# Predicting League of Legends World Championship Winner using Machine Learning

## A Data-Driven Approach to Champion, Meta, and Role Analysis

### Group members: Vansh, Joshua, Enya, Derek, and Eric

## Background

*League of Legends* is a globally popular esports game[4], with a **World Championship (Worlds)** featuring teams from major regions like **South Korea, China, Europe**, and **NA**.

The game revolves around two teams, each consisting of 5 players, who must siege towers and destroy the enemy's base ("nexus"). Each team is composed of 5 different roles, mostly corresponding the lane they primarily play in at the beginning of the game: the toplaner, midlaner, botlaner (also commonly known as Attack Damage Carry or ADC), support (who plays alongside the glass cannon ADC), and jungler (who resides in the "jungle," the space between lanes). The players in the lanes are also commonly referred to as simply their lane's position, such as "top" or "bot." It should be noted that although the support and ADC share the bottom lane, for the purposes of this report, the “botlaner” will refer to the ADC.

Current papers have analyzed League with different ML models. (Arık, 2023) studies a LightGBM classifier[1], (Bahrololloomi et al., 2022) utilizes a GradBoost-based model[2], and (Shen, 2022) implements a voting classifier[3] for results prediction of games based on player and game data.

# Problem Definition

- **Problem**: Esports outcomes are difficult to predict due to evolving metas, player performance, and regional differences.
- **Motivation**: Analyzing what makes a player good at their role and the impact of each role can provide deeper insights into team performance–a valuable asset for teams, fans, and analysts

Our original proposal involved predicting **the Worlds winner** via data from the **Worlds patch (14.18)[5]**. First, we wanted to identify meta champions; then, we wanted to analyze **strong players** based on the meta. We wanted to calculate **role impact** to proportionate player strength, **aggregating** to predict team performance at Worlds.

However, we have decided to move away from this exact approach, as we are uncertain about the feasibility of identifying meta champions (due to restrictions based on the changing nature of the game and available data). Our target shifted to finding “strong players” and role impact with our machine learning algorithms, ultimately using this data to predict the winner of Worlds.

In our new approach, we decided to first determine which statistics are the most important for each role in helping a team achieve victory, i.e, what features does an ideal player in their respective role need to excel at in order to provide a team with the highest chance of winning. With this information, we could then assign players a performance rating for each of their games depending on how they performed with regards to each important feature.

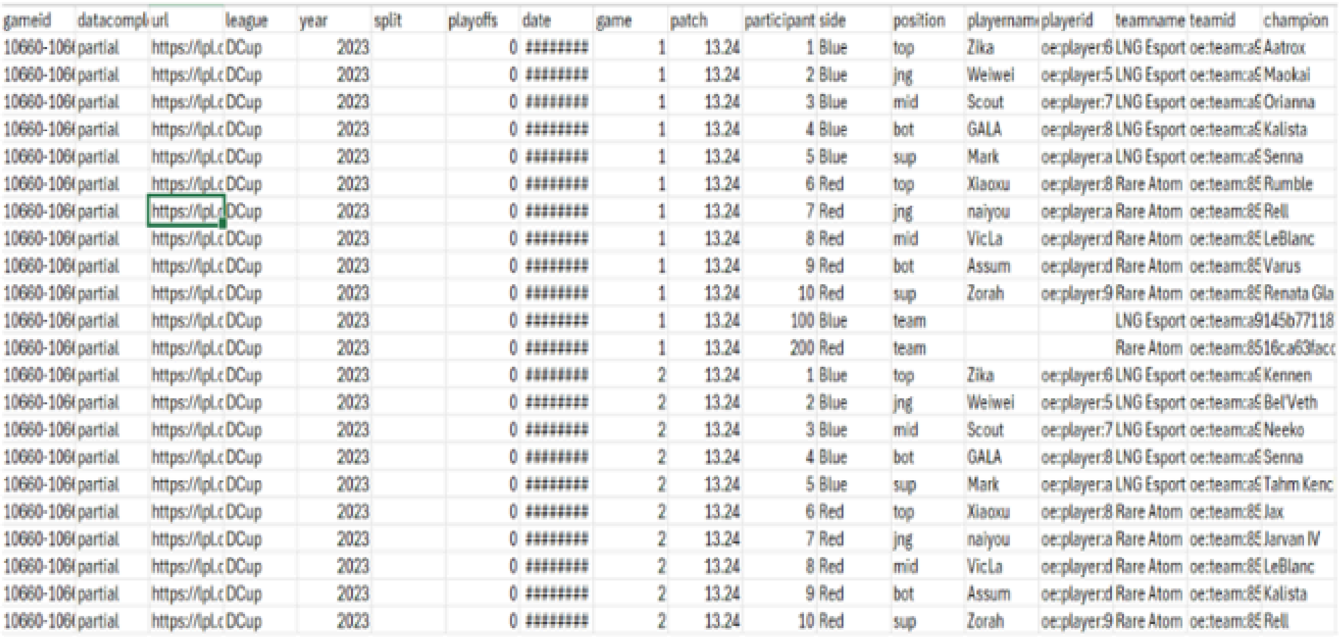
We then employed an additional model to find the impact of each role on the outcome of the game, similar to in our original proposal. By using this information, we could then scale our player performances accordingly and help us better ascertain overall team performance.

Finally to also account for team performance changes over time, we employed a model to predict how each team would perform at worlds given their match history leading up to the tournament.

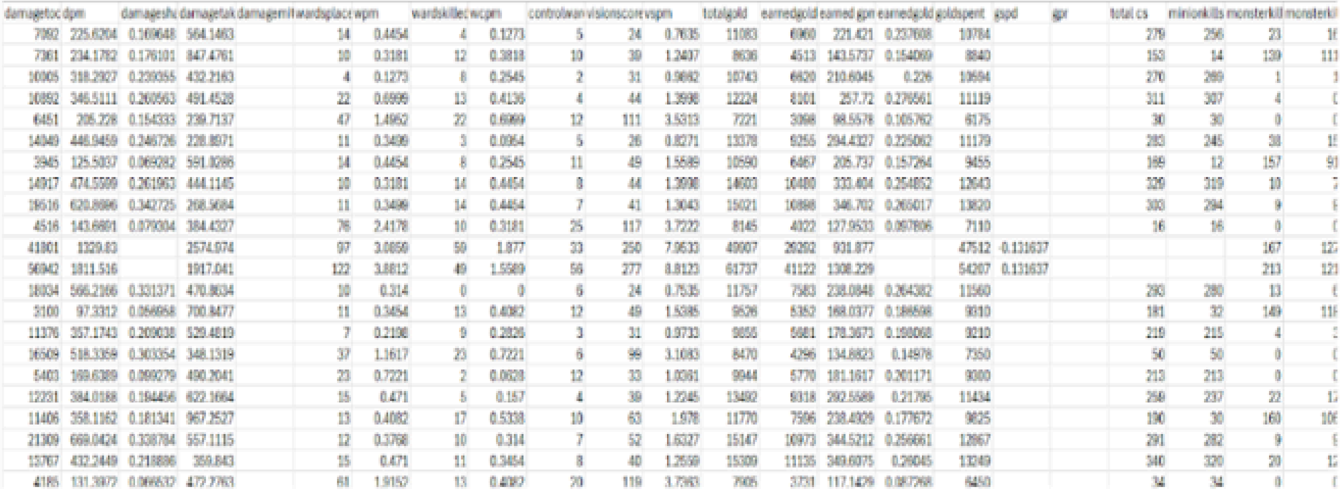
With these 3 models combined, we believed we could achieve a high accuracy in predicting the overall winner of the competition

