Actionable League of Legends Game Outcome Prediction Using Machine Learning Methods

Ryan Stevens, Elle Isabel Angell, Nick Li, Coleman Garnier

Abstract

This project targets multiple different ML approaches (GLM, SVM, RF) to create stable models for prediction and assessment of League of Legends match outcome. Models were assessed across multiple cross validation folds that inform over 75% predictive success on our sample data. These models, chosen for interpretability, are able to provide gameplay insights for specific games using either inherent qualities–like gini impurity for the random forest–or with the assistance of SHAP (SHapley Additive exPlanations), a method that uses monte carlo sampling to provide the additive sum of contribution of each predictor to the overall outcome. After removing colinear terms like turrets destroyed, we found that deaths and gold earned were the two most powerful predictors of gameplay outcome. We believe this is because these factors align with a gameplay style built on consistency. This is further reinforced by our findings that so called "first blood factors" (a one-time event that often occurs toward the beginning of the game) were poor predictors of the final game outcome. This is counter to general community sentiment, where some players will refuse to continue playing the match after one team secures first blood objectives. Considering these factors, our findings can help players achieve higher average win rates by focusing on consistent and measurable improvements in their gameplay quality.

1. Introduction

The game being focused on for this project is League of Legends, a complex, competitive 5v5 multiplayer online battle arena (MOBA) game. Over the last decade, League of Legends has cemented itself as the most popular esport title, attracting millions of players each day to compete for their spot at the top. The game has extremely high global viewership for their annual tournament series. In 2024, the World Finals games reached a peak concurrent global viewership of over 50,000,000 according to lolesports [6]. As such, the value in applying statistical modelling techniques to gain a competitive advantage is high.

The company that created League of Legends, Riot Games, provides high-quality data through its API, gaining access to game data such as player performance and objectives. While there are existing websites to track player statistics such as match history and specific counter pick options, they mainly focus on surface-level statistics and outcomes, not how one's decisions can influence the result of the game.

The main focus of this project is to analyze and interpret player behavior and decision-making to determine how these patterns may impact the outcome of the game. By examining factors such as participation levels, whether their team had more objectives, and experience/gold levels over the course of the game, key behavioral patterns and the player's impact on the game can be interpreted and identified. With these insights, it may be possible to assist professional teams or individual players to outperform the competition.

Table 1: Datasets

Data	Size	Type
Small Dataset	n=250 (2,500 players)	Matches, Timelines
Large Dataset	n=5000 (50,000 players)	Matches, Timelines
Matches Dataset	n=50,000 (500,000 players)	Matches

2. Methods and Analysis

2.1 Data

All data was gathered through the Riot games developer API on patch 25.22, November 1st through 3rd, 2025. This was accomplished using a data collector written in python that collected corresponding game data from two different api endpoints. First, the Riot match data, which is composed of end-of-game summary statistics. This includes which team won the match, the characters (or Champions) played by each team, the ending gold quantity for each team, and other descriptive statistics. Second, the Riot timeline data, which includes snapshots of the game state in one-minute intervals throughout the match. This data includes (for every minute in the game) information such as the position of each player or inventory of each player. Finally, all data was collected from games played where the average MMR (match making rating) of the participating players was 'Diamond II', which consists of players in approximately the top 1.5% of skill rating. The primary motivation for this choice was to focus on high-quality yet still plentiful match data. Had a lower skill bracket been selected, the results would be less interpret-able within the context of an organized environment. Had a higher skill bracket been selected, data sparsity would have likely become a problem.

Using the data described above, three primary data sets were collected for modelling and exploring data. First, a small set of n=250 matches (2,500 total players) was used for general data analysis and data exploration this was the 'small dataset'. Second, a larger set of n=5,000 (50,000 total players) made up the 'large dataset'. Both the small and large datasets consist of combined match and timeline data. Finally, a matches-only dataset n=50,000 (500,000 total players) collected on the "Diamond IV" skill bracket was used to gain insights on the impact of team composition on the overall win but did not directly contribute to our results.

2.2 Preprocessing

The Riot Games developer API returns a highly nested sequence of DTOs (Data Transfer Objects). These structures contain match level, team level, as well as player level level data embedded within multiple, tightly coupled transfer layers, requiring complicated extraction before modeling. All desired JSON responses were flattened into a simpler form by isolating the relevant participant objects and merging them with corresponding match metadata. To accomplish this, data was cleaned using a python script (matchcleaner.py).

```
##  [1] "deaths"               "dragonKills"          "baronKills"
##  [4] "firstBloodKill"       "firstBloodAssist"     "kills"
##  [7] "lane"                 "neutralMinionsKilled" "visionScore"
## [10] "goldEarned"           "timeCCingOthers"      "enemyMissingPings"
## [13] "clearedPings"         "totalPings"           "wardsPlaced"
## [16] "wardsKilled"          "detectorPlaced"       "duration"
## [19] "win"
```

