Machine Learning Prediction of League of Legends Match Outcomes: A Multifaceted Approach

Megh Bhavsar
Student at the University of Guelph
University of Guelph
Guelph, Ontario
mbhavsar@uoguelph.ca

Richard Dimaria
Student at the University of Guelph
University of Guelph
Guelph, Canada
rdimaria@uoguelph.ca

***Abstract*—Our research explores the application of Machine learning algorithm to accurately predict the outcome of a League of Legends match. By using a diverse set of parameters, we created multiple Machine learning algorithms to analyze and find out different patterns in game dynamics. We used a wide variety of parameters and statistics from 40 percent of the game of 10,000 different league of legends matches to create these models. The parameters include game data such as kills, assists, special monsters killed, turrets destroyed, etc., along with the players experience.**

We will leverage the logistic regression model along with the data from the first 40 percent of the game to create a model that will effectively predict the game outcome. We will then take that dataset and combine it with player-champion experience, mastery the player has with that champion and player level, to try to create a model that is extremely efficient at predicting the outcome of a league of legends match. We will then leverage another neural network model, K-Nearest-Neighbours to compare and contrast the effectiveness of the two models.

The results we got showed the efficacy of our multifaceted approach, showcasing improved accuracy compared to methods we found in articles. As we took a dataset and combined it with more features that we felt would help in creating a more accurate model that predicts outcomes of league of legends matches. We then implemented other models to try to see which model is the best. We believe that our findings will prove to be valuable in League of legends game analytics and trying to predict matches given just 40% of game data.

I. Introduction

League of Legends (LoL) stands as a competitive computer game, holding the title of the most largely played Multiplayer Online Battle Arena (MOBA) globally.[3] The ranked competitive gameplay consists of ten players, organized into two teams of comparable skill that the matchmaking algorithm assembles them into. These opposing teams compete in a battle to destroy the rival team's base in order to win the match. At the beginning of each match, players select a champion; this decision is influenced by the player's skill with that champion. The proficiency a player demonstrates with a particular champion becomes an important factor in their performance. [3] Elevated levels of mechanical expertise give players the ability to thrive in a fast-paced gaming environment, making swift and informed decisions. Once the match has started, there are several factors that can swing the game in either direction. For example, if mid-game a team kills a dragon(a non-playable character), then the resulting team will get a team-wide buff, which increases the team's overall power and helps them win the game. Consequently, a player's mechanical skills significantly influence both individual performance and the ultimate outcome of the match. We feel that being able to predict the outcome of such a complicated game with a lot of different factors will give us a lot of insight on making a model efficient and accurate. Our goal for this research is to create neural network models that takes in various game data from the 40 percent of the game along with player-champion experience and player experience in general, in order to create an extremely accurate neural network that is extremely accurate in predicting the outcome of a League of Legends match.

Motivation and Objectives

A. Motivations

One of the main reasons behind why we chose this topic for our research is due to the fact that we both enjoy playing the game and it is one of our favourite games of all time. We have both played in the competitive scene of he game and realize how complex the game is. We then noticed that if we can create a neural network or any machine learning model that can accurately predict the outcome of a match for such a complex game, we will learn a lot about how machine learning models work and it will be a huge discovery for the competitive community of League of Legends. Being able to predict the outcome of matches for the game we love was enough motivation for us to conduct this research. After some extensive research we discovered that many datasets and models already exist for predicting a match outcome. Although none that use both in game statistics and out of game statistics. So we were motivated by this to create a new dataset and model that uses both.

B. Objectives

The main objective of this research paper is to create a model that uses just game data from 40 percent of the game. Then try to combine game data and player-champion experience and player experience in general in order to create the most accurate model we can. This model will then be used to predict outcomes of League of Legends matches. We will try to implement other types of neural network models in order to compare them and see which model was the best for predicting the outcome of matches.

II. Methodology

In order to conduct this research we will go through and briefly explain the datasets used, tools used, libraries used,

Identify applicable funding agency here. If none, delete this text box.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

algorithms used, parameter initialization methods, steps taken to preprocess the data and build the actual algorithm.

