Submitted in part fulfilment of the requirements for the degree
of
Master of Science in Business Analytics

League of Legends Victory Prediction Analysis

By

Calum Palmer

Surrey Business School

Faculty of Arts and Social Sciences

University of Surrey

September 2022

Word Count: 13,996

Executive Summary

Esports is a form of competition using video games where participants will compete either individually or in a team for a chance at victory. The recent generation of professional leagues across multiple game titles has lead to an industry boom of esports worldwide. League of Legends is a MOBA game that is currently one of the most popular titles globally with over 180 million monthly players, and a peak of 73.8 million concurrent viewers (Riot Games 2021, McLaughlin 2021). The objective of this paper is to use machine learning algorithms in order to predict the outcome of League of Legends esports matches by using in-game information at specific in-game intervals. Using this type of predictive analytics, teams could use these predictive systems to optimise and streamline their gameplay to improve their overall win rates and highlight key weaknesses in an opposing team's game plan. With this in mind, studies related to predictive analytics has begun rising with a few researchers producing predictive modelling systems for games such as League of Legends and DOTA 2. Various predictive models have been proposed, with the majority of analyses focussed on post-game data prediction or champion selection recommendation systems by mainly using amateur game data (Ani et al. 2019, Shen et al. 2022). Some application of these models are already seen commercially in bookmakers, with a growing number of esports titles to bet on these bookmakers have the challenge of creating odds that are representative of the outcomes that will occur, and therefore require precise predictive models or will suffer large financial losses.

This study analysed data that was acquired from the website Oracle's Elixir, and is based on the 2021 League of Legends competitive leagues found

globally. As the study is based upon the match predictions, the result variable is used a binary target variable, making this a classification task. The dataset included over 10,000 unique matches, and 105 in-game statistic features that can be used to predict the result. Using Python, the data was explored, cleansed, transformed using feature selection techniques, and validated before being split into smaller subsets based on the in-game timer, resulting in data subsets based on the intervals of 10, 15 and 20 minutes. A combination of Logistic Regression models and Gradient Boosting Classifiers were applied to this newly preprocessed esports data. Using these machine learning models, it was determined that at the three intervals of the 10, 15, and 20 minutes, the models achieved accuracies of 75.16%, 77.06% and 85.99%. When compared to other studies that have attempted to predict results at various time intervals, the results presented in this study show improvements up to 7% on previous works by Lee et al. (2020) and Silva et al. (2018).

Feature analysis was also completed. The features that play key roles in a game's outcome were calculated using relative importance, and a discussion of the reasoning behind why each feature was picked at each interval. The results of this feature analysis showed that during the first ten to fifteen minutes of the game, players should be focussing on three key metrics - minimising deaths, maximising the number of kills they participate in and the most important of all, the differential in the number of minions killed between their lane opponent. Secondary metrics such as taking the first tower are then the next level of importance, where teams should be aiming to destroy the opponents mid-lane tower over the top-lane or bot-lane towers as early as possible. Once the in-game timer passes the 20-minute interval, players should then focus on slaying the neutral monster called 'Baron' whenever possible, as it is exceedingly the most influential feature in the data. Using the buff that is granted to players after this, teams should aim to expand their gold and experience leads as much as possible, by destroying as many turrets as they can, with the first three turrets predictor also showing to be highly influential here. Using this predictive modelling system, an average

layman with no prior knowledge of the game could theoretically predict any given League of Legends match 22% more accurately than the no information rate benchmark, when given the data at the 10-minute mark. Ultimately, the results verify that reliable match result prediction is possible in League of Legends.

Declaration of Originality

I hereby declare that this thesis has been composed by myself and has not been presented or accepted in any previous application for a degree. The work, of which this is a record, has been carried out by myself unless otherwise stated and where the work is mine, it reflects personal views and values. All quotations have been distinguished by quotation marks and all sources of information have been acknowledged by means of references including those of the Internet. ***I agree that the University has the right to submit my work to the plagiarism detection sources for originality checks.***

Name: Calum Palmer

Signature:

Date: 07/09/2022

Contents

List of Figures

List of Tables

Acronyms

AUC Area Under the ROC curve

FN False Negative

FP False Positive

FPR False Positive Rate

MCTS Monte Carlo Tree Search

MOBA Multiplayer Online Battle Arena

NLP Natural Language Processing

RNN Recurrent Neural Networks

ROC Receiver Operating Characteristic

TN True Negative

TP True Positive

Glossary

baron A neutral monster that spawns after the 20-minute mark that will give a powerful buff when slain. 4, 42, 46

buff A broad term to describe when something is made stronger. 41

champion A unique player-controlled character possessing a distinct set of abilities and attributes. 3, 5

dragon A neutral monster that spawns every 5 minutes that will give a moderate team-wide buff when slain. 4

gank When a surprise attack is made upon a player, often made by a jungler or support. 4

inhibitor A structure that causes super minions to spawn in its respective lane. At least one enemy inhibitor must be destroyed for the Nexus to become vulnerable. 42

jungle A section of the map where neutral monsters spawn that can be slain for gold, experience and buffs. 3

meta The most effective strategy for winning. 2

minion A unit that periodically spawns from the Nexus, advances along a lane towards the enemy Nexus and engages with any enemy they encounter. 3, 42

nexus A structure that serves as the primary objective of the game. When the enemy Nexus is destroyed, victory is achieved. 4

patch A version of the game with a set of changes made to the game to update, improve or balance. 5, 6

rift herald A neutral monster that spawns between the 8 and 20-minute mark that can be used as a powerful tower sieging tool when slain. 4

tower A structure that deals damage to enemies that come into its radius and must be destroyed in order to reach the Nexus. 4, 42

ward A deployable unit that grants vision of the surrounding area for a duration, they are typically used to gain valuable information on the enemy. 3, 4

Chapter 1

Introduction

1.1 Context

Esports is a form of competition using video games where participants will compete either individually or in a team for a chance at victory. These competitions attract millions of viewers, with estimates of 532 million spectators by the end of 2022, and this value is expected to grow annually at a value of roughly 8.7% (Newzoo 2022). The rapid growth in esports has led to the industry becoming professional, with hundreds of players contracted on full-time contracts competing for prize pools of up to $40 million (Esports Earnings n.d.). According to Newzoo (2022) this viewership will help the industry generate over $1.38 billion in revenue by the end of 2022. As the esports industry continues to grow, so does the importance on teams to win and remain relevant in the industry.

In traditional sports, analytics has become an extremely popular field with teams investing heavily in some form of analytics. These analytics can be used from evaluating opposing teams, to individual player forecasting and even used to decide signings or team selection (Sarlis & Tjortjis 2020, Apostolou & Tjortjis 2019). Apostolou & Tjortjis (2019), Sarlis & Tjortjis (2020) shows that these analytics can be applied for both teams and individual athletes, giving an accurate estimation of key metrics such as goals scored per

season or the expected number of shots attempted in a given match. This form of sports analytics allow teams to judge both themselves and opposing team performances, calculating the likelihood of shot conversion or the current overall form of a team's performance, giving their team an advantage through prior preparation. This same methodology could be applied to esports, using these machine learning techniques could highlight specific factors both pre-game and in-game, helping analysts and coaches refine strategies within the game.

The ease of data collection coming from each match has led to a rise in esports analytics. In-depth analysis of matches, teams and pre-game factors become key techniques for teams to gain this advantage over their competitors, with teams being required by their leagues to have at least one dedicated coach and analyst similar to traditional sports teams (LoLEsports 2022). These coaches and analysts use predictive analytics to maximise their team's likelihood of winning by altering numerous features related to pre-game and in-game strategies, current meta analysis and common patterns of their competition (Kokkinakis et al. 2021). However, this analysis is often completed manually by watching key highlights of matches using the analyst's intuition and using rudimentary analysis of in-game factors.

If matches can be accurately predicted using machine learning techniques, then analysts can provide new opportunities to optimise player strategies and can lead their teams to better outcomes. Applying the same findings found in Gray & Wert-Gray (2012), it can be seen that the overall performance and fan satisfaction with a sports team's performance has a measurable impact on revenue via fan attendance and their media response. Esports fans also appear to increasingly demand skillful performances especially from players that are deemed as *'superstars'*, with these players being more likely to attract new viewers, thus increasing the economic gain of the market (Mangeloja 2019, Ward & Harmon 2019). It would then be in the interest of both teams and individual players to maximise their abilities and career longevity using these advanced analytics, so they can fully realise their potential; es-

pecially when the volatility of a players job security results in only the top 10% of players having lasting, stable careers (Ward & Harmon 2019).

1.2 League of Legends

1.2.1 General Information

League of Legends is a Multiplayer Online Battle Arena (MOBA) game developed by Riot Games released in 2009, it is one of most popular esports games in the world with over 180 million monthly players and a peak of 73.8 million concurrent viewers (Riot Games 2021, McLaughlin 2021). A MOBA is fusion genre of real-time strategy, role-playing and action games in which two sets of teams will compete in a known arena. The objective of each game is to defeat the opposition by destroying the enemy's base. Each player will select and control a unique champion with their own set of distinct abilities, this champion will be selected before the game starts and cannot be changed until the game has ended - this will be covered further in Section 1.2.2. Players can strengthen their champions by gaining experience and gold, this can be done by slaying enemy minions, jungle monsters, enemy structures or enemy champions. This gold can be spent in the shop allowing players to purchase items that enhance the attributes of their champion, as well as various utility items such as wards.

A map of League of Legends can be seen in Figure 1.1. There are three lanes, Top, Middle and Bottom, with the jungle filling the space between these lanes. Typically, a player will be assigned to each of these lanes including the jungle, the exception being two players assigned to the bottom lane. The roles are typically labeled as follows:

- Top laner - Starts in the top-lane, often a champion who has larger health and resistance to damage with the ability disrupt enemy players.
- Jungler - Roams in the jungle, they help their laners whenever possible

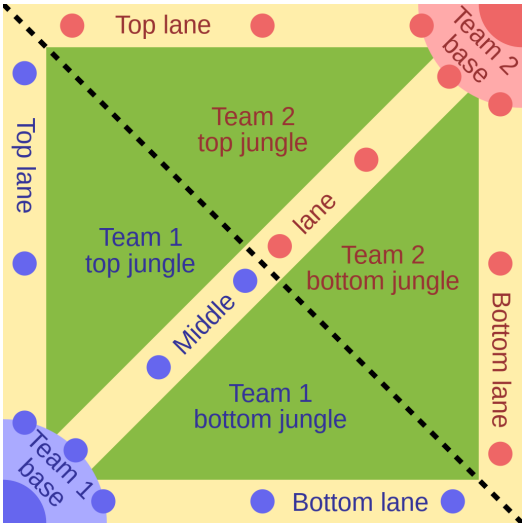


Figure 1.1: Map of League of Legends

often using surprise attacks on enemy laners commonly referred to as a gank.

- Mid laner - Starts in the mid-lane, often a champion who is a spell-caster that can cause magic damage over a wide area or has high single-target burst damage.
- Attack Damage Carry (ADC) - Starts in the bottom-lane, often a long-ranged champion that requires gold in order to deal massive damage towards the latter stages of the game.
- Support - Also starts in the bottom-lane, often a champion that can provide aid to the ADC throughout the game via protective abilities and utility such as wards.

Each coloured dot represents a tower that must be taken in order to reach the enemy Nexus. A river separates the territories between the Blue (Team 1) and the Red team (Team 2) along the dotted black line seen in Figure 1.1. In this river you can find Baron or Rift herald in top-side and the Dragon in the bottom-side, they are key objectives that will often be contested.

1.2.2 Champion Selection

Champion Selection plays an important part in every game of League of Legends. Certain champions have inherent synergies with one another, meaning they are beneficial to be picked with each other. Likewise, some champions are considered counter matchups when they are good at stopping another champion. This means that picking a good mixture of champions that are solid synergistically, whilst also ensuring the opponents champions do not counter yours is vital. These ideas are the fundamentals of champion selection, and they are what professional coaches and analysts attempt to solve each week. Factors such as player champion experience, the current game balance patch or a champion's ability to be flexible across different lanes will change champion select from game to game.

As seen in Figure 1.2, the current draft phase works as follows:

- Ban Phase 1 begins with the Blue team, in turn each team bans three champions from the pool.
- Pick Phase 1 begins with a singular pick from the Blue side, followed by two picks from the Red side. Blue side will get two more picks, followed by a singular pick from Red side for three picks each.
- It will then enter Ban Phase 2. Here both teams will ban two more champions in turn, with Red side starting.
- Pick Phase 2 will begin. Here Red side get their fourth champion pick, followed by the final two picks from Blue side and finally Red side pick their final champion.

This champion selection structure leads to clear opportunities for teams to ban out champions that are deemed too strong in Ban Phase 1. Blue side getting the first pick gives them a chance to pick any champion that is deemed too strong that still remains after Ban Phase 1. Whilst Red side getting the last pick gives a defined opportunity to pick a counter match-up to any given lane. These factors can give one side the edge based on the

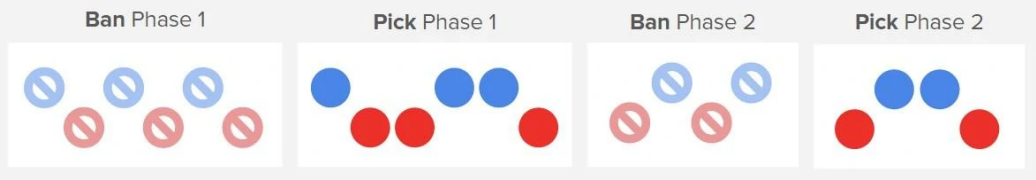


Figure 1.2: A depiction of the Champion Selection phase

Patch, leading to varying strength levels of Blue side vs Red side and can make side selection important.

1.3 Structure, Aims, and Objectives

The following section explores an overview of this dissertation:

The next chapter, Chapter 2, attempts to review the previous literature in the predictive analytics field, exploring the use in traditional sports for both individual and team performance. It then moves on to how these predictive models have been used in newer esports studies, with games such as League of Legends and DOTA 2, highlighting key techniques that were used to get a better understanding of key predictors and better performance metrics. The relation of commercial predictive models were briefly discussed in the betting industry with the popularisation of esports betting, and the industry's need for highly accurate models for their odds-makers.

Chapter 3 follows the techniques and tools used for predicting the effect of pre-game choices on the outcome of the match. Beginning with an overview of the research philosophy, approach and strategy that is used during the course of this dissertation. Followed by a brief data description, showing how the data is viewed when it is first accessed as well as the meaning behind many of the features. The whole data preparation phase is then reviewed, followed by how the data was modelled and the technique that were used were explained.

Chapter 4 then proceeds with a discussion of how the results that were

produced were going to be evaluated, and how these performance metrics would ultimately judge the implication of these results. The results for each time interval model is then presented, with the feature importance of each model being shown. An interpretation of what these performance metrics mean, and a discussion of why certain features are favoured over others. The results section is then wrapped up with a general overview of all the results, an implication of what they mean for this study and how they impact the field of predictive analytics inside esports.

Finally, Chapter 5 will conclude the dissertation giving a summary of the work completed, potential limitations of the works completed, as well as any further opportunities for research.

Having pre-established the landscape of esports and its relationship with analytics, it is clear that refinement in the way that this industry uses its highly available data is needed. Many academics have predicted the outcomes of matches in esports titles such as Ani et al. (2019) and Lin (2016). However, performing these studies, few academics attempted to predict these match results using data from specific time intervals within the game, with most using data from the match result. Predicting using statistics not only will cause the conclusions made from the studies to be highly skewed towards certain factors, but will also miss out on some decisions made inside the early stages of the game that will cause the swing in momentum for one team. This study also uses updated esports data, with most studies completed using either amateur player data or massively outdated esports data from between 2015 and 2018. In contrast to other studies, this study uses a much larger, updated dataset and will concentrate more on the overall effects of the game choices that a team will make during the course of the game, and how teams can apply these choices in order to increase their probability of winning. Therefore, the research topic that will be addressed is as follows:

> *Can the outcome of a League of Legends match be predicted using machine learning?*

Chapter 2

Literature Review

This section aims to review previous literature of prediction models in the esports industry that was previously explored during Chapter 1. These literatures will help create a better working understanding of how to properly model, extract and gain knowledge from the datasets. It will cover papers from traditional sports where prediction models are widespread and lesser researched esports models, to research on the betting industry with its new relation to esports. These studies will then be contextualised to form a set of research questions and aims, that will be answered through this paper.

