Unveiling META Champions in League of Legends: A Data-Driven Analysis of the 13.24 Patch

Julius Maliwat[a]

[a] *University of Milano-Bicocca, Data Science*

Abstract

In the realm of League of Legends, a competitive 5v5 game, achieving balance among numerous champions, items, and evolving game dynamics is a significant challenge. This study focuses on the 13.24 patch, aiming to identify META champions—those exhibiting superior performance. By leveraging scraping techniques and APIs from Lolalytics, Blitz.gg, Op.gg, and Riot Games, we collected and integrated data with a focus on consistency. The resulting analysis successfully identifies META champions, providing valuable insights into the intricate dynamics of this gaming environment.

Contents

1. Introduction

League of Legends (LoL), a highly popular online multiplayer game, is characterized by its strategic team-based gameplay. In this competitive arena, two teams of five players each strive to destroy the opposing team's Nexus, a central structure located within their respective bases. LoL stands out for its diverse array of champions, each with unique abilities, roles, and playstyles (1). Success in this game hinges on effective teamwork, strategic decision-making, and individual player skill.

The game's complexity is largely attributed to the extensive selection of champions and in-game items, coupled with the dynamic nature of its gaming environment. Regular updates, which introduce new champions and modify existing in-game items, pose a significant challenge in maintaining game balance. These updates, known as game patches, often shift the relative strength of champions, leading to the emergence of a "META", an acronym for "Most Effective Tactics Available". The META represents the dominant strategies, champion selections, and item builds deemed most effective in the current game version. It is a dynamic concept, evolving with each game patch or balance update, and reflects the changing strategic landscape of the game. Understanding the META is vital for players and teams as it guides them in adapting their gameplay to the most successful strategies, thereby gaining a competitive edge.

This project aims to utilize web scraping and API data retrieval techniques to gather comprehensive information about champions in League of Legends, with a particular focus on the 13.24 patch. The objective is to identify and analyze the META champions within the current gaming environment. Recognizing META champions provides players with a strategic advantage, enhancing their chances of success by aligning their gameplay with the most powerful and influential characters in the latest game version. The project endeavors to equip players with critical insights, aiding them in navigating the dynamic and evolving world of League of Legends.

Our approach to this project involves a structured progression in data management. We start with the data acquisition phase, employing web scraping and API data retrieval methods to compile a detailed dataset for the 13.24 patch of League of Legends. In the data integration phase, we amalgamate information from multiple sources, creating a comprehensive dataset for analyzing the in-game performance of champions. This is followed by data cleaning, where we refine the dataset to enhance its quality and coherence, ensuring accuracy in our analysis. We place a particular emphasis on data quality and improvement, focusing on completeness and consistency to establish a robust dataset. For data storage, we adopt a relational database model, which allows for efficient organization and retrieval of relevant information. Finally, structured queries are developed to extract insightful information about META champions, their roles, and effective counter-strategies.

This project aspires to provide an in-depth understanding of META champions in League of Legends, offering players a strategic advantage in this constantly evolving game.

2. Software Architecture

In this project, we employed a suite of technologies and libraries to manage, analyze, and store data effectively. Centered around Python, our tech stack was carefully chosen for its versatility and strength in data science applications:

- Python: the primary programming language for the project, ideal for its extensive libraries and data handling capabilities.
- Pandas: Crucial for data cleaning and table operations, Pandas in Python is known for its powerful data manipulation and analysis features.
- BeautifulSoup: A Python library used for web scraping, essential for parsing HTML and XML documents to extract data from websites like Lolalytics.
- Selenium: Employed for automating web browser interactions, particularly useful in managing dynamic content during our web scraping process.
- MySQL Connector: This Python library facilitated interactions between Python and MySQL databases, allowing us to execute SQL operations directly from Python scripts.
- MySQL Workbench: Used for database management and design, MySQL Workbench provided a visual interface for working with our MySQL databases, enhancing the ease of managing and querying our stored data.

Together, these technologies formed a robust foundation for our project, enabling efficient handling and analysis of complex datasets and contributing significantly to our ability to extract meaningful insights

3. Data Acquisition

This phase involves a systematic collection of relevant data, ensuring a comprehensive and diverse dataset for subsequent in-depth analysis. In this project, we have employed two primary methods of data acquisition: utilizing Application Programming Interfaces (APIs) and implementing web scraping techniques.