A. Dataset

We used a Zenodo dataset[1] that contained information based on the percentage of the game that was complete, we decided to combine it with the dataset that contained game data from 40 percent of the game as we came to the conclusion that this was the best amount of data that we need in order to create the most accurate model. Anything more would be too much information and the model would be inaccurate and any less would be too little which would also cause the model to be inaccurate. We also used data from the riot api[6] to retrieve information for the matches in the Zenodo dataset, such as player level and player mastery with the champion. Originally we wanted to get more information such as amount of games played with that champion, player-champion win rate, and amount of recent games played with that champion. This was done by using the matchIDs found in the dataset to then make 10000 calls to riots API to fetch additional data required to do this. However due to restrictions with the amount of calls we can make, we could only make 100 calls per 2 minutes, and due to time restrictions we could not add those 3 parameters, as extra calls needed to be made and that would take too long.[5] So instead we settled with the player level and player-champion mastery as we thought that would be enough more information to improve the model.

B. Features

The total amount of features used for our models ended up being 36 for the unmodified dataset, and 40 for the improved dataset. For each feature they're are two versions, one for the blue team and one for the red team. Included here:

Team Champion Kills - A positive integer representing the number of total kills the team had.

Team First Blood - A boolean indicating if the specified team had the first kill in the game.

Team Dragon Kills - A positive representing how many times this team has killed a dragon. Note there are specific types of dragons which are also features. They are, fire, hex tech, air, earth, water, and elder.

Team Tower Kills - A positive integer representing how many enemy towers this team destroyed.

Baron Kills - A positive integer representing how many times this team killed an NPC enemy called the baron.

Rift Herald Kills- A positive integer representing how many enemy towers this team destroyed.

Inhibitor Kills - A positive integer representing how many Inhibitors this team has destroyed.

Team Total Gold - A positive integer representing the sum of the gold each player has on this team.

Minions Killed - A positive integer representing how many enemy team minions were killed .

Jungle Minions Killed - A positive integer representing how many jungle minions this team has killed.

Average Player Level - A float representing the average level of each player on this team.

These last four features were added to make the models predict more accurately and not used for the first logistic regression model.

Total Champion Mastery Points - A positive integer signifying the total sum of each player on this teams lifetime experience as the champion they are playing on. The champion mastery points are acquired by playing matches with that champion.

Total Summoner Level - A positive integer representing the the total sum of each player on this teams summoner level. The summoner level is increased as a player plays any game with any champion. In other words, the sum of each players collective experience playing League of Legends.

All of these features above represented what the state of the game looks like after 40% of the total time has elapsed. [1] The exception are the last four features for summoner level and champion mastery points per team. These are known prior before the match actually begins.

C. Python Tools and Libraries

In order to create the neural network models we had to use various Python tools and libraries to help us accomplish our task. We used Pandas, Numpy, Cborn, Matplotlib, scikit-learn and Keras. We used Pandas to store the datasets into data-frames that we could use to create the models and preprocessing. Used Numpy for any numerical and mathematical operations that were needed. We used Cborn to display data in heat graphs. We used Matplotlib to to plot the data visually so we can get a better understanding of what the results mean. We used scikit-learn for the logistic regression model and getting other helpful data about the model, while Keras was used for the K-Nearest-Neighbours model and getting relevant information for it.

D. Models

For our paper, we decided to explore the use of three different algorithms. These algorithms being Logistic Regression, K-Nearest-Neighbours and Deep Neural Network model.

Logistic Regression is a statistical method that is most commonly used for binary classification and is used to predict the probability of something happening, since our models final goal is to classify who will win and lose, we figured this would be the perfect model for our research. For implementing this model with our dataset we used the python library sciket-learn.[2]

K-Nearest-Neighbours is a simple machine learning algorithm that is commonly used for classification and regression, since our objective revolves around classification we decided to use this model. We used sciket-learn tool on python to implement this model.[7]

Deep neural network model, is an artificial neural network with multiple hidden layers in between the input and output layers. Here we used tensor flows library Keras a machine learning API, to implement a deep neural network model. We stuck to the same hyper parameters as found in the first paper as we felt that would work the best and then tinkered them to see if we could make our model more accurate.[8]