2.1 Predictive Analytics in Traditional Sports

In the mid to late 2000s the use of prediction modelling exploded academically, being applied to a multitude of fields from biology and medicine to political sciences or sports. This academic surge of interest caused thousands of studies all using various forms of predictive modelling. Not only did this evolve many practises across these fields, but it also helped develop the techniques of predictive modelling today. This is section we will explore literature related to how predictive analytics are used in sports, and how insights can be gained from this type of modelling. Sarlis & Tjortjis (2020) studied the impact of how various basketball related performance evaluation metrics can be used to identify dominant attributes that will help predict

candidates for the Most Valuable Player (MVP) award, as well as Defender of the Year in the National Basketball Association. They concluded that using their Propagation Neural Networks with 8 years of training data, that their model was able to accurately predict the MVP of the season with 100% accuracy, and could also predict the Defender of the Year. Similar studies have been performed with football such as Pantzalis & Tjortjis (2020), where they analysed 4 top football leagues in Europe and predicted the final standings with up to 70% accuracy. Scelles et al. (2021) shows us that transferring this level of analysis from a traditional sports team to an esports team is highly supported, with claims that 'esports... could provide some insights about the future development of sport'. This suggests that prediction modelling is likely to work when used to predict both player performances, whilst potentially being able to predict standing of esports leagues globally.

Wilkens (2021) analysed professional tennis matches over the last decade, using player, match and betting data to produce an ensemble machine learning model to investigate how the informational content of these data points in relation to how they can help predict match outcomes. They used their findings to evaluate whether bettors were able to achieve consistently positive returns. Prediction accuracy was found to reach about 70% with a complex model filled with all data previously mentioned, whilst a simpler baseline model using current world rankings was able to determine the result 65% of the time. Despite these results, they found potential returns of 10% from a long-term betting strategy, however the volatility, total liquidity required and model risk makes the sentiment behind the model questionable at best. A similar model is likely to translate well over to League of Legends, using potential features such as current league standings as a strong predictor variable as well as the historic match data that has already been collected.

Thabtah et al. (2019) proposed a new machine learning framework that would help predict results in the Nation Basketball Association, and discover the most influential features that affect the outcome of NBA games. Some models used throughout include Naïve Bayes, neural networks, and decision

trees. They concluded that there were five key performance metrics that were the most influential to any game played in the NBA. Using these features they were able to achieve a prediction accuracy of up to 83% using the Logistic Model Tree method.

These studies largely suggest that using predictive analytics for both player performance predictions, and team-wide predictions in traditional sports is highly successful, with studies concluding that they are able to predict match outcomes, league standings and league MVP votes consistently. The majority of these studies also concluded that using either a classifier such as Random Forest and Gradient Boosting or Logistic Regression techniques were often optimal for maximising accuracy and AUC. Scelles et al. (2021) proposed a relationship between traditional sports and esports that suggests that similar predictive analytics techniques would have large crossover, and understanding this potential is fundamental in forming a basis for the value of the work in the esports field.

2.2 Predictive Analytics in Esports

2.2.1 Match Outcome Prediction

The work of predictive modelling in esports only started in the mid 2010s, with one of the original papers by Lin (2016) exploring match outcomes in League of Legends. This work focused on the relationship between the potential feature-sets that a game such as League of Legends can create, and how they impact the predicted match outcome both pre-game and in-game. The data was collected using the Riot API from matches with average ranked players, with the in-game data being extracted from the statistics published at the end of a match. It becomes apparent that the data used in this initial study appears to be highly correlated with the match result, and this is reflected by the 95% success rate on prediction using in-game data. Yang et al. (2016) also investigated the match outcomes in another esports MOBA game, DOTA 2. They considered pre-match features from individual play-

ers' historical performance data, as well as real-time data obtained during the match progression. The effectiveness of the pre-match features seemed to be highly variable, with outcome prediction probabilities quoted at up to 71.49%, and a model of combined pre-match and in-match data reaching up to 93.73% accuracy at the 40th minute. Interestingly, for the initial 15 minutes, the prediction accuracy for the real-time model is below that of the pre-match prediction values, potentially suggesting that the pre-match prediction accuracy could be inflated due to issues of interactions between features.

Ravari et al. (2017) once again considered match outcomes, but this is based upon First-Person Shooter game, Destiny. Classification models were created based on the different game modes available, with a model based on specific game modes and a combined model. They obtained results that highly favour models built upon data from specific game modes, concluding that the behavioural difference seen in players depending on the game mode is the main attributing factor. These models were built upon the Random Forest algorithm and the Gradient Boosting classifier.

Two studies built upon the previous research just two years later. Silva et al. (2018) introduces the use of Recurrent Neural Networks (RNN) to predict esports matches through time intervals between the 0 and 25 minute mark, and uses data from matches between 2015 to early 2018. They concluded that these RNNs were capable of obtaining prediction accuracy of between 63.91% to 83.54%, depending on the time interval in-game. Their study presented evidence of the predictability of each match and its positive correlation with the in-game timer, showing that the accuracy of prediction becomes stronger as matches progress, reflecting the snowballing nature of the game itself. However, it should be noted that the study ignores the presentation of each feature's weighting and thus does not provide any insight into what features are most influential in these victories.

Gaina & Nordmoen (2018) built upon the study from Lin (2016), analysing

the features that predict match outcomes at the 10-minute interval. Moreover, the correlation between the impact of early performance on the match result for each individual player in all 5 roles was found to be 'a medium correlation', with 'a weaker one' when overall team performance is compared. Other scholars such as Ani et al. (2019) have presented findings that suggest that prediction model accuracy can reach performance levels of 99.75%, with a combination of pre-game and in-game predictors using the Random Forest algorithm. An interesting note about Ani et al. (2019) is whilst using Adaboost, Gradient Boosting and Extreme Gradient Boosting they were only able to achieve an accuracy of 57.22% to 65.67% using only pre-game features which is drastically lower than the Random Forest algorithm. However, whilst this study mentions the idea of feature selection, it appears to ignore the idea of correlated features, picking only those who have performed best in Recursive Feature Elimination and Gini importance tests. This has likely led to a feature-set that contains strong predictors that are highly correlated with each other, and are plagued with multicollinearity issues. In the literature, both the Random Forest algorithm and Gradient Boosted Trees are commonly used machine-learning strategies that are used inside prediction models. These methods are both decision tree based in which a target variable is predicted based on a number of decision rules inferred from the feature-set, with the main downside is potential overfitting of the data (scikit-learn n.d.). This level of accuracy from pre-game features is more in line with other studies and will be discussed further in Section 2.2.2.

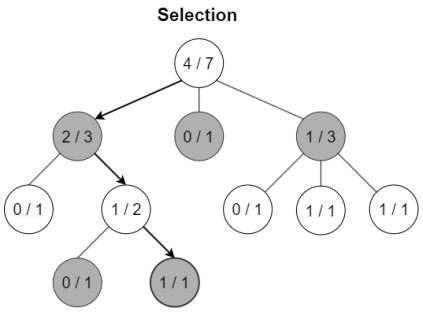
Lee et al. (2020) also produced a match outcome prediction model, using data from the Riot API of high level players from the ranked ladder. Similar to their predecessors of Silva et al. (2018), they tested their dataset across game time intervals, as well as the importance of each feature in the dataset. They achieved a prediction accuracy of between 62.25% and 96.08%, which is an improvement over the RNN used in previous studies. Gold difference and the number of Towers taken appear to consistently be the two most important features across most studies referenced, with gold difference having a relative feature importance calculated up to 43.08% at the 10-minute mark

and turrets having up to a 22% importance (Lee et al. 2020, Ani et al. 2019, Gaina & Nordmoen 2018). Another interesting study comes from Novak et al. (2020), here they apply a coach-centred approach to modelling performances seen at the 2018 League of Legends World Championship. Three coaches rated the proposed feature set using a correlation scale of 1 to 10, with median ratings for features equal or over 6 being retained for modelling. After multicollinearity checks, only 14 predicting features remained and the strongest fixed effect was 'Tower Percentage', closely followed by 'Inhibitors Taken' and allowed the model to achieve prediction accuracy of 95.8% - which remains consistent with previous studies.

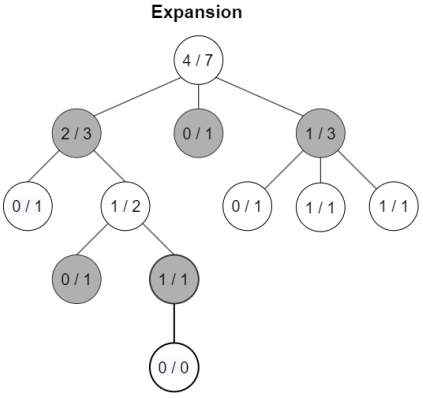
2.2.2 Champion Selection Prediction

As described earlier in Section 1.2.2, champion selection is a key part of any MOBA game and has the ability to give your team an inherent advantage in-game before the game even begins. Previous works have explored the ideas of character recommendation systems with two key methodologies in mind - models based upon historical win rates of each character and an association rule model based on common selection frequencies. Chen et al. (2018) proposed a Monte Carlo Tree Search (MCTS) recommendation model in another popular MOBA game - DOTA 2, that suggests characters to add to the team in the draft phase that will maximise the team's victory probability. The draft phase is approximated to a combinatorial, sequential, zero-sum game with perfect information and deterministic rewards. Therefore, an optimal pick at each stage will be the highest predicted win-rate character at a given stage, when both teams behave optimally in the process. Due to the large branching factor of the draft process, the decision-making scales exponentially larger as the draft process progresses and the computational power required does too. The MCTS is a tree search algorithm that is commonly used to solve deterministic games such as Chess, Tic Tac Toe and Go (Duckett 2016). As seen in Figure 2.1, this algorithm explores the choice tree from root to leaf, selecting child nodes that represent the best winning outcomes.

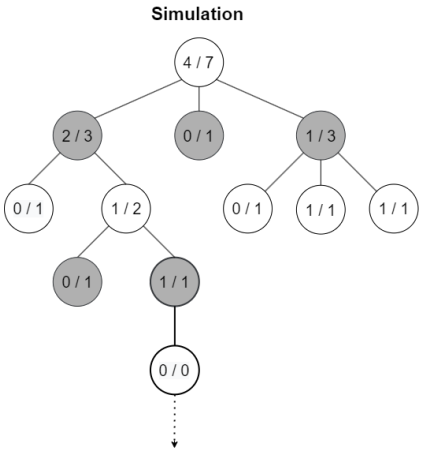
If the leaf node does not terminate the game, it will create the next step



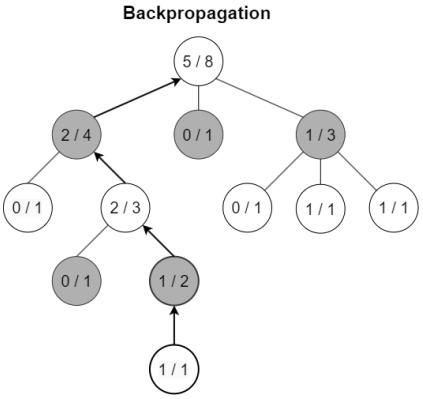
(a) Selection step



(b) Expansion step



(c) Simulation step



(d) Backpropagation step

Figure 2.1: A graph showing the steps of a Monte-Carlo Tree Search

with more child nodes throughout the tree, and if this simulated child node gives optimistic result probabilities it will update and back-propagate up the tree towards the root. At its core, the MCTS chooses the optimal move from the current state of a game's tree with the help of reinforcement learning. Using this system, Chen et al. (2018) were able to achieve a win-rate of up to 88.0% versus a standard association rule draft recommendation system used in studies such as Hanke & Chaimowicz (2017).

Recently, Shen et al. (2022) initiated studies into the champion recommendation using a mix of Natural Language Processing (NLP) techniques and Bidirectional long-short term memory models. They found that their proposed recommendation mechanisms were effective in both the coverage of champions and user satisfaction Whilst this paper reiterated the fact that the champion selection phase can be predicted, it gives a unique perspective of adding a level of subjectivity to model by using user feedback and offers a wider-coverage of champions who are deemed less popular picks. But this lack of objective measure gives it no quantifiable metric onto which this study can extract valuable information from.

These examples of predictive analytics for video games all employed machine learning strategies from Random Forest to RNNs, and show that all are applicable to this task. Demonstrated by many of these papers, the Random Forest performed particularly well with high predictive accuracy when applied to League of Legends or other similar games. This leads to the conclusion that a decision tree based algorithm such as the Random Forest algorithm will likely be the best performing technique, with less overall evidence to support more linear methods such as Logistic Regression.

2.3 Esports Betting

As esports has grown, so has the ability of bookmakers to capitalise on a new developing market and create revenue. According to Absolute Reports (2022), the global esports betting market has been estimated to be worth

up to $10 Billion in 2021, and is forecasted to double by the year 2028 with a compound annual growth rate of 13.1%. A new focus of esports has grown throughout traditional bookmakers, with an increasingly larger number of esports titles to bet on and some bookmakers even sponsoring some events (Byrne 2019). With the large variety of titles comes the challenge of creating odds that are representative of the outcomes that will occur. There are numerous studies showing how gambling markets in most traditional sports can change how a given team is evaluated, often with regard to sentiment bias (Feddersen et al. 2018, Na et al. 2019). This means that odds-makers must create predictive models that can accurately replicate the true likelihood of match outcome in order to ensure money is continually being made. According to the efficient-market hypothesis, sport bets should be subject to all available information that may be publicly available and this information will be reflected in the odds themselves(Even & Noble 1992). The betting market is therefore thought of as a fair and efficient market in which match outcomes can be accurately predicted.

Betting within esports has taken on many forms. Money line bets and proposition bets are the most common types of bets. Money line bets are bets that are placed on the outcome of a specific match, with payouts based on the odds that are created by the odds-makers using their internal prediction models. Proposition bets are bets made based on whether a specific event will take place within the game itself while the match is ongoing. A common proposition bet is whether a team would achieve the first kill of the game - commonly referred to as First Blood. All these types of bets require highly calculated odds ensuring the bookmakers will make money. With esports betting being in its early form, there are no regulatory structures put in place to effectively to keep match integrity in place similar to those found within most sports (Dos Reis 2017). This lack of match integrity causes potential match-fixing scandals which threatens the competitive integrity of both the league and the gambling market, as well as ruining games for spectators alike. Whilst studies such as Abarbanel & Johnson (2019) claim that esports spectators aren't deeply concerned about poten-

tial match-fixing, with most spectators being willing to forgive infractions that have occurred previously. Cases of match-fixing have already been investigated, with a Chinese player- 'Bo' being suspended after being coerced into match-fixing in the Chinese academy leagues, subsequently causing a league-wide large-scale investigation (Dot Esports 2021). This caused a call for harsher punishments and stricter measures to ensure infractions like these would become highly disincentivised, however no regulatory structures apart from the league's punishment systems currently exist.

2.4 Summary

This chapter provided an overview of key literature in predictive analytics and its relation to both sports and esports. Starting with a background on predictive analytics in sports, with the various uses scholars have modelled with and how their success can be replicated in the esports space. This was followed with studies showing how these predictive analytics can be applied to various esport titles. The various machine learning techniques that have been used previously were explained such as the Monte-Carlo Tree Search and the Random Forest classifier, with an exploration into how different machine learning techniques differ with results. The betting industry was then explored, showing its relation to the world of esports and how they use prediction modelling for their business. With the information gained throughout this literature review, the following research questions and research aims have been created:

> *Research Question 1: To what extent does the prediction accuracy of a League of Legends esports match outcome model increase through game time and why?*
>
> *Research Question 2: How strongly afflicted are League of Legends esports prediction models with multicollinearity issues and why?*
>
> *Research Question 3: How does the prediction accuracy of a League of Legends esports match outcome model vary based on the machine learning technique used?*

Research Aim 1: To assess the change in prediction accuracy of a League of Legends esports match through the progression of the match.

