League of Legends Match Winner Predictor

Caleb Wagner

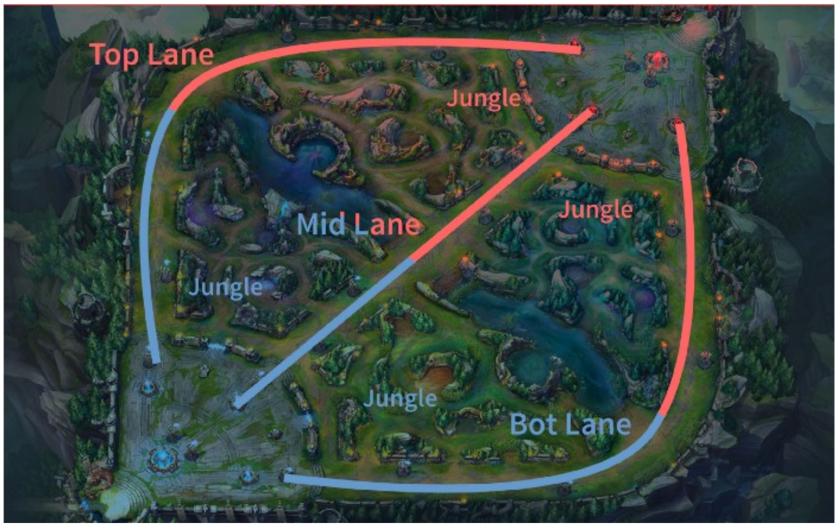
December 7, 2020

Abstract

League of Legends is one of the most popular games of all time with 200 million monthly players. This game lacks prediction models, technology and I am here to assist in this lacking area. The data used was 10 years of professional League of Legends championship data with over 51,000 games. Four of the models used in this project were kNN, Neural Network MLP, Decision Tree, and AdaBoost. I found the top 3 features to be First Dragon, Number of towers destroyed and the Number of inhibitors destroyed. Our top 2 models were kNN and Multi-Layer Perceptron classifiers with ~96.3% accuracy with 6 features (3 from each team). This is an interesting comparison to the 62% accuracy of the best human predictor of League of Legends matches. Some of the information may be too accurate because or overfit because we have an accuracy of over 95% for some of the models. There is room for improvement in the parameter tuning for the models. I could not find a way to get the confidence of the winner prediction other than using the percentage of the nearest neighbor in kNN. It is with bringing up that I did work on this project alone.

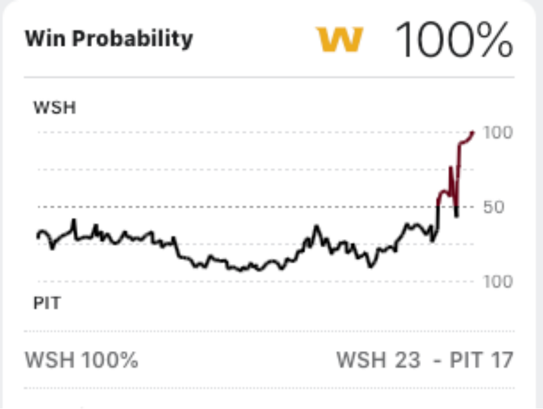
Introduction

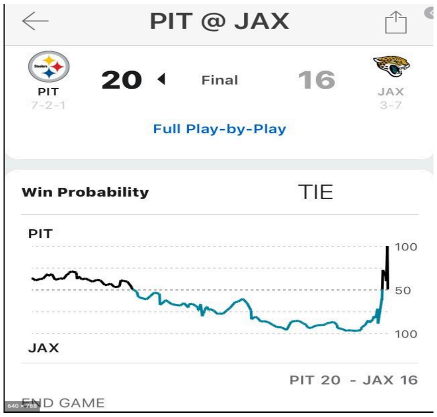
Now "what is League of Legends?", and "why is this video game important?". With over 200 million monthly players, League of Legends is in the top ten most-played video games, and currently ranks in top 5 monthly played games. The game is a 5 vs. 5 massive multiplayer online "battle arena" (MOBA) free game developed and managed by Riot Games. The game was originally released on October 27, 2009. There are 2 main different game modes in League of Legends: ARAM and Summoner's Rift. ARAM stands for "All Random All Mid", and all players get assigned a random character. Every game will have a Red Team and a Blue Team that start on opposite sides of the map. The "Summoner's Rift" game mode will be the only one we will be focused in this report. The name of the map played on during this game mode is also called Summoner's Rift, and a map is show below in *Figure 1* below.