E. Parameter Initialization Methods

The two machine learning algorithms don't require Parameter Initialization method. However for the deep neural network model we used the He normal initialization. In this initialization method weights are initialized with values from a gaussian distribution with a mean of 0 and variance of $2/n_{in}$ for the ELU activation method.[12]

F. *Steps Taken to Preprocess the Data*

The 3 models required a some preprocessing in order for the models to work. For all of the models we trimmed the useless data such as, time in milliseconds, time percents, winner and match id as these parameters would not affect who would win the game at all, and if the model knows the winner, then there is no point in having the model at all. Next we split the dataset so that 80% of it was training data and 10% of it was testing data and the other 10% was the validation data set, this gives us 8,000 training data and 2000 testing data. For the K-Nearest-Neighbours model we had to apply the standard scaler, this standardizes features of the dataset. This preprocessing method transforms the dataset so it ends up with a mean of 0 and standard deviation of 1, also known as z-score normalization. For the Deep neural network model we converted all of the xtrain and xtest data from integer to float. Once the preprocessing was finished we were able to implement the algorithm using the tools and libraries mentioned earlier.

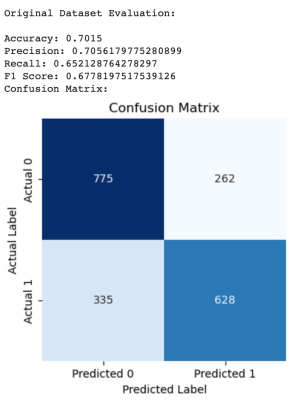
III. RESULTS

A. *Hyperparameters*

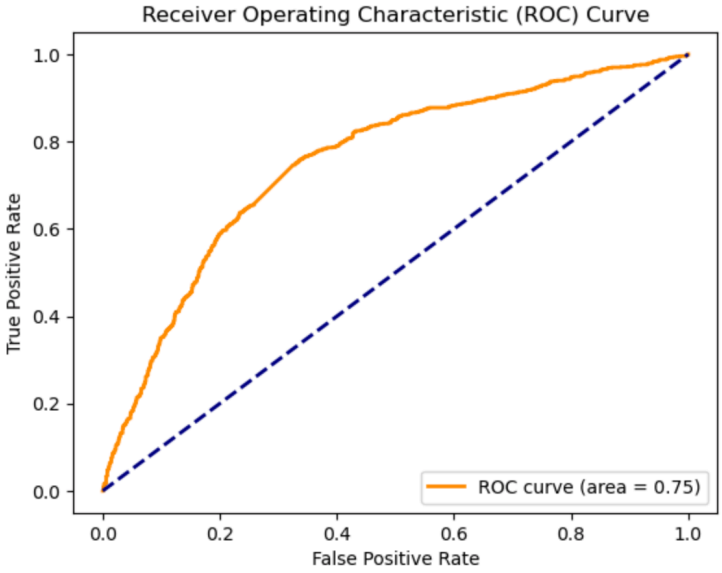
The logistic regression algorithm did not require any hyper-parameters. For our K-nearest neighbours algorithm However for the deep neural network we had the Learning rate set to default, epochs set to 30 and batch size set to 30. Moving, we had 5 dropout layers with a dropout rate of 0.69, 5 normalization layer set to batch normalization and 5 dense layers. Each of these dense layers used exponential linear unit activation and he initialization. The last layer was a 1x1 dense layer with sigmoid activation. These were the same hyper parameters used in the arxiv article.[3]

B. *Algorithm Performance*

The accuracy for the model with just the in game data had an accuracy of 70%, we can call this model 1. When it came to performance of the algorithms that used game data and player champion experience, we noticed that the best algorithm was the logistic regression model at 93% accuracy, we can call this model 2. While the deep neural network had the second best performance at 90% accuracy, we can call this model 3. Lastly, the K-Nearest-Neighbours model with only 67.8% accuracy, we can call this model 4. In addition, in order to better visualize our results we created an abundance of graphs and plots.



This figure shows the different statistics and confusion matrix for the logistic regression model that just used 40% of the game data to predict the outcome. From the confusion matrix we can see that the amount of instances that were correctly predicted was higher than the instances that were not correctly predicted.

