# Studying the Effects of Team Composition in Role-based Competitive Video Games

Amy Cadman
Supervisor: Gaetano Romano

MSci Hons Mathematics with Statistics
Lancaster University
May 2024

**Abstract**

This dissertation explores associations between champions and the influence of team composition on match outcome in League of Legends, through comprehensive data analysis and machine learning techniques. Utilising Riot Games' Application Programming Interface (API), a dataset comprising of 96,000 matches across various patches and player ranks was collected. Exploratory analysis revealed insights into champion popularity and win rates, showing Yuumi had the lowest win rate of 37.955% within the Iron rank. Association rule learning, using an Apriori algorithm, uncovered strategic associations among champions. Notably, in lower ranks, the presence of Caitlyn often implied the selection of Lux, a pattern not observed in higher ranks. Furthermore, association rule learning identified champions perceived as easier to play, such as Lux, compared to more complex champions such as Kai'Sa and Vel'Koz. Clustering analysis, using both k-means and hierarchical methods, detected interchangeable usage of Fighter and Slayer classes, with minimal impact on match outcomes. Logistic regression models underscored significant interactions between champion performance and player rank, emphasising the importance of skill-based matchmaking. Overall, this research enhances understanding of team compositions and match dynamics in League of Legends, with implications for strategic decision-making and game balance. It underscores the value of statistical methods in the continuous evolution of the eSports industry.

# Contents

# 1 Introduction

The world of competitive video gaming, also known as eSports, has seen a large increase in popularity over the past decade with the global eSports market being valued at $1.98 billion (USD) in 2023 as given by [Shewale, 2024]. With this surge in interest, there has been an increasing need to understand the factors that contribute to success in these games. One such factor that requires further exploration is the composition of a team in role-based video games.

Role-based video games are a sub-genre of competitive games, where each player assumes a specific role that contributes to the overall strategy of the team. These roles can vary widely, from damage dealers and defenders to supporters. The composition of these roles within a team is believed to significantly impact the team's performance and outcomes in competitive matches. I hypothesise that certain compositions may be more effective than others, and understanding these dynamics could provide valuable insights for players, coaches, and game developers.

Some examples of role based competitive video games are League of Legends, DotA 2, Valorant, Overwatch2, and Rainbow Six Siege. This work will concentrate on League of Legends due to the game's widespread popularity, as well as my personal familiarity with it, having devoted over 3,400 hours to gameplay since 2018. This extensive experience provides me with a unique perspective and in-depth understanding of the game's mechanics and strategies. Furthermore, the game has a larger number of available characters in comparison to the others previously mentioned. For example, Valorant has 22 'agents' and Overwatch2 has 40 'heroes'. Since League of Legends has 165 'champions', the game possesses a greater complexity which makes for a more interesting statistical analysis. In addition, there is a large amount of match data that is readily available online through both existing datasets and from the game's web application programming interface (API).

League of Legends (LoL), developed by Riot Games, has attracted significant attention in both the gaming community and academic research due to its strategic depth and team-based gameplay. The game has steadily grown in popularity and now has over 152 million monthly players as can be seen in Figure 1.1. In addition, it is a globally recognised eSports game with 3,781 teams registered. The [eSports Charts, 2024] states, from November 2016 to today, the most profitable League of Legends team is T1 from South Korea who have won $8,816,493 (USD), demonstrating the significance of the game. Furthermore, even those who do not play at a competitive level have an established interest in the game. This is evidenced by the fact that the League of Legends 2023 World Championship peaked at 6.4 million viewers according to [Statista, 2024]. For a more in-depth explanation of the game's mechanics and objectives, refer to Section 2.

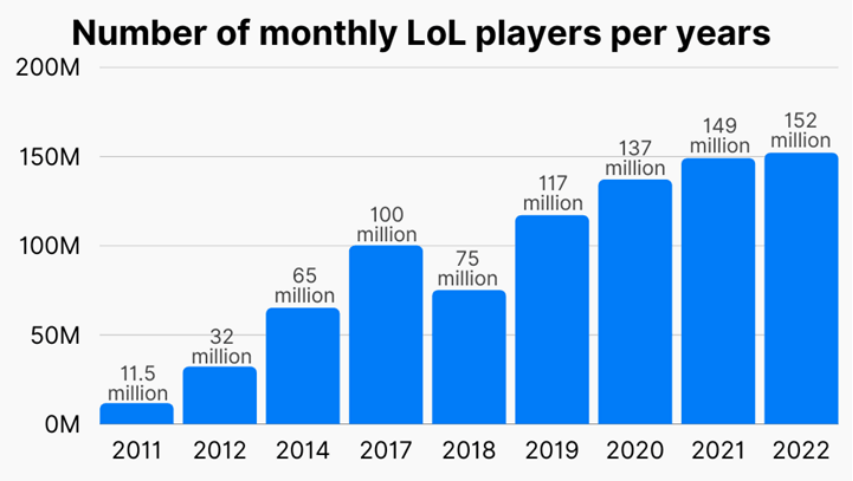


Figure 1.1: Number of monthly players from [lolvvv, 2023].

## 1.1 Aims of Dissertation

This dissertation aims to identify associations among champions to uncover potentially powerful team compositions and explore their impact on the game's outcome. I decided to concentrate on the ranked game mode, as it seemed to yield the most representative results in terms of strategic decisions. Although the game is played globally, the data was collected entirely from Europe for simplicity, assuming that the play style would not significantly differ across regions. It is crucial to acknowledge that these decisions introduced some limitations, which will be addressed in Section 4.

There are several online sources that already give qualitative advice surrounding selecting the best team composition. Although the majority of these are based upon expert opinion, they do not utilise statistical methods. Furthermore, the advice offered in these resources tends to be rather general and lacks specificity, particularly when it comes to individual champions. The limited amount of existing websites that provide information for individually selected champions suggests a potential area for further research and tool development in the field. For example, the online blog by [Burton, 2023] was written four years ago when there were only 145 champions. The blog emphasises that experience and knowledge play a crucial role in selecting a good team composition and improving at League of Legends. This suggests that understanding team compositions and their impacts on gameplay is an essential aspect of mastering the game.

Furthermore, numerous online resources are available to aid players in making strategic decisions within the game. For instance, the website [OPGG, 2024] provides recommendations for champions that are both weak against and strong against a specified champion. It also allows users to look up specific players, enabling them to analyse their personal statistics, identify their most frequently played champions, and determine their win rates with those champions. This feature facilitates an individual analysis of a player's preferred play style.

Similarly, [Mobalytics, 2024] provides information on champions that are weak against and strong against a particular champion. However, it goes a step further as you can select a champion that you wish to play and it will rank champions based on how similar their play styles are in case the chosen champion is unavailable. This dissertation will later investigate the concept of similar play styles using the classes explained in Section 2.2. The website also provides an overall difficulty rating for each champion which it defines as "how hard they are to immediately pick up and learn". However, there is no definitive explanation on how this is measured so it may be using data analysis, expert opinion or a combination. The difficulty ratings can be particularly useful for less experienced players who are still learning the game and its controls. Despite the wealth of information these websites provide, they do not offer team composition recommendations based on multiple champions. Instead, they focus on providing information for individually selected champions.

The following Section 3 will review the literature surrounding various machine learning methodologies, with a particular focus on association rule learning, clustering, and logistic regression. Each subsection will provide an overview of the method, possible algorithms, its applications and overarching limitations. Following this, Section 4 will present the methodology for data collection. Section 5 will then present the initial exploratory analysis, aimed at validating assumptions about the gathered data. Subsequent sections, namely 6, 7, and 8, will explain the data analysis conducted utilising association rule learning, clustering, and logistic regression, respectively. Each of these sections will be structured by starting with the relevant methodology before presenting the results. Section 9 will then discuss the findings and explore the implications of the results for the future of competitive gaming. Lastly, Section 10 provides a comprehensive summary of the key information and results, concluding the dissertation.

All analyses were conducted in the programming language R [R Core Team, 2023] using version 4.3.2. Any results presented in this work shall be rounded to three decimal places.

# 2 Contextual Background: League of Legends

## 2.1 Objectives of the Game

League of Legends is a popular multiplayer online battle arena game, commonly known as MOBA. The game mode focused on was the ranked solo/duo queue which involves ten players who are divided into two teams of five. The players compete on a map called "Summoner's Rift", as shown in Figure 2.1. The teams are distinguished by colours: the blue team, whose base is located in the bottom left corner, and the red team, whose base is in the top right corner.

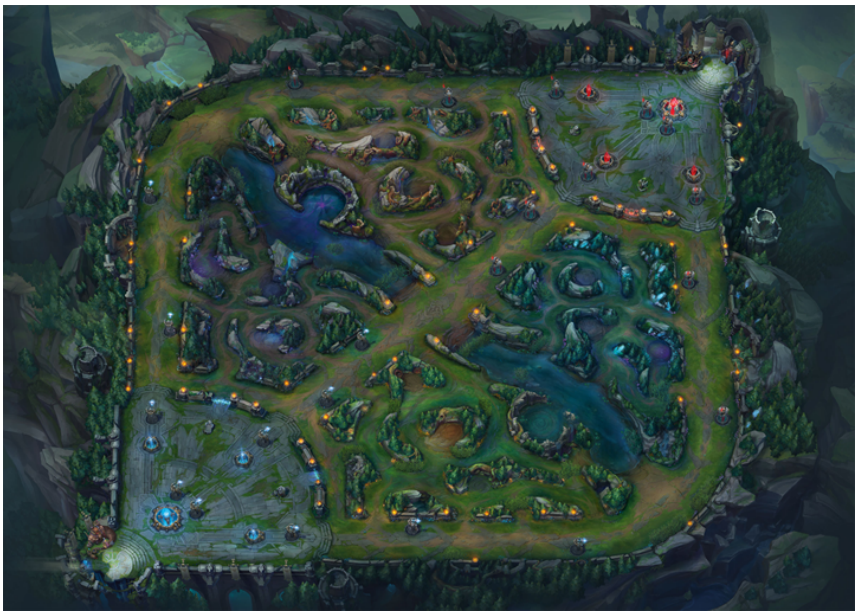

Figure 2.1: Summoner's Rift map from [League of Legends Wiki, 2024b].

Each player controls a unique character, referred to as a champion, which possesses unique abilities and skills. The game is centred around five positions: Top, Jungle, Mid, Bot, and Support. The Top player operates in the top lane, the Mid player is in the middle lane, and the Bot and Support players are in the bottom lane. The Jungle player roams the area between the lanes, providing assistance where needed. Despite these positions, players are not strictly confined to their lanes and can move between them to support their teammates.

The ultimate goal of the game is to traverse the map and destroy the opposing team's base. This objective can typically be accomplished within a time frame of 30 to 45 minutes. However, each base is fortified by a series of turrets and defended by continuously spawning minions, posing a challenge to the attacking team.

The results have certain constraints due to the exclusion of some game mechanics, a measure taken to limit complexity. I will now outline these potential influencing factors. The game incorporates an economic system that enables players to purchase items to enhance their champion's attacks and abilities. This mechanic significantly complicates the analysis due to the availability of over 150 different items, with a maximum of six that can be purchased per player. Additionally, the game features objectives such as elemental dragons, Rift Herald and Baron Nasher that, when defeated, provide your team with additional attributes. Moreover, the level of communication within a team can significantly impact the outcome. For instance, strategies such as coordinating attacks, sharing information about enemy locations, and devising tactical plans can all offer a substantial advantage. However, in higher ranks there is often more consistent communication patterns, which might mitigate this effect compared to lower ranks, where communication effectiveness can vary significantly among teams. This motivates conducting an analysis stratified by ranks and subsequently focusing on those high ranks.

At the time of data collection, players could choose from 165 unique champions. When selecting a team of five, this results in approximately 958 million potential team compositions. Therefore, strategies to reduce the number of choices are highly beneficial. During champion selection, each player has the option to ban a champion they prefer not to compete against, slightly reducing the options. Hence, the most strategic choice may not always be available if it has been banned. Teams alternate in selecting champions, allowing for informed decisions based on the opposing team's choices. Therefore, strategic choices may include choosing champions that synergise well with the choices already made by your team or counter the opposing team's choices. To streamline the selection process, champions can be categorised in various ways which is explained in detail in Section 2.2.

It is important to note that when new players create an account in League of Legends, they initially have access to a limited number of champions through the free rotation system. Additional champions can be unlocked using Blue Essence, an in-game currency earned through levelling up or participating in matches. This progression system allows players to gradually expand their champion pool and explore diverse gameplay options as they advance in the game.

In the ranked version of the game, players can ascend through ten different ranks based on their match victories, serving as a measure of skill. These ranks, in ascending order, are: Iron, Bronze, Silver, Gold, Platinum, Emerald, Diamond, Master, Grandmaster, and Challenger. Additionally, these ranks undergo periodic resets, often at the start of a new competitive season, to maintain the competitiveness and integrity of the ranking system.

## 2.2 Categorising the Champions

Although some champions are capable of playing multiple positions, they are still able to be categorised in this way. Another way to categorise champions is by the type of damage that they inflict. Some champions possess high Attack Damage (AD), which can be effectively countered by items that boost armour or health. Conversely, champions with high Ability Power (AP) can be countered by items that provide high magic resistance. It is important to note that there are a number of hybrid champions, which combine both AD and AP attributes.

Alternatively, champions can be classified into seven distinct classes. First, there are 21 choices for the Controller class, which are champions that focus on crowd control and are generally weak when alone. Second, the Fighter class has 31 choices and includes short-range champions that excel at both dealing and surviving damage. Third, the Mage class, with 30 choices, comprises champions who typically possess great reach and ability-based area of effect damage. Fourth, the Marksman class has 21 choices and is made up of ranged champions who focus on basic attacks. Fifth, the Slayer class, with 28 choices, includes highly mobile champions that focus on burst damage. Sixth, the Specialist class has 14 choices and consists of champions that do not fit into any of the other categories due to their unique abilities. Finally, the Tank class has 20 choices and includes tough melee champions that can survive a large amount of damage. The data analysis will focus on both individual champions and classes. This is motivated by the lack of academic research on statistical analysis with relation to League of Legends, as highlighted in Section 2.3. The champions can also be split into the following sub classes: Catchers, Enchanters, Juggernauts, Divers, Artillery, Burst, Battlemages, Assassins, Skirmishers, Vanguards, Wardens, Marksman, and Specialists. However, this was not studied in this work.

A table containing all the champions in alphabetical order with their ID numbers, class, sub-class and most popular position can be seen in Appendix A.

## 2.3 Statistics in League of Legends

This section will look into a specific source that has analysed match data to draw conclusions about effective team compositions and its impact on game outcomes. The composition of a team, including the specific champions selected and the positions they play, is a critical factor in determining the outcome of a match. In addition, the source highlights the importance of strategic diversity, champion synergy, and counter-picking. Afterwards, a comprehensive overview of what is currently known about the effects of team composition in League of Legends will be presented to identify the gaps that this dissertation aims to address.

The website [LoLTheory, 2024] provides insights into entire team compositions in League of Legends. Users can input specific champions and subsequently receive recommendations and win rates. This feature can be customised for different ranks, although it groups the top four ranks together. The website also offers patch comparisons, which can be particularly useful for tracking how changes in the game impact team compositions and their success rates. Notably, the website is updated frequently, specifically every three hours when a new patch is released, and then once a day thereafter. The data used by the website is sourced from Riot's API, similar to the approach used in this dissertation. The frequent updates and use of official game data increase the reliability of this source of information for studying the effects of team composition in League of Legends. Although, there is no explanation of what methods are used on the data collected.

Overall, there is a lack of academic research done in this area with the majority of the resources available providing no insight into their specific statistical methods. A pivotal factor to consider is the dynamic nature of the game, characterised by frequent updates. Consequently, these resources quickly become outdated, diminishing their relevance over time. This dissertation, therefore, seeks to delve deeper into the concept of team composition, aiming to provide a more detailed definition.

# 3 Literature Review

## 3.1 Machine Learning Methods

This section provides a comprehensive overview of the existing research on machine learning methods and their applications. It helps to establish the foundation for the current study and identify which methods may be applicable to my research.

Machine learning is a subset of artificial intelligence that involves the use of algorithms and statistical models to enable computers to perform tasks without explicit programming. There are two main types of machine learning: supervised and unsupervised. In supervised machine learning, algorithms are trained on labelled datasets that include tags describing each piece of data. As a result, these models are often used for prediction and classification purposes. Whereas, unsupervised machine learning uses unlabelled datasets to train algorithms. As discussed by [Coursera Staff, 2024], unsupervised learning can help to identify patterns within large, unlabelled datasets quickly and efficiently. Both types have significant applications in various fields of mathematics, including optimisation, statistics, and data analysis.