# Dataset

Our dataset includes statistics from players and overall games from this year in .csv format. This dataset is extremely comprehensive, and features data garnered from several sources, including official Riot API and Match History pages. The dataset contained data for all league of legends matches played in the past year, detailing which teams played each other for which match, what date they played, which tournament they played in and also various statistics for each player during the game.



| gameid | datacompl | url | league | year | split | playoffs | date | game | patch | participant | side | position | playernam | playerid | teamname | teamid | champion |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 1 | Blue | top | Zika | oe:player:6 | LNG Esport | oe:team:a | Aatrox |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 2 | Blue | jng | Weiwei | oe:player:5 | LNG Esport | oe:team:a | Maokai |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 3 | Blue | mid | Scout | oe:player:7 | LNG Esport | oe:team:a | Orianna |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 4 | Blue | bot | GALA | oe:player:8 | LNG Esport | oe:team:a | Kalista |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 5 | Blue | sup | Mark | oe:player:a | LNG Esport | oe:team:a | Senna |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 6 | Red | top | Xiaoxu | oe:player:8 | Rare Atom | oe:team:85 | Rumble |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 7 | Red | jng | naiyou | oe:player:a | Rare Atom | oe:team:85 | Rell |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 8 | Red | mid | VicLa | oe:player:d | Rare Atom | oe:team:85 | LeBlanc |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 9 | Red | bot | Assum | oe:player:d | Rare Atom | oe:team:85 | Varus |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 10 | Red | sup | Zorah | oe:player:9 | Rare Atom | oe:team:85 | Renata Gla |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 100 | Blue | team | | | LNG Esport | oe:team:a9145b77118 | |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 1 | 13.24 | 200 | Red | team | | | Rare Atom | oe:team:8516ca63facc | |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 1 | Blue | top | Zika | oe:player:6 | LNG Esport | oe:team:a | Kennen |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 2 | Blue | jng | Weiwei | oe:player:5 | LNG Esport | oe:team:a | Bel'Veth |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 3 | Blue | mid | Scout | oe:player:7 | LNG Esport | oe:team:a | Neeko |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 4 | Blue | bot | GALA | oe:player:8 | LNG Esport | oe:team:a | Senna |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 5 | Blue | sup | Mark | oe:player:a | LNG Esport | oe:team:a | Tahm Kenc |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 6 | Red | top | Xiaoxu | oe:player:8 | Rare Atom | oe:team:85 | Jax |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 7 | Red | jng | naiyou | oe:player:a | Rare Atom | oe:team:85 | Jarvan IV |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 8 | Red | mid | VicLa | oe:player:d | Rare Atom | oe:team:85 | LeBlanc |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 9 | Red | bot | Assum | oe:player:d | Rare Atom | oe:team:85 | Kalista |
| 10660-106( | partial | https://lpl.c | DCup | 2023 | | 0 | ######## | 2 | 13.24 | 10 | Red | sup | Zorah | oe:player:9 | Rare Atom | oe:team:85 | Rell |



