PROJECT: Victory
A Machine Learning Approach to Predicting Game Outcomes

- Aaron Dominic DeCosta

Introduction and Inspiration

As a long-time player of the popular MOBA game League of Legends and an avid viewer of the LoL Esports scene, there was a strong urge to apply Machine Learning models to the game to make predictions on which team would win a game of LoL. This year's World Championships did just that by introducing a "Win Probability" metric. Using live data from professional games, the model made predictions as to which team would win the game as time went on.

According to the Riot Games developer, John "Riot JPham" Pham, WP is a machine learning-based tool that indicates how often teams won given a similar situation previously. WP in League's case is expressed as a percentage and is calculated by comparing the current game situation to similar historical situations.

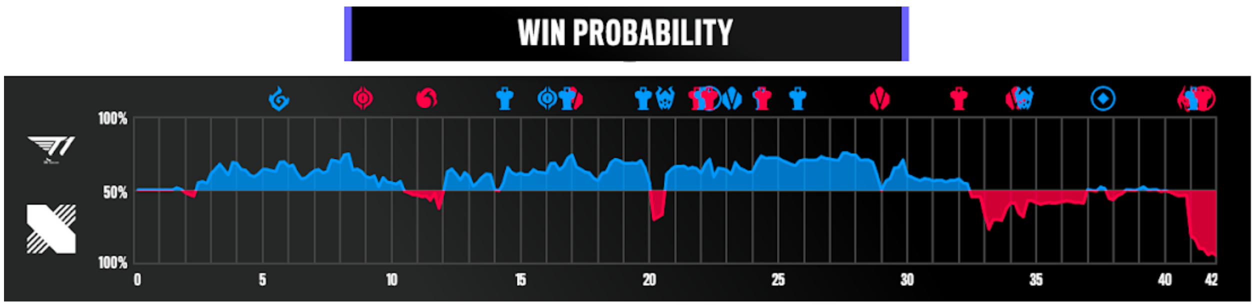

Figure 1. Win Probability model developed by Riot Games. It depicts the probability of winning for the winning team over time.

Over the course of the tournament, this feature implemented by Riot Games, has received heavy backlash from the fanbase, suggesting that model does not produce any worthwhile results until the end of the game, where win percentage reaches a 100%. This project seeks to examine said claims.

Using Riot Games' Win Probability as inspiration, three models were trained: A Linear regression model, a Logistic regression model, and a Softmax regression model, using game data from this year's LoL Esports competitions to predict the winner of a LoL Esports game given the game's state. However, unlike the tool developed by Riot Games, the models developed only output a binary yes or no as to whether the team in question will win, given the current state of the game. The three models use data from professional games and train with their respective loss functions using Mini-Batch Stochastic Gradient Descent to learn a prediction function.

We hypothesize that the linear regression model will have the worst test performance with the lowest accuracy. The logistic regression and softmax regression models using their respective cross-entropy losses will have approximately equal performances on the test with similar accuracies. This is because linear regression models perform better in producing a continuous variable outcome compared to the other two models which learn a binary predictor.

Problem Formulation

The models developed, in contrast to the Win Probability model which draws a percentage, solve a Binary Classification problem. The input to the models would be some parameters related to the game, which is discussed in the next section. Given the state of the game and the team in question, the models would output a 1 if the team was more likely to win the game, or a 0 if the team was more likely to lose the game.

Each model will use their respective loss functions for their gradient, ie; linear regression will use mean squared error gradient, logistic regression, and softmax regression will use cross-entropy loss function gradient. When calculating the losses, each model will dichotomize their predictions to be either a 1 if the prediction is $\geqslant 0.5$ or a 0 otherwise.

The live data used by Riot Games during the Worlds 2023 Event has been made available only to entrusted third-party developers through the Bayes Esports data provider. Thus the game data used to train our models is constrained to only the mid-game and end-game data, instead of a per-second basis.

As a result, we use two different data sets to predict which team will win the game. The first dataset is the mid-game dataset which includes data of the first 20 minutes of the game. The second dataset is the end-game data, which includes all relevant data up until the game is finished.

Splitting the dataset this way also comes with the advantage of making better predictions as the game progresses. This is because as the game progresses towards the end, one team will generally build up more of an advantage for themselves, be it through getting more turrets, drakes, or barons.