Research Aim 2: To explore how the issue of multicollinearity in a League of Legends outcome model affects the prediction accuracy.

Research Aim 3: To understand how the machine learning technique used in a League of Legends esports match outcome model can change its performance?.

Chapter 3

Methodology

In this chapter, the methodology that is used to research the topic is presented. This study attempts to produce a predictive analytics model that uses a machine learning approach that can predict the match outcomes of a League of Legends match. Firstly, an explanation of the pre-processing data-pipeline - how the data was collected, a description of the dataset, and how the data is prepared and cleansed for modelling. The subsequent section then follows the dataset through the prediction methodology, and ends with how the predictive capability of the model is tested and assessed.

3.1 Philosophy, Approach and Strategy

This section outlines the research philosophy under which the research is conducted, and how the methodology will be approached during the course of this study. The framework behind the methodology explored during this section is built upon the works of Saunders et al. (2007). They proposed a methodological framework made up of 6 steps, that underpin any piece of research that they labelled the 'research onion'. This dissertation will mostly follow the research philosophy of pragmatism, this ideology aims to find more of a practical point of view where both subjectivity and objectivity matter, and form a practical combination of methods that will be used to ultimately answer the research questions. The approach to the research

will be quantitative, and mainly follow a deductive approach, with the work completed being based upon previous theories and conclusions that previous researchers have created. This approach will mean that there will be analysis of numerical results, that are specific to the situation using statistical modelling and mathematics. The strategy applied to this research will be a mono-method experimental research strategy. This involves manipulating independent variables to observe a change in the dependent variable, in order to determine the relationship between the variables in a dataset.

3.2 Data Pre-processing

3.2.1 Data Overview

The data used in this project is based upon the match data collected from all the League of Legends esports leagues found globally in 2021. Oracle's Elixir is a website that collects all the esports match data provided by Riot Games; the publisher of League of Legends, and aggregates them into datasets that are freely downloadable and offered to coaches, analysts and fans alike (Oracle's Elixir n.d.). These datasets are updated daily, and go back until 2014. When the dataset is downloaded, it is given in a .csv format that can be opened Microsoft Excel to get an easier perspective of the data. Once opened, it can be seen that it is a huge dataset contained within one spreadsheet with 149,496 rows and 123 columns, and will have to be prepared in order to be ready for modelling. The rows contain data from a unique player within a given match, this starts off with the Blue side Top Lane player and continues serially until the Red side Support player is reached, it will then contain a row for each team's collective data. This means there are 12 rows for each unique match before reaching data for the next match, this can be seen in Table 3.1.

The target variable is the intended prediction variable, and is found at the eighteenth column named 'Result'. It is a binary value and simply denotes whether a team wins or loses using 1 and 0 respectively. Preceding this column are the match descriptor variables, here lies the unique match id,

Table 3.1: An excerpt from the dataset.

Participant	Side	Position	TeamName
1	Blue	top	DWG KIA
2	Blue	jng	DWG KIA
3	Blue	mid	DWG KIA
4	Blue	bot	DWG KIA
5	Blue	sup	DWG KIA
6	Red	top	Nongshim RedForce
7	Red	jng	Nongshim RedForce
8	Red	mid	Nongshim RedForce
9	Red	bot	Nongshim RedForce
10	Red	sup	Nongshim RedForce
100	Blue	team	DWG KIA
200	Red	team	Nongshim RedForce

the league in which the game is played, the date, the game number for best of 5 series, and other information such as match length and those found in Table 3.1. One of the key columns that will be used is the 'datacompleteness' variable, this contains 4 options of 'complete', 'found', 'partial' and 'reparse', and describes the state of the data in a given row. Following these 18 match descriptor variables, are the 105 in-game statistic variables that have been recorded. The values in these columns will be the key predictors in our match-outcome prediction model, however the usefulness of each variable will be decided later in Section 3.2.2. These predictors range from simple statistics such as the total number of kills or deaths in a given match, to more advanced statistics such as GSPD - The average gold spent difference between teams. Many of these statistics also come in the form such as 'goldat10' and 'goldat15', which is simply the amount of gold obtained by the 10 and 15-minute mark in-game respectively. These time-based statistics will be important in factoring how much the match outcome varies by the in-game time. Another note about the data, is the fact that it tracks the opponent statistics for all of these statistics inside each row, so this should allow the use of data from only one side whilst maintaining all the statistics from both teams.

3.2.2 Data Preparation

It is vital for the data gathered to be both reliable and of high validity, this is ensured by a rigorous data preparation stage. In order for the data to be prepared for modelling, coding is required. The coding language used in this study is Python, it is a well-rounded language that is opensource and allows access to great analytical libraries, visualisations and is well documented online whilst being the main analytics language used in industry (TIOBE n.d.). Jupyter Notebooks are used as a personal preference, but any IDE could be used.

The CSV file of all the match data for 2021 is read into the notebook as a dataframe. The data-types of the dataframe are checked to ensure that they are as intended. Once again the data is inspected and a large proportion of missing data becomes visible, many features such as 'turretplates' and 'elementaldrakes' had well over 100,000 null values and were not fully complete. Features like these will likely be removed during feature selection due to their lack of data, otherwise it would ultimately harm the modelling process - the feature selection process will be covered later. Firstly, the dataframe is sorted for the feature 'datacompleteness' and opts to reject any row that is not 'complete', this massively helps reduce the number of null values found in the data. Additionally, another filter is used on the 'position' feature, reducing our data to only rows with team information instead of a mixture of individual and team data. This will help focus on the overall impact of the team's actions in response to an outcome of a match. Now the dataframe should only consist of team-base data, but will contain rows for both the blue-side and the red-side teams for each individual 'gameid'. The problem here is that only one side of data is required to predict with, here it is chosen to predict the outcome of a blue-side team using blue-side match data. This issue is easily subverted by dropping any duplicate 'gameid' rows, leaving the dataframe with only team-based data from the blue-side team. As the dataset is now made-up of only team-based data, no duplicate rows exist and contains no null values, descriptive statistics are used to further in-

spect the data to search for potential outliers that could corrupt our model. This is completed using the 'describe' function and manually searching for data minimums and maximums that look suspicious. It can be seen that the 'gamelength' feature has an unusually high maximum value when compared to both the median value and the 75th percentile value. This is then removed by removing any rows that exist above the 99th percentile for 'gamelength', reducing the maximum gamelength from over 40,700 minutes, down to a maximum of 47.2 minutes; a gamelength that is as expected. After these steps are complete, the dataset has been properly cleansed and now 11,145 rows remain.

Then comes the step of feature selection that aims to reduce the overall dimensionality of the dataset, increase the computational cost of modelling and to improve the performance of the model (Brownlee 2019). All the features that include opposition team data are removed as they will cause issues with multicollinearity later, as well as prediction from opposition statistics being deemed redundant for understanding how a given team could improve. After removing many redundant features, a correlation matrix is created between the remaining features in the data subset. Finding the correlation allows for the measure of how features share an interdependence between one another. Here the Kendall correlation is calculated between features using the 'corr()' function and visualised using the 'heatmap()' function from the 'Seaborn' package. Correlation values between the target and themselves are deemed better predictors when this value moves closer to 1 or -1. Therefore, those features with values close to zero should be removed from the dataset as they likely provide no benefit to the model, whilst adding further complexity. On the contrary, features with high correlation values with other predicting features should be removed or reworked due to the issue of multicollinearity (Alin 2010). If not dealt with, this causes problems when trying to interpret what features actually are influential on the output of a model.

The correlation matrix can be found at Appendix D.1. Here it can be observed that the correlations with our target - 'result', range anywhere

between an absolute minimum value of 0.17 from the 'firstdragon' and 'assistsat10' features, to 0.66 from the 'firstbaron' variable. These correlations suggest that most features have a weak-moderate, with a few stronger predictors such as 'firstbaron' and 'firstthreetowers'. When the correlation between features is examined, features containing assists and kills at the 10 and 15 minute mark are highly positively correlated with a values of 0.80 and 0.78. When correlations are high between features, these features should be either be ignored or transformed. Here another test for multicollinearity is also used, called the Variance Inflation Factors or VIF. This tests each feature for how much variance the feature adds to the overall variance of the model, and generally describes how collinear a feature is with its other features. The results can be seen in Table 3.2, where it is deemed that any variable above 10 may be an issue.

Here a feature is deemed an issue if its VIF value exceeds 10, which many variables do - including those that also have large correlation values. Using the knowledge gained from both these tests, it is chosen to transform the kill and assist features, as they both are deemed useful to the model due to their correlation to the target variable. The 'xpdiff' and 'golddiff' features will also be removed. These four features will simply be combined to create two new features called 'KillParat10' and 'KillParat15', and were constructed as such:

$$Kill\ Participations = Kills + Assists$$

Checking the correlation of these new features now shows that the relationship between the target remains stable, whilst removing the problematic large correlations between predictor features in the dataset. The VIF test is then repeated to ensure that the dataset no longer contains any potential multicollinearity issues, and every feature passes with a sub-10 VIF as seen in Table 3.2. Another correlation issue is seen between recurrent features at different minute intervals in-game. This is fixed by splitting these features into different datasets which in-turn will create separate models based on the statistics from the 10-minute mark, the 15-minute mark and the 20-minute

Table 3.2: A table showing the results of the VIF test

Feature Name	**VIF Before**	**VIF After**
playoffs	1.217690	1.235018
patch	12.464378	3.160019
gamelength	1.006723	8.216505
firstblood	2.814385	2.447549
firstdragon	1.876673	1.804742
firstherald	4.543882	3.472183
firstbaron	2.484599	2.493748
firsttower	4.868430	3.944228
firstmidtower	4.583104	4.571757
firsttothreetowers	5.468883	5.380049
golddiffat10	15.818089	-
xpdiffat10	8.758268	-
csdiffat10	13.385455	4.371368
killsat10	54.651732	-
assistsat10	18.985235	-
deathsat10	30.215561	8.042111
golddiffat15	20.547368	-
xpdiffat15	17.244354	-
csdiffat15	16.369323	4.886503
killsat15	76.322270	-
assistsat15	27.789294	-
deathsat15	40.345937	9.899573
KillParat10	-	6.116191
KillParat15	-	7.917371

mark. A description of the data found across all three data subsets can be seen in Table 3.3.

Balancing the data was deemed unnecessary due to the target variable - 'result', having a 52.8 - 47.2% win to loss balance as seen in Figure 3.1. This means that the dataset does not require any balancing techniques such as SMOTE to be applied, as it already should give reliable and decisive outputs.

Finally, the data is put through the normalization process, re-scaling the data in each column between the values of 0 and 1. This step is completed to

Table 3.3: A table describing the data found within the dataset

Feature Name	**Description**
Playoffs	If the match is a playoff game
Patch	The game patch the match is played on
Gamelength	The duration the game was played
First Blood	If the first kill was obtained by the blue team
First Dragon	If the first Dragon of the match was slain by the blue team
First Herald	If the first Herald of the match was slain by the blue team
First Baron	If the first Baron of the match was slain by the blue team
First Tower	If the first Tower of the match was destroyed by the blue team
First Mid Tower	If the first Mid-lane Tower of the match was slain by the blue team
First to Three Towers	If the first three Towers of the match were destroyed by the blue team
CS Diff at X	The differential in number of minions slain between the teams at a given interval
Deaths at X	The number of deaths by the blue team at a given interval
Kill Participations at X	The number of kill participations of the blue team at a given interval

reduce the effects of the magnitude of feature scale on the model, potentially creating faster converging and better performing models (Jayant Verma n.d.).

3.3 Data Modelling

Now that the dataset has been properly collected, cleansed and processed, the data can enter the data modelling process. With this data the target is a binary classification problem, meaning that a classification algorithm is required to help generate a probability for our target variable based on our features. As mentioned in Section 2, previous works have used a variety of methods such as Naïve Bayes classifiers, the Random Forest algorithm,

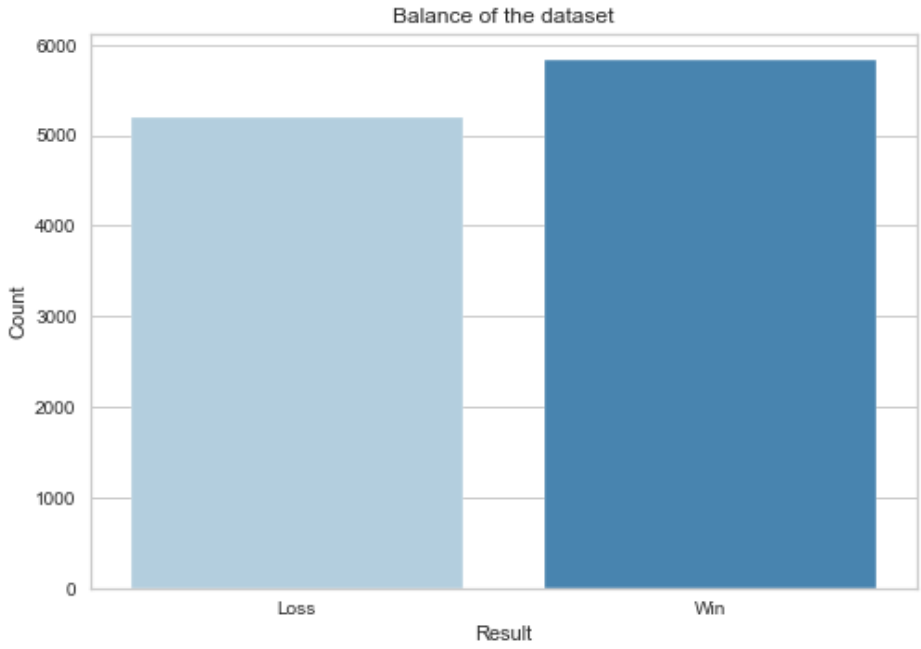


Figure 3.1: A graph showing the target balance of the dataset

Recurrent Neural Networks and Gradient Boosting. For this instance, it is chosen to use the Python Libraries - 'scikit-learn' and 'pycaret', they both offer a wide range of machine learning techniques After reviewing the possible models, it was chosen that the Random Forest algorithm would be used to create the baseline model. Then the PyCaret library would be used to test a larger pool of techniques and ultimately tune the model until the final model is chosen. The Random Forest algorithm is a commonly used supervised machine learning algorithm that creates an ensemble of decision trees and uses a bagging, majority voting system to help classify our problem - see Figure 3.2. This technique ensures that the chance of overfitting is kept to a minimum, as well as allowing for easy determination of importance from any given feature in the model.

A binary logistic regression model was also developed following the Random Forest Model. Like the previous model, a logistic regression classifier can estimate the probability of a target binary response variable using a set of predictor variables. This machine learning technique is based upon the

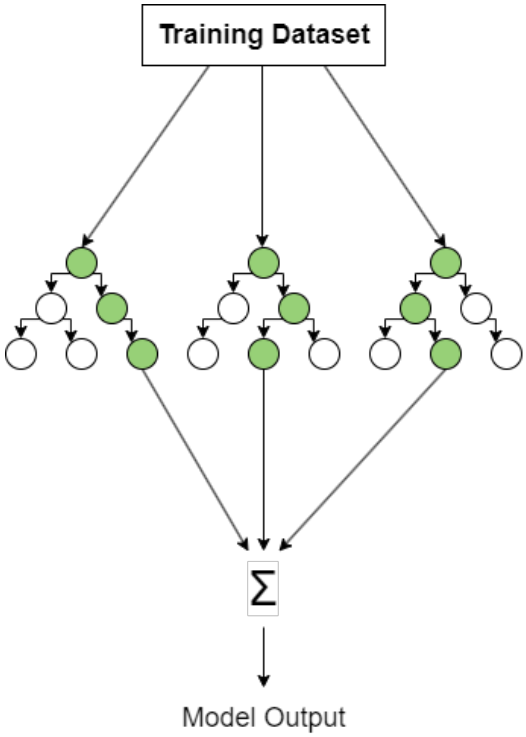


Figure 3.2: A diagram showing how the Random Forest algorithm classifies

natural logarithm of the odds; commonly referred to as a logit or log-odds, system for each event with a logistic function converting these log-odds into a probability. This logistic function is a sigmoid function that takes inputs between the bound of zero and one, it is defined as follows:

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Here β_0 is the intercept from the linear regression equation which is solved using numerical methods, and $\beta_1 x$ is the regression coefficient. With x being a combination of values from the predictor variables. Once solved for $p(x)$, it will provide the probability for the Blue-side team winning a given match.