2.3 Model Evaluation

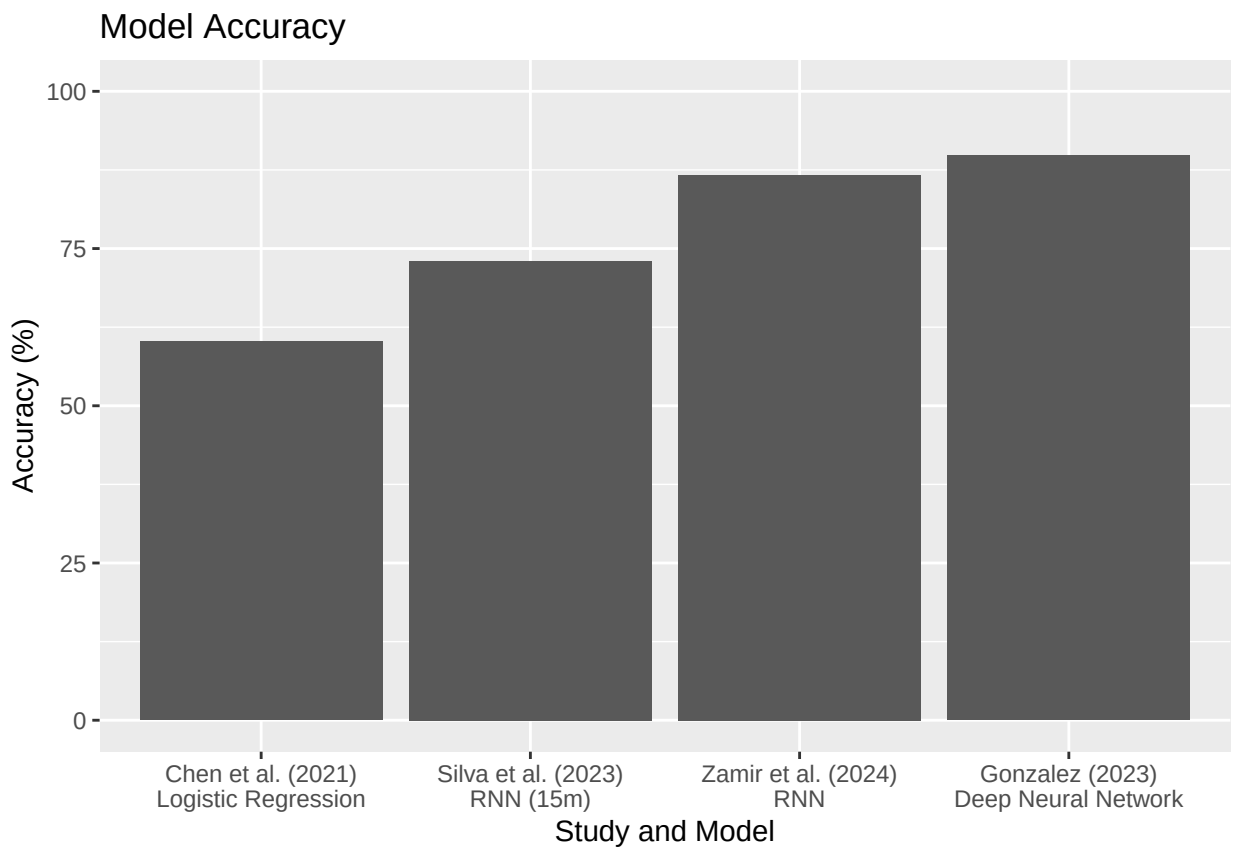
Among studies published performing similar League of Legends prediction tasks with similar datasets, top performing models typically used some sort of nonlinear architecture such as an RNN [4], or other form

of deep neural network [3]. However, such an approach is not acceptable for our purposes as the extra predictive power comes at too great a detriment of interpretability. [5] Analyzed a set of model performances on NBA data and found that tree-based models such as random forests outperformed non-tree alternatives. Furthermore, a birds-eye analysis of remaining League of Legends-based papers reveals a similar story: Random forests and logistic regression often comprise the top performing linear models.