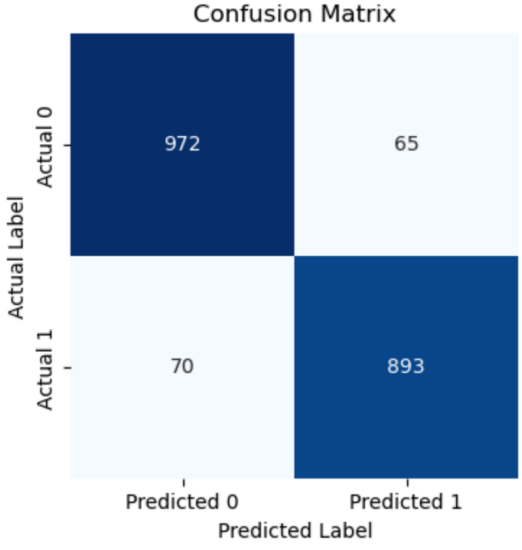


AUC-ROC Score: 0.7525191987831341

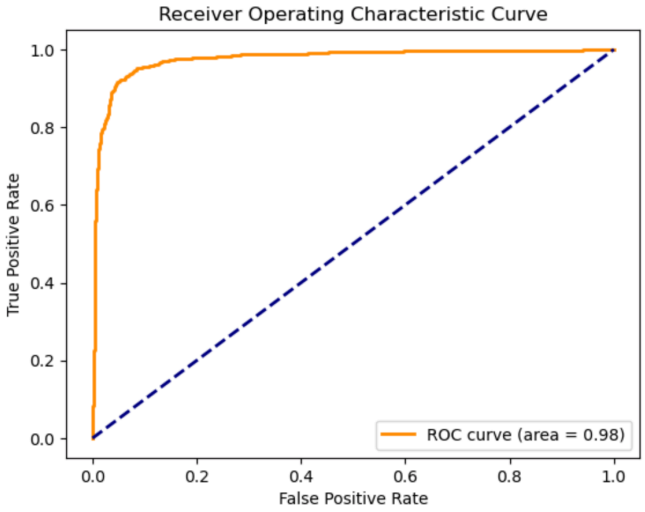
This figure shows the Receiver Operating Characteristic (ROC) curve for the logistic regression model that only used the game data. The ROC curve is the visual representation of the true positive rate versus the false positivity rate. The Area under the ROC curve (AUC) is a metric that summarizes the performance of a model across all possible thresholds, ranging from 0 to 1, where 0.5 represents a random classifier and 1 represents a perfect classifier. A score of 0.7525 shows that model performs decently well in trying to predict the match outcome accurately. Since the curve is closer to the top left of the graph, this means that the model is more accurate.

Modified Dataset Evaluation:

Accuracy: 0.9325
Precision: 0.9321503131524008
Recall: 0.9273104880581516
F1 Score: 0.9297241020301926
Confusion Matrix:



This figure shows the different statistics and confusion matrix for the logistic regression model that used 40% of the game data and the player-champion experience to predict the outcome. From the confusion matrix we can see that the amount of instances that were correctly predicted was way higher than the instances that were not correctly predicted. We can conclude this model was extremely accurate



AUC-ROC Score: 0.9760672360461472

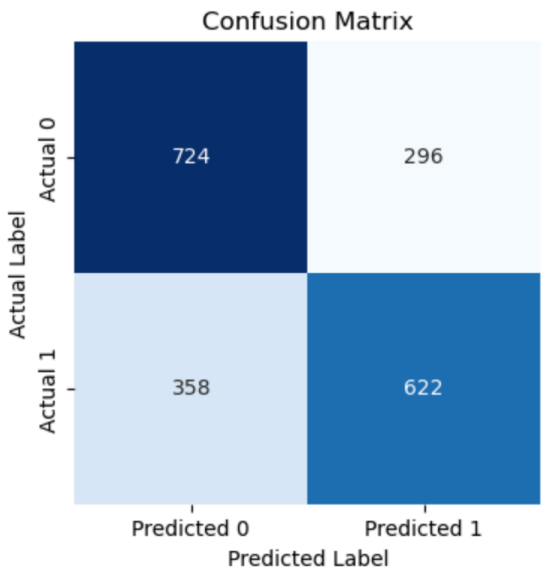
This figure shows the Receiver Operating Characteristic (ROC) curve for the logistic regression model that used the game data and player-champion experience data from the riot api. The ROC curve is the visual representation of the true positive rate versus the false positivity rate. A score of 0.98 shows that model performs extremely well in trying to predict the match outcome accurately. Since the curve is extremely close to the top left of the graph, this means that the model is very accurate.