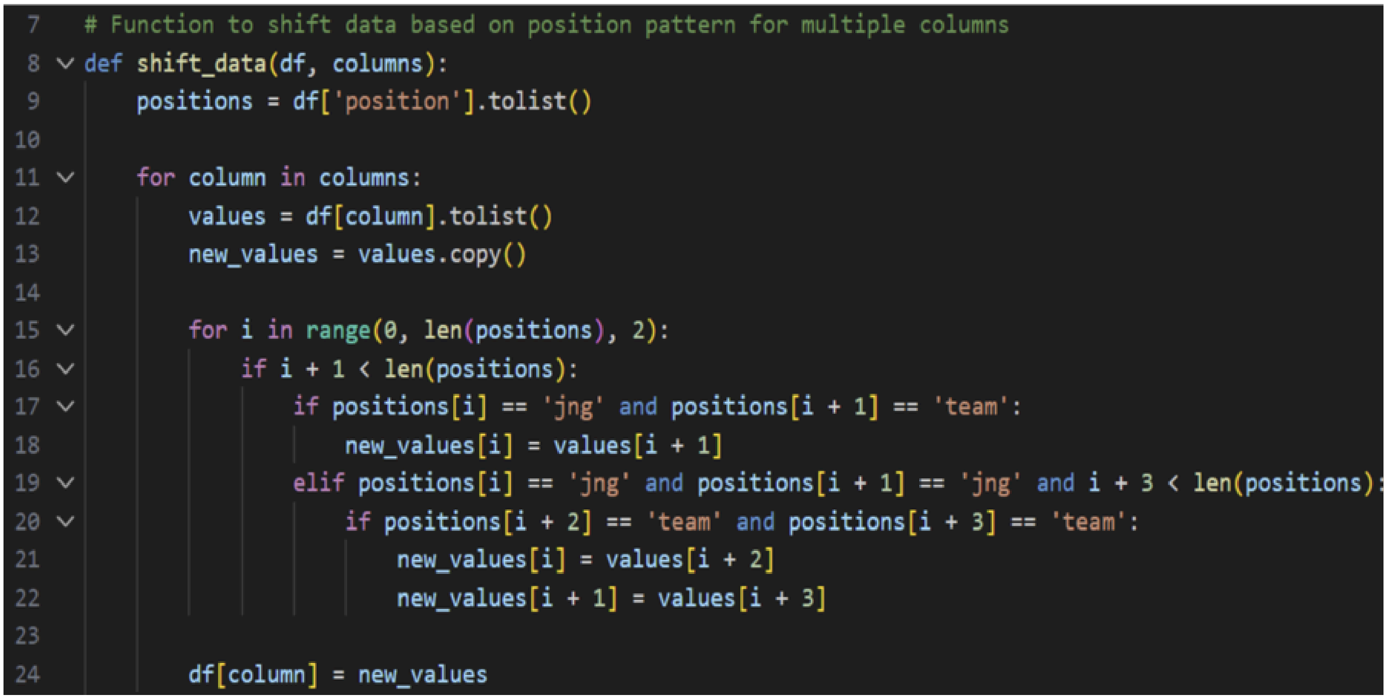
| damagetoc | dpm | damageshi | damagetak | damagemit | wardsplace | wpm | wardskille | wcpm | controlwar | visionscore | vspm | totalgold | earnedgold | earned gpm | earnedgold | goldspent | gspd | gpr | total cs | minionkills | monsterkill | monsterkil |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7092 | 225.6204 | 0.169648 | 564.1463 | | 14 | 0.4454 | 4 | 0.1273 | 5 | 24 | 0.7635 | 11083 | 6960 | 221.421 | 0.237608 | 10784 | | | 279 | 256 | 23 | 16 |
| 7361 | 234.1782 | 0.176101 | 847.4761 | | 10 | 0.3181 | 12 | 0.3818 | 10 | 39 | 1.2407 | 8636 | 4513 | 143.5737 | 0.154069 | 8840 | | | 153 | 14 | 139 | 111 |
| 10005 | 318.2927 | 0.239355 | 432.2163 | | 4 | 0.1273 | 8 | 0.2545 | 2 | 31 | 0.9862 | 10743 | 6620 | 210.6045 | 0.226 | 10694 | | | 270 | 269 | 1 | 1 |
| 10892 | 346.5111 | 0.260563 | 491.4528 | | 22 | 0.6999 | 13 | 0.4136 | 4 | 44 | 1.3998 | 12224 | 8101 | 257.72 | 0.276561 | 11119 | | | 311 | 307 | 4 | 0 |
| 6451 | 205.228 | 0.154333 | 239.7137 | | 47 | 1.4952 | 22 | 0.6999 | 12 | 111 | 3.5313 | 7221 | 3098 | 98.5578 | 0.105762 | 6175 | | | 30 | 30 | 0 | 0 |
| 14049 | 446.9459 | 0.246726 | 228.8971 | | 11 | 0.3499 | 3 | 0.0954 | 5 | 26 | 0.8271 | 13378 | 9255 | 294.4327 | 0.225062 | 11179 | | | 283 | 245 | 38 | 15 |
| 3945 | 125.5037 | 0.069282 | 591.0286 | | 14 | 0.4454 | 8 | 0.2545 | 11 | 49 | 1.5589 | 10590 | 6467 | 205.737 | 0.157264 | 9455 | | | 169 | 12 | 157 | 91 |
| 14917 | 474.5599 | 0.261963 | 444.1145 | | 10 | 0.3181 | 14 | 0.4454 | 8 | 44 | 1.3998 | 14603 | 10480 | 333.404 | 0.254852 | 12643 | | | 329 | 319 | 10 | 7 |
| 19516 | 620.8886 | 0.342725 | 268.5684 | | 11 | 0.3499 | 14 | 0.4454 | 7 | 41 | 1.3043 | 15021 | 10898 | 346.702 | 0.265017 | 13820 | | | 303 | 294 | 9 | 6 |
| 4516 | 143.6691 | 0.079304 | 384.4327 | | 76 | 2.4178 | 10 | 0.3181 | 25 | 117 | 3.7222 | 8145 | 4022 | 127.9533 | 0.097806 | 7110 | | | 16 | 16 | 0 | 0 |
| 41801 | 1329.83 | | 2574.974 | | 97 | 3.0859 | 59 | 1.877 | 33 | 250 | 7.9533 | 49907 | 29292 | 931.877 | | 47512 | -0.131637 | | | | 167 | 122 |
| 56942 | 1811.516 | | 1917.041 | | 122 | 3.8812 | 49 | 1.5589 | 56 | 277 | 8.8123 | 61737 | 41122 | 1308.229 | | 54207 | 0.131637 | | | | 213 | 121 |
| 18034 | 566.2166 | 0.331371 | 470.8634 | | 10 | 0.314 | 0 | 0 | 6 | 24 | 0.7535 | 11757 | 7583 | 238.0848 | 0.264382 | 11560 | | | 293 | 280 | 13 | 6 |
| 3100 | 97.3312 | 0.056958 | 700.8477 | | 11 | 0.3454 | 13 | 0.4082 | 12 | 49 | 1.5385 | 9526 | 5352 | 168.0377 | 0.186598 | 9310 | | | 181 | 32 | 149 | 116 |
| 11376 | 357.1743 | 0.209038 | 529.4819 | | 7 | 0.2198 | 9 | 0.2826 | 3 | 31 | 0.9733 | 9855 | 5681 | 178.3673 | 0.198068 | 9210 | | | 219 | 215 | 4 | 3 |
| 16509 | 518.3359 | 0.303354 | 348.1319 | | 37 | 1.1617 | 23 | 0.7221 | 6 | 99 | 3.1083 | 8470 | 4296 | 134.8823 | 0.14978 | 7350 | | | 50 | 50 | 0 | 0 |
| 5403 | 169.6389 | 0.099279 | 490.2041 | | 23 | 0.7221 | 2 | 0.0628 | 12 | 33 | 1.0361 | 9944 | 5770 | 181.1617 | 0.201171 | 9080 | | | 213 | 213 | 0 | 0 |
| 12231 | 384.0188 | 0.194456 | 622.1664 | | 15 | 0.471 | 5 | 0.157 | 4 | 39 | 1.2245 | 13492 | 9318 | 292.5589 | 0.21795 | 11434 | | | 259 | 237 | 22 | 12 |
| 11406 | 358.1162 | 0.181341 | 967.2527 | | 13 | 0.4082 | 17 | 0.5338 | 10 | 63 | 1.978 | 11770 | 7596 | 238.4929 | 0.177672 | 9825 | | | 190 | 30 | 160 | 106 |
| 21309 | 669.0424 | 0.338784 | 557.1115 | | 12 | 0.3768 | 10 | 0.314 | 7 | 52 | 1.6327 | 15147 | 10973 | 344.5212 | 0.256661 | 12867 | | | 291 | 282 | 9 | 5 |
| 13767 | 432.2449 | 0.218886 | 359.843 | | 15 | 0.471 | 11 | 0.3454 | 8 | 40 | 1.2559 | 15309 | 11135 | 349.6075 | 0.26045 | 13249 | | | 340 | 320 | 20 | 12 |
| 4185 | 131.3972 | 0.066532 | 472.2763 | | 61 | 1.9152 | 13 | 0.4082 | 20 | 119 | 3.7363 | 7905 | 3731 | 117.1429 | 0.087268 | 6450 | | | 34 | 34 | 0 | 0 |

# Methods

# Data Preprocessing 1

Several Data Preprocessing methods were implemented for our model to function on proper data, including **Data Cleaning**. Our dataset contained several empty columns and unnecessary information, such as data regarding teams that did not qualify for Worlds in 2024; these were removed. This left us with data for 1900 games. Additionally, we had numerous rows of missing data due to incomplete data from games in China. This included first bloods, as well as data for in-game objectives like rift heralds and dragons per team. We manually filled in this data by watching replays of the games.

The data for the role of Jungler had to be specifically modified as well, which we accomplished through a script. Some of the features attributed to Junglers were listed under the team data, so we had to shift those specific rows/columns. A sample of our script is as follows:



```
# Function to shift data based on position pattern for multiple columns
def shift_data(df, columns):
    positions = df['position'].tolist()

    for column in columns:
        values = df[column].tolist()
        new_values = values.copy()

        for i in range(0, len(positions), 2):
            if i + 1 < len(positions):
                if positions[i] == 'jng' and positions[i + 1] == 'team':
                    new_values[i] = values[i + 1]
                elif positions[i] == 'jng' and positions[i + 1] == 'jng' and i + 3 < len(positions):
                    if positions[i + 2] == 'team' and positions[i + 3] == 'team':
                        new_values[i] = values[i + 2]
                        new_values[i + 1] = values[i + 3]

        df[column] = new_values
```

Additionally, we employed Feature Selection. This included deleting features that were unhelpful towards determining individual player performance, such as team-based statistics like team kills per minute and game length.