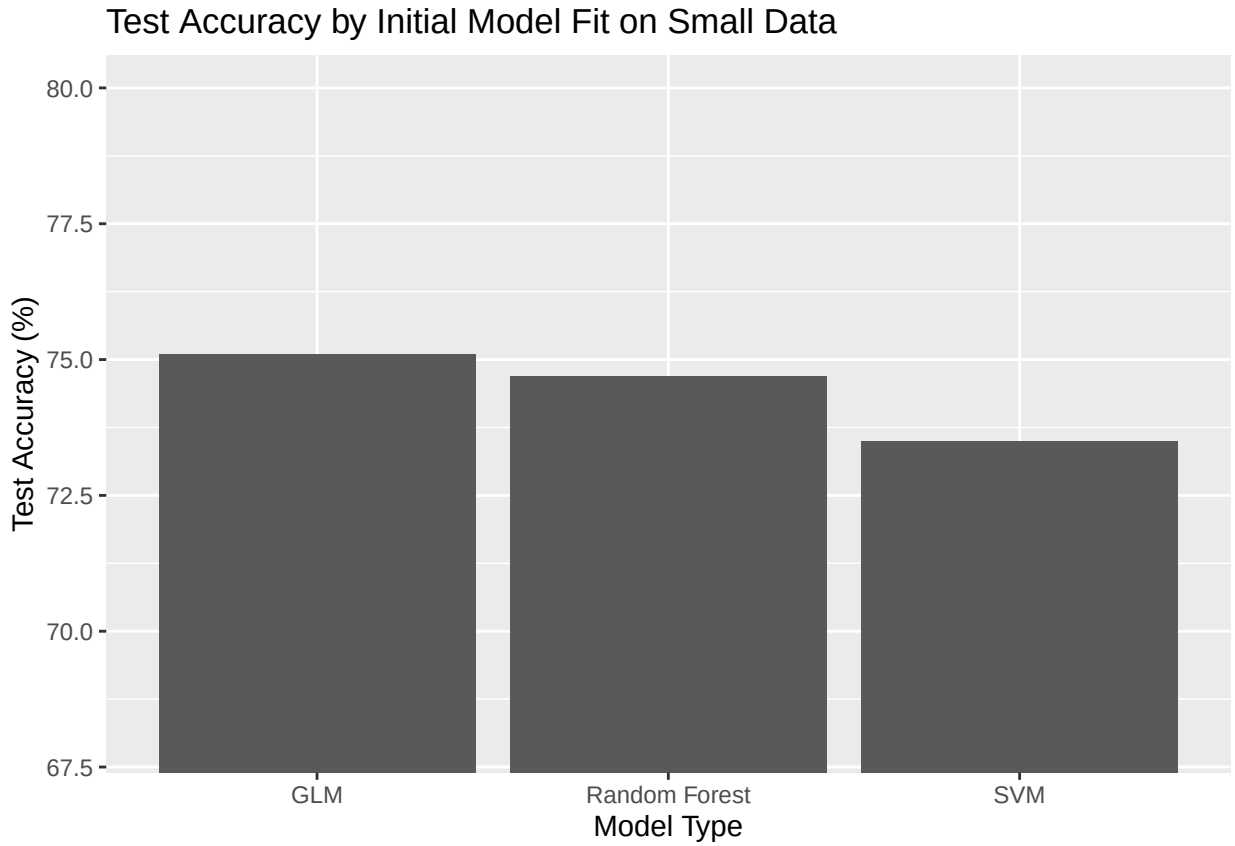
```
library(ggplot2)

data <- data.frame(
  Study = c(
    "Gonzalez (2023)\nDeep Neural Network",
    "Zamir et al. (2024)\nRNN",
    "Silva et al. (2023)\nRNN (15m)",
    "Chen et al. (2021)\nLogistic Regression"
  ),
  Accuracy = c(89.8, 86.7, 73.0, 60.24)
)

ggplot(data, aes(x = reorder(Study, Accuracy), y = Accuracy)) +
  geom_bar(stat = "identity") +
  labs(
    x = "Study and Model",
    y = "Accuracy (%)",
    title = "Model Accuracy"
  ) + ylim(0, 100)
```