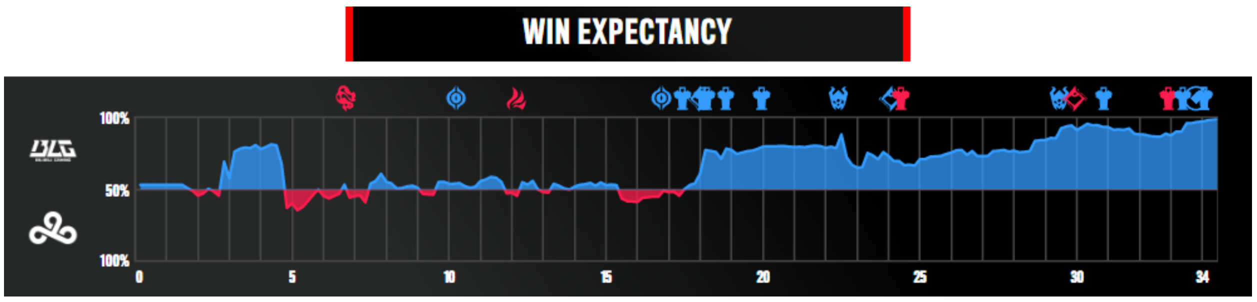

Figure 2. Win Probability graph for the final game between C9 and BLG at MSI 2023. The win percentage at the mid-game stages is relatively even hovering around a 50-50 split. The win percentage skews more toward BLG as the game progresses towards the final minutes.

The Win Probability metric employed by Riot Games uses the following parameters to train their model:

- Game time (the in-game time)
- Gold % (player gold / total gold in-game)
- Total team XP
- # of players alive
- Tower kills
- Dragon kills (whether a team has a dragon soul or not)
- Herald trinket in Inventory
- Inhibitor timers (how long until an inhibitor respawns)
- Baron timers (time until Baron buff expires for the team)
- Elder timer (time until Elder Dragon buff expires for the team)
- # of players with Baron active
- # of players with Elder active

The last 5 parameters used are constantly changing with game times and hence we cannot use them to train our models. With the split we have in data between mid-game and end-game data, we use a different set of parameters to train the three models.

For the mid-game data, we use the following parameters to train the predictor:

- side: blue side or red side, mapped as 0 or 1
- first blood: whether the team was able to secure the first kills
- first dragon: whether the team was able to secure the first dragon
- first herald: whether the team was able to secure the first herald
- first baron: whether the team was able to secure the first baron
- first tower: whether the team was able to secure the first tower
- first mid-tower: whether the team was able to secure the first mid-tower
- first to 3 towers: whether the team was able to secure 3 towers first
- turret plates: # turret plates was the team able to secure (14 min)
- opp turret plates: # turret plate was the enemy able to secure (14 min)
- gold diff at 10: gold difference at 10 mins
- xp diff at 10: experience difference at 10 mins

- cs diff at 10: difference in minion kills at 10 mins
- kills at 10: kills at 10 mins
- assists at 10: assists at 10 mins
- opp kills at 10: enemy team's kills at 10 mins
- opp assists at 10: enemy team's assists at 10 mins
- gold diff at 15: gold difference at 15 mins
- xp diff at 15: experience difference at 15 mins
- cs diff at 15: difference in minion kills at 15 mins
- kills at 15: kills at 15 mins
- assists at 15: assists at 15 mins
- opp kills at 15: enemy team's kills at 15 mins
- opp assists at 15: enemy team's assists at 15 mins

For the end-game data, we use the following parameters to train the predictor:

- side : blue side or red side, mapped as 0 or 1
- kills: # kills
- deaths: # deaths
- assists: # assists
- dragons: # dragons taken by the team
- opp_dragons: # dragons taken by the enemy team
- elders: # elder dragons taken by the team
- opp_elders: # elder dragons taken by the enemy team
- heralds: # heralds taken by the team
- opp_heralds: # heralds taken by the enemy team
- barons: # barons taken by the team
- opp_barons: # barons taken by the enemy team
- towers: # towers taken by the team
- opp_towers: # towers taken by the enemy team
- inhibitors: # inhibitors taken by the team
- opp_inhibitors: # inhibitors taken by the enemy team
- gold_diff: # gold difference between the two teams

The models were trained using the two different sets of game data from the LEC, LCK, LCS, and MSI 2023 seasons. This data was made available to the public by Tim Sevenhuysen, an independent developer who trained a similar model on their site, Oracle Elixir.

Using this data, 2470 samples (2 per game; win and loss samples) were recorded for each dataset of which 1482 samples were used for training the models, 494 samples for validation, and 494 samples for testing, making a 60-20-20 split of the sampled data.