3.1. API from Riot Games

An API (Application Programming Interface) acts as a bridge between different software applications, allowing them to communicate and share data seamlessly. Riot Games, the developer of League of Legends, provides such an API, offering access to a wealth of game-related data (2).

For the purpose of our project, we utilized the Riot API to fetch the official list of League of Legends champions along with their basic statistics, which are not directly linked to in-game match performances. The data retrieved from the Riot API is presented in JSON format, featuring a hierarchical structure. In this structure, each champion's name serves as a key, linked to a dictionary containing detailed information about that champion. This includes essential attributes such as the champion's name, ID, description, roles, type, basic difficulty level, and base statistics, like attack, defense, and speed.

To facilitate subsequent analysis, we transformed the nested dictionary structure into a more manageable format. This involved flattening the hierarchical data, converting each champion's detailed information into a list. This process yielded a structured table with a single-tier format, encompassing all the necessary data in a clear and concise manner. The resulting table includes 30 columns, each representing different attributes of the champions' basic information.

The final dataset, named champion_list.csv, was prepared using the pandas library, a powerful tool in Python for data manipulation and analysis. This dataset is comprehensive, containing 30 columns that encapsulate various aspects of the champions' basic profiles. It consists of 166 rows, corresponding to the total number of champions available in League of Legends at the time of the study.

3.2. Scraping

In our project, scraping techniques were employed to augment the dataset with diverse and detailed information about League of Legends champions. Scraping, a common practice in web development, entails the automated extraction of data from websites. This process requires parsing the HTML markup of web pages to retrieve relevant data for further analysis.

For the implementation of scraping, the BeautifulSoup library in Python was chosen for its powerful and user-friendly capabilities in parsing HTML and XML files. In instances where web pages involved dynamic content loading, Selenium, a Python library designed for automating web applications, was used. This combination allowed for efficient extraction of data even from web pages that required interactive elements to be triggered for data access.

The websites targeted for scraping included Lolalytics, Op.gg, and Blitz.gg. Each of these platforms provides unique and valuable insights into champion statistics, playing patterns, and player preferences, which are critical for a comprehensive understanding of the game's META. The scraping process involved systematically navigating these websites, extracting key information such as champion win rates, pick and ban rates, and player rankings, and then methodically organizing this data to complement the information obtained from the Riot Games API.

3.2.1. LoLalytics

LoLalytics, a prominent online platform for League of Legends statistics, played a significant role in the data acquisition process of our project (3). It stands out for its comprehensive analysis of in-game champion statistics in ranked matches. For our purposes, LoLalytics offered invaluable data, particularly regarding how champions perform in different lanes and across various game ranks.

The platform categorizes champion statistics based on their lane assignments (top, jungle, mid, adc, support) and the rank of the game (ranging from 'all' to specific tiers like 'challenger', 'master', 'diamond', 'emerald', 'platinum', 'gold', 'silver', 'bronze', and 'iron'). This level of detail was crucial for our analysis, allowing us to understand the nuances of champion performance in diverse gaming scenarios.

To effectively scrape data from LoLalytics, we utilized the Python library BeautifulSoup for parsing HTML content. Additionally, Selenium was employed to manage dynamic content on the website, ensuring accurate data retrieval from dynamically loaded tables.

The scraping procedure entailed making HTTP requests to various URLs on LoLalytics, each differing by lane and rank parameters. This approach enabled us to gather a wide range of data across all possible combinations of lanes and ranks. For each lane and rank combination, we scraped a tier list dataset of champions. To ensure a comprehensive analysis, we concatenated rows from these individual datasets, resulting in an extensive, unified dataset.

Key metrics extracted from LoLalytics included:

- Grade:The performance rating assigned by LoLalytics to each champion, considering their effectiveness in the specified rank and lane.
- Champion Name: The name of the champion for which data was collected.
- Position: The position of the champion within the tier list for the specified rank and lane;
- Lane: Indicates the lane in which the champion was played (e.g., top, jungle, mid, adc, support);
- Win Rate: The percentage of games won by the champion under the specified conditions.
- Pick Rate: The frequency with which the champion was chosen by players;
- Ban Rate: The frequency with which the champion was banned in games;
- Games Played: The total number of games in which the champion participated in the specified rank and lane;
- Ranking: Indicates the Ranking in which the champion was played (Challenger, Master, diamond, etc.)

The culmination of this process was the creation of a comprehensive dataset, saved as a .csv file. The dataset, with 3656 rows, was the result of 50 distinct URL calls, each representing a unique combination of lane and rank. Each row captures the detailed in-game performance metrics of a specific champion based on its designated lane and ranking.