The data then needs to be split into training data and testing data. Here a subset of the total dataset is split purely to train the machine learning model, before it is deployed to unseen training data. This help guarantees

the model ensuring that there is no overfitting to the seen data, and validates the quality of the model. The data split ratio is chosen at a 70:30 train-test data split. This allows a larger split to train the model, and the test data will be tested on a more robust, accurate model because of this. A technique called Stratified K-Fold cross validation is used to get an accurate representation of the overall population balance from the target variable - reflecting the 52.8 - 47.2% win-loss balance, as well as evaluating a truer accuracy by summarising models built upon K subsets of data. This leads to a model that is less biased overall and more realistic in its prediction accuracies.

Table 3.4: A table describing the models that were built

Model	Technique	Dataset
1	Logistic Regression	10-minute
2	Logistic Regression	15-minute
3	Gradient Boosting Classifier	20-minute

Table 3.4 shows the three models which were built for analysis purposes. The first two models were built using the Logistic Regression with the 10, and the 15 minute datasets, and the final model was built with the Gradient Boosting Classifier using the 20 minute dataset.

3.4 Summary

In this Chapter, the data extracted from Oracle's Elixir was explored and described in a data overview. The methodology behind the data preparation were rationalised during the data cleansing and pre-processing subsections, highlighting key techniques used to ensure the data modelling process would work correctly. This helped reduce the dataset from over 140,000 rows with 100+ features to a more succinct and useful 11,000 rows with less than 20 features. This was then followed by the data modelling subsection, that explained the machine learning techniques used to create the predictive mod-

elling system. Following this chapter is Chapter 4, which will present and analyse the results of this work.

Chapter 4

Results and Analysis

After establishing the theories and methodologies behind our model, this chapter introduces the performance metrics of the models and how the results from these models will be assessed. The performance metrics of the prediction models will then be analysed, using figures and tables to visualise the data. The machine learning algorithms are evaluated to determine the best model at predicting on unknown data. The feature importance of each model will then be evaluated, and are further contextualised as to why they are relevant within the model.

4.1 Performance Metrics

The performance of any algorithm is evaluated by a few factors that are measured on the test dataset, where the trained model attempts to correctly predict the target field and this performance is then compared to its true outcome. This assessment method is the norm on any binary classification problem and often will be visualised using a confusion matrix. The confusion matrix is a simple visualisation in which the four possible permutations of predictions are displayed. As seen in Figure 4.1, it compares the number of predictions of wins and losses made by the model versus the number of real wins and losses found in the dataset.

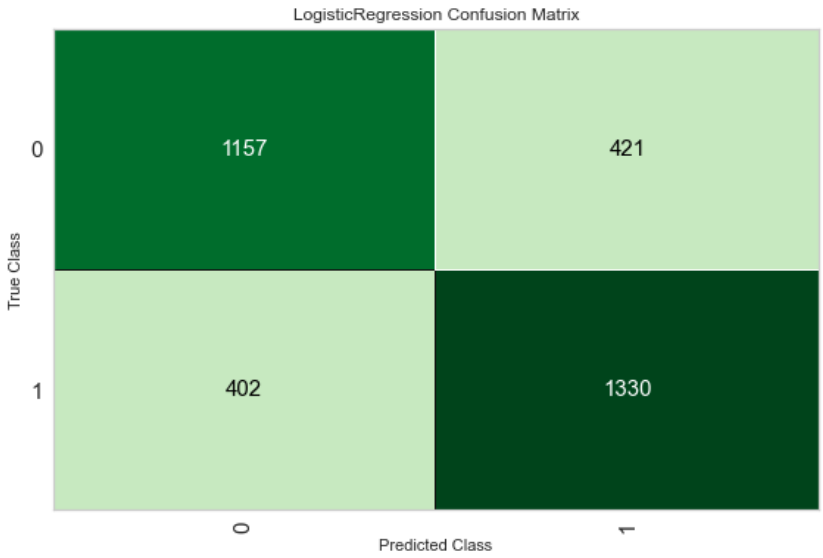


Figure 4.1: A confusion matrix of a Logistic Regression Model trained on the 10 minute dataset

On the y-axis the true class of the result can be seen, and on the x-axis there is the model's predicted class. These four possible permutations are commonly referred to as follows:

- True Positive (TP) is seen in the bottom right of the matrix, this is when the predicted win actually reflects the true win.
- True Negative (TN) is seen in the top left of the matrix, this is when the predicted loss actually reflects the true loss.
- False Positive (FP) is seen in the top right of the matrix, this is when the predicted win is incorrect and the true class is a loss.
- False Negative (FN) is seen in the bottom left of the matrix, this is when the predicted loss is incorrect and the true class is a win.

By collecting these outcome values, the effectiveness of each model is then able to be assessed via many performance metrics that are calculated using these values. These measures each give an understanding of a model, with different implications depending on the objective of the project being

assessed. In this study, the focus is on assessing the correct classifications and there is little to no consequences for misclassification. Therefore, the main criteria needed are the proportion of true classifications - aka the number of TPs and TNs compared to the whole prediction group. This calculation is also known as the Accuracy of the prediction, and is calculated as follows:

$$Accuracy = \frac{\sum TP + \sum TN}{\sum TP + \sum TN + \sum FP + \sum FN}$$

This is considered the key metric for this problem, as our dataset is pretty balanced so the other metrics such as Precision and Recall are not as important. Here Accuracy can help decide how well a given model is predicting the correct outcomes of an esports League of Legends match. The Accuracy of a model is only an effective metric if higher than the balance of the dataset, thus must be greater than 52.8% to show any improvement over guessing the balance of the data. Other metrics such as AUC, Precision, Recall are also calculated and evaluated, these metrics are calculated as such:

$$Precision = \frac{\sum TP}{\sum TP + \sum FP}$$
$$Recall = \frac{\sum TP}{\sum TP + \sum FN}$$
$$FPR = \frac{\sum FP}{\sum FP + \sum TN}$$

A Receiver Operating Characteristic (ROC) curve is a graph that plots the Recall versus the False Positive Rate (FPR), it plots these performance metrics between the bounds of zero and one. Area Under the ROC curve (AUC), is a measure that understandably calculates the area under the ROC curve which provides a measure of performance across all possible classification thresholds (Google n.d.). Essentially, AUC is the probability that a model can distinguish between a positive and negative class - in this study it represents the ability to distinguish wins from losses. Therefore, to con-

clusively accept a model as a feasible option, the Accuracy and AUC are the two core metrics that a model will have to score sufficiently on and will ultimately be the performance indicators to evaluate with.

4.2 Model Results

4.2.1 10 Minute Model

As seen in Table 4.1, most of the machine learning algorithms score quite similarly. The four metrics of Accuracy, AUC, Recall and Precision all are within a 5% range for every algorithm tested. Here the best performing machine learning algorithm is the Logistic Regression model, scoring an accuracy of 75.06% and an AUC of 82.30%. This is closely followed by the Linear Discriminant Analysis at an identical accuracy, but with an AUC of 82.28%.

Table 4.1: A table showing the model results for the 10 minute dataset

Model	**Accuracy**	**AUC**	**Recall**	**Precision**
Logistic Regression	0.7506	0.8230	0.7798	0.7574
Linear Discriminant Analysis	0.7506	0.8228	0.7813	0.7566
Ada Boost Classifier	0.7469	0.8181	0.7747	0.7546
Naive Bayes	0.7458	0.8128	0.7613	0.7599
Gradient Boosting Classifier	0.7458	0.8407	0.7759	0.7529
Quadratic Discriminant Analysis	0.7456	0.8363	0.7601	0.7604
SVM - Linear Kernel	0.7436	0.0000	0.7769	0.7492
Light Gradient Boosting Machine	0.7357	0.8316	0.7630	0.7453
Random Forest Classifier	0.7316	0.8230	0.7498	0.7462
Extra Trees Classifier	0.7248	0.8116	0.7459	0.7385
K Neighbors Classifier	0.7061	0.7794	0.7276	0.7212
Decision Tree Classifier	0.6804	0.6794	0.6969	0.6998

Interestingly enough the decision tree based algorithms that were commonly used in previous studies such as Random Forest, appear to perform

slightly worse than algorithms such as Logistic Regression or Linear Discriminant Analysis. These techniques are techniques based on linear regression analysis that opt to find a linear combination of features that cause separation in a problem. These results then may indicate that the problem of match outcomes in League of Legends is linearly separable. The model for the 10-minute mark was chosen to be built upon a logistic regression technique. Firstly, the model is tested across the data split by the stratified k-fold cross-validation and tuned afterwards. The resultant performance metrics can be seen in Table 4.2.

Table 4.2: A table showing the mean metrics for the 10 minute logistic regression model after 10-fold cross-validation

	Accuracy	**AUC**	**Recall**	**Precision**
Mean	0.7516	0.8226	0.7618	0.7682
Std	0.0150	0.0169	0.0285	0.0116

Table 4.2 shows very similar results to the previous stated values, meaning that the model tuning had very little effect on improving the model's performance. It also gives a standard deviation on the core metrics of under 2%. During the modelling, the feature importance of the features included in the dataset were calculated. Figure 4.2 shows the average relative importance of each feature when using the 10-minute dataset. The top 10 features by feature importance are presented down the y-axis in decreasing order according to their relative importance values. The most important features are those related to the player performance inside their lanes such as 'csdiffat10', 'deathsat10' and 'KillParat10', with more objective team-based features having lower importance in predicting the resultant outcome of a given match. At first this seems counter-intuitive, especially when given that teams must destroy towers in order to eventually win the game. However, it is likely that at this stage of the game there is very low likelihood that teams can effectively destroy towers, so the focus should be on progressing their individual leads inside their lane. These leads that they garner are reflected in the three

features mentioned previously, and should translate into further advantages in more objective oriented features in a later game-state.

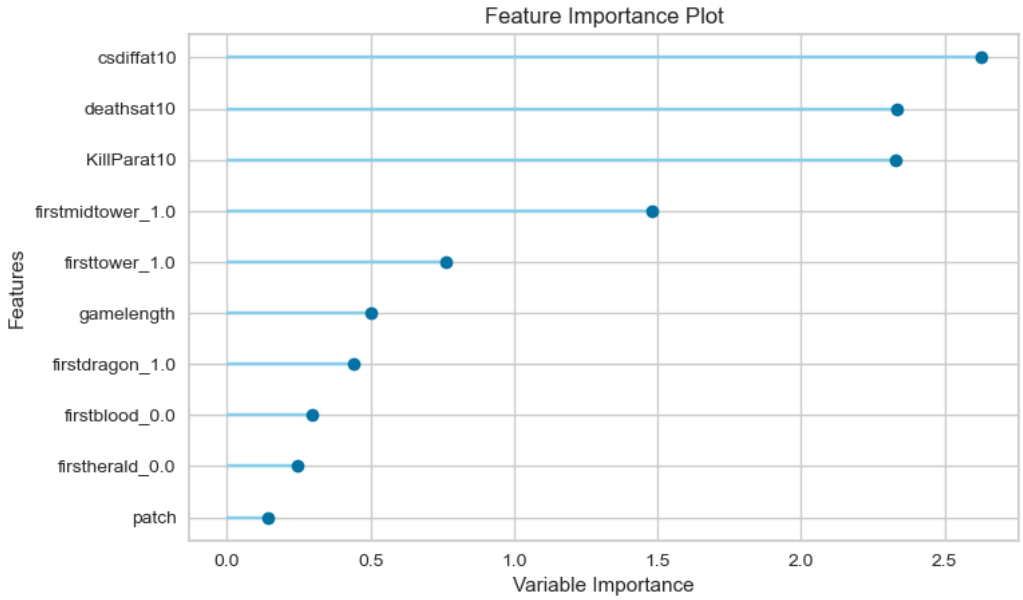


Figure 4.2: A graph showing the Relative Importance of the features in the Logistic Regression Model using the 10 minute dataset

4.2.2 15 Minute Model

Table 4.3 highlights that once again that all but one of the machine learning algorithms score within a 5% range of the four key metrics tracked. The two best performing machine learning algorithm remain the Logistic Regression model and the Linear Discriminant Analysis model, scoring an accuracy of 76.84% and 76.87%, and AUC values of 84.74% and 84.53% respectively. Once again the decision tree based algorithms that were commonly used in previous studies appear to vastly under-perform when compared to Logistic Regression or Linear Discriminant Analysis, with the Decision Tree Classifier falling behind by roughly 10% across all the four key metrics calculated. It is once again chosen for the Logistic Regression model to be the final model for the 15-minute dataset. The model is once again built and a stratified k-fold cross-validation is applied. The final performance metrics can be seen in Table 4.4.

Table 4.3: A table showing the model results for the 15 minute dataset

Model	**Accuracy**	**AUC**	**Recall**	**Precision**
Linear Discriminant Analysis	0.7687	0.8453	0.7994	0.7727
Logistic Regression	0.7684	0.8474	0.7972	0.7733
Ridge Classifier	0.7681	0.0000	0.7986	0.7721
Ada Boost Classifier	0.7662	0.8418	0.7899	0.7741
Quadratic Discriminant Analysis	0.7643	0.8578	0.7794	0.7772
Gradient Boosting Classifier	0.7638	0.8601	0.7935	0.7690
Naive Bayes	0.7589	0.8362	0.7694	0.7746
Light Gradient Boosting Machine	0.7577	0.8551	0.7877	0.7635
SVM - Linear Kernel	0.7562	0.0000	0.7835	0.7642
Random Forest Classifier	0.7519	0.8483	0.7745	0.7623
Extra Trees Classifier	0.7431	0.8359	0.7652	0.7545
K Neighbors Classifier	0.7307	0.8067	0.7449	0.7470
Decision Tree Classifier	0.6870	0.6854	0.7125	0.7021

Table 4.4 shows an accuracy increase of 0.22% after tuning, bumping the Logistical Regression model accuracy value to 77.06%. The change in the accuracy of the model between the 10 to 15 minute models only increases by

roughly 2%. This increase is much lower than expected when compared to previous studies such as Silva et al. (2018), Lee et al. (2020) achieving increases of 5-6% between the same period in game time, albeit whilst achieving accuracy levels on par or greater than its predecessors. These differences between studies are likely due to differences in the feature selection stage, as well as both the quality of the data collected and the metrics that the data offer. In the dataset used in this study, there are many binary objective-type features that are labelled as to which team had completed them first. These variables are often important milestones that help a team towards victory throughout the game, but most importantly they often can reveal which team is in control at a given game-time and thus be a key predictor into the victor of the overall match.

Table 4.4: A table showing the mean metrics for the 15 minute Logistic Regression model after 10-fold cross-validation

	Accuracy	**AUC**	**Recall**	**Precision**
Mean	0.7706	0.8478	0.7752	0.7889
Std	0.0200	0.0173	0.0289	0.0198

Again, the feature importance of the features included in the dataset were calculated and visualised. Figure 4.3 shows the average relative importance of each feature when using the 15-minute dataset. It remains that the most important features are those related to the player performance inside their lanes such as 'csdiffat15', 'deathsat15' and 'KillParat15', with slightly more emphasis on 'deathsat15' when compared to the previous model. In fact the relative importance for these features has roughly doubled with more objective team-based features having lower importance in predicting the resultant outcome of a given match. An interesting feature that appears here is in fact when the blue side team loses their first tower. The model believes this to be an important factor in the blue side achieving a victory, which is the opposite notion to what the previous model suggests. However, the model still believes that destroying the mid-lane tower is the prime team-based

objective on the map within the first 15 minute mark. This could lead to the conclusion that as long as the blue side team obtains the first mid-lane tower, but allows the opposition to take the first tower in either the top or bottom lane, that this will still lead to a higher probability of a match victory.