Several methods have been applied in mathematical research in both supervised and unsupervised contexts. These include decision trees [Rokach and Maimon, 2014], regression models [Freedman, 2009], neural networks [Bishop, 2006], clustering [Cattell, 1943] and association rule learning [Agrawal et al., 1993]. Each method has unique algorithms, applications, advantages, and limitations. These will be covered in detail for association rule learning, clustering and logistic regression models.

Machine learning has found various applications across the sports industry. For example, in sports science, [Kipp et al., 2018] employed a Neural Network (NN) to predict hip, knee, and ankle Net Joint Moments during a weightlifting exercise. This has significant implications, as it enables the refinement of training programs to enhance performance and mitigate injury risks. Moreover, machine learning methodologies have been utilised for predictive analytics in sports, such as the development of logistic regression models by [Prasetio and Harlili, 2016] to forecast football match outcomes in the Barclays' Premier League season. Over time machine learning techniques have also been extended into the eSports industry. The case study by [Hodge et al., 2021] highlights the advantages of employing machine learning for eSports analysis. One advantage is that using machine learning algorithms can improve win prediction abilities in comparison to more traditional methods.

Despite the extensive research into machine learning, there are still gaps in the literature and limitations of the methods. For instance, the large computational demands of some of the methods.

Overall, considering the objectives of this dissertation and the binary nature of the data the following methods have been utilised: association rule learning (Section 6), clustering (Section 7) and logistic regression models (Section 8). Each section will cover the methodology used and present the relevant results. Additionally, a discussion can be found in Section 9, which will consolidate the key findings, discuss their interconnections, and explore their implications for the game. For further exploration of the literature underpinning these methods, refer to the subsequent sections.

### 3.1.1 Association Rule Learning

Association rule learning is a rule-based unsupervised machine learning method for discovering interesting relations or associations between variables in large databases. As discussed by [Piatetsky-Shapiro, 1991] the method is intended to identify strong rules discovered in databases using some measures of relevance, based on the concepts of support and confidence.

There are many algorithms that can be used to implement association rule learning. The main methodologies are summarised in Figure 3.1. As introduced by [Agrawal and Srikant, 1994], there are three significant algorithms: Apriori, AprioriTID, and Apriori hybrid. Apriori, a classic algorithm, iteratively generates candidate itemsets and prunes those below the minimum support threshold to mine frequent itemsets efficiently. AprioriTID, an extension of Apriori, directly mines frequent itemsets from transaction identifier lists (TID-lists) without candidate generation, optimizing performance. Apriori hybrid combines the strengths of both Apriori and AprioriTID algorithms. Additionally, FP-growth, as detailed in [Han et al., 2000], offers an alternative approach for mining frequent itemsets without candidate generation. It employs a compact data structure called the FP-Tree to achieve efficient mining. To determine the most suitable algorithm among these options, one must carefully consider the trade-off between accuracy and computational demand.

COMPARISON OF ASSOCIATION RULE MINING ALGORITHMS

| **Characteristics** | **AIS** | **SETM** | **Apriori** | **Aprioritid** | **Apriori hybrid** | **FP-growth** |
|---|---|---|---|---|---|---|
| Data support | Less | Less | Limited | Often suppose large | Very Large | Very large |
| Speed in initial phase | Slow | Slow | High | Slow | High | High |
| Speed in later phase | Slow | Slow | Slow | High | High | High |
| Accuracy | Very less | Less | Less | More accurate than Apriori | More accurate than Aprioritid | More accurate |

Figure 3.1: Comparison of association rule learning algorithms from [Kumbhare and Chobe, 2014].

Originally the method was proposed in [Agrawal et al., 1993] for discovering regularities between products in large-scale transaction data recorded by supermarkets. For example, a rule could indicate that if a customer buys both onions and potatoes, then they are likely to also buy hamburger meat. Nowadays, the method is used to improve decision making in the applications such as disease diagnosis [Nahar et al., 2013], building intelligent transportation systems [Yang et al., 2012], and fraud detection [Metwally et al., 2005].

This paper [García et al., 2007] discusses the following limitations of the method: discovering an excessive number of rules, many of which may not be relevant. This issue can be addressed to some extent through rule filtering. Additionally, some rules are difficult to understand and lack comprehensibility, particularly in large rules with multiple items.

### 3.1.2 Clustering

Clustering, also referred to as cluster analysis, is a form of unsupervised learning. The goal is to group a collection of objects so that objects within the same group, referred to as a cluster, exhibit more similarity to each other than to objects in other groups. As discussed by [Estivill-Castro, 2002], the implementation of clustering encompasses a range of algorithms, largely owing to the inherent ambiguity surrounding the precise definition of a 'cluster'.

Each of the following algorithms uses the idea of distance to measure the similarity between two points, meaning that points with a small distance are more similar. Typically distance is calculated using metrics like Euclidean or Manhattan distance, which are formally defined in Section 7.1. Hierarchical clustering as presented in [Nielsen, 2016] operates by forming a hierarchy of clusters, merging or splitting them based on proximity either using Euclidean or Manhattan distance. K-means, as explained by [MacQueen, 1967], partitions data into $K$ clusters, where each point is assigned to the cluster with the closest mean. It uses primarily Euclidean distance to measure dissimilarity and iteratively minimises within-cluster variance. DBSCAN as proposed in [Ester et al., 1996], on the other hand, identifies dense regions by grouping nearby points within a specified radius (epsilon). Distance calculation, often employing Euclidean distance, helps discern the density around each point, with regions meeting a minimum point threshold forming clusters and isolated points treated as noise. Since these calculate distance in distinct ways, the algorithms can potentially result in diverse cluster formations. This diversity underscores the complexity and versatility of clustering algorithms in data analysis.

Cluster analysis has a variety of applications. For example [Hodge et al., 2021] used cluster analysis to find patterns in the behaviours of players in the games Tera and Battlefield: Bad Company 2. This was then used to develop behavioural based profiles of how people play these games. Therefore, clustering provides a way to reduce the dimensionality of a dataset in order to find the most important features, and locate patterns which are expressed in terms of user behaviour as a function of these features, which can be acted upon to test and refine a game's design.

While clustering is a powerful analytical tool, it is not without its drawbacks. A significant challenge is the manual specification of the cluster count, which can greatly impact the outcome and is often unknown beforehand. Moreover, as highlighted in [Peña et al., 1999], clustering algorithms like the k-means algorithm, being iterative techniques, are sensitive to initial starting conditions. Outliers pose another challenge as they can drastically distort clustering results by either

pulling the centroids or creating their own cluster. The issue of high-dimensional data introduces further complications for clustering, such as the convergence of point distances, visualisation difficulties, and the curse of dimensionality. Certain methods necessitate numerous arbitrary decisions, like the selection of a linkage method in hierarchical clustering, which can substantially alter the results.

Despite these challenges, clustering continues to be a beneficial tool when utilised effectively. This involves a thorough understanding the assumptions of the clustering algorithm, suitable data preprocessing, and result validation using various techniques and metrics.

### 3.1.3 Logistic Regression

Logistic regression as explained by [James et al., 2013], a type of generalised linear model (GLM), is utilised to model the probability of a binary response variable, $Y$, following a Bernoulli distribution. Consequently, the dependent variable exhibits only two possible outcomes, commonly represented as '0' and '1', 'Yes' and 'No', or 'True' and 'False', depending on the context. Employing a logit link function, the model estimates the likelihood of these outcomes based on one or more independent variables, also known as predictor variables. These predictors may be numerical or categorical, granting logistic regression the flexibility to accommodate various data types. One of its primary strengths lies in its ability to provide probabilities and facilitate the classification of new data points, leveraging both continuous and discrete measurements.

Logistic regression finds applications in diverse binary classification problems. For example, it has been employed in predictive models during eSports matches, such as DotA 2 by [Yang et al., 2016]. In this study, integrating in-game metrics (e.g., gold, experience, and deaths per minute) with pre-existing features (e.g., player information and hero selection) achieved an impressive accuracy of up to 93.73% in predicting outcomes by the 40th minute of the game.

However, logistic regression is not without limitations. The model assumes a linear relationship between the logit of the response and predictor variables, as well as independence and identical distribution of errors. Consequently, results can be inaccurate if these assumptions are violated. Moreover, logistic regression is often sensitive to overfitting which [Hosmer and Lemeshow, 2000] claim is "typically characterised by unrealistically large estimated coefficients and/or estimated standard errors". This sensitivity can lead to poor generalisation performance, especially in high-dimensional datasets. Nevertheless, regularisation techniques can mitigate overfitting by penalising excessive model complexity.

Regularisation techniques such as Lasso (L1 regularisation, Least Absolute Shrinkage and Selection Operator) and Ridge (L2 regularisation) are frequently utilised to mitigate overfitting in logistic regression models. Lasso regularisation introduces a penalty term to the logistic regression cost function, which coerces the coefficient estimates toward zero, thereby facilitating feature selection by driving certain coefficients to precisely zero. Conversely, Ridge regularisation penalises large coefficients by augmenting the cost function with the squared magnitude of coefficients, thus promoting smaller yet non-zero coefficient values. Formal definitions of these techniques are given in Section 8.1.

To ensure the selection of the optimal model, various goodness-of-fit summary measures can be used. In [Hosmer and Lemeshow, 2000] the main measures are highlighted as the Pearson chi-square statistic, classification tables, and the area under the Receiver Operating Characteristic (ROC) curve before suggesting their own method of Hosmer-Lemeshow tests. These measures collectively aid in evaluating a model's performance and its ability to accurately predict the outcome.

# 4 Data Collection

As highlighted in Section 2.3, the frequent updates to the game render data quickly outdated and potentially irrelevant. The data under consideration was gathered during the period of League of Legends Patches 13.22 and 13.23. However, it is important to note that the game has since been updated to Patch 14.9. This signifies that ten subsequent patches, each with their own modifications, have been implemented since the data was originally collected. One feature of these patches is that there has been two new champions added, making the total number now 167. For detailed information about the changes introduced in these patches, visit the official website [Riot Games, 2024a] and navigate to the 'Patch Notes' section.

After reviewing various match data available online, the decision was made to collect new data. The reasoning behind this was that the data available was often outdated or did not include any background information regarding the collection methods. Therefore, there was a large potential for bias that could not be controlled. Hence, data was gathered using the online Application Programming Interface (API) provided by the game developer, Riot Games. This was completed using the following packages: `httr`, `jsonlite`, and `tidyverse`.

The API documentation [Riot Games, 2024b] outlines parameter constraints, such as the requirement to select one out of ten available regions where the players are located. The data was collected entirely from Europe West, under the assumption

that player skills would be comparable across regions. Additionally, the API necessitated the specification of the match type. The 'ranked queue solo/duo 5v5' was chosen as it directly impacts player statistics. I hypothesised that players would approach these matches with more seriousness compared to other types, leading to results that are more reflective of the game's mechanics rather than chance. The API imposed rate limitations, allowing only 100 match requests every two minutes. Consequently, data collection spanned multiple weeks, encompassing two patches. While, ideally, data from a single patch would have been preferable, this was impractical due to the aforementioned constraints. However, it is worth noting that Patch 13.23, being the penultimate patch of 2023, was relatively minor. It introduced no new champions and only implemented slight balance adjustments.

Given that the data collection process extended over multiple weeks, there is a possibility that some players may have experienced changes in their ranks during this period. Nonetheless, it is likely that the ranks of the majority of players remained stable. Furthermore, since there had not been a recent reset of the ranks, it was inferred that most players had maintained their current ranks for a considerable duration. Additionally, the later analysis only considered the highest four ranks which are most difficult to achieve so it is unlikely more than a few people would have changed in or out of this rank.

To ensure a diverse dataset, stratified sampling was used across the ten ranks. Within each rank, players were randomly selected and their Player Universally Unique Identifiers (PUUIDs) were collected. The most recent match history from Patches 13.22 and 13.23 was then retrieved for each player. Once the match lists were generated, they were cross-verified to ensure there were no repeated matches that could introduce bias into the sample. This process resulted in the collection of 4000 matches for each rank, totalling 40,000 matches. While the raw data was preserved, given its voluminous nature, only the most relevant and interesting aspects were chosen for further study. Hence for each match, the following data was extracted: Match ID, PUUIDs of both teams, champions selected by each team, bans chosen by each team, and the outcome of the match.

After gathering the data from 40,000 matches, the rank distribution was inspected. The version utilised was the most recent one accessible in September 2023. As illustrated in the Figure 4.1, the distribution maintains a consistent shape over time, with only small deviations at any rank. To make the data more applicable to a broader context, the decision was made to merge some of the ranks. The ranks were grouped as follows: Iron, Bronze, Silver, Gold, Platinum/Emerald, and Diamond/Master/Grandmaster/Challenger (henceforth referred to collectively as the 'Top 4'). Despite Iron representing less than 8% of players, I chose to keep it as a separate category because my experience suggests that many players at this rank

are newcomers to the game. Consequently, their decisions are often less informed and exhibit greater variability. Conversely, the higher ranks predominantly consist of highly committed players or professionals. This method of rank consolidation simplified the analysis, as my objective was to explore differences between ranks and dealing with 10 separate ranks would have been cumbersome.

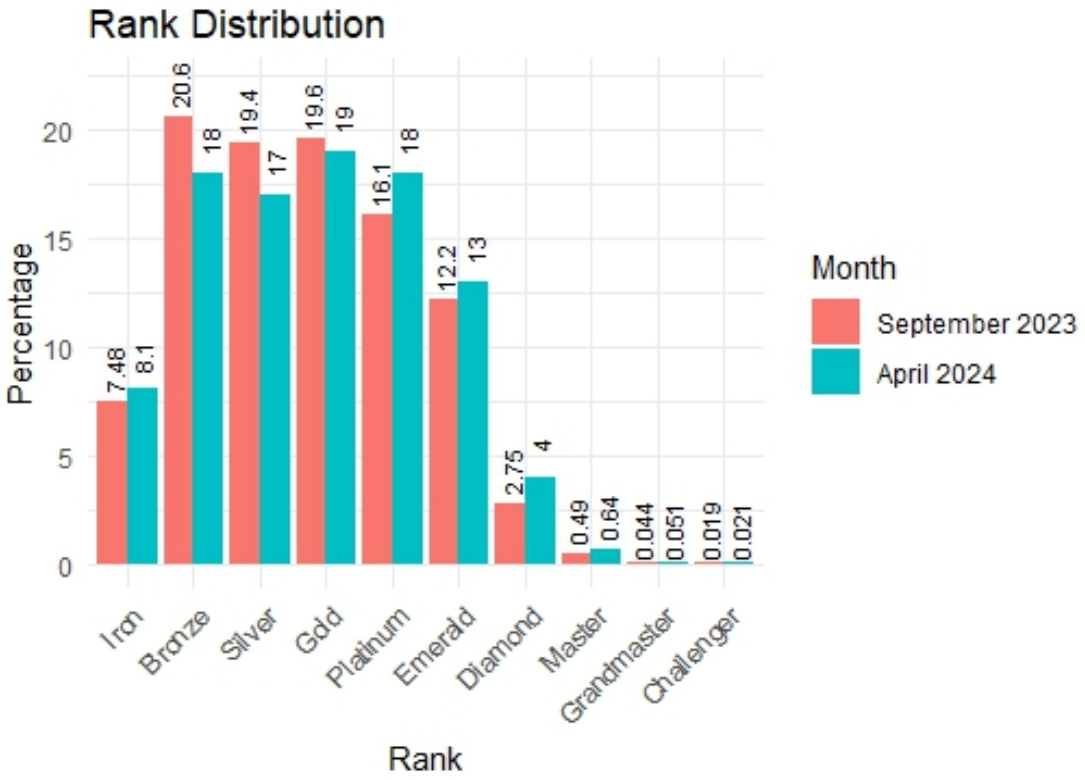


Figure 4.1: Comparison of worldwide player percentages across ranks at data collection and dissertation completion, using data from [Milella, 2024].

This consolidation, however, led to an uneven distribution of matches across the groups. To rectify this, a second round of data collection was undertaken, employing the same methodology as before. This ensured that each rank group contained 16,000 matches, culminating in a comprehensive dataset of 96,000 matches. Although I would have liked to accumulate further data, time constraints and API software issues (specifically its occasional unavailability and the fact it was prone to crashes) meant that this was not feasible.