3.2.2. op.gg

Op.gg stands as a pivotal resource in the League of Legends community, renowned for its extensive player and champion statistics (4). In our project, op.gg was crucial for gathering data on primary counters and favorable matchups for each champion, a key aspect of strategic gameplay.

The data collection from op.gg was meticulously planned to align with the combinations of lanes, rankings, and champions previously identified in LoLalytics, resulting in a total of 3656 unique requests. However, this high frequency of API calls presented a significant challenge, often leading to server errors, primarily HTTP status codes 429 (Too Many Requests) and 502 (Bad Gateway). These errors were indicative of the heavy load imposed on op.gg's servers by our extensive querying. To address this issue and streamline the scraping process, we implemented a strategic timeout mechanism. Upon encountering errors like 429 or 502, the scraping program would automatically pause for a predefined period. This approach not only alleviated the stress on op.gg's servers but also ensured a more efficient and uninterrupted data collection process.

The data extracted from op.gg was particularly focused on identifying the main counters and favorable matchups for each champion. The specific fields collected were:

- Grade: The overall performance rating of a champion as determined by op.gg.
- Counter: A list of champions that are considered effective counters to the specific champion.
- Win Rate Weak: a list of Win rates when the champion faces its counters, as listed in the 'Counter' column.
- Games Played Weak: Represents a list indicating the number of games played by the specific champion against its counters, as listed in the 'Counter' column;
- Strong Against: A list of champions that are typically weaker against the specific champion.
- Win Rate Strong: a list of Win rates when the champion faces those listed in the 'Strong Against' column.
- Games Played Strong: Represents a list indicating the number of games played by the specific champion against the campions listed in the 'Strong Against' column;
- Champion Name: Identifies the specific champion for which statistics were gathered.
- Ranking: Indicates the Ranking in which the champion was played (Challenger, Master, diamond, etc.)
- Lane: Indicates the lane in which the champion was played (e.g., top, jungle, mid, adc, support);

It's important to note that the dataset, in its current form, represents list-type data as strings, for instance, "['element1', 'element2']". A crucial step in the upcoming data cleaning phase will be to parse and transform these string representations into a structured format conducive to analysis. The final step in the scraping process from op.gg was the compilation and storage of the data in a .csv file format. This dataset mirrors the structure obtained from LoLalytics, encompassing 3656 rows across 10 different columns, each offering valuable insights into champion matchups and performance metrics

3.2.3. blitz.gg

Blitz.gg, a renowned online platform, offers key insights and tools to enhance gameplay in League of Legends (5). In our project, we leveraged blitz.gg to gather data on champion tier classifications, counters, and win rates, providing valuable insights for strategic gameplay analysis.

The scraping methodology for blitz.gg paralleled the approaches used for LoLalytics and op.gg. We focused on collecting data across various combinations of lanes, rankings, and champions, as identified earlier in LoLalytics. Utilizing Beautiful Soup for direct HTML parsing was the primary method for scraping blitz.gg, thanks to its efficient loading times and the minimal issues encountered with the volume of requests. Nevertheless, we occasionally faced response failures, prompting the implementation of a retry mechanism to ensure thorough data collection.

The key pieces of information extracted from blitz.gg are as follows:

- Champion Name: Identifies the specific champion for which statistics were gathered;
- Ranking: Indicates the Ranking in which the champion was played (Challenger, Master, diamond, etc.);
- Lane: Indicates the lane in which the champion was played (e.g., top, jungle, mid, adc, support);
- Grade: Represents the overall grade assigned to a champion by blitz.gg based on its performance;
- Counters List: Lists the champions that are considered counters to the specific champion;
- Win Rate List: Represents a list of win rates corresponding to the champion when facing its respective counters listed in the 'Counter List' column;

It is important to note that the 'Counters List' and 'Win Rate List' columns are currently formatted as strings representing lists. The upcoming data cleaning phase will include parsing these string formats into structured data that can be effectively analyzed.

The final dataset from blitz.gg was saved as a .csv file. This file, encompassing tier information, counters, win rates against these counters, and role specifics, consists of 3656 rows and 6 columns.

4. Data Integration

Data integration involves the combination of data from diverse sources into a unified, coherent dataset. This process is essential for providing a comprehensive view of the data, enabling more accurate and insightful analysis.