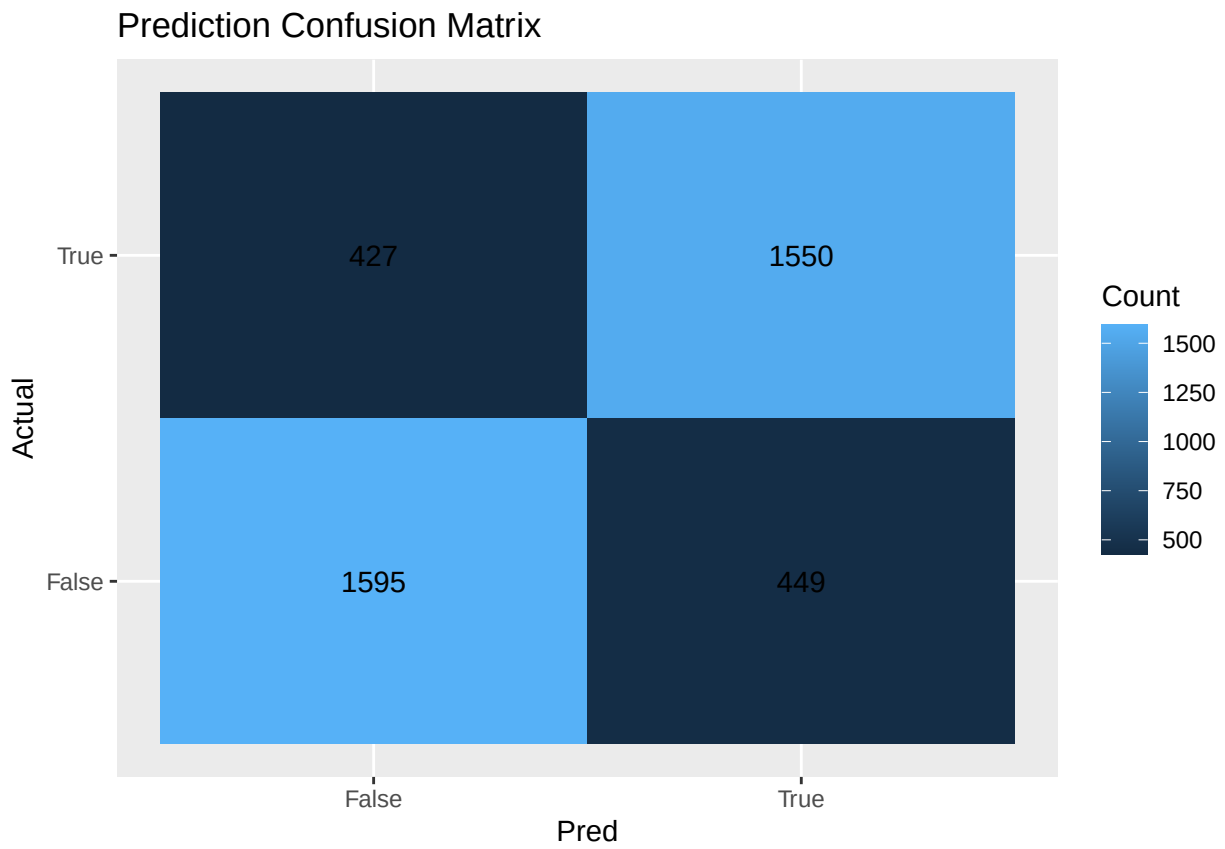
Three options were chosen to fit and compare for the sake of the project: Random Forest, Logistic Regression, and Support Vector Machine models. Initially, these were fit with a small dataset in order to conserve needed compute time. Models were evaluated with a 70/30 train/test split and 5-fold cv for hyperparameter tunings.



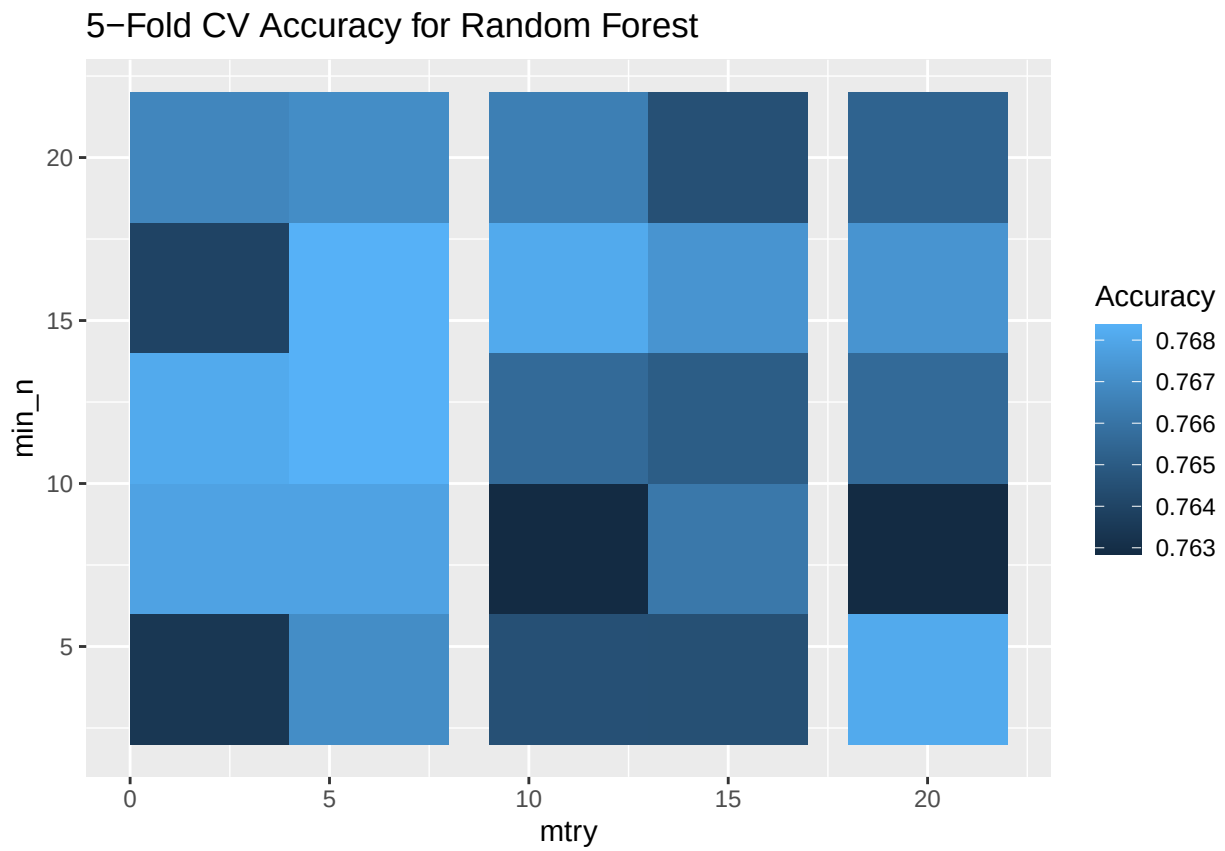
The above barchard depicts the test accuracy of each model type on a 70/30 split of the small dataset. Some additional changes to the input of each model was performed, restricting the number of predictors given to the SVM model to 6 ('deaths', 'dragonKills', 'baronKills', 'firstBloodKill', 'firstBloodAssist', 'kills'), which ultimately increased the accuracy of the model by removing dimensions which the SVM struggled with. The GLM and RF models were trained on the full set of 19 predictors. The GLM model outperformed the other two model types on the small dataset, achieving a test accuracy of 75.1%, trailed by the other two at 74.7% and 73.5% respectively.