Summoner’s Rift is the most played and most competitive game mode in League of Legends. In this game mode, there exists 9 towers and 4 inhibitors. The goal of the game is to destroy all of the opposing team’s 4 inhibitors before they destroy yours; moreover, the team that has all of their inhibitors destroyed first loses. The 5 roles that exist in the game are as follows: Top, Mid, ADC, Support, and Jungle. There are 3 main lanes to traverse the map and there is also a jungle on each side. The characters that each player chooses to use are called “champions”, and each champion has different abilities/passive abilities. Before the match starts, there is a champion selection period where each person playing will chose which champion they want to play as. There is also a banning period where both teams vote on what 3 champions they want to ban for the opposing team. During this time, there are also about 10 summoner spells and each player gets to choose 2 of these. This is what would be considered the pre-match data and many websites like to record and show the best builds for each champion, best champion counters, and show the win rates for each of the champions based on the current season. Some of these builds include items that one can upgrade their champion that is purchased at the shop using the in-game reward currency called gold.

As a fan of the National Football League and the National Basketball Association, I always find the statistics and graphs fascinating. This brings me to a graph that I see all the time that displays the current chance of a team winning the match. This model predicts as the game is going on and will give an accuracy percentage for the prediction. Two images of these models are directly shown below.





These two images display a model/representation that will be my goal for this project.

Problem Statement

Professional E-sports have been around for only a short amount, the market is over a billion dollars. There are a ton of new technologies and applications from other professional sports being applied to the E-sports. Many information and statistics are out there for games such as League of Legends. I also knew that there is an opportunity to look at the statistics and focus on the most impactful stats to win a professional match in League of Legends. The game's meta or most important information to win could change the way we play the game. I also believe that this information will be conjoined with the structure of the professional playstyle. When these teams know what are the most important aspects of their game, they are more likely to go for those to try and secure the win.

I also feel that these models can bring more attention to the gambling market with e-sports, as there already exist websites where you can bet on professional e-sports games and then watch them live. Large tech companies would love to draw more people to the League of Legends, and grow the gambling industry – this is more money that both companies and players will make. Many of these constructs already exist in modern professional sports and there is room for the E-sports market to grow. These markets even play a role in the decisions involving the placement of the Washington Football Team's new stadium. The main appeal for Washington Football Team moving back to Virginia is the fact that Virginia just passed legislation to allow gambling on professional sports. This could play a role in the future in e-sports and will affect the industry in a major way.

However, there is some criticism for this stance, and the benefits may not be that clear cut when utilized on the professional stage. I believe that there is a chance that when this information is used it will change some playstyles of the industry and that could change how important these features are towards win. With each season of the game, the champions and playstyle change and what works for last season might not work this season. Another thing to note is also that we have 51, 000 entries in the data set from over 10 years of data. When comparing this to other sports that have over 50 years of data, e-sports are still a new and growing market.