In our project, the data integration phase focused on harmonizing the datasets obtained from Lolalytics, op.gg, and blitz.gg. The primary key for merging these datasets was based on three fields: the champion, the lane, and the ranking. However, this integration faced challenges, particularly regarding the data columns related to counters. The 'Counter' and 'Win Rate Weak' columns from op.gg and the 'Counters List' and 'Win Rate List' from blitz.gg presented overlapping information, necessitating a conflict resolution strategy.

To address this, we employed the "Trust Your Friends" strategy. This approach entailed prioritizing and retaining specific columns from one source over the other. In our case, we chose to keep the 'Counter' and 'Win Rate Weak' columns from op.gg, while discarding the corresponding columns from blitz.gg. This decision was influenced by the broader gaming data coverage of op.gg, which includes statistics from all League of Legends games, as opposed to blitz.gg's focus on data from users of the Blitz application.

Another challenge arose in reconciling the different grading systems used by each platform. To create a unified grading metric, we adopted the "Meet in the Middle" strategy. This approach involved calculating an average grade based on the individual grades provided by LoLalytics, op.gg, and Blitz.gg. Each platform has its unique grading criteria: op.gg combines factors such as 'Win Rate', 'Pick Rate', and 'Ban Rate'; blitz.gg uses various in-game metrics from Blitz users; and LoLalytics evaluates based on metrics like 'Win Rate', 'Pick Rate', 'Ban Rate', and the champion's potential as observed from top-performing players. By averaging these grades, we aimed to achieve a balanced evaluation that encompasses diverse perspectives on each champion's performance.

The final result of our data integration was the 'complete.csv' file. This dataset brings together champion stats, counters, and grades from all three platforms. It's key for our detailed analysis, helping us understand the best champions in League of Legends Patch 13.24.

5. Data Cleaning

Data cleaning, an essential step in the data management process, involves refining and structuring the dataset for efficient storage and analysis. In this chapter, I will outline the procedures undertaken to clean and organize the datasets obtained from the RIOT API and our scraping efforts. This sets the groundwork for the relational database model, which will be elaborated in Chapter 7.

We started with organizing the 'champion_list.csv' file, provided by the RIOT API. This file gave us a lot of detailed information about the game's champions. We split this into five specific tables: 'champion_name', 'champion_base_info', 'champion_base_stats', 'champion_tags', and 'tag'. The first three tables cover a range of champion data, from basic names to detailed stats. 'Champion_tags' links champions to different tags showing their features, and 'tag' lists all unique tags used in 'champion_tags'.

We also used the 'complete.csv' file, which brought together data from LoLalytics, blitz.gg, and op.gg. From this, we made the 'lane' and 'ranking' tables. These tables help organize the game's strategic roles and the ranking system. Next, we created two important tables: 'game_stats' and 'matchup'. 'Game_stats' tracks champions' in-game stats based on their lane and rank. 'Matchup' goes into detail about how champions perform against others, showing win rates and games played for different matchups. In the 'game_stats' table, we adjusted the 'position' column to fit with our new grading system.

When building the 'matchup' table, a key challenge was managing columns that contained lists of data formatted as strings. The data included lists of counter champions, their win rates against specific champions, and the number of games played, all stored as strings within single columns. To enhance the usability and analysis of this data, we transformed these string lists into a more structured format. Each element of the list within a string was separated out to create individual rows. This meant that for every unique combination of a champion, lane, and ranking, we created separate rows for each counter champion they face, recording the specific win rate and games played. For instance, initially, a single row might have a string listing multiple counter champions. After transformation, this same data would generate multiple rows, with each row dedicated to one counter champion and their associated statistics against a particular champion. This reorganization process led to an expansion of the dataset from 3,656 original rows to 30,925 rows. Now, each row provides a clear and detailed account of how one champion fares against another, vastly improving the depth and clarity of the data available for our analysis.

Our final set of nine interconnected tables from 'champion_list.csv' and 'complete.csv' now gives us a complete view of champion data, crucial for our analysis of the META in League of Legends Patch 13.24.

6. Data quality

Data quality refers to the overall reliability, accuracy, and completeness of the dataset. High-quality data ensures that the analyses conducted are sound and the decisions made based on them are reliable. In our project, we emphasized two key dimensions to evaluate the quality of our dataset: Completeness and Consistency.

6.1. Completeness

In our data quality assessment for the League of Legends dataset, completeness is a crucial metric. We define completeness as the dataset's ability to inclusively cover all necessary information. Specifically, for the purpose of our analysis, we have structured the dataset to include 5 counter champions and 5 strong-against champions for each combination of a specific champion, lane, and ranking. This structuring is pivotal in ensuring that our analysis comprehensively covers the strategic aspects of matchups in the game.