# 5 Exploratory Analysis

The exploratory analysis utilised the `tidyverse` and `ggplot2` packages. Preprocessing of the data was the initial step by transforming the dataset, originally listing champions played, into a wide binary format. This format, explained in [Wickham and Grolemund, 2017], refers to a data structure where each observation is represented by a single row, and each variable is stored in its own column. This transformation resulted in 330 columns, each representing a champion's presence in either Team 1 or Team 2 during a match. An additional column was incorporated to denote the victory (1) or defeat (0) of Team 1.

A supplementary binary data frame was created to represent the presence of champion classes (as detailed in Section 2.2) in each match for both teams. This was done to investigate whether the team composition of classes could potentially influence the match outcome.

The analysis began by conducting frequency counts for each champion to determine their popularity, as shown in Table 5.1. Lux emerged as the most popular champion at lower ranks, likely due to her ease of play. Additionally a limited number of champions are available to new players, and Lux is frequently either in the free rotation or easily acquirable. Conversely, Kai'Sa was the most popular champion at higher ranks, attributed to her high-damage abilities despite being challenging to play due to the skill shots required. Across all ranks, there is a range of least popular champions. It is possible that less popular champions, such as Skarner, may not be compatible with more recently released champions due to their limited reworks since their release. For example, Skarner has had limited reworks since his 2011 release (Skarner has been reworked as of April 2nd, 2024). Overall, several factors could contribute to this variation in champion popularity. These factors include the inherent strengths of certain champions, ease of gameplay, versatility in different positions, frequency of updates, and the limited selection of champions available to new players until they unlock additional options.

| **Rank** | **Most played** | **Count** | **Least played** | **Count** |
|---|---|---|---|---|
| Iron | Lux | 4681 | Corki/Skarner | 87 |
| Bronze | Lux | 4472 | Skarner | 87 |
| Silver | Lux | 4031 | Corki | 95 |
| Gold | Kai'Sa | 3391 | Rek'Sai | 141 |
| Platinum/ Emerald | Kai'Sa | 3586 | Skarner | 127 |
| Top 4 | Kai'Sa | 4779 | Corki | 65 |

Table 5.1: Frequency counts of most and least played champions across different ranks.

To investigate the possibility of systematic biases in team assignments, win rates were calculated for both teams. The results revealed Team 1 and Team 2 boasting win rates of 49.26% and 50.74% respectively. This shows evidence that match outcomes were more likely influenced by natural variations rather than the system's team labelling. Next, the overall win rate for each champion was determined, with Nilah having the highest win rate of 53.222%, while Yuumi had the lowest win rate of 43.253%. This suggests that certain champions may possess inherent advantages, indicating an expected lack of perfect balance in the game, as supported by the updates to the game that subtly modify champions' abilities.

Moreover, win rates were computed within each rank for every champion, as depicted in Table 5.2. Excluding the Top 4, a pattern emerges: as the rank increases, the highest win rate decreases and the lowest win rate increases. This suggests that as players hone their skills, the impact of individual champions diminishes. However, this trend does not apply to the Top 4, implying that beyond a certain skill level, the choice of champion regains significance. An intriguing observation is that in the Iron rank, Corki, despite being one of the least popular champions (as per Table 5.1), has the highest win rate. A potential explanation for this anomaly could be that some seasoned players create new accounts for easier matchups. Consequently, it's plausible that the players opting for Corki are more skilled than the average player at that rank, thereby inflating Corki's win rate. On the other hand, Yuumi's low win rate aligns with the win rates calculated across all the ranks.

| **Rank** | **Highest WR** | **Champion** | **Lowest WR** | **Champion** |
|---|---|---|---|---|
| Iron | 60.920 | Corki | 37.955 | Yuumi |
| Bronze | 56.095 | Maokai | 37.5 | Rek'Sai |
| Silver | 57.543 | Kassadin | 40.272 | Gragas |
| Gold | 56.069 | Janna | 42.085 | Elise |
| Platinum/ Emerald | 55.690 | Udyr | 44.490 | Katarina |
| Top 4 | 56.774 | Vel'Koz | 39.286 | Shyvana |

Table 5.2: Champions with the highest and lowest win rates across different ranks.

An investigation followed to determine if a champion's class exerted a greater influence on win rates than the champions themselves. Figure 5.1 visually presents the win rates across all ranks, showing consistent median values but slight variations in interquartile ranges. The Controller class exhibited the smallest range at 0.009, while the Specialist class had a value of 0.025. Additionally, a specific box plot for the Top 4, depicted in Figure 5.1, revealed wider win rate ranges for each class compared to the overall plot. Notably, the Specialist class exhibited the smallest interquartile range at 0.021, while the Fighter class had the largest at 0.035. These findings underscore the influence of a player's rank on game outcomes.

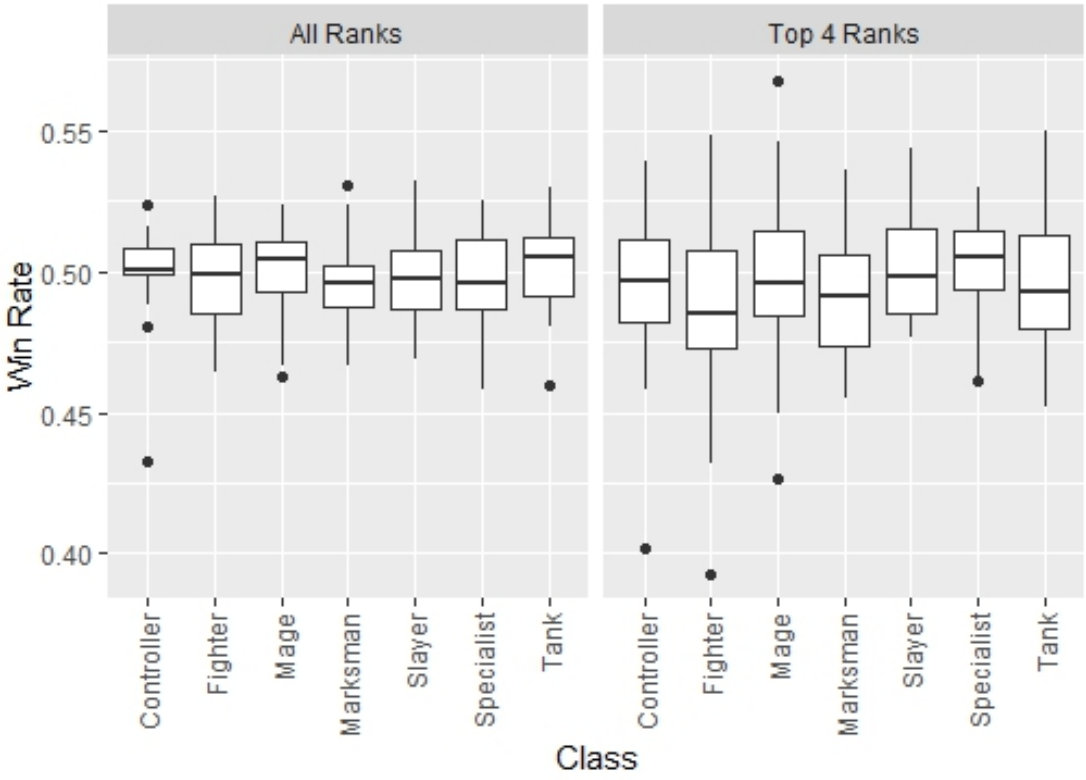


Figure 5.1: Win rate box plots for classes.

# 6 Association Rule Learning

## 6.1 Methodology

Association rule learning operates on a rule-based system with the primary function of uncovering intriguing relationships between variables in extensive databases. The rules that are generated from this process highlight the connections between variables and provide a quantifiable measure of these connections' strength.

Following the original definition by [Agrawal et al., 1993], $I = \{i_1, i_2, ..., i_n\}$ is defined as a set comprising $n$ binary attributes. Hence each $i$, commonly referred to as items, takes on either the value 1 or 0 (or equivalently true or false). Then $D = \{t_1, t_2, ..., t_m\}$ is defined as a set of transactions, collectively referred to as a database. Each transaction within $D$ possesses a unique transaction ID and includes a subset of $I$. A rule is then formulated as $X \rightarrow Y$ where both antecedent $X$ and consequent $Y$ are subsets of $I$. The statement can be read as if $X$ then $Y$.

In this dissertation, the dataset comprises 165 unique items (champions) and includes 16,000 transactions (matches) for each rank. A small scale example in Table 6.1 is used to illustrate the concept of a rule. The set of all potential items is denoted $I = \{Akali, Ekko, Gangplank, Morgana, Skarner\}$. The dataset consists of 3 transactions, each representing a unique match. For example, Match1 can be represented as $t_1 = \{Akali, Morgana, Skarner\}$. One possible rule is $\{Morgana, Skarner\} \rightarrow \{Akali\}$, indicating that if both Morgana and Skarner are picked, then Akali is likely to be picked as well. This rule has an antecedent of size 2 $\{Morgana, Skarner\}$ and a consequent of size 1 $\{Akali\}$.

| | **Akali** | **Ekko** | **Gangplank** | **Morgana** | **Skarner** |
|---|---|---|---|---|---|
| **Match1** | TRUE | FALSE | FALSE | TRUE | TRUE |
| **Match2** | FALSE | TRUE | FALSE | FALSE | FALSE |
| **Match3** | TRUE | FALSE | TRUE | TRUE | TRUE |

Table 6.1: Example of a small potential dataset.

The results were calculated using the `arules` [Hahsler et al., 2023] package. This package offers both the Apriori and Eclat algorithms. The Apriori algorithm was selected due to its satisfactory computational speed and ease of implementation due to example code given by [Hahsler et al., 2005]. The algorithm also provides updates on which stage of the process it is currently computing which is beneficial for the debugging process.

The Apriori algorithm identifies frequent individual items in the database and extends them to larger itemsets, provided these itemsets appear with sufficient frequency. The method, depicted in Figure 6.1 uses a greedy algorithm which [Black, 2005] defines as an algorithm which always takes the best immediate, or local, solution while finding an answer. To understand the diagram the following definitions are required:

**Definition 6.1** (Support for Itemset)**.** *The support of an itemset $X$ in a transaction database $D$ with $m$ transactions is the proportion of transactions containing $X$, denoted as:*

$$supp(X) = \frac{|\{t \in D : X \subseteq t\}|}{m}.$$

**Definition 6.2** (Minimum Support Threshold)**.** *The minimum support threshold is the user-defined parameter indicating the minimum level of support required for an itemset to be considered frequent.*

**Definition 6.3** (Confidence)**.** *Confidence is the conditional probability that the consequent of a rule occurs in a transaction given that the antecedent of the rule occurs in the transaction. For a rule $X \to Y$ in a transaction database $D$, confidence is calculated as:*

$$conf(X \to Y) = \frac{supp(X \cap Y)}{supp(X)}.$$

**Definition 6.4** (Minimum Confidence Threshold)**.** *The minimum confidence threshold is the user-defined parameter indicating the minimum level of confidence required for an association rule to be considered significant.*

**Definition 6.5** (Single Itemset)**.** *A single itemset is a set containing one item.*

The process illustrated in Figure 6.1 involves an algorithm begins by setting minimum support values. It then forms single item subsets from all transactions and prunes those that do not meet the minimum support. From the remaining subsets, frequent itemsets are formed. This approach, often referred to as the "bottom-up" approach, extends frequent subsets one item at a time, and the algorithm ceases operation when no further successful extensions are found. After the frequent itemsets have been found, the algorithm will generate association rules from these frequent itemsets. These rules are added to the list of rules only if they meet the minimum confidence, otherwise, they are eliminated.

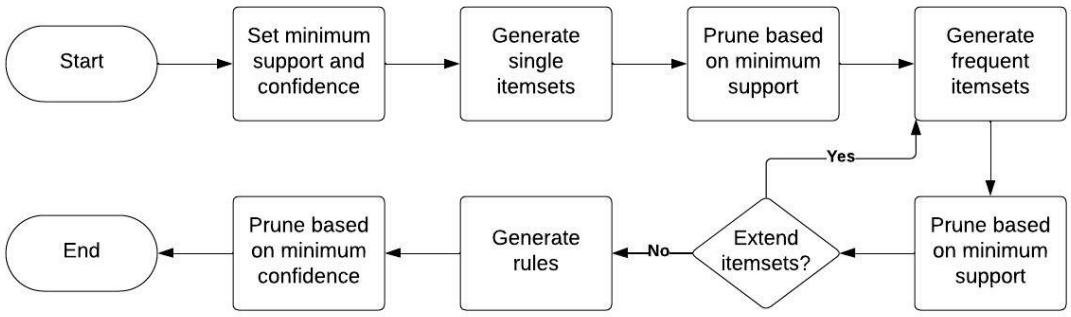


Figure 6.1: Apriori algorithm flowchart.

The Apriori algorithm relies on the anti-monotonicity property of the support measure. This property dictates that if an itemset is frequent, all its subsets must also be frequent, and conversely, if an itemset is infrequent, all its supersets will be infrequent. This characteristic significantly reduces the search space and enhances the efficiency of the algorithm. Figure 6.2 provides an illustration of this principle. To understand the diagram, the following definitions are required:

**Definition 6.6** (Infrequent Itemset)**.** *An itemset is considered infrequent if its support is below the minimum support threshold. In other words, an itemset $X$ is infrequent if $supp(X) <$ minimum support threshold.*

**Definition 6.7** (Superset)**.** *Given two sets $A$ and $B$, $B$ is called a superset of $A$ if every element of $A$ is also an element of $B$. Formally, $B$ is a superset of $A$ if $A \subseteq B$.*

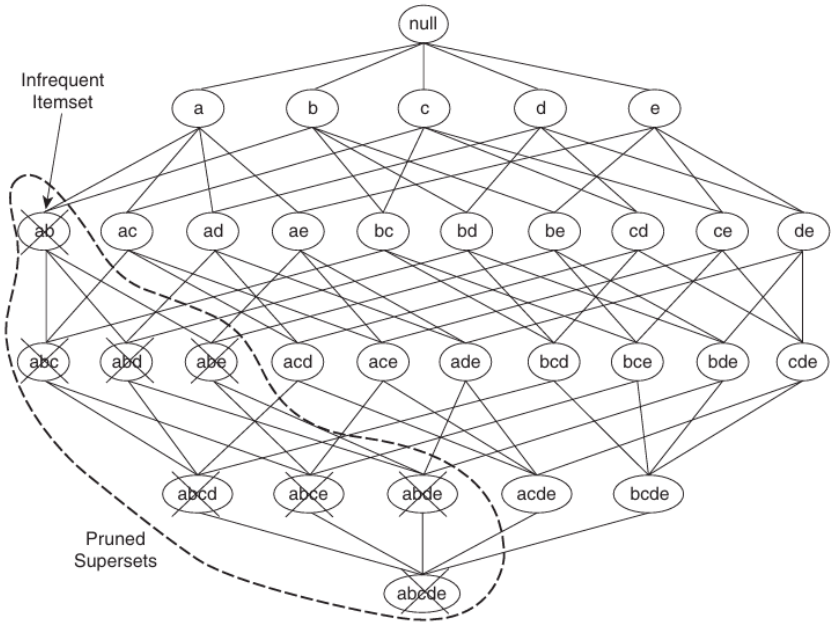


Figure 6.2: Illustration of support-based pruning from [Tan et al., 2005].

In Figure 6.2, the null denotes the empty set at the top. The solid lines represent the potential ways to add different items to this set. Thus, the circles below the empty set represents the five different items $I = \{a, b, c, d, e\}$. Adding another item yields an itemset of size two. The itemset *ab* is infrequent, leading to the pruning of all its supersets within the dotted line. Consequently, the largest possible itemset after support-based pruning will be of size 4, either *acde* or *bcde*.

The data was formatted appropriately during the exploratory analysis stage but a modification was later made to ensure that the champion columns were coded as factors. In addition, some of the rules required the labelling of Team 1 and Team 2 to be removed. The support metric, calculated using Definition 6.1, was set to a minimum value of 0.01 in this study. This was due to the large number of possible combinations relative to the number of transactions. The confidence metric, calculated using Definition 6.3, was set to a minimum value of 0.1. This was to ensure that the rules generated were relevant. Once the rules were found, they

were filtered in various ways to identify any of particular interest. For instance, filters were applied to identify only those rules that contained the match outcome within the set $Y$.

The results were analysed by interpreting the rules and assessing the quality of the rules using both their support and confidence values. This was partially accomplished by ordering the rules in descending order of confidence.