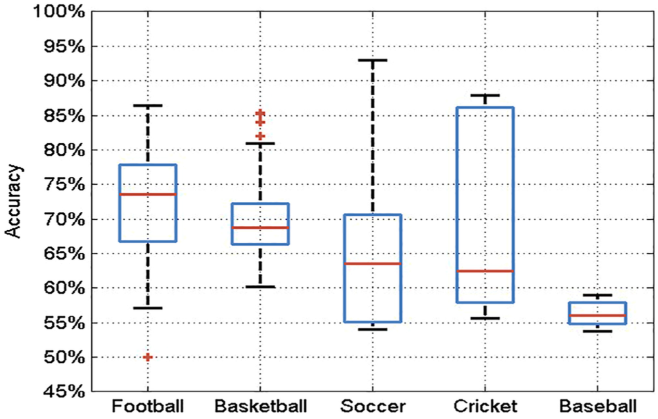
Literature Review

Looking through many other academic papers, I am not the only individual who wants to try and predict sports games. My main goal with the research was to see what people study this application of Data Mining and what problems may arise in this situation. Looking into the different types of models used assisted me greatly in deciding which types of models to use for the best results. I ended up looking into reports on a similar project for League of Legends but also looking at how these results compare across different sports.

The first resource that taught me and educated me in this niche field was about predicting winning teams based on performance in League of Legends. This dataset was different than the one that I used and also analyzed the data. This helped to shine some light on how to interpret the bounteous features in the data and what some of the patterns mean. One of the points in the article points out that some of the stats like k/d may be skewed if the players or teams did not have many games under their belts or just simply disconnected. I also saw that 98% was the accuracy that his Random Forest Classifier model was able to achieve, and with different data, I wanted to set that as a goal for my model. It is important to note that his model tried to remove some of the end-game stats to predict the game because it is too late to predict. This information served me well when developing my structure for my models and feature selection.

The second article that I investigated was "Predicting NBA Games Using Neural Networks", and helped give me other ideas for what model to use. I saw that the main area of emphasis the NBA and the post-season games. Many people bet money and try to predict who will win the tournament and what order the teams will be knocked out. I think that this applies to my situation and can help me refine my models and my data. The study was of 620 games and the models that were mainly used were variations of neural networks, which surprised me because I didn't see any other mentions of this model until I read this article. The article also compared the accuracy of the model with the professional sports announcer's prediction accuracy and the results are quite shocking. The neural network predicted the winning team with 74.33% accuracy and the sportscasters had an average accuracy of 68.67%. That is over a 5% accuracy gain over the top experts in the field.

The third article that I reviewed was comparing the machine learning accuracy across 5 different sports: Baseball, Football, Basketball, Soccer, and Cricket. This article helped to give me perspective on how these leading prediction models compare across different sports and sometimes it is easier or harder to predict games based on the metadata. This article also used neural networks to do most of the work and used k cross-valuation to obtain the accuracy metric for the models. The results were interesting as the average prediction for a football game was higher but the highest model was in soccer. See *Figure 2* on the Right (which is from this article).



Methods and Techniques

There is a need for a model to predict who will be the winner during a game that is occurring. Many of these tools exist within modern sports. These tools are lacking in modern e-sports and help to predict the winner of professional games. Basically, the tool takes in real-time data in intervals and uses historical data to predict the winner. There is also a "Win probability" percentage that shows the likelihood that the team will win. I used this model as inspiration for my model because this is not just a simple match predictor, provides an accuracy metric, and is not definite. This technology is extremely important to the professional aspect, this helps to decide the winner game and allows more critical analysis of the game. I think that this adds another aspect for people to bet and gamble on these games, as previously stated.

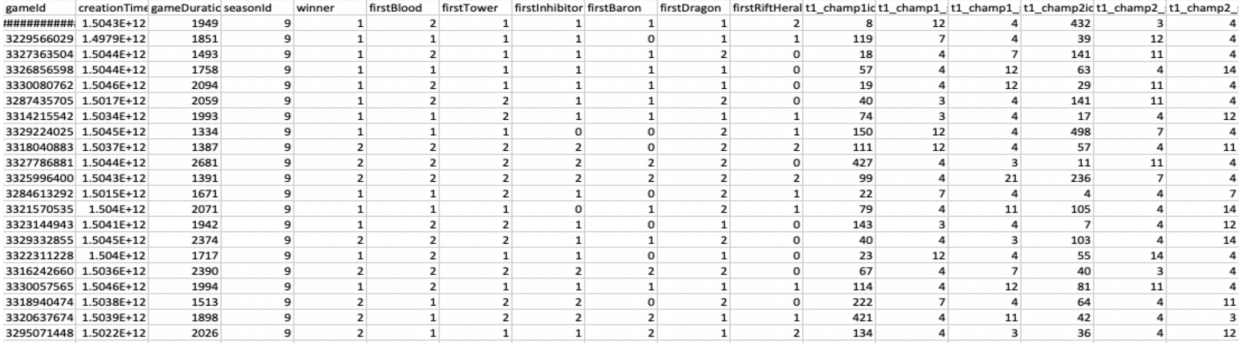
Another aspect of the game that I can test is predicting the match-winner strictly by the pre-match data. I know that some data exists related to this online as champions have a certain win rate. However, by predicting the game with pre-match data, the matchups and summoned spells can be optimized even further.