However, given the small difference in performance between GLM and RF models, we chose the RF model for more extensive work as it has better interpretability for the problem at hand. The following is the results from the Random Forest model:

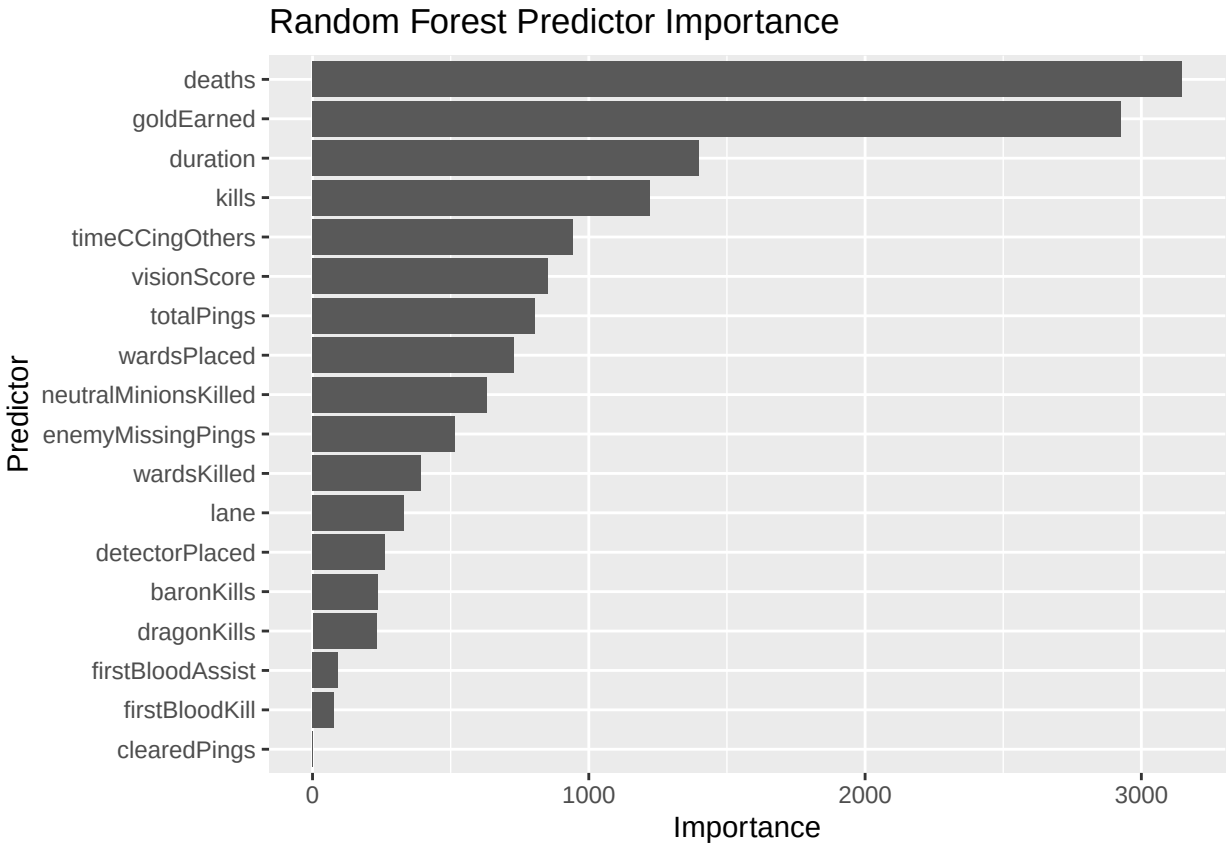
```
## [1] "Test Accuracy:  0.782143745336981"
```