The first type of rules examined in this study generated two sets of rules: one for Team 1 and another for Team 2. The algorithm was applied twice, once with the itemset defined as all champions in Team 1 and then with it defined as all champions in Team 2. The dataset remained consistent, comprising all available matches for both teams. This approach aimed to identify any differences between the two teams and assess the impact of team labelling.

The distinction between Team 1 and Team 2 was then removed, effectively doubling the size of the dataset. Each original match now contributed two rows: one labelled as Team 1 and the other labelled as Team 2. This step allowed for a broader analysis, considering each champion's potential selection regardless of team affiliation.

Next, the labelling of the two teams was reinstated to find rules across teams. Consequently, the itemset encompassed all champions available to both teams, resulting in a size of 330. This analysis aimed to predict enemy selections based on teammates' choices.

Afterwards, the outcome variable for Team 1 (win or lose) was incorporated into the itemset. This addition aimed to evaluate how each team's selections might influence match outcomes.

In a later stage, a transformed dataset was utilised, categorising champions into the seven classes outlined in Section 2.2. Each original match was transformed into two rows, one for each team, resulting in a database of 32,000 matches. The absence of team labelling in this dataset allowed for an exploration of class impact on match outcomes.

## 6.2 Results

To further consider if there were any systematic differences between the labelling of the two teams, rules for Team 1 and Team 2 were generated separately. This gave the highly similar results shown in Table 6.2 where, as expected, there does not appear to be any major differences between them teams.

After finding this, it was investigated if there were any associations when the teams were unlabelled. An example of this would be if one person chooses champion A then it is likely that another person will choose champion B. The information from both teams was used which resulted in a total of 32,000 transactions. This gave a variety of different amounts of rules as seen in Table 6.4a. In Table 6.4b, you can see the highest confidence and the highest support rule for each rank. There was one rule "Caitlin $\rightarrow$ Lux" which was common across all ranks except the highest. Therefore, implying if one person chose to play Caitlin then another player on that team would choose to play Lux. This could allow the opposing team to select a champion that is a counter to Lux in anticipation of her selection. However, both the support and confidence values decreased for this rule as the ranks increased. The highest confidence rule overall "Xayah $\rightarrow$ Rakan" was exclusively in the top two ranks, with a confidence value of 0.302. This suggests that the champions Xayah and Rakan synergise well together, though this synergy may depend on the player's skill level. The presence of matching 'skins,' which are additional customisations to their appearance, further supports this notion as it indicates that the game designers expect these champions to be played together.

I then looked at rules that could suggest potential common counter picks when the two teams were labelled. This gave a larger volume of rules than only considering one team which can be seen in Table 6.5a. As before, the rules with the highest confidence and support values can be seen in Table 6.5b. Some rules were common across ranks while others only appeared in the top ranks. However, one interesting thing to note is that all the champions present in these rules are primarily categorised as either Bot or Support champions. This suggests that the associations are based on the two positions that have the most connected gameplay. It also suggests that the other positions (Jungle, Mid and Top) may select independently of these two positions.

Finally, it was rules where the teams were labelled and the outcome of the match was included. The purpose of this was to investigate if there are stronger choices for champions in general or if certain combinations of champions are beneficial to the team. The number of rules in each rank is shown in Table 6.6a and the rules with the highest confidence and support values are in Table 6.6b. In the Iron rank, the rule "Team 1 Yuumi $\rightarrow$ Team 2 Wins" is supported by Yuumi having the lowest win rate as seen previously in Table 5.2. Two reoccurring rules are "Team 1 Lux $\rightarrow$ Team 1 Wins" and "Team 1 Kai'Sa $\rightarrow$ Team 2 Wins". These are the two champions which were often present in the Table 5.1. However, it is noteworthy that the Kai'Sa rule appears unexpected, considering her high number of games played. One would anticipate that her presence would positively impact the team she belongs to. These findings will be examined in greater detail in Section 9.1.

In the association rule analysis, a notable trend emerged: the maximum confidence value tended to rise with higher ranks. Most rules exhibiting high confidence were concentrated within the Top 4. Acknowledging this pattern, the analysis redirected its attention to the Top 4, offering a more precise depiction of the game mechanics irrespective of skill level.

Upon closer examination of unlabelled team rules within the Top 4, I noticed that all rules had size 1 antecedents and consequents. To further analyse this, champions that frequently appeared together were combined to make a new item. Among the association rules for the Top 4, three rules stood out with confidence levels exceeding 0.25 (as shown in Table 6.7a). These three pairs were then chosen to be merged. For example, the two items $\{Xayah, Rakan\}$ became a single item $\{Xayah/Rakan\}$. Although several larger rules were identified (as depicted in Table 6.7b), their confidence levels were relatively low, prompting it to not be pursued further. This can be attributed to the vast number of possible combinations, resulting in many teams having very low support values.

Given the minimal impact of individual champions, the analysis changed focus to the seven classes of champions. My aim was to determine whether the combination of classes held more significance than the specific champions themselves. For unlabelled teams, I examined if the selection of a class would suggest the choice of another specific class. Furthermore, the match outcome was included to ascertain if specific class combinations could lead to superior results. The analysis yielded 100 rules based on the classes. Unfortunately, all these rules only implied implications with set $Y$ having a size of 1. Despite this, several intriguing rules emerged, as depicted in Table 6.3.

From this second analysis, it emerged that the choice of a single champion could impact the team's chances of winning. Specifically, selecting a Mage champion was associated with the highest confidence of securing a win, with a value of 0.506. On the other hand, opting for a Fighter champion yielded the lowest confidence, with a value of 0.490. The implications of this will be discussed within Section 9.1.

When composing a team consisting of a Controller, Mage, Marksman, and Tank, the confidence levels for selecting a Fighter (0.363) or a Slayer (0.356) as the final choice were notably similar. However, opting for a Slayer champion yielded a slightly higher confidence value of 0.542 for a win, compared to the Fighter's value of 0.512. Potential explanations for this discrepancy are explored in Section 9.1.

In contrast, for a team composition of Fighter, Mage, Marksman and Slayer, there was a significant difference in confidence levels when choosing between a Controller

(0.571) and a Tank (0.266). Interestingly, despite this difference, the impact on the team's winning confidence levels was very similar, with the Tank yielding a slightly higher value of 0.507 compared to the Controller's value of 0.506. This suggests that the choice between a Controller and a Tank does not significantly affect the team's chances of winning. The implications of this are discussed in Section 9.1.

| **Rank** | **Team 1** | | **Team 2** | |
|---|---|---|---|---|
| | **N.o. Rules** | **Confidence** | **N.o. Rules** | **Confidence** |
| Iron | 26 | 0.173 | 22 | 0.161 |
| Bronze | 20 | 0.176 | 17 | 0.159 |
| Silver | 12 | 0.162 | 13 | 0.167 |
| Gold | 8 | 0.269 | 5 | 0.148 |
| Platinum/ Emerald | 8 | 0.245 | 5 | 0.267 |
| Top 4 | 48 | 0.306 | 45 | 0.298 |

Table 6.2: Number of rules and maximum confidence levels for Team 1 and 2.

| **Rule** | **Support** | **Confidence** |
|---|---|---|
| Mage → Win | 0.368 | 0.506 |
| Fighter → Win | 0.294 | 0.490 |
| Controller,Mage,Marksman,Tank → Fighter | 0.029 | 0.363 |
| Controller,Mage,Marksman,Tank → Slayer | 0.029 | 0.356 |
| Controller,Mage,Marksman,Slayer,Tank → Win | 0.016 | 0.542 |
| Controller,Fighter,Mage,Marksman,Tank → Win | 0.015 | 0.512 |
| Fighter,Mage,Marksman,Slayer → Controller | 0.073 | 0.571 |
| Fighter,Mage,Marksman,Slayer → Tank | 0.034 | 0.266 |
| Fighter,Mage,Marksman,Slayer,Tank → Win | 0.017 | 0.507 |
| Controller,Fighter,Mage,Marksman,Slayer → Win | 0.037 | 0.506 |

Table 6.3: Interesting rules for classes in the Top 4.

| Rank | Number of Rules | Maximum Confidence |
|---|---|---|
| Iron | 26 | 0.167 |
| Bronze | 13 | 0.175 |
| Silver | 14 | 0.163 |
| Gold | 8 | 0.146 |
| Platinum/Emerald | 6 | 0.247 |
| Top 4 | 46 | 0.302 |

(a) Number of rules and maximum confidence values.

| Rank | Rule | Support | Confidence |
|---|---|---|---|
| Iron | Caitlin → Lux | 0.016 | 0.167 |
| | Miss Fortune → Lux | 0.020 | 0.144 |
| Bronze | Jinx → Lux | 0.012 | 0.175 |
| | Caitlin → Lux | 0.014 | 0.143 |
| Silver | Jhin → Lux | 0.012 | 0.163 |
| | Caitlin → Lux | 0.014 | 0.139 |
| Gold | Jhin → Lux | 0.011 | 0.146 |
| | Caitlin → Lux | 0.013 | 0.130 |
| Platinum/Emerald | Xayah → Rakan | 0.012 | 0.247 |
| | Caitlin → Lux | 0.012 | 0.130 |
| Top 4 | Xayah → Rakan | 0.019 | 0.302 |
| | Nautilus → Kai'Sa | 0.026 | 0.268 |

(b) Interesting rules with their support and confidence values.

Table 6.4: Rules for the unlabelled teams across the ranks.

| Rank | Number of Rules | Maximum Confidence |
|---|---|---|
| Iron | 129 | 0.185 |
| Bronze | 88 | 0.183 |
| Silver | 67 | 0.178 |
| Gold | 28 | 0.133 |
| Platinum/Emerald | 30 | 0.157 |
| Top 4 | 206 | 0.334 |

(a) Number of rules and maximum confidence values.

| Rank | Rule | Support | Confidence |
|---|---|---|---|
| Iron | Team 1 Ashe → Team 2 Lux | 0.018 | 0.185 |
| | Team 1 Miss Fortune → Team 2 Lux | 0.020 | 0.144 |
| Bronze | Team 1 Morgana → Team 2 Lux | 0.014 | 0.183 |
| | Team 1 Miss Fortune → Team 2 Lux | 0.018 | 0.135 |
| Silver | Team 1 Morgana → Team 2 Lux | 0.012 | 0.178 |
| | Team 1 Caitlin → Team 2 Lux | 0.014 | 0.132 |
| Gold | Team 1 Jhin → Team 2 Caitlin | 0.010 | 0.133 |
| | Team 1 Ezreal → Team 2 Caitlin | 0.012 | 0.129 |
| Platinum/Emerald | Team 1 Vayne → Team 2 Kai'Sa | 0.011 | 0.157 |
| | Team 1 Ezreal → Team 2 Kai'Sa | 0.015 | 0.138 |
| Top 4 | Team 1 Xayah → Team 2 Kai'Sa | 0.022 | 0.334 |
| | Team 1 Ezreal → Team 2 Kai'Sa | 0.025 | 0.208 |

(b) Interesting rules with their support and confidence values.

Table 6.5: Rules for the labelled teams across the ranks.

| Rank | Number of Rules | Maximum Confidence |
|---|---|---|
| Iron | 367 | 0.651 |
| Bronze | 381 | 0.577 |
| Silver | 387 | 0.575 |
| Gold | 420 | 0.603 |
| Platinum/Emerald | 445 | 0.599 |
| Top 4 | 425 | 0.532 |

(a) Number of rules and maximum confidence values.

| Rank | Rule | Support | Confidence |
|---|---|---|---|
| Iron | Team 1 Yuumi → Team 2 Wins | 0.022 | 0.651 |
| | Team 1 Lux → Team 1 Wins | 0.081 | 0.557 |
| Bronze | Team 1 Zyra → Team 1 Wins | 0.029 | 0.577 |
| | Team 1 Lux → Team 1 Wins | 0.072 | 0.501 |
| Silver | Team 1 Yuumi → Team 2 Wins | 0.011 | 0.575 |
| | Team 1 Lux → Team 1 Wins | 0.069 | 0.526 |
| Gold | Team 1 Yuumi → Team 2 Wins | 0.010 | 0.603 |
| | Team 1 Lux → Team 1 Wins | 0.054 | 0.505 |
| Platinum/Emerald | Team 1 Olaf → Team 1 Wins | 0.017 | 0.599 |
| | Team 1 Kai'Sa → Team 2 Wins | 0.079 | 0.525 |
| Top 4 | Team 1 Lissandra → Team 1 Wins | 0.010 | 0.532 |
| | Team 1 Kai'Sa → Team 2 Wins | 0.073 | 0.492 |

(b) Interesting rules with their support and confidence values.

Table 6.6: Rules for the labelled teams and outcomes across the ranks.

| Rule | Support | Confidence |
|---|---|---|
| Xayah → Rakan | 0.019 | 0.302 |
| Karma → Ezreal | 0.013 | 0.274 |
| Nautilus → Kai'Sa | 0.026 | 0.268 |

(a) Highest confidence rules.

| Rule | Support | Confidence |
|---|---|---|
| Xayah/ Rakan → Syndra | 0.002 | 0.121 |
| Karma/ Ezreal → Orianna | 0.002 | 0.119 |
| Nautilus/ Kai'Sa → Orianna | 0.003 | 0.117 |
| Karma/ Ezreal → Syndra | 0.002 | 0.117 |
| Nautilus/ Kai'Sa → Syndra | 0.003 | 0.106 |

(b) Rules after combining common pairs from highest confidence rules.

Table 6.7: Rules for the unlabelled teams in the Top 4.

# 7 Clustering

## 7.1 Methodology

Both hierarchical and k-means clustering techniques were used on various binary datasets. Both methods require establishing a measure of similarity or dissimilarity between data points or clusters. Typically using distance metrics like Euclidean or Manhattan distance as defined below. These definitions were adapted from [Suhaeri et al., 2021] as the notation has been adjusted to better match this work.

**Definition 7.1** (Euclidean Distance)**.** *The Euclidean distance between two points* $p_1 = (x_1, x_2, \ldots, x_n)$ *and* $p_2 = (y_1, y_2, \ldots, y_n)$ *in an n-dimensional space is calculated as the square root of the sum of the squared differences between their corresponding coordinates:*

$$\textit{Euclidean distance} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}$$

**Definition 7.2** (Manhattan Distance)**.** *The Manhattan distance between two points* $p_1 = (x_1, x_2, \ldots, x_n)$ *and* $p_2 = (y_1, y_2, \ldots, y_n)$ *in a n-dimensional space is calculated as the sum of the absolute differences of their corresponding coordinates:*

$$\textit{Manhattan distance} = |x_1 - y_1| + |x_2 - y_2| + \ldots + |x_n - y_n|$$

These distance measures provide a quantitative way to compare the dissimilarity between data points or clusters. The choice of distance metric depends on the characteristics of the data and the clustering objectives. For instance, Euclidean distance is commonly used for continuous data, while Manhattan distance may be preferred for data with categorical or ordinal attributes. Hence, when investigating the relationship between frequency played and win rate, as shown in Figure 7.3, using k-means clustering Euclidean distance was used. Whereas, Manhattan distance was used for the other k-means clustering and all of hierarchical clustering.

### 7.1.1 K-means Clustering

K-means clustering is a machine learning algorithm used for unsupervised learning. Unlike supervised learning, where algorithms are guided by labelled data, clustering attempts to uncover hidden structures and relationships. The primary aim of clustering is to group data points into distinct clusters based on similarities in their features, thereby revealing underlying patterns that may not be immediately apparent. By iteratively partitioning the dataset into clusters and refining their centroids to minimise within-cluster variance, k-means clustering tries to organise complex data into meaningful groupings, facilitating insights and further analysis. This method was chosen because of its computational efficiency and ability to handle large datasets. In addition, the output is easy to interpret as it assigns each data point to a single cluster making it useful for visualisation.

K-Means clustering is a partitioning method that divides a set of $n$ observations into $k$ non-overlapping clusters. Each observation belongs to the cluster with the nearest mean, which serves as a prototype of the cluster. The objective of k-means is to minimise the within-cluster variance (see Definition 7.3). Understanding the workings of the algorithm is facilitated by referring to the provided pseudo-code (Algorithm 1), which draws upon the methodology outlined in [MacKay, 2005]. The algorithm converges when there is no significant change in the cluster assignments or the positions of the centroids between iterations, typically determined by monitoring changes in the within-cluster variance or by setting a maximum number of iterations.