My original idea was to start with kNN, because of the volume of data I believe that this might be the best way to predict the games because it is directly using the previous games and seeing what game is historically more likely to win. I was originally thinking about the most recent clustering algorithms, but these models did not make much sense to me. Most of these algorithms are not meant to handle a ton of new data points. I do not think that the clustering algorithms make sense because the training would have to be rerun so often it would defeat the purpose. I think that because we are only classifying 2 classes, simpler algorithms would also make the most sense.

Looking at many of the academic papers, many suggest a neural network solution and I am not opposed to this idea. There are quite a few features in this dataset and I am interested to see what these perceptrons interpret these features as I did not have much success in previous assignments with this type of model. I think that because there are so many features in this dataset, there will be a benefit by using a neural network solution. The idea of also using an SVM comes to mind, but for me, the results always take too long to finish running.

Discussion and Results

The first starting point is the data set, which is directly from Riot Games and has data since 2010. I first downloaded this information in a CSV format, so I decided to use the pandas library to import this information as a data frame. The data has 64 features, with some of the information not about the actual match. I know that I will have to significantly reduce the number of features using various feature selection methods so that I can have fast and accurate results. I am expecting the Dragon, first blood, and Baron kills to be the most noteworthy features.



gameId	creationTime	gameDuratic	seasonId	winner	firstBlood	firstTower	firstInhibitor	firstBaron	firstDragon	firstRiftHeral	t1_champ1ic	t1_champ1_	t1_champ1_	t1_champ2ic	t1_champ2_	t1_champ2_
###########	1.5043E+12	1949	9	1	2	1	1	1	1	2	8	12	4	432	3	4
3229566029	1.4979E+12	1851	9	1	1	1	1	0	1	1	119	7	4	39	12	4
3327363504	1.5044E+12	1493	9	1	2	1	1	1	2	0	18	4	7	141	11	4
3326856598	1.5044E+12	1758	9	1	1	1	1	1	1	0	57	4	12	63	4	14
3330080762	1.5046E+12	2094	9	1	2	1	1	1	1	0	19	4	12	29	11	4
3287435705	1.5017E+12	2059	9	1	2	2	1	1	2	0	40	3	4	141	11	4
3314215542	1.5034E+12	1993	9	1	1	2	1	1	1	1	74	3	4	17	4	12
3329224025	1.5045E+12	1334	9	1	1	1	0	0	2	1	150	12	4	498	7	4
3318040883	1.5037E+12	1387	9	2	2	2	2	0	2	2	111	12	4	57	4	11
3327786881	1.5044E+12	2681	9	2	2	2	2	2	2	0	427	4	3	11	11	4
3325996400	1.5043E+12	1391	9	2	2	2	2	2	2	2	99	4	21	236	7	4
3284613292	1.5015E+12	1671	9	1	1	2	1	0	2	1	22	7	4	4	4	7
3321570535	1.504E+12	2071	9	1	1	1	0	1	2	1	79	4	11	105	4	14
3323144943	1.5041E+12	1942	9	1	2	2	1	0	1	0	143	3	4	7	4	12
3329332855	1.5045E+12	2374	9	2	2	2	1	1	2	0	40	4	3	103	4	14
3322311228	1.504E+12	1717	9	1	2	1	1	0	1	0	23	12	4	55	14	4
3316242660	1.5036E+12	2390	9	2	2	2	2	2	2	0	67	4	7	40	3	4
3330057565	1.5046E+12	1994	9	1	2	1	1	1	1	1	114	4	12	81	11	4
3318940474	1.5038E+12	1513	9	2	1	2	2	0	2	0	222	7	4	64	4	11
3320637674	1.5039E+12	1898	9	2	1	2	2	2	1	1	421	4	11	42	4	3
3295071448	1.5022E+12	2026	9	2	1	1	1	2	1	2	134	4	3	36	4	12