Above we can see the performance of our model on the medium test set. This is an increase of 3-4 percentage points over the initial model, due mostly to having access to a larger train set. Importantly, the model performs similarly on positive and negative truth values, meaning it is well balanced.



Additionally, the model does not perform significantly different across various CV hyperparameters. This indicates the model is relatively stable overall.



Finally, as seen above, we can see feature importance calculated using GINI impurity. This is a major benefit of the RF model. The data reveals that deaths and goldEarned are both highly important predictors. This supports existing domain intuition that choosing a gameplan that maximizes the number of opportunities the player has throughout the game is vital. Deaths minimize such possibilities, and goldEarned allows the player to better capitalize on potential win conditions when they become available. First blood kill and assist, on the other hand, do not seem to be as important.

2.4 SHAP

SHAP or (SHapley Additive exPlanations) is a way to explain the results of a machine-learning model by assigning components of the prediction to individual features. SHAP performs monte carlo sampling to measure the effects of a randomly selected permutation of the features, adding each feature one at a time, accumulating the delta contribution of each feature as it is added: Contributions are then averaged across all permutations to gain the monte carlo estimate. Not all SHAP variations use monte carlo sampling, however it is mandatory here as SHAP explanations are already by far the most computationally expensive part of this project.

2.5 Interpretability

The approach used builds heavily on accurately classifying game outcomes to categorize samples as either win or defeat. However, in order to determine which factors led to the outcome of the match, SHAP was used to quantify each feature's contribution to the model's predictions. SHAP values provide a game-theoretic interpretation of model behavior by breaking down each each individual prediction into additive contributions from each feature. This allows interpretability in two ways:

- Global interpretability, where SHAP summarizes which features most strongly influence predictions across all matches
- Local interpretability, where SHAP explains why the model predicted a win or loss for a specific match.

A primary advantage of using SHAP over model-specific analysis is that it is agnostic to the type of model being used and also provides more in-depth insights.

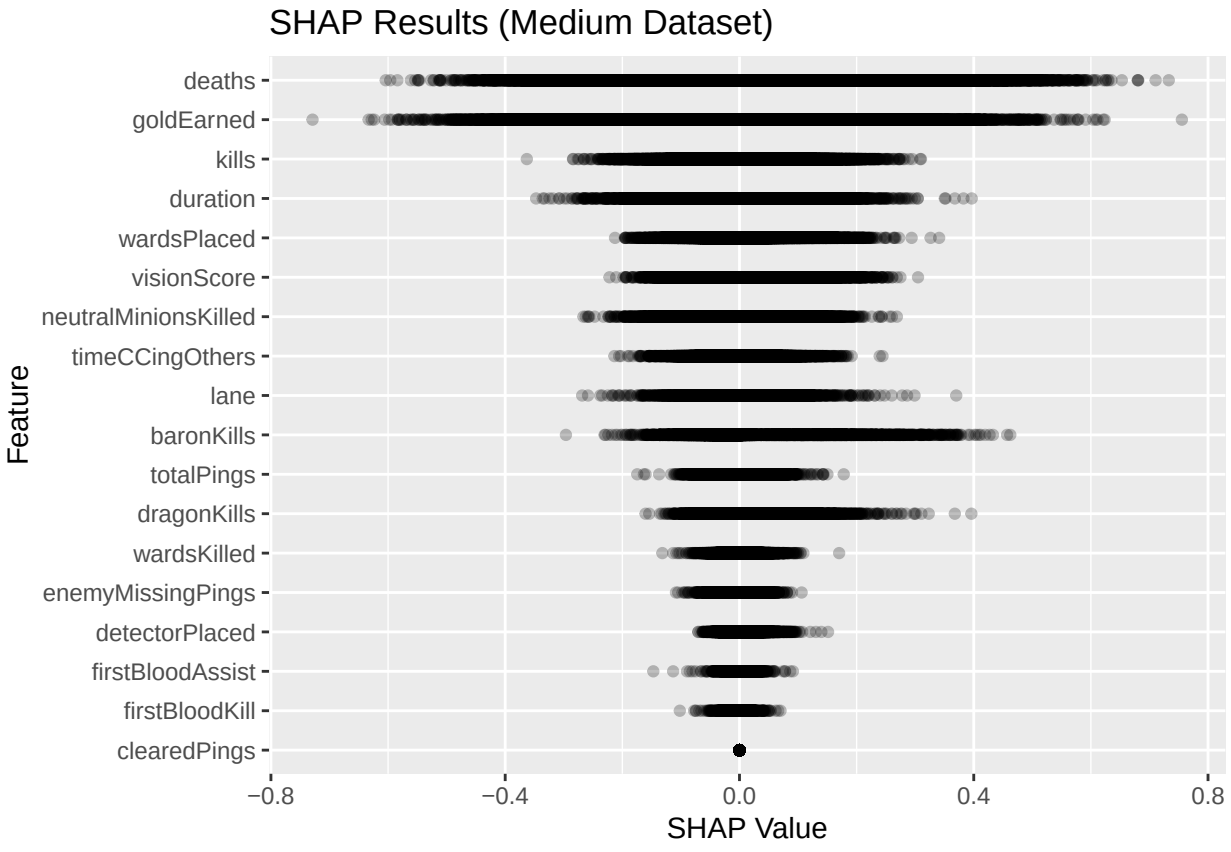
Futhermore, by looking at the local prediction for a specific match, we can begin to provide feedback to the player on things to improve. This aligns with the stated goals of this project and will be explored further later in the paper.

```
## [1] ".pred_False" ".pred_True"
```