**Definition 7.3** (Within-cluster variance)**.** *Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of $n$ observations, where each $x_i$ is a $d$-dimensional data point. Then the within-cluster variance is*

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \phi(x, \mu_i)$$

*where:*

- *$k$ is the number of clusters;*
- *$C_i$ is the $i$-th cluster;*
- *$\mu_i$ is the centroid (mean) of the $i$-th cluster;*
- *$\phi(x, \mu_i)$ is the distance measure.*

**Algorithm 1** K-means clustering

**Require:** Data points $X = \{x_1, x_2, ..., x_n\}$, number of clusters $k$.
**Ensure:** Cluster centroids $C = \{\mu_1, \mu_2, ..., \mu_k\}$.

```
Randomly initialise k centroids μ_1, μ_2, ..., μ_k.
repeat
    for each data point x_i do
        Calculate the nearest centroid: μ_nearest = arg min_j dist(x_i, μ_j).
        Assign x_i to the nearest centroid: x_i → μ_nearest.
    end for
    for each centroid μ_j do
        Update centroid: μ_j = (1/|C_j|) Σ_{x_i ∈ C_j} x_i, where C_j is the set of points assigned
        to μ_j.
    end for
until Convergence
return Cluster centroids C.
```

The results were completed using the following packages: `plotly`, `factoextra`, `cluster` and `tidyverse`.

### 7.1.2 Exploring Optimal Cluster Number with the Elbow Method

In the analysis, the elbow method was tested to determine the ideal number of clusters. This method aids in identifying the point at which adding more clusters ceases to significantly enhance the reduction of within-cluster variance. By systematically increasing the number of clusters and observing how the within-cluster variance decreases, you seek the point where additional clusters yield diminishing returns in variance reduction.

To construct an elbow plot, one plots the number of clusters ($k$) against their corresponding within-cluster variance. The plot typically exhibits a sharp decrease in within-cluster variance as the number of clusters rises, followed by a more gradual decline. The "elbow" point on this plot represents the optimal number of clusters, as it signifies the juncture where the rate of variance reduction diminishes significantly. An example of an elbow plot can be seen in Figure 7.1, where the optimal number of clusters is 4.

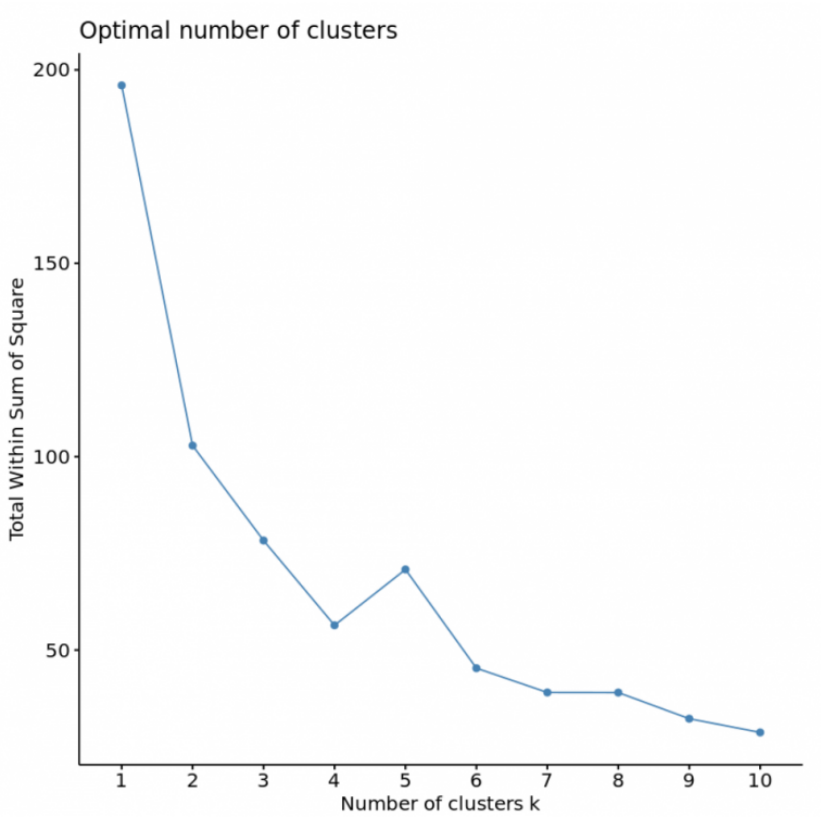


Figure 7.1: Elbow plot with an optimal number of 4 clusters from [Bobbitt, 2022].

### 7.1.3 Hierarchical Clustering

Hierarchical clustering is a data analysis technique that sorts data points into clusters based on either Euclidean (Definition 7.1) or Manhattan (Definition 7.2) distance. There are two main types of hierarchical clustering: agglomerative and divisive, both of which can be seen in Figure 7.2. In agglomerative clustering, also known as bottom-up clustering, each data point initially forms its own cluster, and pairs of clusters are iteratively merged based on their similarity until all points belong to a single large cluster. Algorithm 2 provides pseudocode for the agglomerative clustering method, as this is the method used within the analysis. Conversely, divisive clustering, or top-down clustering, starts with a single cluster containing all data points and recursively splits it into smaller clusters until each cluster contains only one data point.

**Algorithm 2** Agglomerative Hierarchical Clustering

**Require:** Data points $X = \{x_1, x_2, ..., x_n\}$
**Ensure:** Dendrogram representing hierarchical clustering
1: Initialise each data point as a separate cluster: $C_1, C_2, ..., C_n$
2: Compute the distance matrix $D$ between all pairs of clusters using a chosen distance metric
3: **while** more than one cluster remains **do**
4: Find the pair of clusters with the smallest distance: $(p, q) = \arg\min_{i,j} D_{ij}$
5: Merge clusters $C_p$ and $C_q$ to form a new cluster $C_{pq}$
6: Update the distance matrix $D$ to reflect the merging of clusters $C_p$ and $C_q$
7: **end while**
8: **return** Dendrogram representing the hierarchical clustering

One critical aspect influencing the behaviour of hierarchical clustering algorithms is the choice of linkage criterion, which determines how the distance between clusters is computed. Common linkage criteria include single linkage, complete linkage, average linkage, centroid linkage, and Ward's linkage. Using [Jarman, 2020] as starting point, these can be defined as:

**Definition 7.4** (Single Linkage)**.** *The single linkage criterion defines the distance between two clusters $C_i$ and $C_j$ as the shortest distance between any two points in the clusters:*

$$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} dist(x, y).$$

**Definition 7.5** (Complete Linkage)**.** *The complete linkage criterion defines the distance between two clusters $C_i$ and $C_j$ as the maximum distance between any two points in the clusters:*

$$d(C_i, C_j) = \max_{x \in C_i, y \in C_j} dist(x, y).$$

**Definition 7.6** (Average Linkage)**.** *The average linkage criterion defines the distance between two clusters $C_i$ and $C_j$ as the average distance between each point in one cluster to every point in the other cluster:*

$$d(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} dist(x, y).$$

**Definition 7.7** (Centroid Linkage)**.** *The centroid linkage criterion defines the distance between two clusters $C_i$ and $C_j$ as the distance between their centroids (means):*

$$d(C_i, C_j) = dist(centroid(C_i), centroid(C_j)).$$

**Definition 7.8** (Ward's Linkage)**.** *Ward's linkage is a variance-minimising approach where the distance between two clusters $C_i$ and $C_j$ is based on the increase in variance resulting from merging them.*

These different linkage criteria can lead to distinct cluster structures and may be chosen based on the specific characteristics of the dataset or the objectives of the analysis. Within this analysis, ward's linkage was used because the data was binary.

One common way to represent the results of hierarchical clustering is through a dendrogram, which is a tree-like diagram, where each node represents either a single data point or a cluster of data points. An example dendrogram can be seen in Figure 7.2. The structure of the dendrogram reflects the order in which clusters were merged or divided during the clustering process. Interpreting the dendrogram involves tracing the paths from the leaf nodes to the root node, which represents the final, all-encompassing cluster. The height at which branches merge or split corresponds to the level of dissimilarity at which these operations occur. Thus, the structure of the dendrogram provides insights into the hierarchical organisation of the data, revealing clusters at different levels of granularity.

A notable advantage of hierarchical clustering is its flexibility regarding the number of clusters. Once the dendrogram is constructed, slicing it horizontally defines individual clusters at varying levels of granularity. This flexibility allows for the exploration of sub-clusters and adjustments to the clustering granularity.

The reason for using hierarchical clustering is that it is useful for uncovering the underlying structure of data and identifying outliers. The agglomerative approach was used in this dissertation due to its computational efficiency, surpassing the divisive method in terms of resource utilisation.

The analyses were conducted using the `cluster` and `tidyverse` packages. A total of 15 clusters were chosen to strike a balance between granularity and cluster size, aiming to uncover meaningful patterns while maintaining manageable cluster sizes.

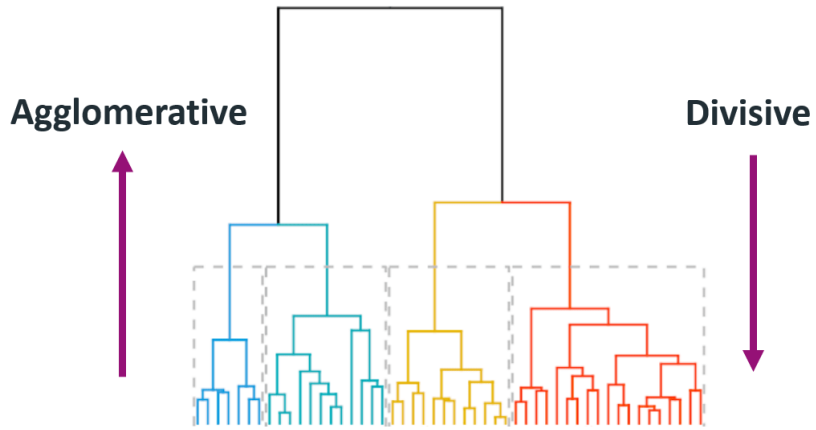


Figure 7.2: Agglomerative and divisive clustering example dendrogram from [ClicData, 2020].

## 7.2 Results

### 7.2.1 K-means Clustering

The analysis began with clustering on champions based on their frequency and win rate which can be seen in Figure 7.3. In cluster 2, there are champions that are more frequent and seem to possess middling win rate, meaning they lost and won similar amounts. Then cluster 1 denotes champions that were not played a lot and seemed to lose when they were played. Interestingly, cluster 3 contained champions that were not played that much but had a high win rate when they were.

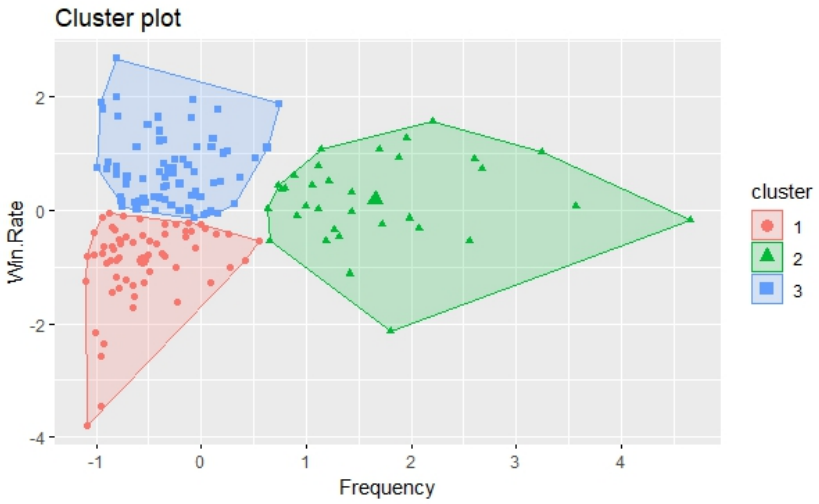


Figure 7.3: K-means clustering on champions using win rate and frequency.

The elbow plot in Figure 7.4 was examined, but there was no optimal number identified using it. It suggested that around 3 or 4 would be appropriate. After testing both, three clusters were chosen for Figure 7.3 as it provided the best visualisation and logical output. Since there was little to be gained from the individual champions, the focus shifted to the classes of champions instead.

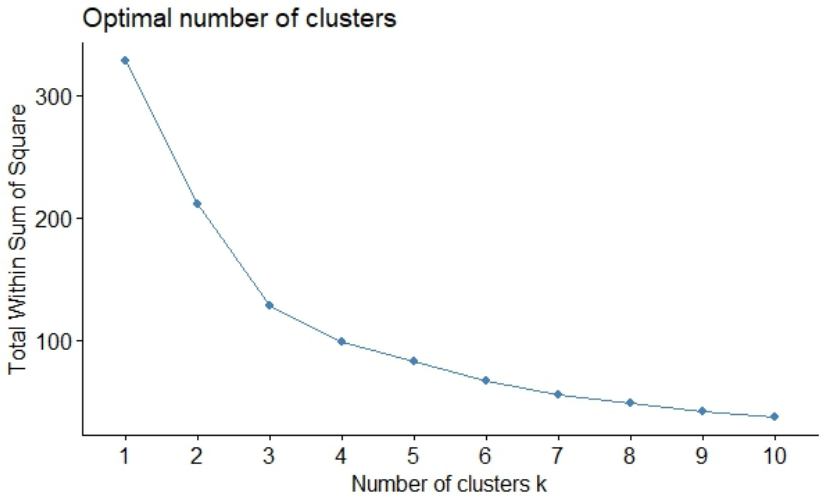


Figure 7.4: Elbow plot for k-means clustering on champions.

The matches were used to cluster the different classes of champions, resulting in Figure 7.5. A variety of different numbers of clusters were attempted; however, as before, the elbow plot did not reveal anything useful, so it has not been included here. Nonetheless, any attempt with more than four clusters did not converge, as the clusters would change each time the algorithm was run. Across all 4 clusters, there were similar outcomes for the team of approximately 0.47 to 0.5. Cluster 1 and 2 were similar except that cluster 1 had a Fighter but no Slayer, while cluster 2 had a Slayer but no Fighter. Cluster 3 and 4 were also similar except that cluster 3 had a Controller and cluster 4 had no Controller but a much larger emphasis on a tank. Across all the 4 clusters the Specialist class was infrequent. However, as seen in Table 7.1 Specialists are the least popular class of champion overall in the Top 4.

| **Class** | **Controller** | **Fighter** | **Mage** | **Marksman** | **Slayer** | **Specialist** | **Tank** |
|---|---|---|---|---|---|---|---|
| **Percentage** | 66.884 | 59.969 | 72.647 | 79.338 | 66.125 | 26.378 | 46.419 |

Table 7.1: Percentage of matches in the Top 4 containing each class.

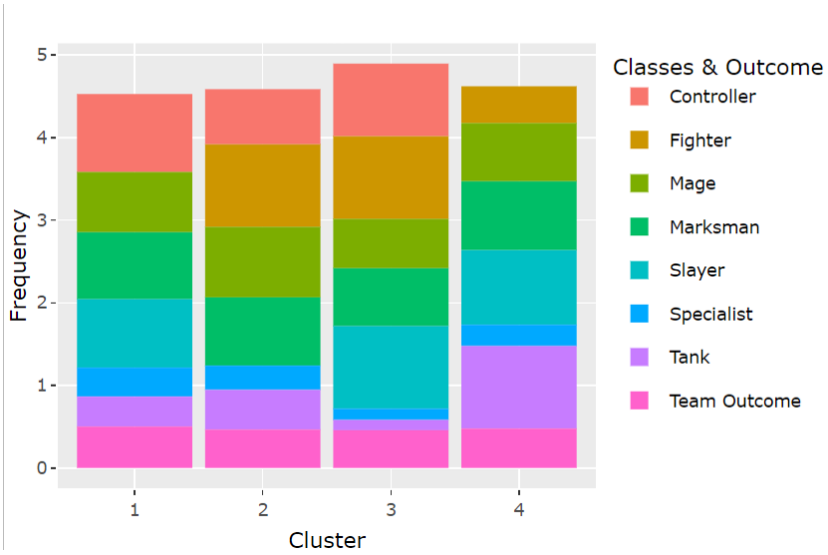


Figure 7.5: K-means clustering of classes using matches.

### 7.2.2 Hierarchical Clustering

The initial approach involved applying hierarchical clustering on individual champions. However, this method was not pursued due to its high dimensionality, which led to significant computational times and a lack of interpretability. Therefore, an alternative approach was considered, which involved performing hierarchical clustering on the different classes. This was done on unlabelled teams meaning that the outcome was if the given team won. The results of the clustering can be seen in Table 7.2.