First, I started by cleaning the data and had edited a column that had negative numbers and float numbers. These columns need to be fixed in-order to try and start using some of the feature selection libraries from sklearn, as well as the value normalizer for different classifiers such as MLPClassifier and AdaBoost. From the beginning, I knew that I need to remove the first 5 columns to create the train and test data, as some of this information does not relate to the winner, and one of the columns is our true values for the winner. I also was trying to find out what would be the best library to help lower down the number of features and I found SelectKBest from sklearn to be the best as I could use many different algorithms to find the most important features. I ended up going with the chi2 algorithm when selecting features. I tried to use a linear When running my first working selection of the features, it was selected that these were the most important: tower kills, inhibitor kills and dragon kills. This came to my surprise and I think that the inhibitor kills stat gives away the winner most of the time. Usually, the inhibitors are destroyed before they can respawn, however on the other side teams can forfeit at 20 minutes if they think they will lose. Only the champions and summoner spells will be included in the pre-match data predictions. I think that the true test will be with the pre-match data, however, it is pretty cool to see what aspects are crucial to a win.

Evaluation Metrics

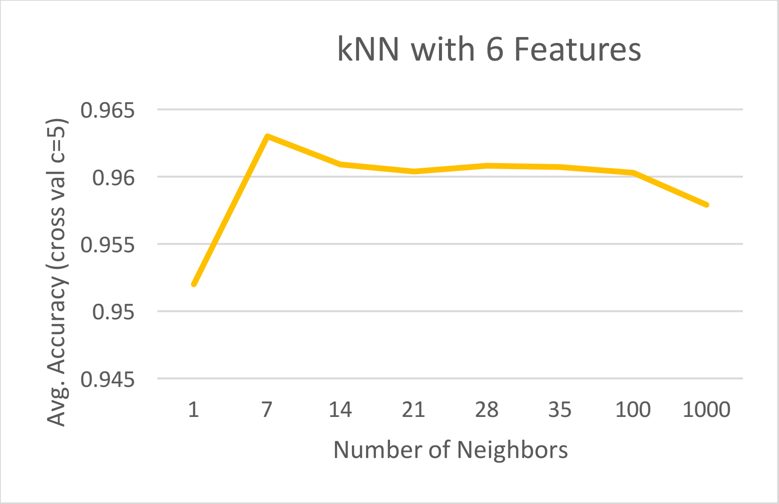
There are a few different ways to make sure that your prediction models are accurate when only using 1 dataset. You can just split the entire dataset into a percentage of train data and test data, and this is called holdout testing. I ended up using a variation of this with cross-validation k times, which splits the data into k even sections and leaves out one part, and uses it as the test set. This way you test all parts of the data but I am a little wary of overfitting the data because I do not have somewhere to test like miner. I will be doing some holdout testing with a 50% split to test the models with minimum test data. There will also be a confusion matrix for showing where the incorrect predictions lie. All of the values in the graphs will be averages of the cross-validation testing with k=5. Accuracy