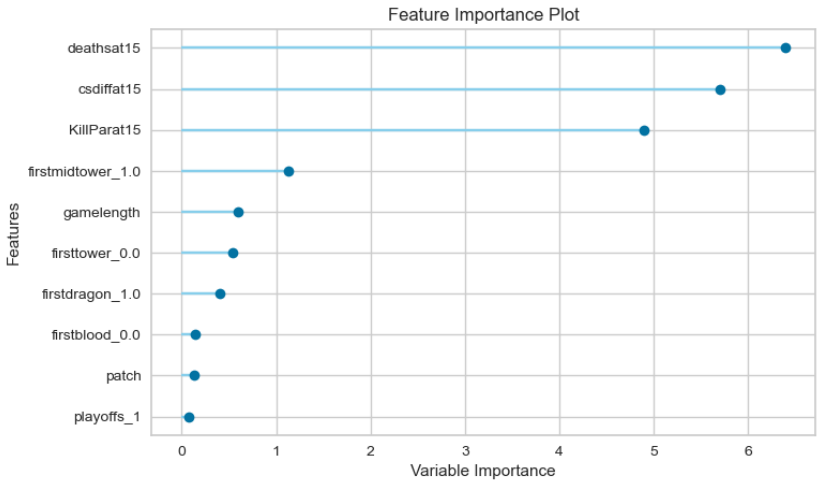


Figure 4.3: A graph showing the Relative Importance of the features in the Logistic Regression Model using the 15 minute dataset

4.2.3 20 Minute Model

Again all the machine learning algorithms score within a performance metric range of 5% as seen in Table 4.5. This just reiterates that there is minimal difference in performance between the machine-learning algorithms tested, with the decision tree based algorithms generally falling behind. This remains true aside from the Gradient Boosting Classifier, where it is in-fact the best performing model overall. It scored an accuracy of 85.85% and an AUC of 94.33%. It was then closely followed by the two previous algorithms mentioned in both Section 4.2.1 and 4.2.2 - the Logistic Regression and the Linear Discriminant Analysis at identical accuracies of 85.83%, but with respective AUC values of 92.66% and 92.57%. This observed sequential increase of accuracy in the models helps answer research question 1, concluding that prediction accuracy would increase by between 2% to 9% every 5 minutes of in-game time with these gains appearing to potentially be exponential. The prediction accuracy of the match outcome model does vary based on the machine learning technique used, as asked in research question 3. When comparing the linear techniques such as the Linear Discriminant Analysis or Logistic Regression, the decision tree-based algorithms tend to be weaker predictors of match outcomes. Therefore it is concluded that the algorithms vary greatly, with a mean increase of 6.93% between the best performing machine learning algorithm and worst algorithm.

For this model, the Gradient Boosting Classifier is chosen as it scores high on all major performance metrics calculated. After the model is tuned, the accuracy increases to 85.99%, an increase of 0.14% - see Table 4.6. The change in the accuracy of the model between the 10 to 15 minute models only increases by roughly 2%. This is much lower than expected when compared to previous studies such as Silva et al. (2018), Lee et al. (2020) achieving increases of 5-6% between the same period in game time, albeit whilst achieving accuracy levels on par or greater than its predecessors.

The average relative importance for the Gradient Boosting Classifier

Table 4.5: A table showing the model results for the 20 minute dataset

Model	**Accuracy**	**AUC**	**Recall**	**Precision**
Gradient Boosting Classifier	0.8585	0.9433	0.8782	0.8586
Logistic Regression	0.8583	0.9266	0.8648	0.8679
Linear Discriminant Analysis	0.8583	0.9257	0.8513	0.8780
Quadratic Discriminant Analysis	0.8582	0.9406	0.8706	0.8634
Ridge Classifier	0.8581	0.0000	0.8511	0.8778
Ada Boost Classifier	0.8559	0.9245	0.8623	0.8655
Random Forest Classifier	0.8525	0.9387	0.8653	0.8581
SVM - Linear Kernel	0.8521	0.0000	0.8504	0.8701
Light Gradient Boosting Machine	0.8494	0.9392	0.8645	0.8538
Extra Trees Classifier	0.8451	0.9315	0.8584	0.8512
K Neighbors Classifier	0.8367	0.9057	0.8433	0.8482
Naive Bayes	0.8284	0.9122	0.8509	0.8300
Decision Tree Classifier	0.8024	0.8017	0.8140	0.8137

Table 4.6: A table showing the mean metrics for the 20 minute Gradient Boosting Classifier model after 10-fold cross-validation

	Accuracy	**AUC**	**Recall**	**Precision**
Mean	0.8599	0.9433	0.8789	0.8602
Std	0.0061	0.0042	0.0130	0.0087

model is shown in Figure 4.4. Now at the 20 minute mark the features that are most influential inside the model have changed. Suddenly the player performance features that were dominating the relative importance previously such as 'csdiffat15', 'deathsat15' and 'KillParat15' have now fallen out of favour versus the team-based map objectives. Here the 'firstbaron' metric becomes the focal feature in whether a team will achieve the victory, with a relative importance three times as large as the next highest. The next two key features are 'gamelength' and 'firsttothreetowers', reiterating the team-based map objectives being key to achieving victory as the gamelength increases. The 'firstbaron' feature being this impactful makes logical sense when accounting for the context behind the powerful buff that this monster grants when slain. This buff gives players the ability to increase the power

of their surrounding minions, and thus a large increase in the capability to destroy enemy towers or inhibitors. Subsequently, the team that slays the baron is granted a large sum of experience and gold through the monster itself, as well as the towers that could also fall. This large spike in income and champion strength likely helps the team to win future fights and then the match, this snowballing effect helps form a reasoning for research question 1.

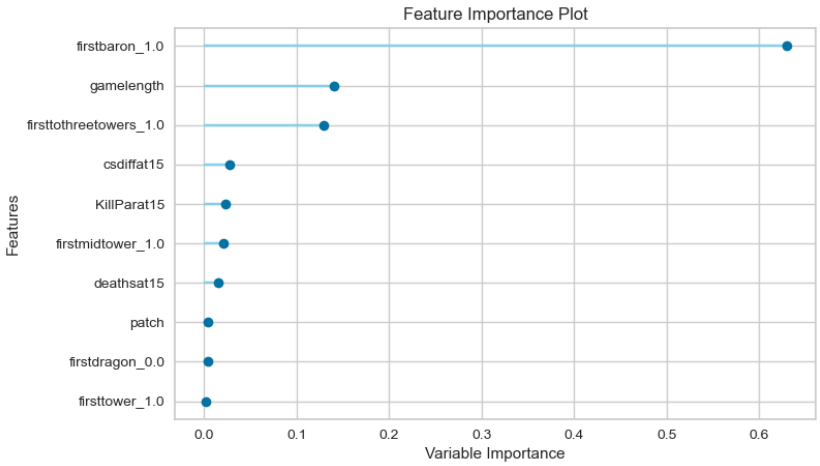


Figure 4.4: A graph showing the Relative Importance of the features in the Gradient Boosting Classifier Model using the 20 minute dataset

4.3 Discussion of Results

The results of these three models indicate that there is potential for a highly efficient League of Legends esports prediction model. Using a mixture of Logistic Regression models and Gradient Boosting Classifiers the accuracy of predictions were calculated between the 10 to 20 in-game minute mark. A summary of these results can be seen in Table 4.7.

The assumption can be made that with no prior knowledge any given League of Legends esports match result can be roughly predicted via coin-flip, with the class balance shown previously in Figure 3.1. However using

Table 4.7: A table comparing the performance metrics for all three models

Model	**Accuracy**	**AUC**	**Recall**	**Precision**
10 Minute LR	0.7516	0.8226	0.7618	0.7682
15 Minute LR	0.7706	0.8478	0.7752	0.7889
20 Minute GBC	0.8599	0.9433	0.8789	0.8602

the models create in this study, an average layman could theoretically predict the match outcome over 75% of the time when given some key features at the 10 minute mark, with this increasing to 86% by the 20 minute mark. Using the key figures seen in Table 4.7, it can be seen that when compared to previous attempts at a similar form of prediction modelling for League of Legends, these models exceed the metrics found by many of its predecessors.

Silva et al. (2018) used RNNs for their prediction modelling using esports data between 2015 and early 2018, similar to this study they produced results for time intervals during the game. Lee et al. (2020) used a Random Forest Classifier for their modelling using ranked data extracted from the Riot API, and once again produced results for varying time intervals in-game. The results published in those studies have been collated into Table 4.8. Here it can be seen that the prediction accuracy is much higher than those stated in previous studies, with improvements up to 7% at the 10 minute interval. This means that the models created during this study not only contribute more proof to the growing numbers of literature surrounding predictive modelling in the esports industry, but are also outperforming the most recent literature in the field. As seen with all the results collated, the improvements to the prediction accuracy increases as the in-game timer increases. All three studies show an accuracy increase of 10% from the 10 minute mark to the 20 minute mark, regardless of the initial accuracy. This reaffirms the aim behind research question 1 stated in Section 2.4, ultimately verifying the idea that a match of League of Legends has a snowballing nature, thus the outcome becomes more predictable as it progresses. Research question 2 can generally be assumed to be true when looking at the correlations between

many of the features during the data exploration phase, these correlation values are highlighted in Figure D. Many of these features have a correlation level between 0.4 and 0.8, which suggests a moderate to strong level of correlation and can lead the overall model to suffer from some level of multicollinearity. This is also verified when looking at the VIF values recorded in Table 3.2, with transformations and removal of many features needed in order to minimise the likelihood of multicollinearity within the model. The best performing models in this study were the Logistic Regression and the Gradient Boosting Classifier, with the Logistic Regression being used for two of three models. Research question 3 wished to explore the various machine learning algorithms, with many of the previous studies concluding that the Random Forest algorithm is the optimal algorithm for the prediction model, This ultimately was proven to be mostly wrong, with conclusive evidence that shows that decision-tree based algorithms perform worse when compared to linear-based algorithms - with the exception of the Gradient Boosting Classifier for the 20 minute model.

Table 4.8: A table comparing the accuracy for different time intervals from other studies (Silva et al. 2018, Lee et al. 2020)

Study	**10 Minutes**	**15 Minutes**	**20 Minutes**
This Study	0.7516	0.7706	0.8599
Silva et al.	0.6869	0.7523	0.8018
Lee et al.	0.6788	0.7318	0.7649

Chapter 5

Conclusion

5.1 Summary of Works

This study aimed to investigate the ability to accurately predict the outcome of an esports match of League of Legends through the use of the in-game statistics at various intervals throughout the match. Whilst a few studies have been completed in academic literature about this subject, this study tested a wider range of machine learning algorithms with data that is updated to match the current state of the game - rather than outdated seven-year-old match data. A combination of Logistic Regression models and Gradient Boosting Classifiers were applied to League of Legends esports data acquired from the website Oracle's Elixir. This data was collected from the 2021 League of Legends competitive leagues globally, made up of over 100 individual in-game data points. The application of the logistic regression model in the 10-minute interval produced an improvement of 22% over the 52.8% no information rate benchmark. This improvement only further increased through the next two models of the 15 and 20 minute interval, with the final gradient boosting classifier achieving a prediction accuracy of 85.99%.

In addition, analysis of the features that play a key roles in a game's outcome were undertaken, calculating the importance of these features and

the implications of what they could possibly mean at every given stage of the game. It was found that in the first 15 minutes of the game, it was clear that individual player statistics were the most impactful in the decision of the game, with Figure 4.2 and 4.3 showing a clear support in the ability for a player to keep their deaths to a minimum, whilst still creating a minion killed differential and playing a part in any kills that can occur for their team. This suggests that the first 15 minutes of the game should be concentrated on generating both a gold and experience lead over the opponent in each position, attempting to grab kills whenever possible. The model also suggests the fact that the mid-lane tower is the most important tower to take on the map, and should be the secondary focus of a team in these first 15 minutes. Once the 20-minute mark is met, the model now proposes that the singular feature that is of importance is now the neutral monster baron. Therefore, confirming the objective of this study - that a League of Legends esports match can in-fact be predicted and with a high prediction rate.

These results are interesting for both players, coaches and managers within the esports industry, given that these findings highlight that this model outperforms the current rivalling academic literature on this subject (Lee et al. 2020, Silva et al. 2018). As esports teams aim for greatness, and to ultimately win both their domestic regional leagues and then qualify to compete for the world championship. They must ensure they are consistently winning matches and are the best in their respective regions, and they can achieve this through efficient practice. Using models such as these, teams could alter the feature set and produce predictive systems to optimise and streamline their gameplay to improve their overall win-rates. These systems could highlight key weaknesses in a teams game-plan, thus helping players practice towards these features of high importance found in these models.

5.2 Research Limitations

There are a few limitations of this dissertation. Firstly, given the fact that multicollinearity plays a major part in which data can or should be used

in a model of League of Legends. The explanatory variables being so interconnected inside the game leads to an issue where it becomes hard for machine learning algorithms to properly distinguish the true effect of a singular variable, as the consequences of a single effect in one variable can cause a cascading effect onto other independent variables. This ultimately can cause massive variance or errors in the predictive model where predictions can be slightly incorrect due to these errors cause inside the feature selection process. Given this knowledge, the results from any prediction modelling about League of Legends and any game within the MOBA category will inherently suffer due to this fact. This is due to the core gameplay mechanics revolving around the increasing power of a champion through two key aims - experience and gold. Despite these key aims not being the core objective to achieve the victory, they are highly influential in achieving victory and thus correlated highly with victory, and this means that a player's intention should be to achieve as much gold and experience as possible whilst also denying their opponents. Moreover, with almost all the actions a player can take in-game resulting in either gold or experience, all the features within a dataset become highly correlated to both gold and experience metrics. In order to minimise this both these were removed from the final dataset to reduce the correlation between data as much as possible, but some moderate correlations still remained.

Secondly, it remains to be seen how effective the feature importance values truly are when these strategies are applied in-game. The idea of relative importance is that it calculates an importance score based on the feature's apparent significance to the model's target variable, therefore a higher score means that a specific feature will contribute more to predicting the target variable than other features. This idea helps the user gain understanding of the relationships between the predicting features and the target variable, whilst also giving insight into irrelevant features in the model. However, whilst this feature importance can highlight which features may be relevant to the target variable, this does not help apply this to the context behind the features. The interpretation of these important features normally requires

the insight of an expert in the given domain, in this case someone with a great understanding of League of Legends. Only then can the findings be properly contextualised, but this examination and application of what these features could mean is largely up to the domain expert's opinion so is highly subjective and can easily change depending on the expert questioned.

5.3 Further Research

For future work, the dataset could be expanded to delve further into pre-game effects such as a further look into the champion select process described in Section 1.2.2. A support tool could be produced for this type of model, using an online web application. This could use champion specific statistics combined with historical match data in order to dictate the correct champion to be selected at a given phase of the champion selection phase, and could predict the highest likelihood champion pick from the opposition. This would allow for an initial prediction value before the game begins, maximising a teams potential for winning before the game truly begins. It could also increase the synergistic effects of a given team composition, drafting champions that inherently work well together using interaction terms between all the possible champion picks.

Another piece of research could be investigating the ability for comebacks in these games, and the tools or objectives that teams from losing positions can use in order to make a comeback for victory. This would analyse winning teams when they fall a certain threshold percentage of team gold behind, and the key features that enabled them to recover from the deficits incurred. It would use a similar methodology, with various machine learning techniques to analyse how an individual players' performance changes whilst in these deficits and the overall chances of winning whilst in those positions. There is room for more of a qualitative study, with psychology of player performance obtained through questionnaire based methodologies.