Modified Dataset Evaluation:

Accuracy: 0.678
Precision: 0.6656282450674974
Recall: 0.6656282450674974
F1 Score: 0.6656282450674974
Confusion Matrix:

This figure shows the different statistics for model 3, as we can see this was the worst model in terms of accuracy, precision, Recall and F1 Score.

Confusion Matrix:



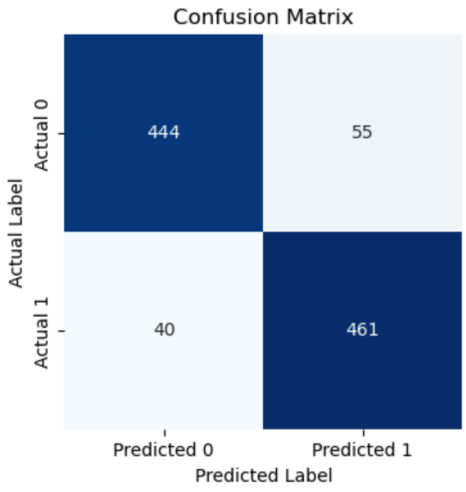
This figure is the confusion matrix for the third model, we can see that the amount of prediction the model got wrong were a lot more than the logistic regression, so we can see that this model is not as accurate

Accuracy: 0.909
Precision: 0.8802336903602727
Recall: 0.9387331256490135
F1 Score: 0.9085427135678392

This figure shows the different statistics for model 4, as we can see this was the second best model in terms of accuracy. The confusion matrix shows that it predicted correctly for almost all instances of data.

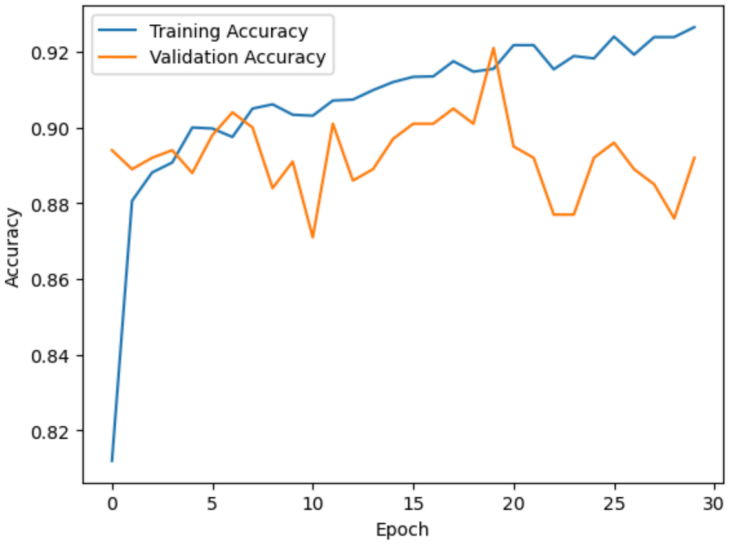
This figure is the confusion matrix for the fourth model,

Confusion Matrix:



we can see that the amount of prediction the model got most

of the predictions correct, so we can see that it is very accurate.