Experimental Results

kNN

The first model I have implemented is the kNN, and it is arguably the simplest of them all as well. I went into this project expecting the kNN algorithm to be the best algorithm to use for this data because I believe that the similarities between the games and truly predict the winner. I think that the games that have the same stats, should in theory have the same outcome. I can say that this information was kind of true, however, I think that this model is limited because of the curse of dimensionality. There is a possibility that adding more features could introduce more reveal more information on who may win the game. This classifier took an average of 27 seconds to run with the cross-validation. This ended up just about tied with the Neural Network MLP classifier with a high score of 96.3% accuracy. An interesting result is that k=7 was the best parameter considering other projects had k in the hundreds for the best results. This makes me think that only the closest related games matter, you get noise when trying to grab further out neighbors. It is worth noting that the model significantly slowed down at k=1000. The metric that works best for the confidence of the prediction, was the similarity percentage of the closest neighbor. This ended up being a 60% similarity between the test data and the closest neighbor.

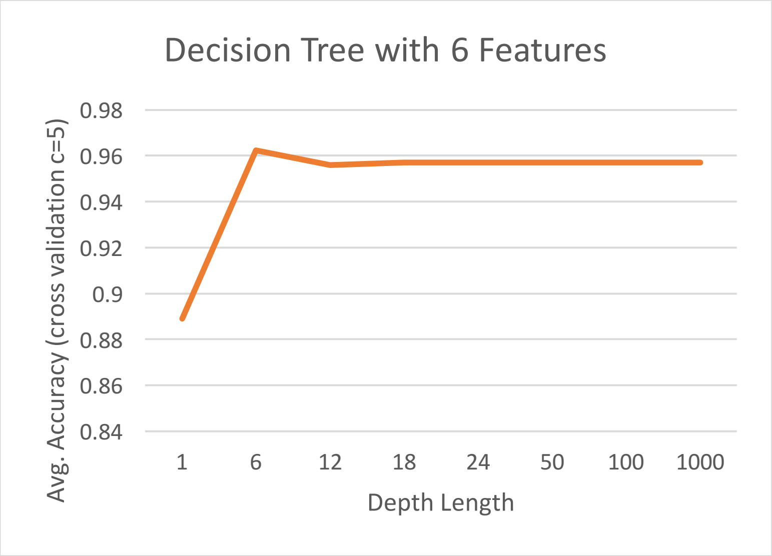
	Predict No	Predict Yes
Actual No	17278	961
Actual Yes	533	17271

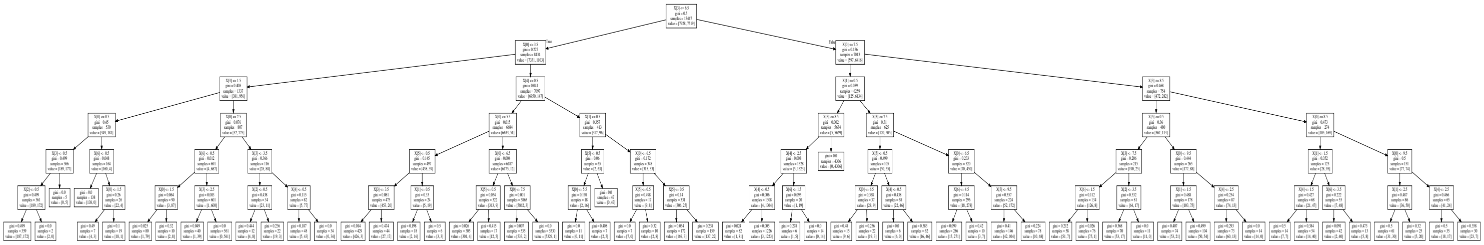


Decision Tree

The second classifier used was a decision tree, which is using DecisionTreeClassifier from the sklearn library. This decision was made as a model that would have a longer training time but very fast prediction values. I thought that there might be some stats that would define who would win and would convert well to a decision tree. The results are excellent and this classifier comes in right above 96% accuracy, and a very quick 2 second run time. I think that this model is the best time and if this was used to predict games in real-time, a Decision Tree would meet the time constraints. The results surprisingly did not change after a depth of 12 for the tree and got essentially the same level of accuracy from 12 and higher. Based on previous projects, I propose the idea that we needed to keep the tree simple and the model got too complex after 12. The tree is below.