Approaches and Baselines

For the three models, we used the following hyperparameters:

- learning rate (alpha): the rate at which updates are made to the weights
- l2 decay (lambda): constraints on the magnitude of the weights
- epoch (max_epoch): the number of epochs to attain convergence
- batch-size: number of data points in each bach

Each of these hyperparameters was first tuned to attain convergence for the linear regression model. Convergence here refers to the highest possible accuracy achieved through the validation, which was used as an evaluation metric. The best decay values were obtained by comparing the best validation accuracies, which were calculated after each epoch. The validation set used was {5, 1, 0.5, 0.1, 0.05, 0.01, 0.005, 0.001, 0.0}

For linear regression, the learning rate, alpha, was tuned to be very low as it appeared that the gradient for the curve was very steep, leading to high magnitudes for the weights. With the low learning rate, the number of epochs and batches were tuned such that convergence was reached by the end of the epoch cycling. Furthermore, as the magnitude of the weights was very low without any regularization, we used a very low decay to prevent relevant features from being excluded.

The same hyperparameters and their values were then used for the logistic regression and softmax regression models. The learning rate was tuned to be faster as convergence was found very late into the epochs despite decreasing with each epoch. Additionally, the best decay was found to be zero, which is explained by the very low-ordered magnitudes found for the weights yielding maximum accuracy. For the softmax regression model, the batch size was increased by a factor of 5, to improve runtimes, however, this did not compromise the validation accuracy.

The best test performance for linear regression was obtained using the following tuned hyperparameters:

- learning rate (alpha): 0.0001
- l2 decay (lambda): 0.01
- epoch (max_epoch): 1000
- batch-size: 10

The best test performance for logistic regression was obtained using the following tuned hyperparameters:

- learning rate (alpha): 0.001
- l2 decay (lambda): 0.0
- epoch (max_epoch): 1000
- batch-size: 10

The best test performance for softmax regression was obtained using the following tuned hyperparameters:

- learning rate (alpha): 0.001
- l2 decay (lambda): 0.0
- epoch (max_epoch): 1000
- batch-size: 50

Evaluation Metrics

The three models implemented were evaluated based on their accuracy on the test split of the datasets; the higher the accuracy the better the performance. The accuracy as an evaluation metric represents the real goal of the task: to make accurate predictions on whether the team will win or not, given the state of the game. A higher accuracy model would thus be a “better” model.

Since we have a balanced data set with a “Win” sample and a “Loss” sample for each game recorded, there is no skewing towards one half of the classification outcomes. Thus accuracy is a good measure of the performance for the dataset used.

Results and Discussion

In our findings, using the three models to predict which team will win, we find that all three models performed significantly better than the trivial baseline accuracy of the binary classification, which is 50% accuracy.

For the mid-game dataset, using data from the first 15-20 minutes of the games, the logistic regression and softmax regression models had the best test performance with accuracies of 83.2% . The linear regression model did not trail far behind, having a test accuracy of 82% . The similarity in accuracies between the cross-entropy loss models and the mean-squared error model shows that

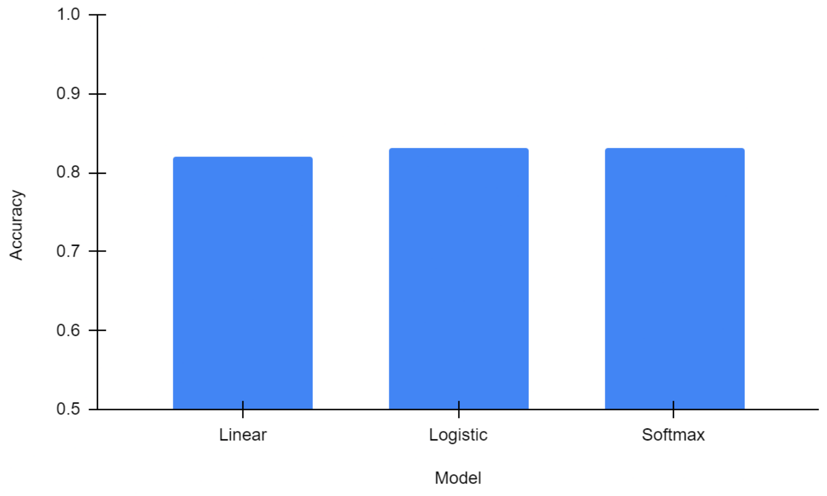

Figure 3. Bar Chart representing test accuracies of the three different models for mid-game dataset. The difference between Linear and both Logistic and Softmax is very insignificant.

For the end-game dataset, using data from the ending portions of the game, the logistic model and softmax model both had 98.8% . The linear regression model had a higher accuracy of 99% .