To assess the completeness concerning the counters and strong-against entries for each champion, associated with a specific lane and ranking, we employed the following formula, as shown in Eq. 1

$$\text{Completeness} = \frac{\text{Number of Rows in Matchup}}{\text{Number of Rows in Game Stats} \times 10} \tag{1}$$

The Equation 1 is based on the expectation that each champion, in relation to their lane and ranking, should have 5 counter and 5 strong-against entries, making a total of 10 entries per champion. The completeness rate is then calculated by dividing the number of rows in the 'Matchup' table by the product of the number of rows in the 'Game Stats' table and 10. The calculation yielded a completeness rate of 84.59%, indicating that while a significant portion of the expected data is present, there are gaps that need addressing.

Regarding the completeness of tables containing basic statistics of champions, we have achieved a 100% rate. This means that every champion in the dataset is fully equipped with all the necessary basic statistics, ensuring both table and tuple completeness. This high level of completeness in basic statistics is essential, as it provides a reliable foundation for our analysis.

6.2. Consistency

Consistency refers to the logical correlation between various data points within the dataset. For our analysis, a key aspect of consistency is ensuring that the win rates are logically aligned with the roles of champions as counters or strong against their opponents.

To maintain consistency, we established two specific criteria:

- Win Rates of Counters: The average win rate of the counters to a specific champion (when in a particular lane and ranking) should logically be less than the overall win rate of that champion. This is based on the premise that a counter champion is expected to have a lower win rate against the champion it counters.
- Win Rates of Strong Against Champions: Conversely, for champions a specific champion is strong against, the average win rate should be higher than the general win rate of the selected champion. This aligns with the understanding that a champion will generally perform better against those it is considered strong against.

To validate these criteria, we calculated the average win rates for both counters and strong against for each champion, considering their specific lane and ranking, within the matchup dataset. This process involved analyzing a total of 3656 rows, each representing a unique combination of champion, lane, and ranking.

Our assessment revealed that approximately 3.4% of the rows showed inconsistencies, not aligning with the expected win rate patterns. However, upon further examination, we identified that the primary reason for these inconsistencies is the small number of games played for certain champions. This factor is particularly influential in cases where data is recorded by platforms like op.gg, which might not immediately reflect recent changes or variations in gameplay.

Small sample sizes can lead to significant fluctuations in win rates, making them less stable and more susceptible to rapid changes. For example, a champion with a limited number of games played might show a temporarily high or low win rate, which can significantly differ from the average as more games are played and recorded. This variance can explain why certain data points did not meet our consistency criteria.

6.3. *Improvement Phase*

The Improvement Phase is a pivotal part of enhancing our League of Legends dataset, focusing on refining the data to ensure its utmost quality. This phase is integral for ensuring that the analyses derived from the data are accurate, reliable, and meaningful, particularly in understanding the META champions and their counters. Our primary objective in this phase is to attain the highest level of consistency. Consistency ensures that the data logically aligns with the expected patterns and behaviors in League of Legends gameplay, especially in the context of champion matchups. To achieve this, we employed a data-driven strategy, leveraging insights from our previous evaluations to address anomalies and irregularities in the dataset. We faced the challenging decision of excluding certain matchups from our dataset. These matchups were identified as inconsistent during our analysis and, as a result, were removed to maintain the integrity of the data. This decision was necessary to ensure that the remaining dataset reflected a high level of consistency.

As a result of these refinements, our dataset now achieves a 100% consistency rate. This is a significant milestone, indicating that the data now fully aligns with logical expectations and patterns within the game. However, it's important to acknowledge that this focus on consistency required a trade-off with completeness. The completeness rate now stands at 83.3%, slightly lower than before but still robust enough to support reliable analyses.

7. Data Storage

The relational database model was selected for the storage phase due to its strength in handling complex queries and the ability to join related data sets, which is critical when dealing with interconnected entities such as champions, their statistics in different lanes and rankings, as well as their performance against counter champions.