	Predict No	Predict Yes
Actual No	2477	395
Actual Yes	1137	16667

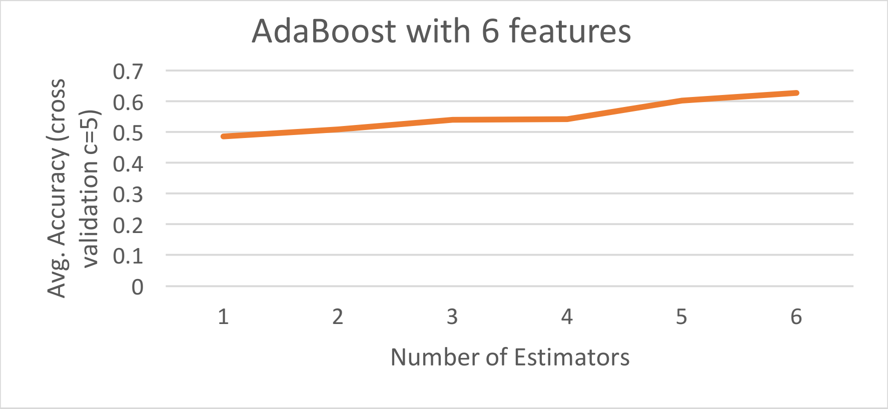




AdaBoost

This model was a top performer in a previous project I have worked on and that did not hold in this project. I wanted to use this model but ran into some problems with the data and realized that I needed to standardize and normalize the data before using the AdaBoost model. The first time I ran the model it ran for over an hour with 1 estimator and never terminated. After some parameter tweaking, I finally got the model to terminate with estimators 1-5 with an average run time of over 20 minutes. I would not recommend this model for our dataset as the runtime is terrible and the diminishing returns are not worth the higher number of estimators.

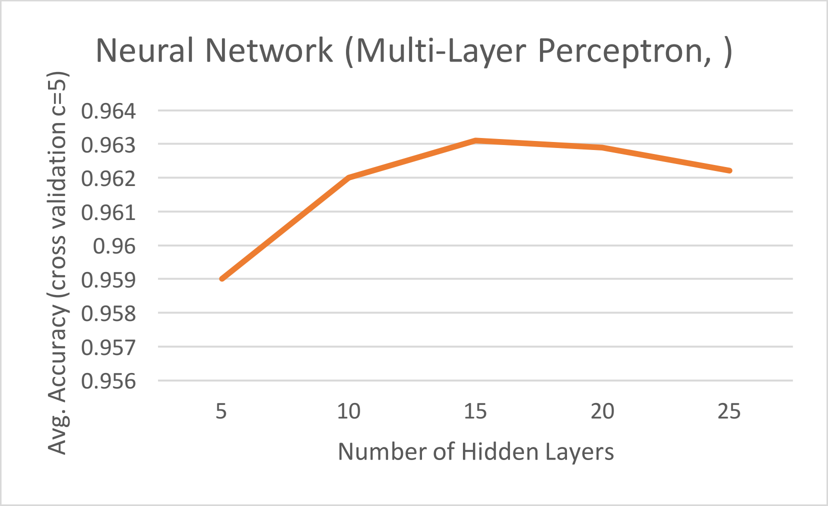
	Predict No	Predict Yes
Actual No	4277	13962
Actual Yes	413	17391



Neural Network – MLP Classifier

The most recommended model based on the academic articles was a Neural Network modifier. I started running this with the Limited-memory BFGS, which is known for running well in data mining applications because it uses the minimal amount of memory. I started out with 10 estimators and a maximum amount of iterations of 100. Every time I was running this algorithm, it was not fully terminating the perceptrons weights. I ended up raising the max iterations to 5000 and this fixed the problem. It is also worth noting that in this model, the data needed to be standardized and normalized. The peak accuracy was on par with the kNN model and had a runtime of 1 minute and 15 seconds for training and predicting. I think that after the model is trained that prediction time is faster than the kNN and more on par with the decision tree. This is in a big tie with kNN.