This figure represents the training accuracy and the validation accuracy over each epoch. The training accuracy is a metric that measures the percentage of correctly classified instances in the training data set. It indicates how well the model is fitting to the training data. In the early stages of training, typically it should quickly increase as the model learns the data, which is what we can see here. As for the validation accuracy it is similar to the training accuracy except it is the separate validation data set that has not been seen during training. The validation accuracy should tend to increase as the model learns the training data, however this model is plateauing and decreasing in our validation set, this means that it is overfitting the data.[13]

C. *Comparative Analysis Results*

Based on the results, we can clearly see that the logistic regression model (model 2) was the most accurate, had the highest precision, Recall and F1 Score among all of the models we created. It had the highest accuracy at 93 percent, the closest to it was the deep neural network with a 90 percent accuracy. The logistic regression model also had the highest correct position predictions with 972, while the deep neural network had the highest correct negative predictions. The logistic regression model (model 1) had the highest precision at 93% compared to the second highest, deep neural network at 88%. However, the deep neural network had the highest recall at 93%, compared to the logistic regression model (model 1) at 92%. From these stats we can confidently say that the logistic regression model works with player-champion experience and player experience in general is the best model.

IV. DISCUSSION

A. *Findings*

From all of the research we conducted and models we made we had a lot of key findings that we would like to mention. We found out that the level the player is and the mastery of that player with the champion they are playing in the current match is extremely important to finding out which team will win the game. This is due to the fact that the logistic regression with just the game data for 40 percent of the game had an accuracy of 70%. Once those two parameters were added to the dataset, the accuracy jumped to 93 percent. Adding these features drastically effective the accuracy. We also found out that out of the three algorithms we used, Logistic Regression, K-Nearest-Neighbours and Deep Neural Network, Logistic regression worked the best for predicting out the match outcome. This could be due to the fact that logistic regression is a simple yet affective algorithm when it comes to binary classification. Our research confirms that the logistic regression model is the best for this type of binary classification. We also found out that oddly enough the K-Nearest-Neighbours model with the player-champion experience data somehow had a worse accuracy then the logistic regression model that did not have the player-champion experience. This tells us that K-Nearest-Model is not that useful for binary classification. There will never be a perfect algorithm for determining the match outcome this is due to other factors that we cannot account for in a machine learning model. These factors include a player can have a bad game, even if the player is more experienced with their champion than the player they are playing against, and has more time in the game in general, that player could just have a bad game and lose anyways. Another factor that could also affect which team wins is if a champion is counter picked by an opposing team player, this could be detrimental for the team, causing them to lose. Our model does not account for the players matchup history, meaning if the player lanes against a particular champion they may not be as effective in their role. However we did not touch on this as that would require to many calls to the riot api. Since we are only allowed to make 100 calls every 2 minutes with they're built in throttling. Making 110,000 calls would take a very long time. We found that 11 for the number of neighbours yielded the highest accuracy yet the algorithm was still underperforming. We left the rest of the parameters for K-nearest neighbours as the default because no matter how we tuned the parameters this model was not predicting our test data better. With all of these findings we can safely say that our research and creating a model that can predict the outcome of a game was successful and those who want to explore more in this field have a better stepping stone to get started.

B. *Interpretation of Results*

Based on the results we can see that since the Logistic regression model that uses our new dataset with extra features had the highest accuracy. This means that when predicting the outcome of a League of Legends game, this model will be the best at doing so given knowing what the stats look like at 40% of the elapsed time. This model also had the best confusion matrix score, with the most outcomes predicted correctly. We can also see the correlation between adding the mastery a player has with the champion they are playing along with the current level, increases the accuracy of the logistic model drastically. This is just as we hypothesized, meaning that as the player has more experience in the game and the champion they are playing, they are more likely to perform better, overall resulting in their team winning the game. We can also see that using just 40 percent of game data to predict the game outcome showed that it was still decently accurate at 70 percent. This was expected as from our previous referenced paper they had similar results. Overall we can confirm that our hypothesis was correct, accuracy in predicting the winning team in a League of Legends game went up drastically when the two experience parameters were added. As we can see from the results K-Nearest-Neighbours model had the worst accuracy,

precision, recall and f1 score, this tells us that when using this algorithm as a binary classifier, the results will not be accurate. From this we can conclude that the driving factors in determining the game outcome of a league of legends rank match were the players on each teams overall experience with their champions and their total experience in league of legend games. The deep neural network was also very accurate, as we added depth in the layers.