After trying a range of choices for the number of clusters, it was decided that 15 would be an interesting number to study. Anymore than 15 would result in $n$ (the number of matches represented in each cluster) becoming too small and would become more likely to be unrepresentative and lack generality. An interesting discovery was made in relation to cluster 3, which demonstrated a good win rate of 0.780. Notably, the team associated with this cluster never included a Mage. It is important to note that this was based on only 1571 matches. In contrast cluster 9, which contained 840 matches, was associated with a 0 win rate and never contained a Mage or a Tank. In addition, it always a Controller, Marksman and Slayer.

Due to computational constraints, the analysis was only conducted on a subset of 16,000 matches. This constraint should be taken into account when interpreting the results.

| Cluster | Outcome | Controller | Fighter | Mage | Marksman | Slayer | Specialist | Tank | n |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.498 | 0.000 | 1.000 | 0.835 | 1.000 | 0.051 | 0.349 | 1.000 | 1118 |
| 2 | 0.466 | 0.992 | 0.996 | 0.000 | 0.867 | 0.031 | 0.530 | 0.487 | 517 |
| 3 | 0.780 | 0.930 | 0.493 | 0.000 | 0.688 | 0.922 | 0.358 | 0.388 | 1571 |
| 4 | 0.442 | 0.463 | 0.000 | 0.884 | 1.000 | 1.000 | 0.000 | 1.000 | 1327 |
| 5 | 0.495 | 1.000 | 0.000 | 1.000 | 1.000 | 0.837 | 0.347 | 0.000 | 1570 |
| 6 | 0.434 | 0.000 | 0.728 | 0.884 | 1.000 | 0.734 | 0.380 | 0.000 | 827 |
| 7 | 0.467 | 0.175 | 0.852 | 0.424 | 1.000 | 1.000 | 0.000 | 1.000 | 1367 |
| 8 | 0.468 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.201 | 0.248 | 1854 |
| 9 | 0.000 | 1.000 | 0.664 | 0.000 | 1.000 | 1.000 | 0.340 | 0.000 | 840 |
| 10 | 0.502 | 0.552 | 0.000 | 1.000 | 1.000 | 0.000 | 0.516 | 1.000 | 611 |
| 11 | 0.478 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 1183 |
| 12 | 0.458 | 0.374 | 0.969 | 1.000 | 0.000 | 0.610 | 0.202 | 0.858 | 893 |
| 13 | 0.504 | 0.810 | 0.000 | 1.000 | 0.000 | 0.899 | 0.357 | 0.515 | 844 |
| 14 | 0.393 | 0.000 | 0.156 | 0.486 | 1.000 | 1.000 | 1.000 | 1.000 | 430 |
| 15 | 0.462 | 1.000 | 1.000 | 0.971 | 0.000 | 0.659 | 0.229 | 0.000 | 1048 |

Table 7.2: Hierarchical clustering on the classes and outcome.

# 8 Logistic Regression

## 8.1 Methodology

Logistic regression is used for analysing the relationship between predictor variables and binary outcomes. This section will present the methodology for modelling match outcomes. Multiple logistic regression models were formulated to predict the probability of a team winning a match. To address potential overfitting and enhance the model's robustness, regularisation can be used:

**Definition 8.1** (Lasso Regularisation)**.** *Lasso regularisation, or L1 regularisation, constrains model coefficients by adding a penalty term to the optimisation objective. In logistic regression, the lasso penalty term is defined as:*

$$Lasso(\beta) = \lambda \sum_{j=1}^{p} |\beta_j|$$

*where $\beta$ are the model coefficients, $p$ is the number of predictors, and $\lambda$ is the regularisation parameter controlling penalty strength. The term $|\beta_j|$ denotes the absolute value of coefficient $\beta_j$. Lasso promotes sparsity by shrinking some coefficients to zero, facilitating feature selection. Higher $\lambda$ values lead to more aggressive coefficient shrinkage, balancing model complexity and accuracy.*

**Definition 8.2** (Ridge Regularisation)**.** *Ridge regularisation, or L2 regularisation, mitigates overfitting in regression models by penalising large coefficients. In logistic regression, the Ridge penalty term is:*

$$Ridge(\beta) = \lambda \sum_{j=1}^{p} \beta_j^2$$

*where $\beta$ are model coefficients, $p$ is the number of predictors, and $\lambda$ controls the penalty strength. Ridge encourages smaller but non-zero coefficients, reducing model complexity.*

Within this analysis, lasso regularisation was chosen to try to reduce the high-dimensionality of the data. For further details on the models' equations and the application of lasso regularisation, refer to Section 8.1.1.

In all the models, the following definitions will apply:

**Definition 8.3** (Logit Function)**.** *The logit function, denoted by logit($p$), is the natural logarithm of the odds of success $p$ in a binary event. It serves as a link function in logistic regression, transforming the probability of success into a linear combination of predictor variables:*

$$logit(p) = \ln\left(\frac{p}{1-p}\right).$$

**Definition 8.4** (Logistic Function)**.** *The logistic function, denoted by logistic($t$), is defined as the sigmoid function, which maps any real-valued input $t$ to the range $(0, 1)$:*

$$logistic(t) = \frac{1}{1+e^{-t}}.$$

**Definition 8.5** (Maximum Likelihood Estimation (MLE))**.** *Maximum Likelihood Estimation (MLE) is a method to estimate the parameters of a statistical model by maximising the likelihood function $L(\theta|X)$ given observed data $X$. It seeks to find the parameter values $\theta$ that maximise $L(\theta|X)$, expressed as:*

$$\hat{\theta}_{MLE} = \arg\max_{\theta} L(\theta|X),$$

*where $\hat{\theta}_{MLE}$ is the maximum likelihood estimate of the parameters.*

The models were fitted using maximum likelihood estimation and these packages: `tidyverse`, `caret`, `pROC`, `MASS` and `glmnet`. To assess model performance, the data was split into training and testing sets at a 7:3 ratio. Evaluation metrics including specificity, sensitivity, and area under the Receiver Operating Characteristic (ROC) curves were employed using the following definitions.

**Definition 8.6** (Sensitivity)**.** *Sensitivity, or the true positive rate, measures the proportion of actual positives correctly identified by the classifier:*

$$Sensitivity = \frac{TP}{TP+FN}$$

*where $TP$ and $FN$ denote true positives and false negatives, respectively.*

**Definition 8.7** (Specificity)**.** *Specificity, or the true negative rate, measures the proportion of actual negatives correctly identified by the classifier:*

$$Specificity = \frac{TN}{TN + FP}$$

*where $TN$ and $FP$ denote true negatives and false positives, respectively.*

**Definition 8.8** (AUC)**.** *The Area Under the Curve (AUC) of a ROC curve quantifies the classifier's ability to distinguish between classes:*

$$AUC = \int_0^1 TPR(f)\, dFPR(f)$$

*where $TPR(f)$ and $FPR(f)$ represent the true positive and false positive rates, respectively, at a given decision threshold $f$.*

The logistic regression coefficient $\beta$ associated with a predictor $X$ is the expected change in log odds of having the outcome per unit change in $X$. So, when you exponentiate this coefficient, it gives you the odds ratio, which represents the multiplicative change in the odds for a one-unit change in the predictor. To interpret the models, the $\beta$ coefficients were exponentiated to give the odds ratios for the corresponding independent variables. In addition, p-values were calculated for the odds ratios to see if any were of statistical significance.

### 8.1.1 Model Specification

The following are the models used within the analysis.

$$logit(p_1) = \beta_0 + \sum_{i=1}^{n_1} \beta_{1i} x_{1i} + \sum_{j=1}^{n_2} \beta_{2j} x_{2j} \tag{8.1}$$

where $logit(p_1)$ describes the log odds of Team 1 winning a match. The model comprises an intercept term $\beta_0$ and indicator variables $x_{1i}$, $x_{2j}$ representing the presence of each "Champion" for both Team 1 and 2. The terms $n_1$ and $n_2$ denote the number of champions available to each respective team. This model was also used with lasso regularisation penalty term $\lambda \sum_{j=1}^{p} |\beta_j|$.

$$logit(p) = \beta_0 + \sum_{i=1}^{7} \beta_{1i} y_{1i} + \sum_{j=1}^{7} \beta_{2j} y_{2j} \tag{8.2}$$

where $logit(p)$ describes the log odds of winning a match. The model includes an intercept term $\beta_0$ and indicator variables $y_{1i}$ and $y_{2j}$ for the presence of a "Class" for both Team 1 and 2. This model was also used with lasso regularisation penalty term $\lambda \sum_{j=1}^{p} |\beta_j|$. In addition, interaction terms between the classes of the two teams were later added.

$$logit(p) = \beta_0 + \beta_1 x + \sum_{i=1}^{5} \gamma_i z_i + \sum_{i=1}^{5} \sum_{j=1}^{5} \delta_{ij} x y_i \tag{8.3}$$

where $logit(p)$ predicts the log odds of winning a match. The model includes an intercept term $\beta_0$, an indicator variable for the presence of "Champion" ($\beta_1 x$), a categorical variable "Rank" ($\gamma_i y_i$), and interaction terms between "Champion" and each "Rank" category ($\delta_{ij} x y_i$).

## 8.2 Results

A summary of how well each model performed, as well as the largest positive and negative predictors within the model can be seen in Table 8.1.

| **Model** | **Negative Impact** | **Estimate** | **Positive Impact** | **Estimate** | **Sensitivity** | **Specificity** | **AUC** |
|---|---|---|---|---|---|---|---|
| Model 8.1 | Team 2 Vel'Koz | 0.320 | Team 1 Yorick | 3.953 | 0.643 | 0.405 | 0.535 |
| Model 8.1 LASSO | Team 1 Trundle | 0.379 | Team 2 Wukong | 1.656 | 0.612 | 0.448 | 0.543 |
| Model 8.2 | Team 2 Mage | 0.868 | Team 1 Tank | 1.140 | 0.848 | 0.190 | 0.531 |
| Model 8.2 LASSO | Team 2 Mage | 0.880 | Team 1 Controller | 1.128 | 0.861 | 0.169 | 0.529 |
| Model 8.2 interactions | Team 2 Specialist | 0.657 | Team 1 Controller | 1.824 | 0.717 | 0.314 | 0.519 |

Table 8.1: Logistic regression model comparison.

The data was divided into a training set and a test set to evaluate the prediction accuracy of my model. Using a cutoff point of 0.5 for my predictions, all the models gave similar AUC values of 0.519-0.543. These results indicate that the models are slightly better than random guessing, but none are them are exceptional prediction tools. An example of a ROC curve (from Model 8.1) can be seen in Figure 8.1. I had attempted to fit Model 8.1 to include interaction terms between the champions, but this was too computationally demanding for my laptop.

Consequently, logistic regression models were not utilised as prediction tools. Hence, the logistic function was not utilised to calculate the probability of a specific team composition winning the match. Instead, logistic regression was used to verify the underlying assumptions and conclusions derived from other methods.

Therefore, Model 8.3 was applied individually to each of the 165 champions to investigate if the assumption that the rank of the player does have an impact on the outcome. In Table 8.2, you can see for each of the chosen champions (those who had appeared multiple times in previous analyses such as win rates and association rules) which terms were significant and at what significance level. In addition, you can see the corresponding odds estimate for how they effect the outcome of

the match. For example, in the Lux model the only term that was significant was the interaction term between the champion Lux and the rank Iron. This was significant at the 0.1 level and has a corresponding odds estimate of 1.079. Hence, playing Lux at the Iron rank increases your teams chances of winning. This result is discussed in more detail in Section 9.1.

Appendix B gives a more detailed breakdown of the estimates and p-values for each of the champions in Table 8.2.

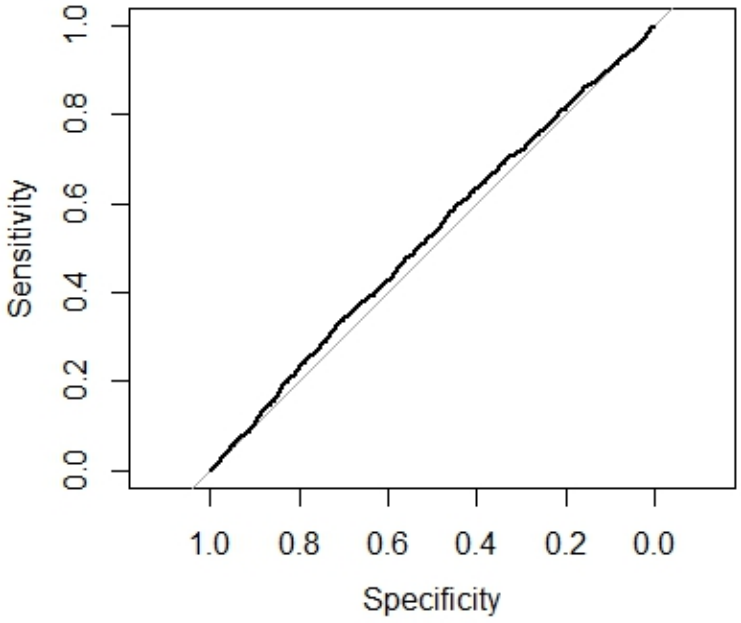


Figure 8.1: ROC curve for Model 8.1.

| **Champion** | **Term** | **Significant at** | **Odds Estimate** |
|---|---|---|---|
| Lux | Lux × Iron | 0.1 | 1.079 |
| Kai'Sa | Kai'Sa × Platinum/ Emerald | 0.1 | 0.905 |
| Corki | Corki × Iron | 0.05 | 1.804 |
| Yuumi | Yuumi | 0.001 | 0.747 |
| | Yuumi × Iron | 0.05 | 0.807 |
| | Yuumi × Silver | 0.05 | 1.275 |
| | Yuumi × Platinum/ Emerald | 0.1 | 1.246 |
| Vel'Koz | Vel'Koz × Top 4 | 0.1 | 1.305 |
| Shyvana | Shyvana × Top 4 | 0.05 | 0.605 |

Table 8.2: Significant terms for selected champions from Model 8.3.

# 9 Discussion

## 9.1 Interpretation of Results

As depicted in Table 5.1, Kai'Sa emerges as the most popular champion among high ranks, in contrast to Lux, who was most popular in lower ranks. This discrepancy suggests that Lux's perceived simplicity might make her a preferred choice for less experienced players. Consequently, Lux appears frequently in association rules without team labels (Table 6.4), except in higher ranks. Moreover, the interaction between Lux and Iron in Table 8.2 suggests her suitability for lower ranks but vulnerability to counters in higher ones.

Conversely, despite their low pick rates across ranks, champions like Corki and Skarner demonstrate unique strategic potentials. Both these champions possess unique attacking patterns that can be countered by many champions, hence they may only be reliable in certain circumstances. For instance, Corki's disproportionately high win rate in Iron (Table 5.2) might indicate specialised usage or the phenomenon of experienced players creating new accounts. However, this underscores a limitation: data from lower ranks may be skewed by players who actually belong in higher tiers, emphasising the need to focus analysis on the Top 4 for more reliable insights.

An intriguing insight from association rules Table 6.5b is the negative impact of having Yuumi on a team in lower ranks, supported by her low win rate in Iron (Table 5.2). However, in the logistic regression model shown in Table 8.2 there was a significant interaction term between Yuumi and the Silver rank, which suggests Yuumi's effectiveness increases in higher tiers, hinting at her complexity and utility at advanced levels. Yuumi plays by primarily attaching herself to another player and boosting their abilities, hence making her relatively simple to play but one challenge is knowing when to detach and when to boost the other player for maximum effect.

Overall, the ascending confidence values in association rules across ranks signify a shift from random selections to strategic decision-making as player skill progresses. Furthermore, the prevalence of rules at higher ranks underscores strategic associations over chance occurrences, affirming the decision to exclusively analyse the Top 4.

Lower ranks display a higher incidence of rules indicating champion choices leading to losses, whereas higher ranks exhibit rules suggesting champion choices resulting in victories. This contrast suggests a discerning preference for stronger champions among higher-ranked players compared to the inclination towards weaker or familiar ones among lower-ranked players.

K-means clustering (Figure 7.5) indicates interchangeability between Fighters and Slayers for similar outcomes, supported by corresponding rules (Table 6.3). A potential explanation for this is that both Fighters and Slayers possess similar characteristics in terms of high damage output, mobility, and versatility in combat. As a result, teams may find success with either class depending on their specific strategies, team compositions, and playstyles. Additionally, the overlapping roles of Fighters and Slayers in engaging and eliminating high-priority targets contribute to their interchangeability in achieving comparable outcomes within matches. Further analysis is needed to explore this further and its implications for team dynamics and strategic decision-making in League of Legends.