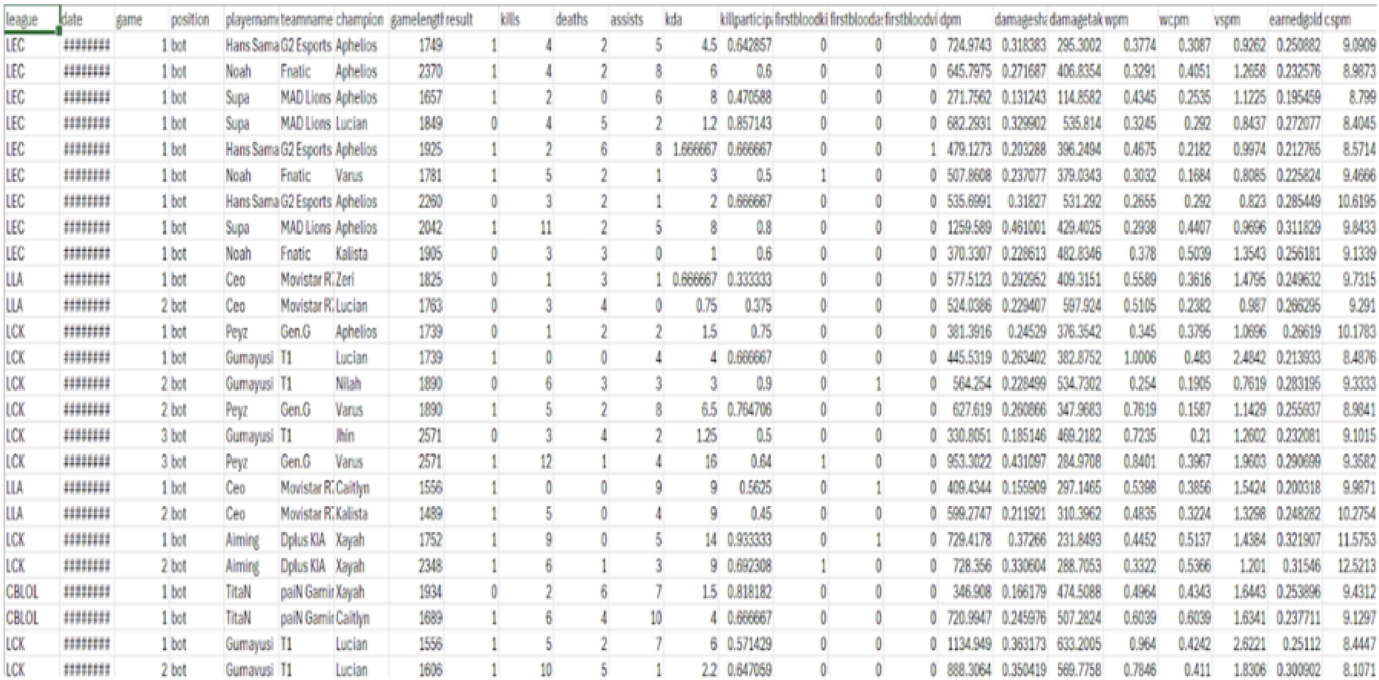
We also implemented Feature Engineering, where we created new features to better capture some statistics that would further elucidate player performance for each role, such as KDA (a statistic determined by summing the kills and assists of a player, divided by their number of deaths + 1), and kill participation, which is the total number of kills and assists a player gets divided by the number of kills the team gets in total.

We had originally planned to use Feature Encoding as well, but found that this was unnecessary as our data was already entirely in numerical format (with no categorical data).

After all our data was processed, we applied Data Transformation where we split the data into 5 separate unique datasets corresponding to each of the 5 roles. This allowed us to run our models on each role separately and draw conclusions based on each individual role rather than all the players as a whole. Additionally, we could then include additional features for the jungle role to train our models as securing objectives such as dragons and void grubs is exclusively a statistic for junglers and not for other roles.

Our final datasets for logistic regression in total each had 16 different features which we considered relevant with the Jungle role having an extra 9 for a total of 25 different features.

Below is an example subset of our processed data for the botlane role.



| league | date | game | position | playernam | teamname | champion | gamelengt | result | kills | deaths | assists | kda | killparticip | firstbloodki | firstblooda | firstbloodvi | dpm | damagesh | damagetak | wpm | wcpm | vspm | earnedgold | cspm |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LEC | ######## | 1 | bot | Hans Sama | G2 Esports | Aphelios | 1749 | 1 | 4 | 2 | 5 | 4.5 | 0.642857 | 0 | 0 | 0 | 724.9743 | 0.318383 | 295.3002 | 0.3774 | 0.3087 | 0.9262 | 0.250882 | 9.0909 |
| LEC | ######## | 1 | bot | Noah | Fnatic | Aphelios | 2370 | 1 | 4 | 2 | 8 | 6 | 0.6 | 0 | 0 | 0 | 645.7975 | 0.271687 | 406.8354 | 0.3291 | 0.4051 | 1.2658 | 0.232576 | 8.9873 |
| LEC | ######## | 1 | bot | Supa | MAD Lions | Aphelios | 1657 | 1 | 2 | 0 | 6 | 8 | 0.470588 | 0 | 0 | 0 | 271.7562 | 0.131243 | 114.8582 | 0.4345 | 0.2535 | 1.1225 | 0.195459 | 8.799 |
| LEC | ######## | 1 | bot | Supa | MAD Lions | Lucian | 1849 | 0 | 4 | 5 | 2 | 1.2 | 0.857143 | 0 | 0 | 0 | 682.2931 | 0.329902 | 535.814 | 0.3245 | 0.292 | 0.8437 | 0.272077 | 8.4045 |
| LEC | ######## | 1 | bot | Hans Sama | G2 Esports | Aphelios | 1925 | 1 | 2 | 6 | 8 | 1.666667 | 0.666667 | 0 | 0 | 1 | 479.1273 | 0.203288 | 396.2494 | 0.4675 | 0.2182 | 0.9974 | 0.212765 | 8.5714 |
| LEC | ######## | 1 | bot | Noah | Fnatic | Varus | 1781 | 1 | 5 | 2 | 1 | 3 | 0.5 | 1 | 0 | 0 | 507.8608 | 0.237077 | 379.0343 | 0.3032 | 0.1684 | 0.8085 | 0.225824 | 9.4666 |
| LEC | ######## | 1 | bot | Hans Sama | G2 Esports | Aphelios | 2260 | 0 | 3 | 2 | 1 | 2 | 0.666667 | 0 | 0 | 0 | 535.6991 | 0.31827 | 531.292 | 0.2655 | 0.292 | 0.823 | 0.285449 | 10.6195 |
| LEC | ######## | 1 | bot | Supa | MAD Lions | Aphelios | 2042 | 1 | 11 | 2 | 5 | 8 | 0.8 | 0 | 0 | 0 | 1259.589 | 0.461001 | 429.4025 | 0.2938 | 0.4407 | 0.9696 | 0.311829 | 9.8433 |
| LEC | ######## | 1 | bot | Noah | Fnatic | Kalista | 1905 | 0 | 3 | 3 | 0 | 1 | 0.6 | 0 | 0 | 0 | 370.3307 | 0.228613 | 482.8346 | 0.378 | 0.5039 | 1.3543 | 0.256181 | 9.1339 |
| LLA | ######## | 1 | bot | Ceo | Movistar R | Zeri | 1825 | 0 | 1 | 3 | 1 | 0.666667 | 0.333333 | 0 | 0 | 0 | 577.5123 | 0.292952 | 409.3151 | 0.5589 | 0.3616 | 1.4796 | 0.249632 | 9.7315 |
| LLA | ######## | 2 | bot | Ceo | Movistar R | Lucian | 1763 | 0 | 3 | 4 | 0 | 0.75 | 0.375 | 0 | 0 | 0 | 524.0386 | 0.229407 | 597.924 | 0.5105 | 0.2382 | 0.987 | 0.266295 | 9.291 |
| LCK | ######## | 1 | bot | Peyz | Gen.G | Aphelios | 1739 | 0 | 1 | 2 | 2 | 1.5 | 0.75 | 0 | 0 | 0 | 381.3916 | 0.24529 | 376.3542 | 0.345 | 0.3795 | 1.0696 | 0.26619 | 10.1783 |
| LCK | ######## | 1 | bot | Gumayusi | T1 | Lucian | 1739 | 1 | 0 | 0 | 4 | 4 | 0.666667 | 0 | 0 | 0 | 445.5319 | 0.263402 | 382.8752 | 1.0006 | 0.483 | 2.4842 | 0.213933 | 8.4876 |
| LCK | ######## | 2 | bot | Gumayusi | T1 | Nilah | 1890 | 0 | 6 | 3 | 3 | 3 | 0.9 | 0 | 1 | 0 | 564.254 | 0.228499 | 534.7302 | 0.254 | 0.1905 | 0.7619 | 0.283195 | 9.3333 |
| LCK | ######## | 2 | bot | Peyz | Gen.G | Varus | 1890 | 1 | 5 | 2 | 8 | 6.5 | 0.764706 | 0 | 0 | 0 | 627.619 | 0.260866 | 347.9683 | 0.7619 | 0.1587 | 1.1429 | 0.255937 | 8.9841 |
| LCK | ######## | 3 | bot | Gumayusi | T1 | Jhin | 2571 | 0 | 3 | 4 | 2 | 1.25 | 0.5 | 0 | 0 | 0 | 330.8051 | 0.185146 | 469.2182 | 0.7235 | 0.21 | 1.2602 | 0.232081 | 9.1015 |
| LCK | ######## | 3 | bot | Peyz | Gen.G | Varus | 2571 | 1 | 12 | 1 | 4 | 16 | 0.64 | 1 | 0 | 0 | 953.3022 | 0.431097 | 284.9708 | 0.8401 | 0.3967 | 1.9603 | 0.290699 | 9.3582 |
| LLA | ######## | 1 | bot | Ceo | Movistar R | Caitlyn | 1556 | 1 | 0 | 0 | 9 | 9 | 0.5625 | 0 | 1 | 0 | 409.4344 | 0.155909 | 297.1465 | 0.5398 | 0.3856 | 1.5424 | 0.200318 | 9.9871 |
| LLA | ######## | 2 | bot | Ceo | Movistar R | Kalista | 1489 | 1 | 5 | 0 | 4 | 9 | 0.45 | 0 | 0 | 0 | 599.2747 | 0.211921 | 310.3962 | 0.4835 | 0.3224 | 1.3298 | 0.248282 | 10.2754 |
| LCK | ######## | 1 | bot | Aiming | Dplus KIA | Xayah | 1752 | 1 | 9 | 0 | 5 | 14 | 0.933333 | 0 | 1 | 0 | 729.4178 | 0.37266 | 231.8493 | 0.4452 | 0.5137 | 1.4384 | 0.321907 | 11.5753 |
| LCK | ######## | 2 | bot | Aiming | Dplus KIA | Xayah | 2348 | 1 | 6 | 1 | 3 | 9 | 0.692308 | 1 | 0 | 0 | 728.356 | 0.330604 | 288.7053 | 0.3322 | 0.5366 | 1.201 | 0.31546 | 12.5213 |
| CBLOL | ######## | 1 | bot | TitaN | paiN Gamir | Xayah | 1934 | 0 | 2 | 6 | 7 | 1.5 | 0.818182 | 0 | 0 | 0 | 346.908 | 0.166179 | 474.5088 | 0.4964 | 0.4343 | 1.6443 | 0.253896 | 9.4312 |
| CBLOL | ######## | 1 | bot | TitaN | paiN Gamir | Caitlyn | 1689 | 1 | 6 | 4 | 10 | 4 | 0.666667 | 0 | 0 | 0 | 720.9947 | 0.245976 | 507.2824 | 0.6039 | 0.6039 | 1.6341 | 0.237711 | 9.1297 |
| LCK | ######## | 1 | bot | Gumayusi | T1 | Lucian | 1556 | 1 | 5 | 2 | 7 | 6 | 0.571429 | 0 | 0 | 0 | 1134.949 | 0.363173 | 633.2005 | 0.964 | 0.4242 | 2.6221 | 0.25112 | 8.4447 |
| LCK | ######## | 2 | bot | Gumavusi | T1 | Lucian | 1606 | 1 | 10 | 5 | 1 | 2.2 | 0.647059 | 0 | 0 | 0 | 888.3064 | 0.350419 | 569.7758 | 0.7846 | 0.411 | 1.8306 | 0.300902 | 8.1071 |