3. Results and Discussion

Model performance across the three supervised approaches initially showed that the GLM outperformed SVM and Random Forest approaches. However, we chose to use results obtained from the Random Forest model for interpretability.

3.1 Global Interpretability



Both SHAP and the random forest internal importance ranking highlight deaths and goldEarned as the two most influential predictors. This is an expected result as the amount of gold earned throughout the course of a match directly leads to the player with high gold earned purchasing more items, and therefore having more

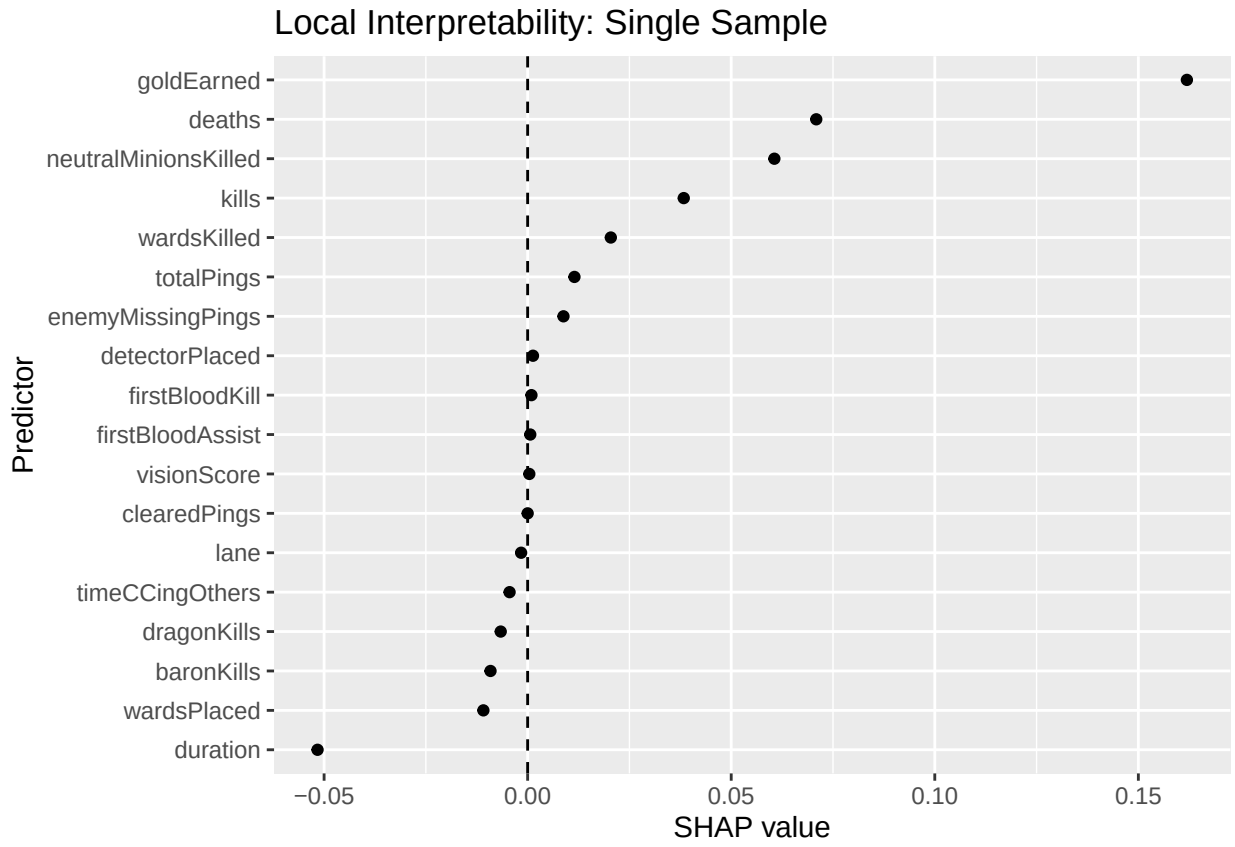
frequent access to victory conditions thoughout the match. What is surpising, however, is that the internal estimate from the random forest model holds the total number of deaths as the strongest predictor. Deaths have two primary side-effects throughout the course of the match. First, the enemy player responsible for the death is rewarded with gold, increasing their total gold earned and providing them with more opportunity to engage with victory conditions. However, this relationship is more directly captured with the goldEarned statistic. The second side-effect of a death is that you lose the ability to interact with any aspect of the game for 6-56 seconds (based on level). This directly impacts the player's presence within the game. In domain-specific terms: deaths decrease your tempo.