The relational model's compatibility with powerful visualization tools like Tableau and the Python library Seaborn was also a key factor in its selection. These tools rely on well-structured relational data to create insightful visualizations, making the relational model a natural fit. Additionally, since the scraping process resulted in tabular data, a relational model seemed most fitting. The schema I developed comprises nine tables:

- champion: Serves as the central reference, containing 'champion_ID' and 'champion_name'.
- champion_base_info: Houses basic information about each champion.
- champion_base_stats: Stores detailed base statistics for each champion.
- champion_tags: Links champions to their respective tags, useful for categorization.
- tag: Stores unique identifiers and names of tags associated with champions.
- ranking: Consists of identifiers and names for different game rankings, along with their value.
- lane: Contains identifiers and names of the various lanes in the game.
- game_stats: Captures in-game statistics, referencing 'champion_ID', 'lane_ID', and 'ranking_ID'.
- matchup: Details champion versus champion statistics across different game conditions.

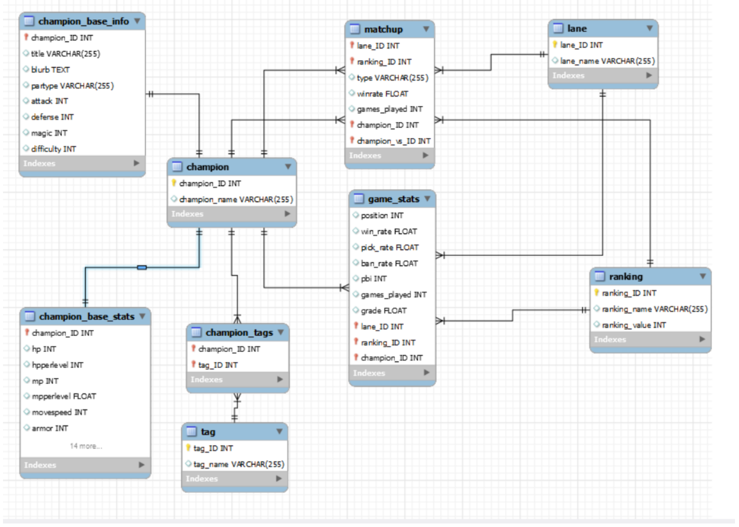

Figure 1: *ER Diagram*

The ER Diagram in Fig. 1 visually represents these tables' relationships and data flow. At the heart of the schema, the 'champion' table's primary key, 'champion_ID', is referenced throughout other tables, establishing a normalized and efficient relational structure. The 'game_stats' table, with its composite primary key, ties directly to the primary keys of 'champion', 'ranking', and 'lane' tables, facilitating in-depth, condition-specific analyses of champion performance.

To classify champions by their tags, the 'champion_tags' table utilizes 'tag_ID' in conjunction with 'champion_ID' as its primary key, allowing each champion to be associated with multiple tags. This arrangement supports complex query filtration. The 'matchup' table, intricate in its design, uses a composite primary key that includes 'champion_ID', 'ranking_ID', 'lane_ID', and a new 'champion_vs_ID'—a reference back to the 'champion' table that denotes the opponent in the matchup.

For actual database implementation, MySQL Workbench was employed. A MySQL database named 'main' was established, and the tables, previously structured in .xlsx format, were migrated using Python scripts and the mysql.connector library. The get_create_table_query function was written to generate the necessary CREATE queries from the Pandas DataFrames, considering primary and foreign keys. Each table was thus created in the MySQL environment.

Populating these tables was achieved with the get_insert_query function, which formulated INSERT queries for the data. The mysql.connector library executed these queries, seamlessly transferring data from the DataFrames into the MySQL tables.

This meticulous design and implementation of the data storage schema ensure that the rich dataset is efficiently stored and readily accessible for complex queries, analysis, and visualization, supporting the overarching goals of our League of Legends project.

8. Queries

In our exploration of League of Legends, we utilize queries to extract insights into META champions and strategic dynamics within the game. These queries serve as a tool, guiding us through the expansive world of League of Legends to answer specific questions and unearth data-driven insights.

8.1. First Query

Our initial query focuses on the Gold rank, identifying the top 5 champions in the Support role. Considering that the ranking influences how powerful a champion might be (reflecting players' proficiency with champions, where gold is considered average), the results offer a glimpse into the most preferred and effective champions for players in the Gold rank playing in the support role. The query is presented in Lst. 1.

The results from this query, presented in Table 8.1, showcase a snapshot of the preferred support champions among Gold-level players, ranked by their grade and number of games played.