Research could even be potentially introduced in the identification of

arbitrage betting through esports bets, using machine learning prediction analytics models to identify when bookmakers are exposed to excess risk due to miscalculation of match or event outcomes. It would use machine learning techniques to analyse the differential in calculated probabilities between a match outcome prediction model, such as the one created in this study, versus the odds published by various bookmakers in order to find a potential profitable betting strategy.

Bibliography

Abarbanel, B. & Johnson, M. R. (2019), 'Esports consumer perspectives on match-fixing: implications for gambling awareness and game integrity', *International Gambling Studies* **19**(2), 296–311.

Absolute Reports (2022), 'Global esports betting market'.

Alin, A. (2010), 'Multicollinearity', *Wiley interdisciplinary reviews: computational statistics* **2**(3), 370–374.

Ani, R., Harikumar, V., Devan, A. K. & Deepa, O. (2019), Victory prediction in league of legends using feature selection and ensemble methods, *in* '2019 International Conference on Intelligent Computing and Control Systems (ICCS)', IEEE, pp. 74–77.

Apostolou, K. & Tjortjis, C. (2019), Sports analytics algorithms for performance prediction, *in* '2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)', IEEE, pp. 1–4.

Brownlee, J. (2019), 'How to choose a feature selection method for machine learning', *Machine Learning Mastery* **10**.

Byrne, L. (2019), 'Betway agrees to a one-year sponsorship deal for BLAST Pro Series', *Esports Insider* . Accessed: 20-07-2022.
URL: *https://esportsinsider.com/2019/03/betway-agrees-to-a-one-year-sponsorship-deal-for-blast-pro-series/*

Chen, Z., Nguyen, T.-H. D., Xu, Y., Amato, C., Cooper, S., Sun, Y. & El-Nasr, M. S. (2018), The art of drafting: a team-oriented hero recommen-

dation system for multiplayer online battle arena games, *in* 'Proceedings of the 12th ACM Conference on Recommender Systems', pp. 200–208.

Dos Reis, V. (2017), 'Q&a: The rise of esports betting and the challenges the industry faces', *Gaming Law Review* **21**(8), 630–633.

Dot Esports (2021), 'FPX's Bo handed 4-month ban for match-fixing in Chinese academy league'. Accessed: 23-07-2022.
URL: *https://dotesports.com/league-of-legends/news/fpx-bo-handed-four-month-ban-match-fixing*

Duckett, C. (2016), 'Google AlphaGo AI clean sweeps European Go champion', *ZDNet* . Accessed: 24-07-2022.
URL: *https://www.zdnet.com/article/google-alphago-ai-clean-sweeps-european-go-champion/*

Esports Earnings (n.d.), 'Largest Individual Tournament Prize Pools'. Accessed: 16-07-2022.
URL: *https://www.esportsearnings.com/tournaments/largest-team-tournaments*

Even, W. E. & Noble, N. R. (1992), 'Testing efficiency in gambling markets', *Applied Economics* **24**(1), 85–88.

Feddersen, A., Humphreys, B. R. & Soebbing, B. P. (2018), 'Sentiment bias in national basketball association betting', *Journal of Sports Economics* **19**(4), 455–472.

Gaina, R. & Nordmoen, C. (2018), 'League of legends: A study of early game impact'.

Google (n.d.), 'Classification: ROC Curve and AUC'. Accessed: 08-08-2022.
URL: *https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc*

Gray, G. T. & Wert-Gray, S. (2012), 'Customer retention in sports organization marketing: examining the impact of team identification and satisfac-

tion with team performance', *International Journal of Consumer Studies* **36**(3), 275–281.

Hanke, L. & Chaimowicz, L. (2017), A recommender system for hero lineups in moba games, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainmen', pp. 43–49.

Jayant Verma (n.d.), '2 Easy Ways to Normalize data in Python'. Accessed: 05-08-2022.
URL: *https://www.digitalocean.com/community/tutorials/normalize-data-in-python*

Kokkinakis, A., York, P., Patra, M., Robertson, J., Kirman, B., Coates, A., Pedrassoli Chitayat, A., Demediuk, S. P., Drachen, A., Hook, J. D. et al. (2021), 'Metagaming and metagames in esports', *International Journal of Esports* .

Lee, S.-K., Hong, S.-J. & Yang, S.-I. (2020), Predicting game outcome in multiplayer online battle arena games, *in* '2020 International Conference on Information and Communication Technology Convergence (ICTC)', IEEE, pp. 1261–1263.

Lin, L. (2016), 'League of legends match outcome prediction', *Comput. Sci. Dept., Univ. Stanford, Stanford, CA, USA, Rep* .

LoLEsports (2022), 2022 LCS Rule Set, Technical report, Riot Games.
URL: *https://lolesports.com/article/2021-lcs-rule-set-and-penalty-index/bltd4266fc4777c19a9*

Mangeloja, E. (2019), 'Economics of esports', *Electronic Journal of Business Ethics and Organization Studies* **24**(2).

McLaughlin, D. (2021), 'Worlds 2021 final draws 73.8 million peak concurrent viewers, Riot reports', *Upcomer* . Accessed: 18-07-2022.
URL: *https://upcomer.com/worlds-2021-final-draws-73-8-million-peak-concurrent-viewers-riot-reports*

Na, S., Su, Y. & Kunkel, T. (2019), 'Do not bet on your favourite football team: the influence of fan identity-based biases and sport context knowledge on game prediction accuracy', *European Sport Management Quarterly* **19**(3), 396–418.

Newzoo (2022), Global Esports & Live Streaming Market Report, Technical report, Newzoo.

Novak, A. R., Bennett, K. J., Pluss, M. A. & Fransen, J. (2020), 'Performance analysis in esports: modelling performance at the 2018 league of legends world championship', *International Journal of Sports Science & Coaching* **15**(5-6), 809–817.

Oracle's Elixir (n.d.), 'Match Data Downloads'. Accessed: 06-06-2022.
URL: *https://oracleselixir.com/tools/downloads*

Pantzalis, V. C. & Tjortjis, C. (2020), Sports analytics for football league table and player performance prediction, *in* '2020 11th International Conference on Information, Intelligence, Systems and Applications (IISA', IEEE, pp. 1–8.

Ravari, Y. N., Spronck, P., Sifa, R. & Drachen, A. (2017), Predicting victory in a hybrid online competitive game: The case of destiny, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment', Vol. 13, pp. 207–213.

Riot Games (2021), 'Total League of Legends Playercount'. Accessed: 18-07-2022.
URL: *https://twitter.com/riotgames/status/1455172784938651649?s=20-&t=dydJavkVHnrR5YnceUFnMw*

Sarlis, V. & Tjortjis, C. (2020), 'Sports analytics—evaluation of basketball players and team performance', *Information Systems* **93**, 101562.

Saunders, M., Lewis, P. & Thornhill, A. (2007), 'Research methods', *Business Students 4th edition Pearson Education Limited, England* .

Scelles, N., Peng, Q. & Valenti, M. (2021), 'Do the peculiar economics of professional team sports apply to esports? sequential snowballing literature reviews and implications', *Economies* **9**(1), 31.

scikit-learn (n.d.), '1.10. Decision Trees'. Accessed: 24-07-2022.
URL: *https://scikit-learn.org/stable/modules/tree.html*

Shen, Y., Zhou, J., Lin, W. & Feng, Z. (2022), A deep learning supported sequential recommendation mechanism for ban-pick in moba games, *in* '2022 IEEE 2nd International Conference on Software Engineering and Artificial Intelligence (SEAI)', IEEE, pp. 259–265.

Silva, A. L. C., Pappa, G. L. & Chaimowicz, L. (2018), 'Continuous outcome prediction of league of legends competitive matches using recurrent neural networks', *SBC-Proceedings of SBCGames* pp. 2179–2259.

Thabtah, F., Zhang, L. & Abdelhamid, N. (2019), 'Nba game result prediction using feature analysis and machine learning', *Annals of Data Science* **6**(1), 103–116.

TIOBE (n.d.), 'TIOBE Index for August 2022'. Accessed: 28-07-2022.
URL: *https://www.tiobe.com/tiobe-index/*

Ward, M. R. & Harmon, A. D. (2019), 'Esport superstars', *Journal of Sports Economics* **20**(8), 987–1013.

Wilkens, S. (2021), 'Sports prediction and betting models in the machine learning age: The case of tennis', *Journal of Sports Analytics* **7**(2), 99–117.

Yang, Y., Qin, T. & Lei, Y.-H. (2016), 'Real-time esports match result prediction', *arXiv preprint arXiv:1701.03162* .

Appendix A

Ethical Approval

SAGE-HDR (v3.5 31/05/22)

Response ID	Completion date
904570-904552-99186934	2 Sep 2022, 12:07 (BST)

1	**Applicant Name**	Calum Palmer
1.a	**University of Surrey email address**	cp00628@surrey.ac.uk
1.b	**Level of research**	Postgraduate Taught (Masters)
1.b.i	**Please enter your University of Surrey supervisor's name. If you have more than one supervisor, enter the details of the individual who will check this submission.**	Dr Colin Fu
1.b.ii	**Please enter your supervisor's University of Surrey email address. If you have more than one supervisor, enter the details of the supervisor who will check this submission.**	c.fu@surrey.ac.uk
1.c	**School or Department**	Surrey Business School
1.d	**Faculty**	FASS - Faculty of Arts and Social Sciences

2	**Project title**	League of Legends Victory Prediction Analysis

3	**Please enter a brief summary of your project and its methodology in 250 words. Please include information such as your research method/s, sample, where your research will be conducted and an overview of the aims and objectives of your research.**	This study aims to investigate the ability to accurately predict the outcome of an esports match of League of Legends through the use of the in-game statistics at various intervals throughout the match. Using this type of predictive analytics, teams could use these predictive systems to optimize and streamline their gameplay to improve their overall win rates and highlight key weaknesses in an opposing team's game plan. Various machine learning techniques will be used, models will be created and evaluated using certain performance metrics, such as Accuracy and AUC. The best machine learning technique will be used according to these metrics, with the preliminary models likely being completed using the Random Forest algorithm. Feature importance analysis will also be completed, highlighting key features that show that they play key roles in a game's outcome. The data will be Using Python, the data will be transformed, cleansed, and prepared before being split into smaller subsets based on the in-game timer, resulting in intervals of 10, 15, and 20 minutes. The objectives are to reach Accuracy and AUC values similar to some of its contemporaries found in previous literature, at roughly 60-70% accuracy depending on the in-game time. Another objective is to find the core features that decide the game at each interval of the game.

4	**Are you planning to join on to an existing Standard Study Protocol (SSP)? SSPs are overarching pre-approved protocols that can be used by multiple researchers investigating a similar topic area using identical methodologies. Please note, SSPs are only being used by 2 schools currently and cannot be used by other schools. Using an SSP requires permission and sign-off from the SSP owner**	NO

5	**Are you making an amendment to a project with a current University of Surrey favourable ethical opinion or approval in place?**	NO

6	**Does your research involve any animals, animal data or animal derived tissue, including cell lines?**	NO

8	**Does your project involve human participants (including human data and/or any human tissue*)?**	NO

9	**Will you be accessing any organisations, facilities or areas that may require prior permission? This includes organisations such as schools (Headteacher authorisation), care homes (manager permission), military facilities, closed online forums, private social media pages etc. This also includes using University mailing lists (admin permission). If you are unsure, please contact ethics@surrey.ac.uk.**	NO

10	**Does your project involve any type of human tissue research? This includes Human Tissue Authority (HTA) relevant, or non-relevant tissue (e.g. non-cellular such as plasma or serum), any genetic material, samples that have been previously collected, samples being collected directly from the donor or obtained from another researcher, organisation or commercial source.**	NO

11	**Does your research involve exposure of participants to any hazardous materials e.g. chemicals, pathogens, biological agents or does it involve any activities or locations that may pose a risk of harm to the researcher or participant?**	NO

12	**Will you be importing or exporting any samples (including human, animal, plant or microbial/pathogen samples) to or from the UK?**	NO

13	**Will any participant visits be taking place in the Clinical Research Building (CRB)? (involving clinical procedures; if only visiting the CRB to collect/drop-off equipment or to meet with the research team (i.e. for informed consent/discussion) select 'NO').**	NO

14	**Will you be working with any collaborators or third parties to deliver any aspect of the research project?**	NO

15	**Are you conducting a service evaluation or an audit? Or using data from a service evaluation or audit?**	NO

16	**Does your funder, collaborator or other stakeholder require a mandatory ethics review to take place at the University of Surrey?**	NO

17	**Does your research involve accessing students' results or performance data? For example, accessing SITS data.**	NO

18	**Will ANY research activity take place outside of the UK?**	NO

19	**Are you undertaking security-sensitive research, as defined in the text below?**	NO

20	**Does your project require the processing of special category1 data?**	NO

21	**Have you selected YES to one or more of the above governance risk questions on this page (Q10-Q20)?**	NO

22	**Does your project process personal data2? Processing covers any activity performed with personal data, whether digitally or using other formats, and includes contacting, collecting, recording, organising, viewing, structuring, storing, adapting, transferring, altering, retrieving, consulting, marketing, using, disclosing, transmitting, communicating, disseminating, making available, aligning, analysing, combining, restricting, erasing, archiving, destroying.**	NO

23	**Are you using a platform, system or server external to the University approved platforms (Outside of Microsoft Office programs, Sharepoint or OneDrive)?**	NO

24	**Does your research involve any of the above statements? If yes, your study may require external ethical review or regulatory approval**	NO

25	**Does your research involve any of the above? If yes, your study may require external ethical review or regulatory approval**	NO

26	**Does your project require ethics review from another institution? (For example: collaborative research with the NHS REC, the Ministry of Defence, the Ministry of Justice and/or other universities in the UK or abroad)**	NO

27	**Does your research involve any of the following individuals or higher-risk methodologies? Select all that apply or select 'not applicable' if no options apply to your research. Please note: the UEC reviewers may deem the nature of the research of certain high risk projects unsuitable to be undertaken by undergraduate students**	NOT APPLICABLE - none of the above high-risk options apply to my research.

28	**Does your research involve any of the following individuals or medium-risk methodologies? Select all that apply or select 'not applicable' if no options apply to your research.**	NOT APPLICABLE - none of the above medium-risk options apply to my research.

29	**Does your research involve any of the following individuals or lower-risk methodologies? Select all that apply or select 'not applicable' if no options apply to your research.**	NOT APPLICABLE - none of the above lower-risk options apply to my research.

30	**Declarations**	• I confirm that I have read the University's Code on Good Research Practice and ethics policy and all relevant professional and regulatory guidelines applicable to my research and that I will conduct my research in accordance with these. • I confirm that I have provided accurate and complete information regarding my research project • I understand that a false declaration or providing misleading information will be considered potential research misconduct resulting in a formal investigation and subsequent disciplinary proceedings liable for reporting to external bodies • I understand that if my answers to this form have indicated that I must submit an ethics and governance application, that I will NOT commence my research until a Favourable Ethical Opinion is issued and governance checks are cleared. If I do so, this will be considered research misconduct and result in a formal investigation and subsequent disciplinary proceedings liable for reporting to external bodies. • I understand that if I have selected 'YES' on any governance risk questions and/or have selected any options on the higher, medium or lower risk criteria then I MUST submit an ethics and governance application (EGA) for review before conducting any research. If I have NOT selected any governance risks or selected any of the higher, medium or lower ethical risk criteria, I understand I can proceed with my research without review and

		acknowledge that my SAGE answers and research project will be subject to audit and inspection by the RIGO team at a later date to check compliance.