## Model 1: Logistic Regression

The first model we implemented was a logistic regression model to determine the weights of each feature, which was determined as a good choice due to several factors.

1. Our model will predict whether a player won or lost in a particular game based on player statistics data. This is a binary classification task between win (1) and loss (0) making logistic regression a natural fit for our problem.
2. Logistic regression provides interpretable coefficients (weights) for each feature. This helps in understanding the impact of each player statistic on the game's outcome. By examining these weights, we can identify which statistics are most important in determining whether a team wins or loses.

3. Additionally, it is computationally efficient, which is advantageous when dealing with our large dataset with many features and data points. This provides quick training and results while preserving performance.
4. Also, due to our chosen features potentially having simple linear relationships, to avoid overfitting, we decided to pick a simpler model as opposed to a model like Random Forest or GradBoost.

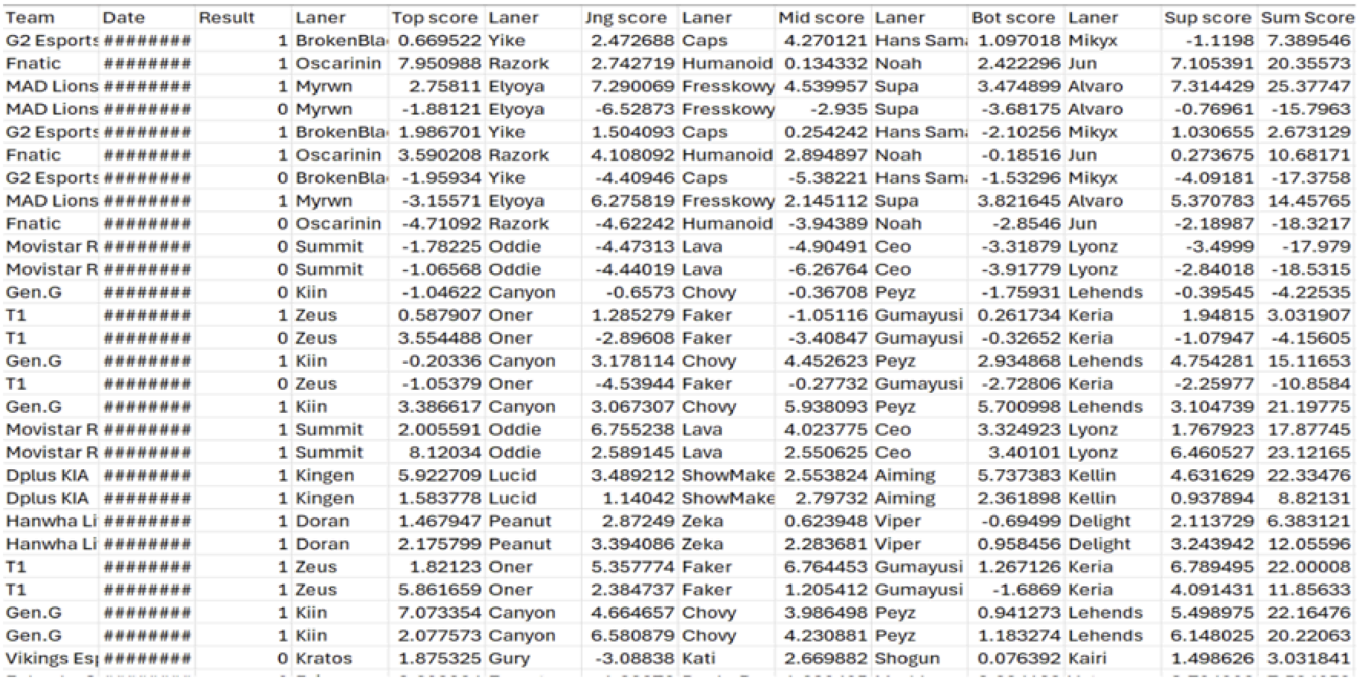
To find the impact of each player statistic for each position, we trained a logistic regression model on five separate datasets for each individual lane (top, mid, bot, jungle, support). In this model, we used 80% of the 1900 games for training and the remaining 20% for testing.