Futhermore, the above figure seems to reveal a slight 'right hand' skew to the impact of deaths on game outcome. This means that in extreme cases, low numbers of deaths seem to have a greater positive impact than the magnitude of the negative impact of high numbers of deaths.

Other notable features of the SHAP importance graph include the predictor baronKills. This predictor represents the number of times a team has achieved a baron kill, which is often thought of as a "win condition", meaning it massively increases the chances of your winning the game. It is very much akin to something like promoting a pawn to a queen in Chess, where it doesnt outright win you anything but regardless has a massive positive impact on your chances of winning the game. This can be seen by how baronKills has the third highest maximum SHAP value, right behind goldEarned and deaths. Notably, the predictor lacks many low SHAP values, hinting that while it provides a massive strategic advantage, achieving a baron kill is not required for winning the game in the same way that your goldEarned is. In other words, promoting a pawn to a queen is not the only way to win a game of Chess.

Finally,, first blood-related statistics were commonly ranked lowest for importance within the models. This suggests that early, isolated advantages do not often lead to determining the match outcome. This is counter to some community sentiment and may be the grounds for future analysis.

3.2 Local Interpretability



We can view the predictor contributions for an individual sample to provide targeted feedback. This provides local, targeted feedback and has direct applications for things such as personalized coaching tools and gameplay feedback systems. Above is a figure detailing the SHAP explanation for the values of a specific game. For example, if we see that deaths and baron kills contributed very positively to the eventual prediction, wheras timeCCingOthers and neutralMinionsKilled were weak points in the gameplay, we can provide actionable feedback to the player: they should continue their habits of not dying while attempting to kill more neutral minions and CC more enemies. Alternatively, if we see that the goldEarned and deaths are both negatively impacting the outcome of a game, we could give the feedback to re-assess the entire strategy.

4. Conclusion

This project demonstrates a pipeline and plausible solution for integrating SHAP and machine learning maodels to create interpretable, actionable insights about League of Legends match wins. Working with a high-level dataset, all three models got high predictive accuracy (even when compared to similar projects in the same area). Vitally, SHAP provided a highly detailed and robust vision of the key parts of the dataset, identifying gold, kills, deaths, vision control and infulential factors and first blood statistics as non-influential.

The hope is that the work done in this project can help form a foundation for future market-targeted platforms such as analytical systems to support competitive teams in esports, automated coaching for players, or personalized platforms such as the already existent mobalytics.gg.

References

[1] M. A. Zamri, M. A. Zamir, et al., "Online game outcome prediction model using weighted-based feature approach," Fusion: Practice and Applications (FPA), vol. 15, no. 2, pp. 132–144, 2024, doi: 10.54216/FPA.150212.

[2] J. A. Hitar-García, L. Morán-Fernández, and V. Bolón-Canedo, "Machine learning methods for predicting League of Legends game outcome," IEEE Transactions on Games, vol. 15, no. 2, pp. 171–181, Jun. 2023, doi: 10.1109/TG.2022.3153086.

[3] R. Leo, "ML-Prediction-LoL," GitHub repository. [Online]. Available: https://github.com/reneleogp/ML-Prediction-LoL

[4] A. L. C. Silva et al., "Continuous outcome prediction of League of Legends competitive matches using recurrent neural networks," 2018. [Online]. Available: https://arxiv.org/abs/1806.09802

(if applicable; otherwise cite as unpublished manuscript).

[5] Springer, "Continuous outcome prediction of League of Legends competitive matches," 2024. [Online]. Available: https://link.springer.com/content/pdf/10.1007/s10115-024-02092-9.pdf

[6] LoL Esports, "Official announcement," X (formerly Twitter), Nov. 2024. [Online]. Available: https://x.com/lolesports/status/1859295721956024521