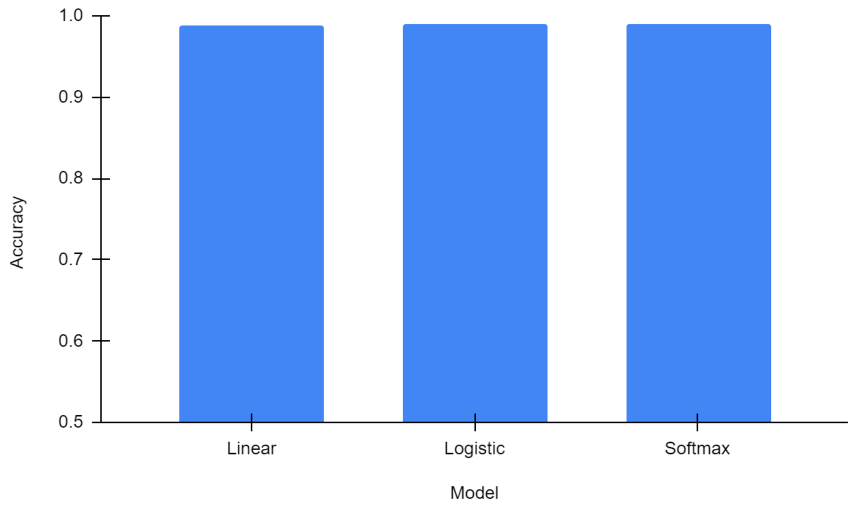

Figure 4. Bar Chart representing test accuracies of the three different models. The difference between Linear and both Logistic and Softmax is very insignificant.

Contrary to our hypothesis, the finding that the linear regression model has a higher test accuracy than the logistic and softmax regression models indicates that the end-game input features used to make the predictions have an approximate linear impact on the binary outcome. The minute disparity in accuracies for the mid-game dataset also illustrate a linear relationship between the outcome’s odds and the feature set.

The near-perfect similarity between the logistic model’s accuracy and the softmax model’s accuracy confirms our hypothesis.

The near 20% change in model accuracy in all three cases explains that shift in win probability as seen in Figure 2. This seems to lead toward the suspicion that the Win Probability model does not produce meaningful results. However, the above 80% accuracy shows that meaningful predictions can be made before the end of the game, and within the 20 minute mark.

The statistic of 80% accuracy also implies that around 80% of the games could be accurately predicted from within the first 15-20 minutes of the game. From the game-design and esports perspective, this would mean a lot of the games broadcasted would be un-entertaining for the viewers at home, as the general consensus is that closer games are more enjoyable compared to one-sided “stomps”.

Though the 80% and 99% accuracies are reasonably decent for the problem in question, many improvements can be made to the model. One such improvement would be introducing lane-matchups, using head-to-head win-rate between the players and their champions. Employing the live data-sets used by Riot Games would also improve accuracies as it would include factors such as jungle proximity and other such statistics. While the data-set focuses primarily on the features developed throughout the game, using data-sets with historical data of the players and their performance in recent times could also help improve accuracy.

Bibliography

Lol Esports. (n.d.). https://lolesports.com/article/dev-diary-win-probability-powered-by-aws-at-worlds/blt403ee07f98e2e0fc

Glossary

Baron:	Refers to Baron Nashor, a neutral objective. Defeating Baron grants powerful buffs to the team.
Drakes:	Short for Elemental Drakes, a neutral objective that grants various elemental buffs to the team.
Elder:	Short for Elder Dragon, a neutral objective. Defeating Elder Dragon grants a team-enhancing buff.
End-Game:	The late stage of a League of Legends match where teams focus on completing objectives and making strategic plays to secure victory.
Gold:	The in-game currency used to buy items for champions, earned by killing minions, monsters, turrets, and enemies.
Herald:	Short for Rift Herald, a neutral objective. Usually taken around the 10 to 20 minute mark.
Inhibitor:	Structures in the enemy base that need to be destroyed to win the game.
Mid-Game:	The phase of the game that occurs between 15-25 minutes.
Minions:	AI-controlled units that spawn regularly and march down the lanes. Players earn gold by killing minions. Kill count on minions is referred to as CS.
MOBA:	Acronym for Multiplayer Online Battle Arena, a genre of video games.
PROJECT:	A popular cosmetics line within the game.
Stomps:	Refers to a one-sided match where one team dominates the other, often resulting in a quick and decisive victory.
Turrets:	Structures located along each lane that must be destroyed to advance and gain control of the game.