataset was submitted to the public library by Elif Ceren Gok. To better simulate the draft experience, many of the elements of the dataset were removed since they are either not known or not manipulatable elements during the draft phase. As such, the modified dataset consists of 4028 games with 10 inputs, one for each player’s selected champion. Each champion was assigned a number which would then be used to represent the champion in the dataset. These datasets were then converted to file types which could be uploaded into a Python library where the models were trained.

## Artificial Neural Network

The game and champion dataset was converted into a

[10, 1, 1, 4028] image library where each of the 10 inputs was given its own channel. It is important to note that, for the purposes of this model, having the 10 inputs be classified as channels as opposed to rows or columns was subjective as the calculations and outcomes will be identical. The constructed neural network model contained three hidden layers of size 512, 256, and 64, respectively. The mode was trained for 500 epochs and was tested using a 90-10 validation split dataset. Due to the relatively low size of the sample data, a batch size of one was used as the training time with the batch size was still reasonable and the most information could be gained from each data point with this batch size.

## Logistic Regression Model

The dataset was split into two groups: data containing the 10 feature inputs, and target that holds the output. The two sets were then split into a training and testing group at a 90-10 ratio. Standard scaling was implemented to standardize the data for both training and testing sets with the 10 input features while the output sets are left unchanged. The logistic regression model was configured with a learning rate of 0.001, and class weight set at balanced. Other parameters were left on default.

## Naïve-Bayes Model

The same approach as the logistic regression takes the same split groups into a training and validation set also at a 90-10 ratio. The size of the dataset is small but the ratio at this size does not change drastically but rather within the margins of one to three percentile. Standard scaling was used to standardize the values in the training and validation group. The Naïve-Bayes model was configured using the default parameters.

# Results and Analysis

The neural network model produced a validation accuracy of 67.7%. Normally, one would expect binary classifying to produce a much greater accuracy. However, for this particular dataset and given how little information the machine learning algorithm has about a game of League of Legends compared to the amount of data that could be used to analyze a game, having a model that, on a relatively small number of games predicts at a better than two-thirds chance of a game’s outcome is very impressive.

The logistic regression model produced a testing accuracy of 51.9% and a validation accuracy of 53.1%. The model had to be given the parameter for the class weight to prevent the issue of zero division warning for the confusion matrix and classification report. The error appears to indicate that the testing set does not have indices when the training set does. Checking the values does appear to have the two outputs in each case but somehow the f1-score, precision, and recall did not count 1 classifier in the result. Without the class weight set to balanced, the result came out to be 77.1% accurate despite not calculating the whole data. The result from this approach makes the game prediction a little above 50% split in chances of winning odds.

The Naïve-Bayes model resulted in a 77.2% accuracy rate. Similar situation as before, the precision, recall and f1-score were set to zero, indicating that the data for 1 output was not appearing in the validation set. This was checked again to see if the data had both output labels for 0 and 1, both of which were confirmed to have the values. If the accuracy result is to be valid then the logistic regression and naïve-bayes method are yielding high results.

##### Lesson Learned and Future Improvements

At the time the dataset was created, there were 4.5 sextillion permutations of potential League of Legends games. Of those, 5.9 billion are what could be considered “realistic” permutations to consider. As such, the dataset’s limited size of just over 4,000 is, overall, a very small sample size. The dataset also does not include champion bans or pick order which is information that is available during the draft phase which could hone the accuracy of the model further. Other factors which the model does not take into consideration would be the name of the opposing team and its players as well as both teams’ win-loss ratio for the season, although some of these factors are very impractical to obtain. Additionally, the dataset is not filled with an equal number of outcomes for each label. Due to the imperfections in the dataset, the outcomes shown in the various models should be viewed as proof-of-concepts and any results derived from these models should be considered with the flawed dataset in mind.

Despite the flaws in the dataset, it would be very difficult to fix many of the flaws with the dataset without the direct cooperation of Riot Games. Riot Games is very protective of the data they release to the public about the information they collect from every game of League of Legends and are very selective as to what companies have access to their API. At the current moment, it seems that the most practical way to gain additional datapoints would be to program a scraper bot which could scrape data from other websites which already have access to Riot Games’ API through features such as individual match histories. With an effective scraper, the dataset could be enhanced with the hundreds of thousands of League of Legends games played all over the world on a daily basis for a more robust and up to date dataset.

Outside of improving the dataset, future improvements would include updating the code to simulate a live-draft scenario that could be used in real time. This would turn the model derived above from valuable insight which could be used in training and reviewing previous games into a product which could be used to make determinations about the game that is currently being played. This would dramatically increase the marketability of the model.

##### References

* Gok, E. C. (2022, February 24). League-of-Legend-High-Elo-Team-Comp--Game-Length. OpenML. Retrieved December 16, 2022, from https://www.openml.org/search?type=data&amp;status=any&amp;id=43842