The query results indicate that Senna and Xerath are at the forefront in terms of grade, pointing to their status as potential META champions in the support role for Gold rank players. The difference in games played may reflect the varying degrees to which each champion is leveraged to exploit the current META at the Gold rank.

```
SELECT position, champion_name as champion, grade,
games_played as GamesPlayed
FROM game_stats
JOIN champion ON
game_stats.champion_ID = champion.champion_ID
JOIN lane ON game_stats.lane_ID = lane.lane_ID
JOIN ranking on
game_stats.ranking_ID = ranking.ranking_ID
WHERE lane_name = 'support' AND ranking_name = 'gold'
AND position BETWEEN 1 AND 5
ORDER BY position ASC;
```

Listing 1: First Query

Position	Champion	Grade	GamesPlayed
1	Senna	1.111	306700
2	Xerath	1.111	198432
3	Lux	1.222	428303
4	Blitzcrank	1.222	226385
5	Zyra	1.222	195733

Table 1: Top 5 Supports in gold rank

8.2. Second Query

Building on insights from Query 1, our second query explores effective strategies for countering Senna, a dominant force in the Gold support lane. This investigation provides strategic guidance for players facing this formidable champion. The query is presented in Lst. 2.

```
SELECT vs_champion.champion_name as Counter, winrate,
games_played as GamesPlayed
FROM matchup
JOIN lane ON matchup.lane_ID = lane.lane_ID
JOIN ranking ON matchup.ranking_ID = ranking.ranking_ID
JOIN champion ON
matchup.champion_ID = champion.champion_ID
JOIN champion AS vs_champion ON
matchup.champion_vs_ID = vs_champion.champion_ID
WHERE ranking_name = 'gold' AND
champion.champion_name = 'Senna' AND
lane_name = 'support' AND type = 'counter'
ORDER BY winrate ASC;
```

Listing 2: Second Query

The results of the query are presented in Table 8.2.

Table 8.2 shows Senna's win rates when facing her counters in the support role at Gold rank. Zyra leads with a win rate of 45.78% across 5100 games, suggesting her as a viable pick against Senna. The data indicates Xerath not only as a top META support champion in the Gold rank but also an effective counter to Senna, highlighting his dual role as both a

Counter	Winrate	GamesPlayed
Zyra	45.78	5100
Xerath	46.38	4644
Vel'Koz	47.05	1526
Amumu	47.36	492
Lux	47.44	10302

Table 2: Champion counters to Senna in the support role at Gold rank

popular and strategic pick in this tier. The presence of Lux in both queries further solidifies her as a pivotal choice, excelling against Senna and within the overall support META. The inclusion of unconventional support choices like Amumu suggests a niche yet effective strategy against Senna, showcasing the diverse tactical approaches players can employ within the Gold rank META. This analysis suggests that champions with strong engage, crowd control, or superior poke are effective strategies against Senna in the support role within the Gold tier.

8.3. *Third Query*

Transitioning to the Bronze rank, the third query unveils the average difficulty and grade of the top 3 META champions in each lane. This analysis offers insights into champion performance in lower skill tiers, benefiting novice players. The query is presented in Lst. 3

```
SELECT lane.lane_name as Lane
AVG(champion_base_info.difficulty) as AvgDiff,
AVG(game_stats.grade) as AvgGrade
FROM game_stats
JOIN ranking ON
game_stats.ranking_ID = ranking.ranking_ID
JOIN lane ON game_stats.lane_ID = lane.lane_ID
JOIN champion_base_info ON
game_stats.champion_ID = champion_base_info.champion_ID
WHERE (position BETWEEN 1 AND 3) AND
ranking_name = 'bronze'
GROUP BY lane.lane_name;
```

Listing 3: Third Query

The results are represented in Table 8.3

Lane	AvgDiff	AvgGrade
jungle	3.8	1.28800
adc	5.2	1.49967
mid	6.0	1.34433
top	5.6	1.19967
support	5.8	1.06653

Table 3: Average Difficulty and Grade of Top 3 META Champions in Each Lane for Bronze Rank

Support champions in Bronze rank, while more complex, are effectively utilized, achieving the best grades. This suggests that players at this level may grasp the role's intricacies well. In contrast, jungle champions, despite being less complex, see lower performance grades, indicating the role's strategic demands may be more challenging for players at this tier.

8.4. *Fourth Query*

Expanding the view beyond individual ranks, the fourth query provides a panoramic overview of the top two champions for each role across all ranks. This comprehensive analysis transcends skill levels, showcasing universally favored champions. The query is represented in Lst. 4

```
SELECT lane_name as Lane, position,
champion_name as Champion, grade
FROM game_stats
JOIN ranking ON
game_stats.ranking_ID = ranking.ranking_ID
JOIN champion ON
game_stats.champion_ID = champion.champion_ID
JOIN lane ON game_stats.lane_ID = lane.lane_ID
WHERE (game_stats.position BETWEEN 1 AND 2) AND
ranking_name = 'all'
ORDER BY game_stats.lane_ID ASC, position ASC;
```

Listing 4: Fourth Query

The results of the query are presented in Table 8.4.