Similarly, the absence of Controllers in cluster 4 in Figure 7.5 prompts Tank selections for comparable match-winning probabilities. This was further evidenced by the associated rules in Table 6.3. A possible explanation for this could be the complementary roles of Tanks and Controllers in team composition. While Controllers excel at crowd control and providing utility, Tanks specialise in absorbing damage and initiating team fights. In the absence of Controllers, teams may opt for Tanks to fulfil similar functions, ensuring crowd control and frontline presence. Additionally, the strategic importance of controlling objectives and dictating the pace of team fights may underscore the preference for Tanks in achieving favourable match outcomes. Further investigation is needed to delve into the specific strategies and synergies that drive this trend in team composition.

Furthermore, Table 6.3 showed that having a Mage was the highest confidence class to imply a winning outcome. This may potentially be due to the fact that Mages excel at dealing burst damage and disrupting enemy formations with crowd control abilities. Their versatility in both offence and defence allows them to contribute significantly to team fights. Additionally, Mages often scale well with items and experience, becoming increasingly powerful as the match progresses. This effect may be emphasised in the Top 4 where matches are often longer.

Logistic regression Model 8.1, as seen in Table 8.1, reveals Vel'Koz's presence on the opposing team diminishes the odds of winning, consistent with his high win rate in the Top 4 (Table 5.2). Notably, in the logistic regression Model 8.3 in Table 8.2, the interaction between Vel'Koz and the Top 4 suggests his effectiveness increases at higher skill levels.

In summary, findings suggest champions like Lux are relatively easy to play compared to Kai'Sa, Yuumi, or Vel'Koz, aligning with [Mobalytics, 2024] ratings. Apart from Yuumi who is rated easy on the website, perhaps because her controls are relatively simple but utilising them to benefit the team is quite difficult. Moreover, the study validates the hierarchical rank system in the game, reflecting varying strategic decisions across ranks.

## 9.2 Limitations

The data collection took place during League of Legends Patches 13.22 and 13.23, which may now be considered outdated. However, while the data itself may be dated, the methodologies used remain relevant. There exists an opportunity for validation and further exploration using more current datasets. Additionally, allocating additional computational resources could enhance the extension of methods, such as incorporating interaction terms within logistic regression.

There are other game mechanics as explained in Section 2.1 such as elemental dragons that can buff the team or the use of purchasing items to enhance abilities, that could also influence the effectiveness of team composition. These were not feasible to include in this dissertation due to the increase in complexity and computational demand.

Several potential sources of bias can influence individual matches. These include instances where players create new accounts, which may not accurately reflect their true skill level, or situations where players become disconnected. Furthermore, intentional under performance or early departure from a match due to dissatisfaction can also introduce bias. For instance, if a player disconnects during a game, their opponent may gain an advantage by more easily levelling up. Moreover, deliberate actions by players to concede kills to the opposing team can distort results, potentially leading to an undeserved loss for their own team.

## 9.3 Implications and Applications

The turn-based champion selection system in League of Legends allows players to strategically counter opponents' choices, hinting at a potential rock-paper-scissors mechanic that could shape team compositions and outcomes. Delving deeper into this dynamic warrants future investigation, subject to gaining access to champion selection order data, currently inaccessible through the API. Hence, advancing the API's capabilities or employing alternative data collection methods becomes important for further exploration.

Furthermore, future research should take into account global player differences, as the current study is limited to data from a single region. It is possible that players from various regions exhibit diverse playstyles. For instance, certain regions may favour a more aggressive approach while others prioritise a defensive position. Nonetheless, the findings presented in this dissertation offer a valuable foundation for cross-regional comparisons, enabling researchers to assess potential variations in playstyles across different regions.

Furthermore, this analysis focused on the individual champions and their classes. This could be extended to sub-classes or other ways of categorising the champions.

# 10 Conclusion

In summary, this dissertation has contributed to the growing body of knowledge at the intersection of mathematics and gaming. By utilising machine learning techniques, I have uncovered valuable insights into understanding the complex strategies in competitive gaming, particularly within League of Legends.

The data gathered consisted of 96,000 matches which were evenly distributed across the different groups of ranks. For each match, the key information gathered was a list of champions played in each team and the final outcome of the match. This was then transformed into a binary dataset. Initially, the ID number of each match was retained to ensure no match was repeated. However, the data was made anonymous during the transformation to a binary dataset.

It is essential to acknowledge the data collection limitations. Factors like potential rank fluctuations during the collection period and the possibility of skewed match outcomes due to intentional poor play. Any matches containing a player that purposely played badly could have skewed a champion's win rates. However, measures were taken to address these limitations, ensuring a more balanced dataset for analysis. These included collecting the data in the shortest feasible time frame with the computational constraints and studying the ranked version of the game where theoretically there should be the fewest purposely lost matches.

The analysis used association rule learning mainly to discover associations between individual champions. Whereas, clustering was most useful for visualising the different classes of champions and logistic regression was useful for investigating the influence of rank within the game. However, a limitation of logistic regression was that it was not beneficial for prediction. Despite this it was beneficial for validating the other results.

The findings highlight several key insights. Notably, difficulty levels were identified among champions, suggesting strategic recommendations for novice players. These include champions like Lux being denoted as relatively easy to play in comparison to Kai'Sa, Yuumi, or Vel'Koz. This is relatively consistent with the difficulty ratings provided by [Mobalytics, 2024]. Moreover, the analysis underscores the significance of rank-based systems in team composition. Alongside with the interchangeable utility of certain champion classes, such as Fighters and Slayers.

Moving forward, there are many possibilities for further investigation. Subsequent studies could broaden their scope by incorporating data from multiple regions to validate the findings. Additionally, enhancing logistic models with interaction terms, given sufficient computational resources, could yield deeper insights into which champions work best together or have negative impacts on the performance

of others. Furthermore, there is potential to delve into other aspects of gameplay dynamics, such as the impact of in-game objectives like the elemental dragons or the use of items. Additionally, examining the temporal stability of the findings amidst game updates could provide valuable insights into the evolving trends of the game. If the game developers are afforded the ability to discover which champions remain strong or weak, this may shed light in the need for a significant change in their ability levels.

Beyond the academic implications, the research carries broader significance for the gaming industry. The methodologies used here could be applied to a variety of other competitive games. The findings underscore the importance of balanced game design and presents opportunities for the development of recommendation systems to enhance player experience. In conclusion, this dissertation not only advances our understanding of gaming dynamics but also validates the utility of statistical methodologies within the eSports domain. It is anticipated that the findings will stimulate further research, inspire novel approaches to data-driven analysis, and foster collaborations between mathematics and gaming communities. Ultimately, this should encourage positive advancements in strategic decision-making, game balance, and player experience within League of Legends and beyond.

# 11 Bibliography

# References


[Agrawal et al., 1993] Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining Association Rules between Sets of Items in Large Databases. *SIGMOD Rec.*, 22(2):207–216. 10, 11, 19

[Agrawal and Srikant, 1994] Agrawal, R. and Srikant, R. (1994). Fast Algorithms for Mining Association Rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499. Santiago. 11

[Bishop, 2006] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. *Springer*, 2:645–678. 10

[Black, 2005] Black, P. E. (2005). greedy algorithm. `https://www.nist.gov/dads/HTML/greedyalgo.html`. Accessed: 13/04/2024. 20

[Bobbitt, 2022] Bobbitt, Z. (2022). How to Use the Elbow Method in R to Find Optimal Clusters. `https://www.statology.org/elbow-method-in-r/`. Accessed: 02/05/2024. 31

[Burton, 2023] Burton, W. (2023). Everything You Need to Know About Team Comps and Teamfighting in League of Legends. `https://mobalytics.gg/blog/everything-you-need-to-know-about-team-comps-and-teamfighting-in-league-of-legends/`. Accessed: 02/04/2024. 4

[Cattell, 1943] Cattell, R. B. (1943). THE DESCRIPTION OF PERSONALITY: BASIC TRAITS RESOLVED INTO CLUSTERS. *Journal of Abnormal and Social Psychology*, 38(4):476–506. 10

[ClicData, 2020] ClicData (2020). How Machine Learning Can Be Used for Real-Time Product Targeting. `https://www.clicdata.com/blog/machine-learning-product-targeting/`. Accessed: 21/04/2024. 34

[Coursera Staff, 2024] Coursera Staff (2024). What Is Machine Learning? Definition, Types, and Examples. `https://www.coursera.org/articles/what-is-machine-learning`. Accessed: 13/04/2024. 9

[eSports Charts, 2024] eSports Charts (2024). LoL esports teams statistics. `https://escharts.com/teams/lol`. Accessed: 02/04/2024. 3

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press. 12

[Estivill-Castro, 2002] Estivill-Castro, V. (2002). Why so many clustering algorithms - a Position Paper. *SIGKDD Explor. Newsl.*, 4(1):65–75. 12

[Freedman, 2009] Freedman, D. A. (2009). *Statistical Models: Theory and Practice*. Cambridge University Press. 10

[García et al., 2007] García, E., Romero, C., Ventura, S., and Calders, T. (2007). Drawbacks and solutions of applying association rule mining in learning management systems. *CEUR Workshop Proceedings*, 305:13–22. 11

[Hahsler et al., 2023] Hahsler, M., Buchta, C., Gruen, B., and Hornik, K. (2023). *arules: Mining Association Rules and Frequent Itemsets*. 20

[Hahsler et al., 2005] Hahsler, M., Gruen, B., and Hornik, K. (2005). arules – A Computational Environment for Mining Association Rules and Frequent Item Sets. *Journal of Statistical Software*, 14(15):1–25. 20

[Han et al., 2000] Han, J., Pei, J., and Yin, Y. (2000). Mining FrequentPatterns without Candidate Generation. *SIGMOD Rec.*, 29(2):1–12. 11

[Hodge et al., 2021] Hodge, V. J., Devlin, S., Sephton, N., Block, F., Cowling, P. I., and Drachen, A. (2021). Win Prediction in Multiplayer Esports: Live Professional Match Prediction. *Transactions on Games*, 13(4):368–379. 10, 12

[Hosmer and Lemeshow, 2000] Hosmer, D. W. and Lemeshow, S. (2000). *Applied Logistic Regression*. Wiley New York, 2nd edition. 13, 14

[James et al., 2013] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning*, volume 112. Springer. 13

[Jarman, 2020] Jarman, A. M. (2020). Hierarchical Cluster Analysis: Comparison of Single linkage,Complete linkage, Average linkage and Centroid Linkage Method. *Georgia Southern University*, 29. 32

[Kipp et al., 2018] Kipp, K., Giordanelli, M., and Geiser, C. (2018). Predicting net joint moments during a weightlifting exercise with a neural network model. *Journal of Biomechanics*, 74:225–229. 10

[Kumbhare and Chobe, 2014] Kumbhare, T. A. and Chobe, S. V. (2014). An Overview of Association Rule Mining Algorithms. *International Journal of Computer Science and Information Technologies*, 5(1):927–930. 11

[League of Legends Wiki, 2024a] League of Legends Wiki (2024a). Champion Classes. `https://leagueoflegends.fandom.com/wiki/Champion_classes`. Accessed: 13/04/2024. 55

[League of Legends Wiki, 2024b] League of Legends Wiki (2024b). Map (League of Legends). https://leagueoflegends.fandom.com/wiki/Map_(League_of_Legends). Accessed: 10/02/2024. 6

[LoLTheory, 2024] LoLTheory (2024). Team Comp Analyzer. https://loltheory.gg/lol/team-comp-analyzer. Accessed: 02/04/2024. 9

[lolvvv, 2023] lolvvv (2023). League of Legends Player Count: Today and Through the Years. https://www.lolvvv.com/blog/league-of-legends-player-count. Accessed: 10/02/2024. 4

[MacKay, 2005] MacKay, D. J. (2005). *Information Theory, Inference, and Learning Algorithms*, chapter 20, pages 285–289. Cambridge University Press, Cambridge, UK, Version 7.2 (fourth printing) edition. 30

[MacQueen, 1967] MacQueen, J. (1967). SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA. 12

[Metwally et al., 2005] Metwally, A., Agrawal, D., and El Abbadi, A. (2005). Using Association Rules for Fraud Detection in Web Advertising Networks. In *VLDB*, volume 5, pages 169–180. 11

[Milella, 2024] Milella, V. (2024). LEAGUE OF LEGENDS RANK DISTRIBUTION IN SOLO QUEUE - APRIL 2024. https://www.esportstales.com/league-of-legends/rank-distribution-percentage-of-players-by-tier. Accessed: 21/04/2024. 16

[Mobalytics, 2024] Mobalytics (2024). League of Legends: Summoner Stats, Match History and Champions Builds. https://mobalytics.gg/lol. Accessed: 02/04/2024. 5, 43, 45

[Nahar et al., 2013] Nahar, J., Imam, T., Tickle, K. S., and Chen, Y.-P. P. (2013). Association rule mining to detect factors which contribute to heart disease in males and females. *Expert Systems with Applications*, 40(4):1086–1093. 11

[Nielsen, 2016] Nielsen, F. (2016). *Hierarchical Clustering*, pages 195–211. Springer International Publishing. 12

[OPGG, 2024] OPGG (2024). OP.GG - Multiplayer Online Battle arena (MOBA) Statistics and Analytics. https://www.op.gg. Accessed: 02/04/2024. 5

[Peña et al., 1999] Peña, J., Lozano, J., and Larrañaga, P. (1999). An empirical comparison of four initialization methods for the K-Means algorithm. *Pattern Recognition Letters*, 20(10):1027–1040. 12

[Piatetsky-Shapiro, 1991] Piatetsky-Shapiro, G. (1991). Discovery, Analysis, and Presentation of Strong Rules. In *Knowledge Discovery in Databases*. AAAI/MIT Press. 10

[Prasetio and Harlili, 2016] Prasetio, D. and Harlili, D. (2016). Predicting Football Match Results with Logistic Regression. In *2016 International Conference On Advanced Informatics*, pages 1–5. 10

[R Core Team, 2023] R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Austria. 5

[Riot Games, 2024a] Riot Games (2024a). League of Legends Patch Notes. `https://www.leagueoflegends.com/en-us/news/tags/patch-notes/`. Accessed: 02/05/2024. 14

[Riot Games, 2024b] Riot Games (2024b). Riot Games API. `https://developer.riotgames.com/apis`. 14

[Rokach and Maimon, 2014] Rokach, L. and Maimon, O. (2014). *Data Mining with Decision Trees*. WORLD SCIENTIFIC, 2nd edition. 10

[Shewale, 2024] Shewale, R. (2024). eSports Statistics In 2024 (Viewers, Growth & Forecast). `https://www.demandsage.com/esports-statistics/`. Accessed: 13/04/2024. 3

[Statista, 2024] Statista (2024). Statistics report on League of Legends. `https://www.statista.com/study/51642/league-of-legends/`. Accessed: 02/04/2024. 3

[Suhaeri et al., 2021] Suhaeri, M. E., Alimudin, Javaid, A., Ismail, M. T., and Ali, M. K. M. (2021). Evaluation of clustering approach with euclidean and manhattan distance for outlier detection. In *AIP Conference Proceedings*, volume 2423, Melville. American Institute of Physics. 29

[Tan et al., 2005] Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining, (First Edition)*, chapter 6, pages 327–414. Addison-Wesley Longman Publishing Co., Inc., USA. 22

[Wickham and Grolemund, 2017] Wickham, H. and Grolemund, G. (2017). Tidy Data. In *R for Data Science*, chapter 12. O'Reilly Media, 1st edition. 17

[Yang et al., 2012] Yang, X., Liu, Z., and Xinxiang, H. (2012). The Application of Association Rules Mining in Building Intelligent Transportation Systems. *Journal of Convergence Information Technology*, 7:575–582. 11

[Yang et al., 2016] Yang, Y., Qin, T., and Lei, Y.-H. (2016). Real-time eSports Match Result Prediction. *arXiv preprint arXiv:1701.03162*. 13