From this, we retrieved the numerical values of each coefficient and graphed them in order to determine the order of importance for these features

## Data Preprocessing 2

With our feature weights from Logistic Regression, by applying them to the actual feature values (e.g. the number of kills or number of assists), we summed them up to get a total "score" of a player. This is reflected in a new data spreadsheet for our next model to work off of.

With these new player performance scores, we created a new dataset with each game on a new row, featuring the team who played (as such, one game will equate to two rows), the date of the game, the result, each player and their respective score, as well as a the sum of these scores for a total team score.



| Team | Date | Result | Laner | Top score | Laner | Jng score | Laner | Mid score | Laner | Bot score | Laner | Sup score | Sum Score |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| G2 Esports | ######## | 1 | BrokenBla | 0.669522 | Yike | 2.472688 | Caps | 4.270121 | Hans Sam | 1.097018 | Mikyx | -1.1198 | 7.389546 |
| Fnatic | ######## | 1 | Oscarinin | 7.950988 | Razork | 2.742719 | Humanoid | 0.134332 | Noah | 2.422296 | Jun | 7.105391 | 20.35573 |
| MAD Lions | ######## | 1 | Myrwn | 2.75811 | Elyoya | 7.290069 | Fresskowy | 4.539957 | Supa | 3.474899 | Alvaro | 7.314429 | 25.37747 |
| MAD Lions | ######## | 0 | Myrwn | -1.88121 | Elyoya | -6.52873 | Fresskowy | -2.935 | Supa | -3.68175 | Alvaro | -0.76961 | -15.7963 |
| G2 Esports | ######## | 1 | BrokenBla | 1.986701 | Yike | 1.504093 | Caps | 0.254242 | Hans Sam | -2.10256 | Mikyx | 1.030655 | 2.673129 |
| Fnatic | ######## | 1 | Oscarinin | 3.590208 | Razork | 4.108092 | Humanoid | 2.894897 | Noah | -0.18516 | Jun | 0.273675 | 10.68171 |
| G2 Esports | ######## | 0 | BrokenBla | -1.95934 | Yike | -4.40946 | Caps | -5.38221 | Hans Sam | -1.53296 | Mikyx | -4.09181 | -17.3758 |
| MAD Lions | ######## | 1 | Myrwn | -3.15571 | Elyoya | 6.275819 | Fresskowy | 2.145112 | Supa | 3.821645 | Alvaro | 5.370783 | 14.45765 |
| Fnatic | ######## | 0 | Oscarinin | -4.71092 | Razork | -4.62242 | Humanoid | -3.94389 | Noah | -2.8546 | Jun | -2.18987 | -18.3217 |
| Movistar R | ######## | 0 | Summit | -1.78225 | Oddie | -4.47313 | Lava | -4.90491 | Ceo | -3.31879 | Lyonz | -3.4999 | -17.979 |
| Movistar R | ######## | 0 | Summit | -1.06568 | Oddie | -4.44019 | Lava | -6.26764 | Ceo | -3.91779 | Lyonz | -2.84018 | -18.5315 |
| Gen.G | ######## | 0 | Kiin | -1.04622 | Canyon | -0.6573 | Chovy | -0.36708 | Peyz | -1.75931 | Lehends | -0.39545 | -4.22535 |
| T1 | ######## | 1 | Zeus | 0.587907 | Oner | 1.285279 | Faker | -1.05116 | Gumayusi | 0.261734 | Keria | 1.94815 | 3.031907 |
| T1 | ######## | 0 | Zeus | 3.554488 | Oner | -2.89608 | Faker | -3.40847 | Gumayusi | -0.32652 | Keria | -1.07947 | -4.15605 |
| Gen.G | ######## | 1 | Kiin | -0.20336 | Canyon | 3.178114 | Chovy | 4.452623 | Peyz | 2.934868 | Lehends | 4.754281 | 15.11653 |
| T1 | ######## | 0 | Zeus | -1.05379 | Oner | -4.53944 | Faker | -0.27732 | Gumayusi | -2.72806 | Keria | -2.25977 | -10.8584 |
| Gen.G | ######## | 1 | Kiin | 3.386617 | Canyon | 3.067307 | Chovy | 5.938093 | Peyz | 5.700998 | Lehends | 3.104739 | 21.19775 |
| Movistar R | ######## | 1 | Summit | 2.005591 | Oddie | 6.755238 | Lava | 4.023775 | Ceo | 3.324923 | Lyonz | 1.767923 | 17.87745 |
| Movistar R | ######## | 1 | Summit | 8.12034 | Oddie | 2.589145 | Lava | 2.550625 | Ceo | 3.40101 | Lyonz | 6.460527 | 23.12165 |
| Dplus KIA | ######## | 1 | Kingen | 5.922709 | Lucid | 3.489212 | ShowMake | 2.553824 | Aiming | 5.737383 | Kellin | 4.631629 | 22.33476 |
| Dplus KIA | ######## | 1 | Kingen | 1.583778 | Lucid | 1.14042 | ShowMake | 2.79732 | Aiming | 2.361898 | Kellin | 0.937894 | 8.82131 |
| Hanwha Li | ######## | 1 | Doran | 1.467947 | Peanut | 2.87249 | Zeka | 0.623948 | Viper | -0.69499 | Delight | 2.113729 | 6.383121 |
| Hanwha Li | ######## | 1 | Doran | 2.175799 | Peanut | 3.394086 | Zeka | 2.283681 | Viper | 0.958456 | Delight | 3.243942 | 12.05596 |
| T1 | ######## | 1 | Zeus | 1.82123 | Oner | 5.357774 | Faker | 6.764453 | Gumayusi | 1.267126 | Keria | 6.789495 | 22.00008 |
| T1 | ######## | 1 | Zeus | 5.861659 | Oner | 2.384737 | Faker | 1.205412 | Gumayusi | -1.6869 | Keria | 4.091431 | 11.85633 |
| Gen.G | ######## | 1 | Kiin | 7.073354 | Canyon | 4.664657 | Chovy | 3.986498 | Peyz | 0.941273 | Lehends | 5.498975 | 22.16476 |
| Gen.G | ######## | 1 | Kiin | 2.077573 | Canyon | 6.580879 | Chovy | 4.230881 | Peyz | 1.183274 | Lehends | 6.148025 | 20.22063 |
| Vikings Es | ######## | 0 | Kratos | 1.875325 | Gury | -3.08838 | Kati | 2.669882 | Shogun | 0.076392 | Kairi | 1.498626 | 3.031841 |

## Model 2: Random Forest

The next model we employed with our new dataset was Random Forest. While the relationship between various statistics is likely linear due to the focus being on a singular lane doing their own job, the relationships between roles and how they affect a game could be nonlinear, with roles interacting differently between each other. For this reason, Random Forest was the perfect choice.

To find how impactful each role is to the outcome of a game, we ran a random forest model with the input being the player score values for each game, as aforementioned. Again, 20% of the data was used for testing, and 80% of the data was used for training out of a total 1900 games.