Lane	Position	Champion	Grade
top	1	Jax	0.88667
top	2	Illaoi	1.11100
jungle	1	Briar	0.88667
jungle	2	Graves	1.00000
mid	1	Sylas	0.88667
mid	2	Fizz	1.11100
adc	1	Vayne	1.00000
adc	2	Jhin	1.11100
support	1	Rakan	0.88667
support	2	Xerath	1.11100

Table 4: Top Two Champions in Each Role Across All Ranks

The fourth query reveals that Jax and Illaoi are the top two champions in the top lane, with Jax leading in performance. In the jungle, Briar and Graves are favored, again with the first holding a better grade. Mid lane choices show Sylas outperforming Fizz, while in the ADC role, Vayne surpasses Jhin in terms of grade. Support lane sees Rakan leading over Xerath. Lower grades across the board suggest these champions are effective across various skill levels, indicating their strong presence in the overall META.

8.5. *Fifth Query*

The final query focuses on identifying potential overpowered champions by pinpointing those with a grade below 1 across all rankings. This critical examination aims to uncover champions that may significantly impact the balance of the game.

```
SELECT champion_name AS Champion, lane_name AS Lane, grade
FROM game_stats
JOIN champion ON
game_stats.champion_ID  = champion.champion_ID
JOIN lane ON game_stats.lane_ID = lane.lane_ID
JOIN ranking ON game_stats.ranking_ID = ranking.ranking_ID
WHERE ranking_name = 'all' AND grade < 1
```

Listing 5: Fifth Query

The results of the query are presented in Table 5.

Champion	Lane	grade
Jax	top	0.88667
Briar	jungle	0.88667
Rakan	support	0.88667
Sylas	mid	0.88667

Table 5: Champions with a Grade Below 1 Across All Ranks

The fifth query results highlight Jax, Briar, Rakan, and Sylas as champions with a grade below 1 across all ranks, potentially marking them as overpowered in their respective roles of top, jungle, support, and mid. The absence of ADC champions from the list of those with a grade below 1 suggests that currently, there are no overpowered ADCs across all ranks. The ADC role has historically been considered the weaker or more neglected in terms of balancing, as it requires precise adjustments to avoid tipping them into being overpowered. The delicate balance is challenging due to ADCs' dependence on item builds and support from the team, which can significantly vary in effectiveness from one rank to another.

9. Conclusions

In this study, we set out to identify META champions in the League of Legends patch 13.24, with the goal of understanding the prevailing strategies and champion selections. We began by gathering data from prominent analytics sources like Riot Games, LoLalytics, Blitz, and Op.gg. Following data acquisition, we integrated the information into a unified dataset and meticulously cleaned it to ensure high quality for analysis.

Throughout the project, we prioritized the consistency and completeness of our data, essential for the integrity of our findings. The queries we ran were aimed at revealing the top-performing champions in various ranks and roles, and also at pinpointing those champions whose grades suggested they were performing above the expected norm.

From our analysis, we discovered certain champions that stand out in their respective roles, some showing potential signs of being overpowered. Interestingly, the ADC role did not feature any champions with exceptionally low grades, suggesting a level of balance in this role.

The project's insights provide a snapshot of the competitive dynamics at play within this patch of League of Legends. Future work, such as developing a Tableau dashboard, could expand on this by offering interactive visualizations, making the insights even more accessible.

By adhering to a structured data management approach, we have achieved our objective of highlighting key patterns in the game's current META. This approach has proven effective in deconstructing the complexities of League of Legends gameplay and strategy, offering players and enthusiasts valuable insights.

References

[1] "League of legends." https://www.leagueoflegends.com/it-it/, 2024. Accessed: 2024-01-23.

[2] "Riot games developer portal: League of legends." https://developer.riotgames.com/docs/lol. Accessed: 2024-01-24.

[3] "League of legends analytics - lolalytics." https://lolalytics.com/. Accessed: 2024-01-24.

[4] "Op.gg - the best lol builds and tier list." https://www.op.gg/. Accessed: 2024-01-24.

[5] "Blitz, the best tracker for players to win league of legends." https://blitz.gg/lol. Accessed: 2024-01-24.