C. *Pros and Cons of Implemented Solution*

Overall this solution has been quite beneficial in creating model that accurately predicts the outcome of a League of Legends game. One pro of our methodology is that, since we did a Logistic Regression just for the game data and then again for game data and player-champion experience and game experience added, we can compare the results of the two models and check to see if adding these extra features actually increased the accuracy of the model, and it did. Another pro is that since we did 2 machine learning models and a deep neural network model, we are able to compare these models and their accuracies and see which ones performed the best and which ones performed the worst. So our solution involves adding on to models to refine them and compare and contrast different models. Another pro is that since we made an abundance of graphs and plots we are able to visualize the data better and actually see how the models performed better. A con of our solution is that since we were relying on the riot api to receive the extra features, we weren't able to get some of them such as amount of games played with that champion, player-champion win rate, and amount of recent games played with that champion, due to the 100 calls per 2 minutes restriction. We believe that if we were able to add these features our model would have been even more accurate. Another con to our solution is that, when trying to predict who will win the game, we don't account for other factors, such as the player having a bad game, if a champion a player is playing is being countered by a champion on the other team. Countering a champion means that the other champions move sets, prevent the other champion using their moves effectively. Since these factors have not been taken into account, our model will never truly be accurate. Overall the pros outweigh the cons, so overall our solution was a success and was able to produce a model that is 93 percent accurate in predicting match outcome which was higher than the other articles we looked at.

V. CONCLUSION

In conclusion, after creating all the models, getting their results, comparing them, we can conclude that logistic regression was the best model for predicting the outcome of the game. We are also able to conclude that adding the mastery of the player with the champion they are playing and overall level in the game, the accuracy of the model increased drastically. So, a players experience is directly correlated in trying to predict the outcome of a game. The deep neural network model was overfitting the data and only receiving a 90 accuracy overall, meaning that with more tinkering the model could increase in accuracy even more, to possibly exceed the logistic regression model. In conclusion League of Legends is Avery complex game and there are so many factors that affect a match, such as in game data, experience, players having bad games, getting countered, however I using game data from 40 percent of the game and players experience with the champion along with the players total level, helped us produce a model that is 93 percent accurate in predicting the outcome of any league of legends match. So we can conclude that our hypothesis was correct, and our research was successful. In conclusion, the accuracy we got with our best model was the best accuracy even compared to the other models in the two articles that we based our paper off of.

REFERENCES

1. Junior, J. B. da S., & Campelo, C. (2023, August 31). *League of Legends Match Data at various time intervals*. Zenodo. https://zenodo.org/records/8303397
2. Brownlee, J. (2023, December 5). *Logistic regression for machine learning*. MachineLearningMastery.com. https://machinelearningmastery.com/logistic-regression-for-machine-learning/
3. Using machine learning to predict game outcomes based on player ... (n.d.-a). https://arxiv.org/pdf/2108.02799
4. *Riot Developer Portal*. (n.d.). https://developer.riotgames.com/
5. *Riot Developer Portal*. (n.d.-b). https://developer.riotgames.com/docs/portal
6. *Riot Developer Portal*. (n.d.-c). https://developer.riotgames.com/docs/lol
7. Harrison, O. (2019, July 14). Machine Learning Basics with the K-Nearest Neighbors Algorithm. *Medium*. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
8. Kunapuli, S. S., & Bhallamudi, P. C. (2021). A review of deep learning models for medical diagnosis. In *Elsevier eBooks* (pp. 389–404). https://doi.org/10.1016/b978-0-12-821777-1.00007-0
9. *News - League of Legends*. (n.d.). https://na.leagueoflegends.com/en-us/news/
10. Lee, S., Hong, S., & Yang, S. (2020). Predicting Game Outcome in Multiplayer Online Battle Arena Games. *Predicting Game Outcome in Multiplayer Online Battle Arena Games*. https://doi.org/10.1109/ictc49870.2020.9289254
11. *League of Legends Diamond Ranked Games (10 min)*. (2020, April 13). Kaggle. https://www.kaggle.com/datasets/bobbyscience/league-of-legends-diamond-ranked-games-10-min
12. *Papers with Code - ELU Explained*. (n.d.). https://paperswithcode.com/method/elu
13. *The differences between training, validation & test datasets*. (n.d.). Kili-website. https://kili-technology.com/training-data/training-validation-and-test-sets-how-to-split-machine-learning-data