31	**If I am conducting research as a student:**	• I confirm that I have discussed my responses to the questions on this form with my supervisor to ensure they are correct. • I confirm that if I am handling any information that can identify people, such as names, email addresses or audio/video recordings and images, I will adhere to the security requirements set out in the relevant Data Protection Policy

Appendix B

Python Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import jinja2
from pycaret.classification import *
from pycaret.utils import check_metric
SEED = 123

# DATA INPUT AND CLEANING
# 123 Columns and 150k rows means that the data
   needs to be simplified and the non-essentials be
   stripped.
df = pd.read_csv("2021
   _LoL_esports_match_data_from_OraclesElixir_20220606
   .csv")
df_copy = df.copy()
df.shape

# Filtering for completed data, team data and
```

```
    removing features that are non-team related or
    deemed useless for our model
df = df[df["datacompleteness"] == "complete"]
df = df[df["position"] == "team"]
df = df.drop_duplicates(subset = "gameid")
col_drop = ["url", "year", "split", "date", "game",
    "participantid", "playername", "playerid", "
    teamid", "champion",
              "ban1", "ban2", "ban3", "ban4", "ban5",
                 "kills", "deaths", "assists", "
                 teamkills", "teamdeaths",
              "doublekills", "triplekills", "
                 quadrakills", "pentakills", "
                 firstbloodkill", "firstbloodassist",
              "firstbloodvictim", "team␣kpm", "ckpm",
                 "dragons", "opp_dragons", "
                 elementaldrakes", "
                 opp_elementaldrakes",
              "infernals", "mountains", "clouds", "
                 oceans", "chemtechs", "hextechs", "
                 dragons␣(type␣unknown)",
              "elders", "opp_elders", "heralds", "
                 opp_heralds", "barons", "opp_barons",
                  "towers", "opp_towers",
              "turretplates", "opp_turretplates", "
                 inhibitors", "opp_inhibitors", "
                 damagetochampions", "dpm",
              "damageshare", "damagetakenperminute", "
                 damagemitigatedperminute", "
                 wardsplaced", "wpm", "wardskilled",
              "wcpm", "controlwardsbought", "
                 visionscore", "vspm", "totalgold", "
                 earnedgold", "earned␣gpm",
```

```
            "earnedgoldshare", "goldspent", "gspd",
               "total␣cs", "minionkills", "
               monsterkills", "monsterkillsownjungle
               ",
            "monsterkillsenemyjungle", "cspm", "
               opp_goldat10", "opp_goldat15", "
               goldat10", "goldat15", "csat10",
            "csat15", "opp_csat10", "opp_csat15", "
               xpat10", "xpat15", "opp_xpat10", "
               opp_xpat15", "league",
            "opp_assistsat10", "opp_killsat10", "
               opp_assistsat15", "opp_killsat15", "
               opp_deathsat10", "opp_deathsat15",
            "gameid", "datacompleteness", "teamname"
               , "side", "position",]
df.drop(col_drop, inplace=True, axis=1)
df = df.dropna()
df_copy = df.copy()

# Correlation of the features is then completed
# If we consider a Correlation of 0.5 or above to be
    a 'Strong' correlation
plt.figure(figsize=(18, 16))
mask = np.zeros_like(df.corr(), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
cmap = sns.diverging_palette(10, 250, as_cmap=True)
sns.heatmap(df.corr(method="kendall"), cmap=cmap,
   annot=True, fmt='.2f', vmin=-1, vmax=1,
            mask = mask, square=True, linewidths=.8,
                center = 0)

from statsmodels.stats.outliers_influence import
   variance_inflation_factor
```

```
X_variables = df[['playoffs', 'patch', 'gamelength',
    'firstblood', 'firstdragon',
        'firstherald', 'firstbaron', 'firsttower', '
           firstmidtower',
        'firsttothreetowers', 'golddiffat10', '
           xpdiffat10', 'csdiffat10',
        'killsat10', 'assistsat10', 'deathsat10', '
           golddiffat15', 'xpdiffat15',
        'csdiffat15', 'killsat15', 'assistsat15', '
           deathsat15']]

vif_data = pd.DataFrame()
vif_data["feature"] = X_variables.columns
vif_data["VIF"] = [variance_inflation_factor(
   X_variables.values, i) for i in range(len(
   X_variables.columns))]
vif_data

# Reworking and dropping any strongly correlated
   features
df["KillParat10"] = df["assistsat10"] + df["
   killsat10"]
df["KillParat15"] = df["assistsat15"] + df["
   killsat15"]
df = df.drop(df[df.patch == 10.25].index)
df["patch"] = (df["patch"] - 11)*100
round(df["patch"], 1)
df_copy = df.copy()
feature_drop = ["assistsat10", "killsat10", "
   assistsat15", "killsat15", "golddiffat10", "
   golddiffat15",
                "xpdiffat10", "xpdiffat15",]
df = df.drop(feature_drop, axis=1)
```

```
plt.figure(figsize=(18, 16))
mask = np.zeros_like(df.corr(), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
cmap = sns.diverging_palette(10, 250, as_cmap=True)
sns.heatmap(df.corr(method="kendall"), cmap=cmap,
   annot=True, fmt='.2f', vmin=-1, vmax=1,
             mask = mask, square=True, linewidths=.8,
                 center = 0)

# Descriptive statistics to find if there are any
   outliers
with pd.option_context('display.max_columns', 15):
    print(df.describe())

# Removing the outliers found in the high end of
   gamelength
df = df.loc[df["gamelength"] <= np.quantile(df["
   gamelength"], q=0.99)]
with pd.option_context("display.max_columns", 15):
    print(df.describe())

# Normalization
#from sklearn import preprocessing
#scaler = preprocessing.MinMaxScaler()
#names = df.columns
#d = scaler.fit_transform(df)
#scaled_df = pd.DataFrame(d, columns=names)
#df = scaled_df

from sklearn.feature_selection import RFE
from sklearn.tree import DecisionTreeClassifier
# define dataset
```

```
y = df.result
X = df.drop("result", axis=1)
# define RFE
rfe = RFE(estimator=DecisionTreeClassifier(),
   n_features_to_select=12)
# fit RFE
rfe.fit(X, y)
# summarize all features
for i in range(X.shape[1]):
    print('Column: %d, Selected %s, Rank: %.3f' % (i
       , rfe.support_[i], rfe.ranking_[i]))

# New correlation matrix with updated feature set
plt.figure(figsize=(18, 10))
mask = np.zeros_like(df.drop("result",axis=1).corr()
   , dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(df.drop("result",axis=1).corr(), cmap=
   cmap, annot=True, fmt='.2f', vmin=-1, vmax=1,
            mask = mask, square=True, linewidths=.8,
                center = 0)
plt.title('Correlation Matrix between Features')

X_variables = df[['playoffs', 'patch', 'gamelength',
    'firstblood', 'firstdragon',
       'firstherald', 'firstbaron', 'firsttower', '
          firstmidtower',
       'firsttothreetowers', 'csdiffat10',
       'KillParat10', 'deathsat10',
       'csdiffat15', 'KillParat15', 'deathsat15']]

vif_data = pd.DataFrame()
vif_data["feature"] = X_variables.columns
```

```
vif_data["VIF"] = [variance_inflation_factor(
   X_variables.values, i) for i in range(len(
   X_variables.columns))]
vif_data

df = pd.DataFrame(data=df)

#  Split the data into separate 10 and 15 minute
   datasets
feature_drop2 = ["firstbaron", "firsttothreetowers"]
at10 = []
for column in list(df.columns):
    if '10' in column:
        at10.append(column)
at15 = []
for column in list(df.columns):
    if '15' in column:
        at15.append(column)
df10 = df.drop(at15, axis = 1)
df15 = df.drop(at10, axis = 1)
df20 = df.drop(at10, axis = 1)
df10.drop(feature_drop2, axis = 1, inplace=True)
df15.drop(feature_drop2, axis = 1, inplace=True)

# Correlation matrix of the 10 minute dataset
plt.figure(figsize=(12, 10))
mask = np.zeros_like(df10.drop("result",axis=1).corr
   (), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(df10.drop("result",axis=1).corr(), cmap=
   cmap, annot=True, fmt='.2f', vmin=-1, vmax=1,
            mask = mask, square=True, linewidths=.8,
                center = 0)
```

```
plt.title('Correlation␣Matrix␣between␣Features␣(10␣
   minute␣dataset)')

# Correlation matrix of the 15 minute dataset
plt.figure(figsize=(12, 10))
mask = np.zeros_like(df15.drop("result",axis=1).corr
   (), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(df15.drop("result",axis=1).corr(), cmap=
   cmap, annot=True, fmt='.2f', vmin=-1, vmax=1,
            mask = mask, square=True, linewidths=.8,
                center = 0)
plt.title('Correlation␣Matrix␣between␣Features␣(15␣
   minute␣dataset)')

# Correlation matrix of the 20 minute dataset
plt.figure(figsize=(12, 10))
mask = np.zeros_like(df20.drop("result",axis=1).corr
   (), dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(df20.drop("result",axis=1).corr(), cmap=
   cmap, annot=True, fmt='.2f', vmin=-1, vmax=1,
            mask = mask, square=True, linewidths=.8,
                center = 0)
plt.title('Correlation␣Matrix␣between␣Features␣(20␣
   minute␣dataset)')

# Now we check to see if the results are balanced
# Slightly skewed but is fine to use.
print("Blue␣side␣win␣rate:␣{0:.1f}%".format(df15["
   result"].sum() / df15["result"].shape[0]*100))
sns.set_style('whitegrid')
fig = sns.countplot(x= df15.result , data=df15,
```

```
   palette= "Blues")
fig.set(xlabel= "Result", ylabel= "Count␣of␣Matches"
   , title = "Balance␣of␣the␣dataset")
fig.set_xticklabels(["Loss", "Win"]),

df15.hist(alpha = 0.9, figsize=(12,10), bins=20);

# Baseline Model for 10 minute data
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

y = df10.result
X = df10.drop("result", axis=1)
train_X, val_X, train_y, val_y = train_test_split(X,
    y, random_state=SEED)
rf_model = RandomForestClassifier(random_state=SEED)
rf_model.fit(train_X, train_y)
pred = rf_model.predict(val_X)
score = accuracy_score(val_y, pred)
print('Accuracy:␣%.2f%%' %(score*100))

# Baseline Model for 15 minute data
y = df15.result
X = df15.drop("result", axis=1)
train_X, val_X, train_y, val_y = train_test_split(X,
    y, random_state=SEED)
rf_model = RandomForestClassifier(random_state=SEED)
rf_model.fit(train_X, train_y)
pred = rf_model.predict(val_X)
score = accuracy_score(val_y, pred)
print('Accuracy:␣%.2f%%' %(score*100))
```

```
# Baseline Model for 20 minute data
y = df20.result
X = df20.drop("result", axis=1)
train_X, val_X, train_y, val_y = train_test_split(X,
    y, random_state=SEED)
rf_model = RandomForestClassifier(random_state=SEED)
rf_model.fit(train_X, train_y)
pred = rf_model.predict(val_X)
score = accuracy_score(val_y, pred)
print('Accuracy: %.2f%%' %(score*100))

# Building a model based on data from the 10 minute
   mark
models=setup(data=df10,
             target="result",
             silent=True,
             session_id=33,
            normalize=True,
            normalize_method="minmax")
# Comparing the different model's final results
model_results=compare_models()
model_results
lr_model=create_model('lr')
lr_tunedmodel=tune_model(lr_model)
plot_model(lr_tunedmodel, plot = 'auc')
plot_model(lr_tunedmodel, plot='feature')
plot_model(lr_tunedmodel, plot = 'confusion_matrix')
predict_model(lr_tunedmodel)
final_lr = finalize_model(lr_tunedmodel)
print(final_lr)
predict_model(final_lr)

df22 = pd.read_csv("2022
```

```
   _LoL_esports_match_data_from_OraclesElixir_20220606
   .csv")
df22.head

# Cleaning data on 2022 data
def data_clean(df):
    df = df[df["datacompleteness"] == "complete"]
    df = df[df["position"] == "team"]
    df = df.drop_duplicates(subset = "gameid")
    col_drop = ["url", "year", "split", "date", "
       game", "participantid", "playername", "
       playerid", "teamid", "champion",
            "ban1", "ban2", "ban3", "ban4", "ban5",
               "kills", "deaths", "assists", "
               teamkills", "teamdeaths",
            "doublekills", "triplekills", "
               quadrakills", "pentakills", "
               firstbloodkill", "firstbloodassist",
            "firstbloodvictim", "team␣kpm", "ckpm",
               "dragons", "opp_dragons", "
               elementaldrakes", "
               opp_elementaldrakes",
            "infernals", "mountains", "clouds", "
               oceans", "chemtechs", "hextechs", "
               dragons␣(type␣unknown)",
            "elders", "opp_elders", "heralds", "
               opp_heralds", "barons", "opp_barons",
                "towers", "opp_towers",
            "turretplates", "opp_turretplates", "
               inhibitors", "opp_inhibitors", "
               damagetochampions", "dpm",
            "damageshare", "damagetakenperminute", "
               damagemitigatedperminute", "
```

```
            wardsplaced", "wpm", "wardskilled",
        "wcpm", "controlwardsbought", "
            visionscore", "vspm", "totalgold", "
            earnedgold", "earned␣gpm",
        "earnedgoldshare", "goldspent", "gspd",
            "total␣cs", "minionkills", "
            monsterkills", "monsterkillsownjungle
            ",
        "monsterkillsenemyjungle", "cspm", "
            opp_goldat10", "opp_goldat15", "
            goldat10", "goldat15", "csat10",
        "csat15", "opp_csat10", "opp_csat15", "
            xpat10", "xpat15", "opp_xpat10", "
            opp_xpat15", "league",
        "opp_assistsat10", "opp_killsat10", "
            opp_assistsat15", "opp_killsat15", "
            opp_deathsat10", "opp_deathsat15",
        "gameid", "datacompleteness", "teamname"
            , "side", "position",]
df.drop(col_drop, inplace=True, axis=1)
df = df.dropna()
df_NO = df.copy()
df["KillParat10"] = df["assistsat10"] + df["
    killsat10"]
df["KillParat15"] = df["assistsat15"] + df["
    killsat15"]
feature_drop = ["assistsat10", "killsat10", "
    assistsat15", "killsat15", "golddiffat10", "
    golddiffat15", "xpdiffat10", "xpdiffat15"]
df = df.drop(feature_drop, axis=1)
df["patch"] = (df["patch"] - 12)*100
round(df["patch"], 1)
return df
```

```
df22 = data_clean(df22)
df22.describe

#  Split the data into separate 10 and 15 minute
   datasets
def data_split(df):
    feature_drop2 = ["firstbaron", "
       firsttothreetowers"]
    at10 = []
    for column in list(df.columns):
        if '10' in column:
            at10.append(column)
    at15 = []
    for column in list(df.columns):
        if '15' in column:
            at15.append(column)
    df10 = df.drop(at15, axis = 1)
    df15 = df.drop(at10, axis = 1)
    df20 = df.drop(at10, axis = 1)
    df10.drop(feature_drop2, axis = 1, inplace=True)
    df15.drop(feature_drop2, axis = 1, inplace=True)
    return df10, df15, df20

df22at10, df22at15, df22at20 = data_split(df22)
df22at10.drop("result", axis = 1)
unseen_predictions = predict_model(final_lr, data=
   df22at10)
check_metric(unseen_predictions["result"],
   unseen_predictions["Label"], metric = "AUC")

models=setup(data=df15,
             target="result",
```

```
              silent=True,
              session_id=33,
             normalize=True,
             normalize_method="minmax")
model_results=compare_models()
model_results
lr_model2=create_model('lr')
lr_tunedmodel2=tune_model(lr_model2)
plot_model(lr_tunedmodel2, plot = 'auc')
plot_model(lr_tunedmodel2, plot='feature')
plot_model(lr_tunedmodel2, plot = 'confusion_matrix'
   )
predict_model(lr_tunedmodel2)
final_lr2 = finalize_model(lr_tunedmodel2)
predict_model(final_lr2)
df22at15.drop("result", axis = 1)
unseen_predictions = predict_model(final_lr2, data=
   df22at15)
check_metric(unseen_predictions["result"],
   unseen_predictions["Label"], metric = "AUC")

models=setup(data=df20,
              target="result",
              silent=True,
              session_id=33,
             normalize=True,
             normalize_method="minmax")
model_results=compare_models()
model_results
gbc_model=create_model('gbc')
gbc_tunedmodel=tune_model(gbc_model)
plot_model(gbc_tunedmodel, plot = 'auc')
plot_model(gbc_tunedmodel, plot='feature')
```