To find the importance of each role in determining game result, we once again retrieved the numerical values of each coefficient of our features (the player performances in each role) and graphed them in order to determine the order of importance for these features

## Data Preprocessing 3

With the player scores obtained from the logistic regression model and how impactful each role is obtained from random forest, we can determine the overall impact of a player on their team, and the team's overall strength. The formula for this is simple: (player_score * role_impact) for each player, summing it to get team strength.

In the next step, we applied Feature Scaling to better balance each team and to account for teams playing in weaker regions. To do this, we took the Regional Strength Score found on the RIOT Games website, which is computed by AWS, to determine how well a region performs overall on the international stage. The multipliers are computed through analyzing historical team performances from the region across all seasons of the game, from the first ever world championship to present.

The region multipliers are summarized in the table below:

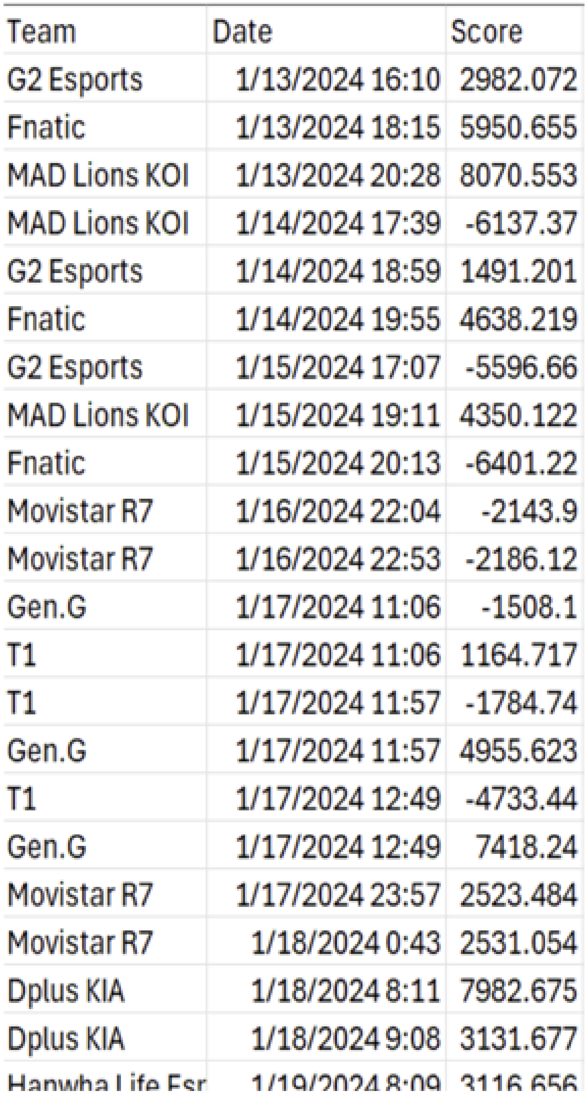
# Region Multipliers Table

| | Region | Multiplier |
|---|---|---:|
| 0 | Korea (LCK) | 1873 |
| 1 | China (LPL) | 1826 |
| 2 | North America (LCS) | 1486 |

| | Region | Multiplier |
|---|---|---|
| 3 | Europe (LEC) | 1542 |
| 4 | Asia-Pacific (PCS) | 863 |
| 5 | Vietnam (VCS) | 839 |
| 6 | Latin America (LLA) | 583 |
| 7 | Brazil (CBLOL) | 634 |

To apply these multipliers, we multiplied our calculated team strengths by the multiplier for each team's respective region. Before applying the multiplier, however, we scaled all the strengths up by adding a constant to them (magnitude of lowest value in our data) so that there were no longer negative team strengths. This prevented the region multipliers from increasing variance in our data.

We then created our final CSV containing only the dates and adjusted performance scores of each team across each game.



| Team | Date | Score |
|---|---|---|
| G2 Esports | 1/13/2024 16:10 | 2982.072 |
| Fnatic | 1/13/2024 18:15 | 5950.655 |
| MAD Lions KOI | 1/13/2024 20:28 | 8070.553 |
| MAD Lions KOI | 1/14/2024 17:39 | -6137.37 |
| G2 Esports | 1/14/2024 18:59 | 1491.201 |
| Fnatic | 1/14/2024 19:55 | 4638.219 |
| G2 Esports | 1/15/2024 17:07 | -5596.66 |
| MAD Lions KOI | 1/15/2024 19:11 | 4350.122 |
| Fnatic | 1/15/2024 20:13 | -6401.22 |
| Movistar R7 | 1/16/2024 22:04 | -2143.9 |
| Movistar R7 | 1/16/2024 22:53 | -2186.12 |
| Gen.G | 1/17/2024 11:06 | -1508.1 |
| T1 | 1/17/2024 11:06 | 1164.717 |
| T1 | 1/17/2024 11:57 | -1784.74 |
| Gen.G | 1/17/2024 11:57 | 4955.623 |
| T1 | 1/17/2024 12:49 | -4733.44 |
| Gen.G | 1/17/2024 12:49 | 7418.24 |
| Movistar R7 | 1/17/2024 23:57 | 2523.484 |
| Movistar R7 | 1/18/2024 0:43 | 2531.054 |
| Dplus KIA | 1/18/2024 8:11 | 7982.675 |
| Dplus KIA | 1/18/2024 9:08 | 3131.677 |
| Hanwha Life Esr | 1/19/2024 8:09 | 3116.656 |

## Model 3: Long Short-Term Memory

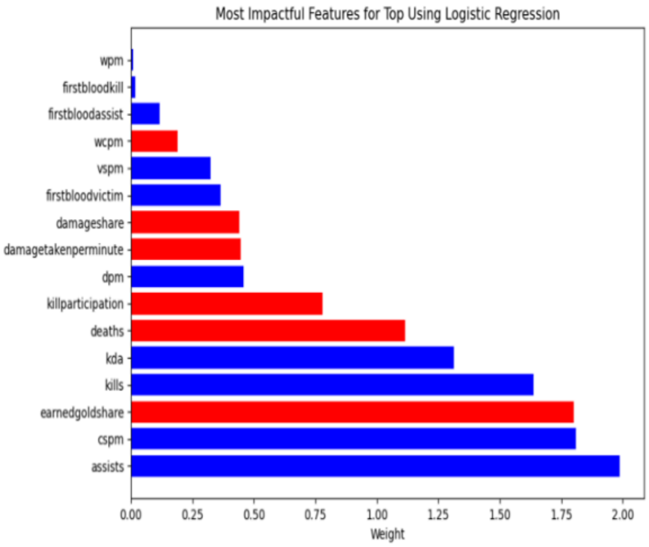
As we have the data of every game throughout the year (excluding Worlds), we decided to use an LSTM model to predict the strength of teams on October 14th, which is around the midpoint of the actual Worlds tournament. As an LSTM is able to take in information that happens over a period of time (as memory), it can capture any trends that appear over the time period and use that to predict a point of time in the future. As the performance of teams is generally variable and not all teams are always performing well or poorly, an LSTM is perfect to find what teams will be strong during Worlds.