# A Champion Information Table

| Champion Name | Champion ID | Class | Sub-Class | Position |
|---|---|---|---|---|
| Aatrox | 266 | Fighter | Juggernaut | Top |
| Ahri | 103 | Mage | Burst | Mid |
| Akali | 84 | Slayer | Assassin | Mid |
| Akshan | 166 | Marksman | Marksman | Mid |
| Alistar | 12 | Tank | Vanguard | Support |
| Amumu | 32 | Tank | Vanguard | Jungle |
| Anivia | 34 | Mage | Battlemage | Mid |
| Annie | 1 | Mage | Burst | Mid |
| Aphelios | 523 | Marksman | Marksman | Bot |
| Ashe | 22 | Marksman | Marksman | Bot |
| Aurelion Sol | 136 | Mage | Battlemage | Mid |
| Azir | 268 | Specialist | Specialist | Mid |
| Bard | 432 | Controller | Catcher | Support |
| Belveth | 200 | Slayer | Skirmisher | Jungle |
| Blitzcrank | 53 | Controller | Catcher | Support |
| Brand | 63 | Mage | Burst | Support |
| Braum | 201 | Tank | Warden | Support |
| Briar | 233 | Fighter | Diver | Jungle |
| Caitlyn | 51 | Marksman | Marksman | Bot |
| Camille | 164 | Fighter | Diver | Top |
| Cassiopeia | 69 | Mage | Battlemage | Mid |
| Cho'Gath | 31 | Specialist | Specialist | Top |
| Corki | 42 | Marksman | Marksman | Mid |
| Darius | 122 | Fighter | Juggernaut | Top |
| Diana | 131 | Fighter | Diver | Jungle |
| Dr. Mundo | 119 | Fighter | Juggernaut | Top |
| Draven | 36 | Marksman | Marksman | Bot |
| Ekko | 245 | Slayer | Assassin | Jungle |
| Elise | 60 | Fighter | Diver | Jungle |
| Evelynn | 28 | Slayer | Assassin | Jungle |
| Ezreal | 81 | Marksman | Marksman | Bot |
| Fiddlesticks | 9 | Specialist | Specialist | Jungle |
| Fiora | 114 | Slayer | Skirmisher | Top |
| Fizz | 105 | Slayer | Assassin | Mid |
| Galio | 3 | Tank | Warden | Support |
| Gangplank | 41 | Specialist | Specialist | Top |
| Garen | 86 | Fighter | Juggernaut | Top |
| Gnar | 150 | Specialist | Specialist | Top |

| Gragas | 79 | Tank | Vanguard | Top |
|---|---|---|---|---|
| Graves | 104 | Specialist | Specialist | Jungle |
| Gwen | 887 | Slayer | Skirmisher | Top |
| Hecarim | 120 | Fighter | Diver | Jungle |
| Heimerdinger | 74 | Specialist | Specialist | Mid |
| Illaoi | 420 | Fighter | Juggernaut | Top |
| Irelia | 39 | Fighter | Diver | Top |
| Ivern | 427 | Controller | Catcher | Jungle |
| Janna | 40 | Controller | Enchanter | Support |
| Jarvan IV | 59 | Fighter | Diver | Support |
| Jax | 24 | Slayer | Skirmisher | Top |
| Jayce | 126 | Mage | Artillery | Top |
| Jhin | 202 | Controller | Catcher | Bot |
| Jinx | 222 | Marksman | Marksman | Bot |
| Kai'Sa | 145 | Marksman | Marksman | Bot |
| Kalista | 429 | Marksman | Marksman | Bot |
| Karma | 43 | Controller | Enchanter | Support |
| Karthus | 30 | Mage | Battlemage | Jungle |
| Kassadin | 38 | Slayer | Assassin | Mid |
| Katarina | 55 | Slayer | Assassin | Mid |
| Kayle | 10 | Specialist | Specialist | Top |
| Kayn | 141 | Slayer | Skirmisher | Jungle |
| Kennen | 85 | Specialist | Specialist | Top |
| Kha'Zix | 121 | Slayer | Assassin | Jungle |
| Kindred | 203 | Marksman | Marksman | Jungle |
| Kled | 240 | Slayer | Skirmisher | Top |
| Kog'Maw | 96 | Marksman | Marksman | Bot |
| K'sante | 897 | Slayer | Skirmisher | Top |
| LeBlanc | 7 | Mage | Burst | Mid |
| Lee Sin | 64 | Fighter | Diver | Jungle |
| Leona | 89 | Tank | Vanguard | Support |
| Lillia | 876 | Slayer | Skirmisher | Jungle |
| Lissandra | 127 | Mage | Burst | Mid |
| Lucian | 236 | Marksman | Marksman | Bot |
| Lulu | 117 | Controller | Enchanter | Support |
| Lux | 99 | Mage | Artillery | Support |
| Malphite | 54 | Tank | Vanguard | Top |
| Malzahar | 90 | Mage | Battlemage | Mid |
| Maokai | 57 | Tank | Vanguard | Top |
| Master Yi | 11 | Slayer | Skirmisher | Jungle |
| Milio | 902 | Controller | Enchanter | Support |

| Miss Fortune | 21 | Marksman | Marksman | Bot |
|---|---|---|---|---|
| Wukong | 62 | Fighter | Diver | Jungle |
| Mordekaiser | 82 | Fighter | Juggernaut | Top |
| Morgana | 25 | Controller | Catcher | Support |
| Naafiri | 950 | Slayer | Assassin | Mid |
| Nami | 267 | Controller | Enchanter | Support |
| Nasus | 75 | Fighter | Juggernaut | Top |
| Nautilus | 111 | Tank | Vanguard | Support |
| Neeko | 518 | Mage | Burst | Support |
| Nidalee | 76 | Specialist | Specialist | Jungle |
| Nilah | 895 | Slayer | Skirmisher | Bot |
| Nocturne | 56 | Slayer | Assassin | Jungle |
| Nunu & Willump | 20 | Tank | Vanguard | Jungle |
| Olaf | 2 | Fighter | Diver | Top |
| Orianna | 61 | Mage | Burst | Mid |
| Ornn | 516 | Tank | Vanguard | Top |
| Pantheon | 80 | Fighter | Diver | Support |
| Poppy | 78 | Tank | Warden | Jungle |
| Pyke | 555 | Controller | Catcher | Support |
| Qiyana | 246 | Slayer | Assassin | Mid |
| Quinn | 133 | Specialist | Specialist | Top |
| Rakan | 497 | Controller | Catcher | Support |
| Rammus | 33 | Tank | Vanguard | Jungle |
| Rek'Sai | 421 | Fighter | Diver | Jungle |
| Rell | 526 | Tank | Vanguard | Support |
| Renata | 888 | Controller | Enchanter | Support |
| Renekton | 58 | Fighter | Diver | Top |
| Rengar | 107 | Fighter | Diver | Jungle |
| Riven | 92 | Slayer | Skirmisher | Top |
| Rumble | 68 | Mage | Battlemage | Top |
| Ryze | 13 | Mage | Battlemage | Mid |
| Samira | 360 | Marksman | Marksman | Bot |
| Sejuani | 113 | Tank | Vanguard | Jungle |
| Senna | 235 | Controller | Enchanter | Support |
| Seraphine | 147 | Controller | Enchanter | Support |
| Sett | 875 | Fighter | Juggernaut | Top |
| Shaco | 35 | Slayer | Assassin | Jungle |
| Shen | 98 | Tank | Warden | Top |
| Shyvana | 102 | Fighter | Juggernaut | Jungle |
| Singed | 27 | Specialist | Specialist | Top |
| Sion | 14 | Tank | Vanguard | Top |

| Sivir | 15 | Marksman | Marksman | Bot |
|---|---|---|---|---|
| Skarner | 72 | Tank | Vanguard | Jungle |
| Sona | 37 | Controller | Enchanter | Support |
| Soraka | 16 | Controller | Enchanter | Support |
| Swain | 50 | Mage | Battlemage | Support |
| Sylas | 517 | Mage | Burst | Mid |
| Syndra | 134 | Mage | Burst | Mid |
| Tahm Kench | 223 | Tank | Warden | Top |
| Taliyah | 163 | Mage | Battlemage | Mid |
| Talon | 91 | Slayer | Assassin | Mid |
| Taric | 44 | Controller | Enchanter | Support |
| Teemo | 17 | Specialist | Specialist | Top |
| Thresh | 412 | Controller | Catcher | Support |
| Tristana | 18 | Marksman | Marksman | Bot |
| Trundle | 48 | Fighter | Juggernaut | Jungle |
| Tryndamere | 23 | Slayer | Skirmisher | Top |
| Twisted Fate | 4 | Mage | Burst | Mid |
| Twitch | 29 | Marksman | Marksman | Bot |
| Udyr | 77 | Fighter | Juggernaut | Jungle |
| Urgot | 6 | Fighter | Juggernaut | Top |
| Varus | 110 | Mage | Artillery | Bot |
| Vayne | 67 | Marksman | Marksman | Bot |
| Veigar | 45 | Mage | Burst | Mid |
| Vel'Koz | 161 | Mage | Artillery | Support |
| Vex | 711 | Mage | Burst | Mid |
| Vi | 254 | Fighter | Diver | Jungle |
| Viego | 234 | Slayer | Skirmisher | Jungle |
| Viktor | 112 | Mage | Battlemage | Mid |
| Vladimir | 8 | Mage | Battlemage | Mid |
| Volibear | 106 | Fighter | Juggernaut | Top |
| Warwick | 19 | Fighter | Diver | Jungle |
| Xayah | 498 | Marksman | Marksman | Bot |
| Xerath | 101 | Mage | Artillery | Support |
| Xin Zhao | 5 | Fighter | Diver | Jungle |
| Yasuo | 157 | Slayer | Skirmisher | Mid |
| Yone | 777 | Slayer | Skirmisher | Mid |
| Yorick | 83 | Fighter | Juggernaut | Top |
| Yuumi | 350 | Controller | Enchanter | Support |
| Zac | 154 | Tank | Vanguard | Jungle |
| Zed | 238 | Slayer | Assassin | Mid |
| Zeri | 221 | Marksman | Marksman | Bot |

| | | | | |
|---|---|---|---|---|
| Ziggs | 115 | Mage | Artillery | Bot |
| Zilean | 26 | Specialist | Specialist | Support |
| Zoe | 142 | Mage | Burst | Mid |
| Zyra | 143 | Controller | Catcher | Support |

This information was originally gathered in December 2023 and was verified on 21/04/2024 using [League of Legends Wiki, 2024a]. Those with multiple classes or positions were categorised based on which is more common in my experience.

## A.1 Explanation of Sub-Classes

In addition to the previously mentioned classes in Section 2.2, there are sub-classes. For Controllers these are Catchers and Enchanters. Catchers specialise in crowd control and map pressure by finding opportunities to catch enemies out. Whereas, Enchanters focus on amplifying their allies by temporarily increasing their statistics or healing and shielding allies to increase their survivability. Secondly, in the Fighter class, there are Juggernauts and Divers. Juggernauts are melee focused bruisers that run down the enemy with the ability to deal and receive significant amounts of damage but they lack mobility. On the other hand, Divers can focus on high priority targets by having the manoeuvrability that the Juggernaut lacks while lacking the durability. Thirdly, the Mage class is split up into Artillery, Burst and Battlemages. Artillery Mages have extensive range for their spells but can easily be punished if an enemy gets too close to them. Burst mages will focus on targeting vulnerable enemies with abilities that can quickly kill the opponent. Battlemages are often on the frontline of fights to provide sustained damage on the enemy team. Fourth, Slayers have sub-classes of Assassins and Skirmishers. Assassins focus on infiltrating the enemy frontline with their mobility and often have the ability to quickly disengage or negate incoming damage. On the other hand, Skirmishers have the ability to cut down enemies looking to duel them and use defensive abilities to push through the frontline. Finally, Tanks have sub-classes of Vanguard and Wardens. Vanguards are offensive and initiate team fights due to their ability to take a large amount of damage. On the other hand, Wardens are defensive champions who hold down the line and defend their more vulnerable allies. Both Marksmen and Specialists possess no sub-classes.

# B Logistic Regression Model 8.3 Tables

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | -0.005 | 0.995 | 0.691 |
| Lux | 0.034 | 1.035 | 0.287 |
| Iron | -0.011 | 0.989 | 0.505 |
| Silver | -0.002 | 0.998 | 0.913 |
| Gold | 0.008 | 1.009 | 0.615 |
| Platinum/Emerald | 0.004 | 1.004 | 0.812 |
| Top 4 | 0.005 | 1.005 | 0.748 |
| Lux × Iron | 0.076 | 1.079 | 0.091 |
| Lux × Silver | 0.018 | 1.019 | 0.692 |
| Lux × Gold | -0.070 | 0.932 | 0.151 |
| Lux × Platinum/Emerald | -0.024 | 0.976 | 0.647 |
| Lux × Top 4 | -0.053 | 0.948 | 0.483 |

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | 0.001 | 1.001 | 0.930 |
| Kai'Sa | -0.012 | 0.988 | 0.766 |
| Iron | 0.000 | 1.000 | 0.994 |
| Silver | 0.008 | 1.008 | 0.622 |
| Gold | 0.003 | 1.003 | 0.879 |
| Platinum/Emerald | 0.012 | 1.012 | 0.490 |
| Top 4 | 0.005 | 1.005 | 0.757 |
| Kai'Sa × Iron | -0.002 | 0.998 | 0.974 |
| Kai'Sa × Silver | -0.091 | 0.913 | 0.102 |
| Kai'Sa × Gold | -0.022 | 0.978 | 0.685 |
| Kai'Sa × Platinum/Emerald | -0.100 | 0.905 | 0.060 |
| Kai'Sa × Top 4 | -0.030 | 0.970 | 0.553 |

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | 0.000 | 1.000 | 0.969 |
| Corki | -0.145 | 0.865 | 0.477 |
| Iron | -0.002 | 0.998 | 0.918 |
| Silver | -0.001 | 0.999 | 0.943 |
| Gold | 0.000 | 1.000 | 0.991 |
| Platinum/Emerald | 0.001 | 1.001 | 0.965 |
| Top 4 | 0.000 | 1.000 | 0.994 |
| Corki × Iron | 0.590 | 1.804 | 0.049 |
| Corki × Silver | 0.378 | 1.460 | 0.193 |
| Corki × Gold | 0.088 | 1.092 | 0.738 |
| Corki × Platinum/Emerald | -0.074 | 0.929 | 0.774 |
| Corki × Top 4 | -0.009 | 0.991 | 0.977 |

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | 0.006 | 1.006 | 0.576 |
| Yuumi | -0.292 | 0.747 | 0.000 |
| Iron | 0.009 | 1.010 | 0.554 |
| Silver | -0.005 | 0.995 | 0.734 |
| Gold | -0.002 | 0.998 | 0.913 |
| Platinum/Emerald | -0.005 | 0.995 | 0.755 |
| Top 4 | -0.004 | 0.996 | 0.799 |
| Yuumi × Iron | -0.215 | 0.807 | 0.034 |
| Yuumi × Silver | 0.243 | 1.275 | 0.033 |
| Yuumi × Gold | 0.018 | 1.019 | 0.874 |
| Yuumi × Platinum/Emerald | 0.220 | 1.246 | 0.053 |
| Yuumi × Top 4 | -0.106 | 0.899 | 0.530 |

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | 0.000 | 1.000 | 0.986 |
| Vel'Koz | 0.009 | 1.009 | 0.907 |
| Iron | -0.001 | 0.999 | 0.975 |
| Silver | -0.001 | 0.999 | 0.955 |
| Gold | 0.000 | 1.000 | 0.997 |
| Platinum/Emerald | 0.001 | 1.001 | 0.930 |
| Top 4 | -0.002 | 0.998 | 0.877 |
| Vel'Koz × Iron | 0.030 | 1.030 | 0.795 |
| Vel'Koz × Silver | 0.043 | 1.043 | 0.699 |
| Vel'Koz × Gold | 0.005 | 1.005 | 0.967 |
| Vel'Koz × Platinum/Emerald | -0.072 | 0.930 | 0.522 |
| Vel'Koz × Top 4 | 0.266 | 1.305 | 0.055 |

| Term | Log Odds | Odds Estimate | P-value |
|---|---|---|---|
| Intercept | -0.001 | 0.999 | 0.901 |
| Shyvana | 0.066 | 1.068 | 0.396 |
| Iron | 0.000 | 1.000 | 0.997 |
| Silver | 0.000 | 1.000 | 0.990 |
| Gold | 0.002 | 1.002 | 0.914 |
| Platinum/Emerald | -0.001 | 0.999 | 0.965 |
| Top 4 | 0.003 | 1.003 | 0.873 |
| Shyvana × Iron | 0.008 | 1.008 | 0.943 |
| Shyvana × Silver | 0.004 | 1.004 | 0.974 |
| Shyvana × Gold | -0.081 | 0.922 | 0.462 |
| Shyvana × Platinum/Emerald | 0.061 | 1.063 | 0.602 |
| Shyvana × Top 4 | -0.502 | 0.605 | 0.034 |