```
plot_model(gbc_tunedmodel, plot = 'confusion_matrix'
   )
predict_model(gbc_tunedmodel)
final_gbc = finalize_model(gbc_tunedmodel)
predict_model(final_gbc)
df22at20.drop("result", axis = 1)
unseen_predictions = predict_model(final_gbc, data=
   df22at20)
check_metric(unseen_predictions["result"],
   unseen_predictions["Label"], metric = "Recall")
```

Appendix C

Data Description

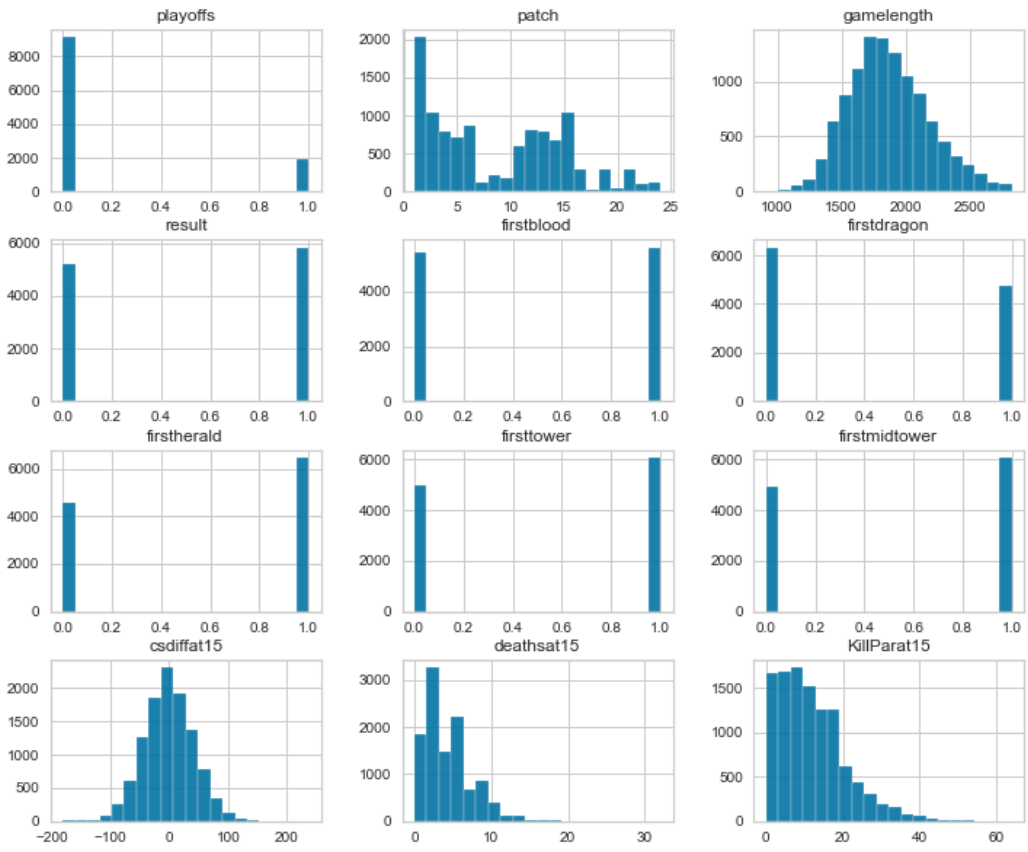


Figure C.1: Graphs showing the distribution of data for the dataset

Appendix D

Feature Correlations

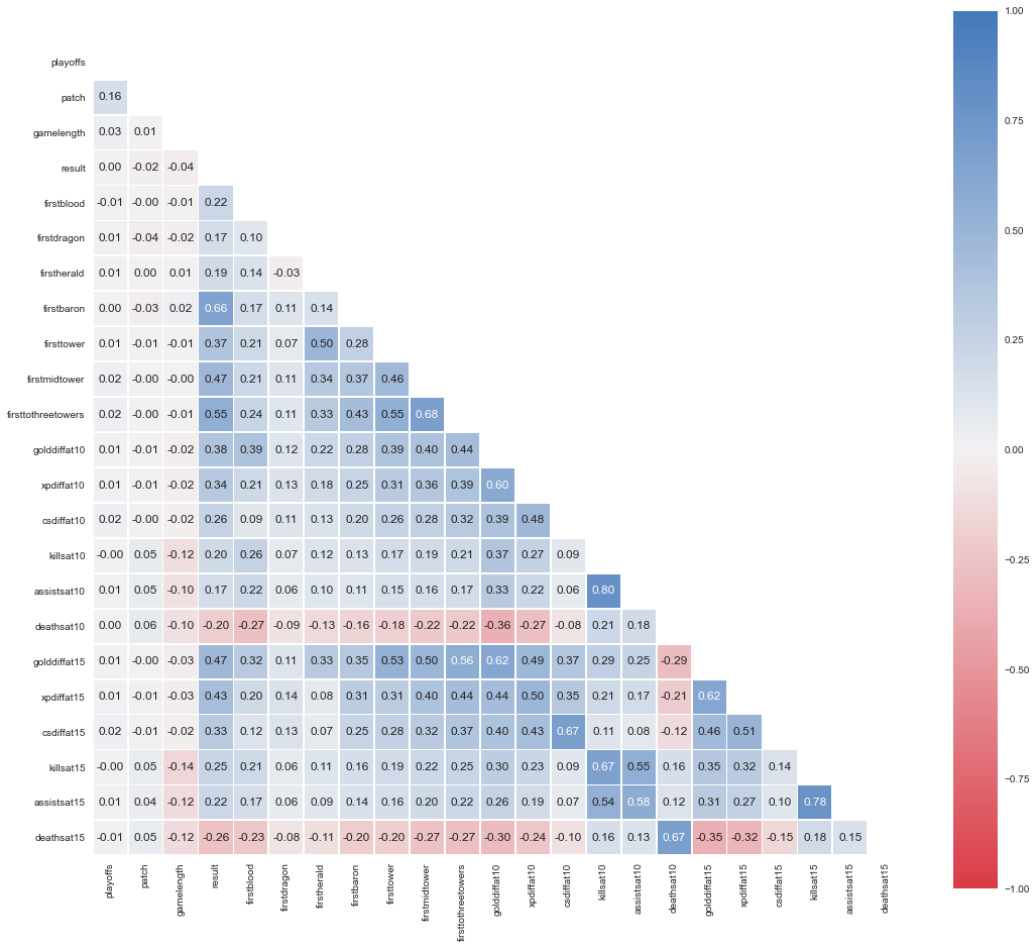


Figure D.1: A matrix of correlations between features in the dataset

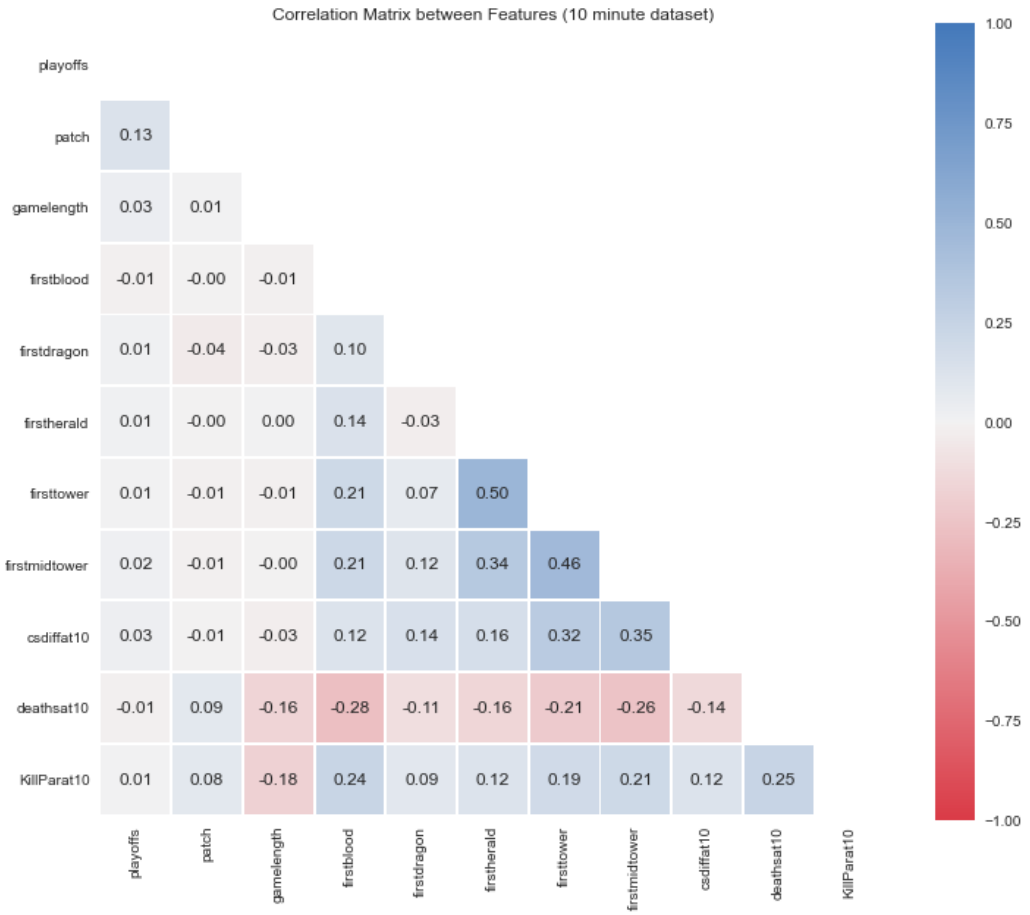


Figure D.2: A matrix of correlations from the 10-minute dataset

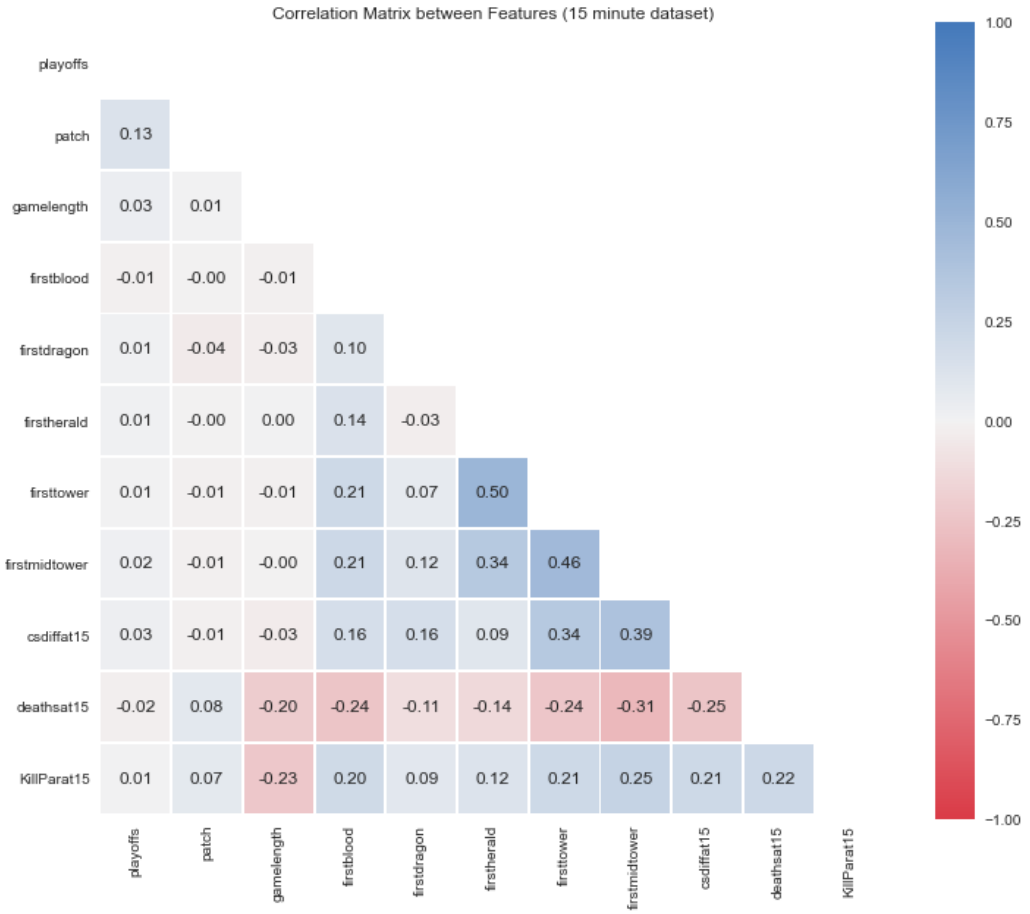


Figure D.3: A matrix of correlations from the 15-minute dataset

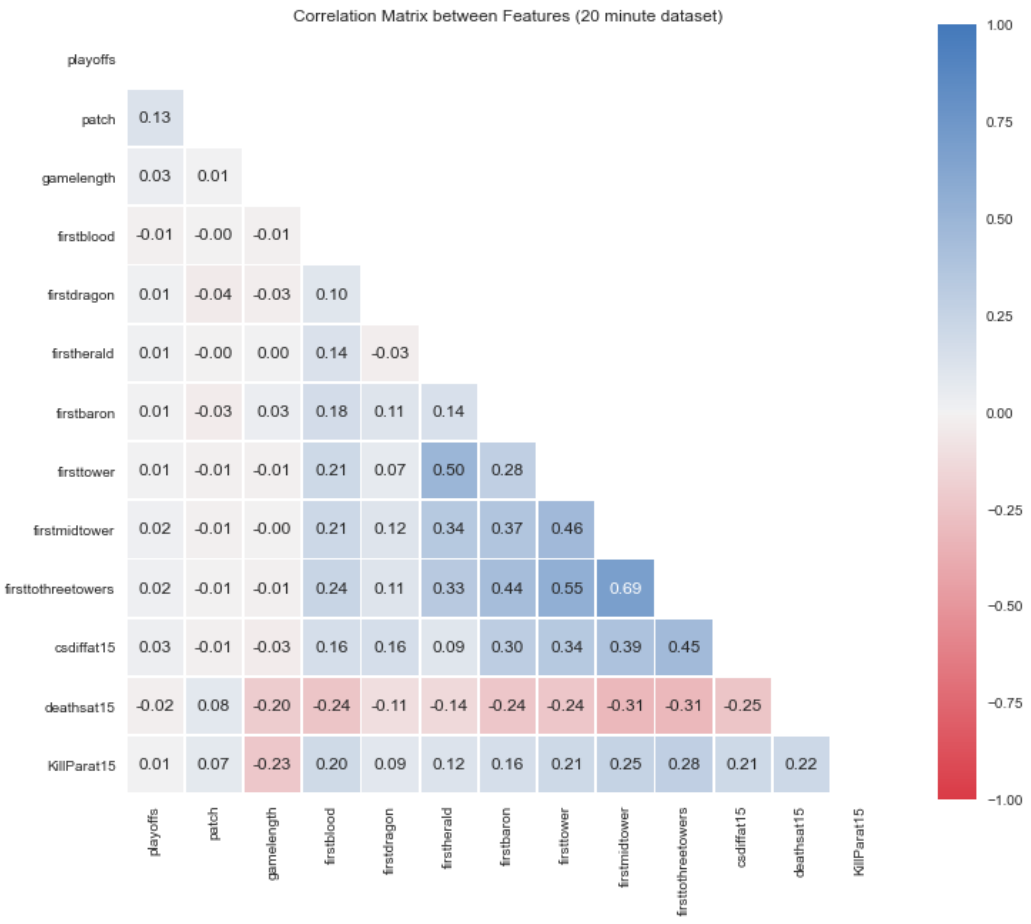


Figure D.4: A matrix of correlations from the 20-minute dataset