	Predict No	Predict Yes
Actual No	4277	13962
Actual Yes	413	17391



Conclusion

Our results showed us that predicting winners in League of Legends games are possible and the data is very accurate. I think that the most important features here are emphasized and can help the professionals take that next step in their game. I found the top 3 features to be First Dragon, the Number of towers destroyed and the number of inhibitors destroyed. Our top 2 models were kNN and Multi-Layer Perceptron classifiers with ~96.3% accuracy with 6 features (3 from each team). There is much more room for improvement and there might be more information on who will win the game hidden in more features, however, in this report, we only tried the top 6 and could not find any better combinations. Some of the information may be too accurate because of overfit model problems, this is because we have an accuracy of over 95% for some of the models. Our models predicted the games much higher than the most accurate human predictor of League of Legends games, his accuracy is 62%. I could not find a way to get the confidence of the winner prediction, other than using the percentage of the nearest neighbor in kNN. It is an important point that I did work on this project alone and did not have help from anyone else. This document took a long time to create by myself.

Future Results

In the future, it would be optimal to record matches of League of Legends that my friends play, or other professional streamers and use the model on their game as play. The model would be run every 5 minutes using the in-game stats recorded. This information was not easy to create as matches usually last 30-45 minutes and I would have to at minimum record 10 games, which takes some time to gather. I would also like to look more into the pre-game data classification only using the Champions' data, and I could use this to predict my friend's games as well and record the results. Another aspect is that there are plenty of other classification models that I can try and work with, and many parameters I could spend time dedicated to get higher results. I want to use these trained models with other datasets to test how much we are overfitting to the train data. There needs to be more research on the confidence percentage of the predicted winner, using the nearest neighbor is not enough for the other models. I think that that was one area of the project I struggled on and did not find sufficient research and time to figure this out. The last point would be to work on this with a partner because I didn't have as much time for fine-tuning parameters and research.

References

Arntzen, H., & Hvattum, L. M. (2020). Predicting match outcomes in association football using team ratings and player ratings. *Statistical Modelling*, 1471082X20929881. https://doi.org/10.1177/1471082X20929881

Chouhbi, K. (2020, March 4). *What is like to be a Data Scientist with a passion for Gaming ….* Medium. https://towardsdatascience.com/what-is-like-to-be-a-data-scientist-with-a-passion-for-gaming-43c067ad6415

Claudino, J. G., Capanema, D. de O., de Souza, T. V., Serrão, J. C., Machado Pereira, A. C., & Nassis, G. P. (2019). Current approaches to the use of artificial intelligence for injury risk assessment and performance prediction in team sports: A systematic review. *Sports Medicine - Open*, *5*(1), 28. https://doi.org/10.1186/s40798-019-0202-3

Horvat, T., & Job, J. (2020). The use of machine learning in sport outcome prediction: A review. *WIREs Data Mining and Knowledge Discovery*, *10*(5), e1380. https://doi.org/https://doi.org/10.1002/widm.1380

Loeffelholz, B., Bednar, E., & Bauer, K. W. (2009). Predicting nba games using neural networks. *Journal of Quantitative Analysis in Sports*, *5*(1). https://doi.org/10.2202/1559-0410.1156

Tipify. Gg free skins & free bets giveaway. (n.d.). *Tipify*. Retrieved December 9, 2020, from https://gleam.io/IXSl6/tipifygg-free-skins-free-bets-giveaway

https://link.springer.com/article/10.1186/s40798-019-0202-3
https://towardsdatascience.com/what-is-like-to-be-a-data-scientist-with-a-passion-for-gaming-43c067ad6415
https://www.degruyter.com/view/journals/jqas/5/1/article-jqas.2009.5.1.1156.xml.xml
https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1380
https://journals.sagepub.com/doi/abs/10.1177/1471082X20929881
https://www.tipify.gg/tipsters/?game_filter=58364