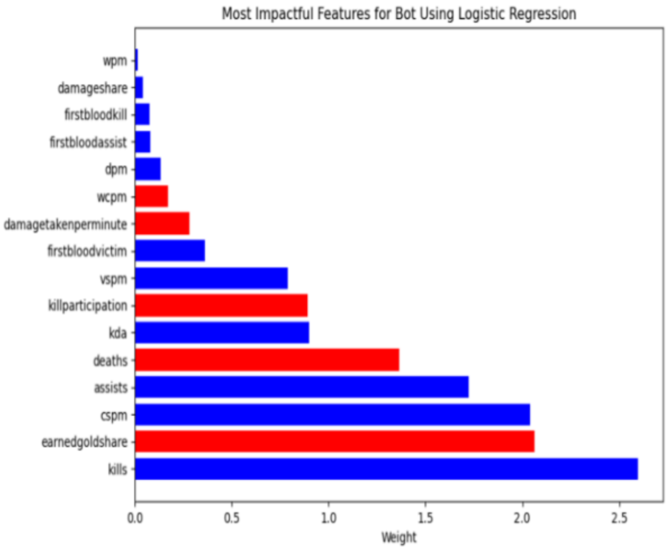
As such, we ran LSTM on each team in order to determine a score for the team during Worlds, which produced a graph for each team and a trendline (with a prediction). Scores were scaled to be between 0 and 1, our batch number was set to 32 and our epoch number was set to 300. With this, we were able to get a prediction, as well as calculate the error in our predictions in the form of MSE and MAPE.

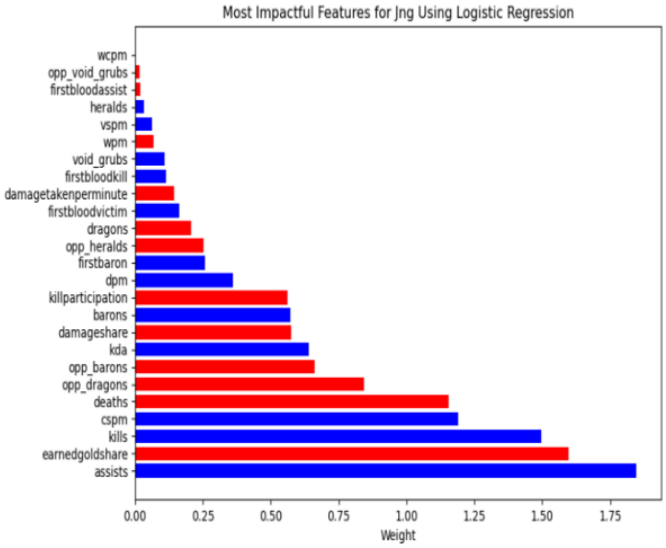
# Results and Discussion

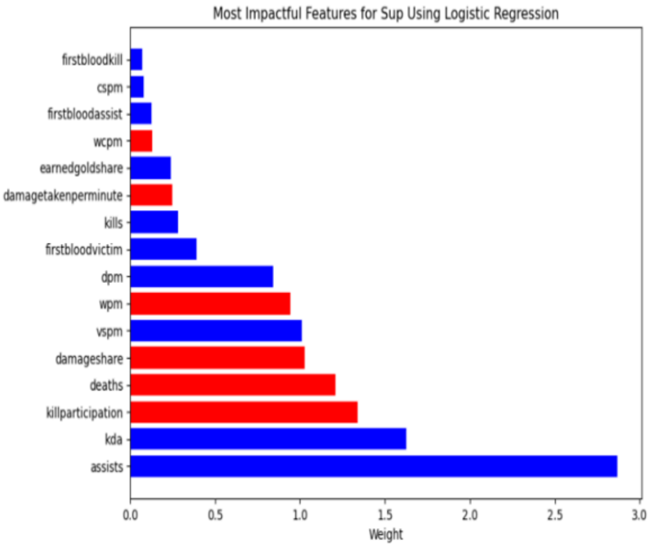
## Model 1: Logistic Regression

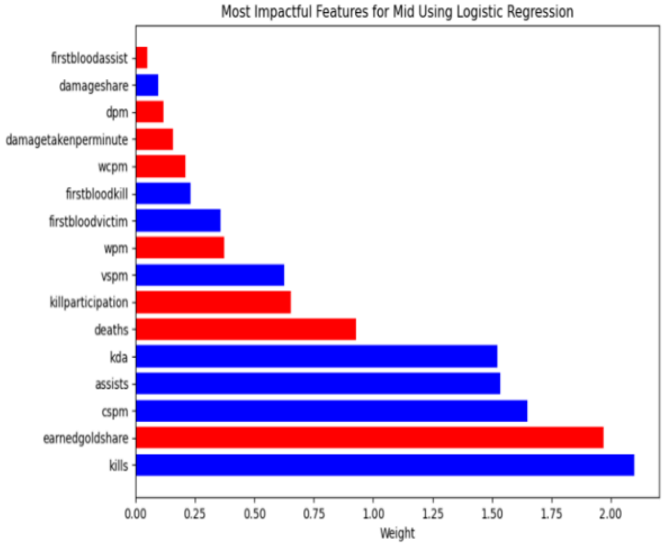
### Data Visualizations











## Discussion

From our model, we can clearly see the most impactful features for each role based on the weight, as denoted. These differ depending on the role; for top, jungle, and support, the feature that matters the most is assists, but for mid and bot, kills matter the most. Per the graphs, it is evident that assists are by far the most important factor for supports, with its weight being about 2.8, while top has a weight of right under 2.0 and jungle only having a weight of about 1.8 for assists. Meanwhile, kills matter more for botlaners compared to midlaners, with a weight of about 2.6 to 2.1, respectively.

We have reason to believe that this is accurate; midlaners and botlaners are often thought to have what is known as “carry-potential,” or the ability to single-handedly win a fight or subsequently, even the entire game. If they are carrying, or in order to carry a game, they would need many kills. As such, kills heavily impacting midlane and botlane performance would be accurate. Meanwhile, toplaners, supports, and junglers often set up the rest of their team for success, so these roles having assists indicate good performance makes sense. The other metrics also line up with what can be reasonably expected from each role.

Some initially surprising features with negative correlation include kill participation and gold share. However, upon deeper analysis, this makes sense. In regards to kill participation, for very one-sided games, kills are usually very high on one team and very low on the other. In games like these, some players on the winning team have a low kill participation as it naturally gets difficult to maintain a high ratio when there is a large amount of total kills. On the other hand, the losing team usually has a higher kill participation as there are not very many kills to begin with, so even having 1 kill or assist could result in a large kill participation. As such, having lower kill participation can very well indicate the team will win the game, so there is a negative correlation.

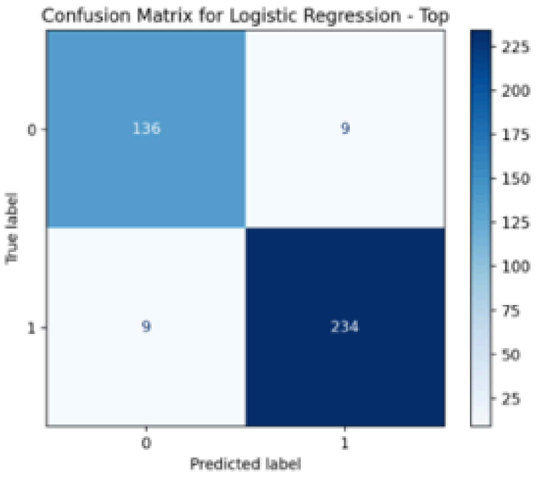
Meanwhile, a similar explanation exists for gold share. As gold share is proportional, if one player has low gold share, it can be inferred that gold share is very skewed; as such, the other 4 players would have most of the gold, which can indicate good performance from them. If the gold share is relatively even among the players and your gold share is average, then it means that everyone on the team is performing about the same, so no one really has a big advantage and thus the team as a whole is likely not ahead of their opponents. As such, low gold share can point to better team performance.

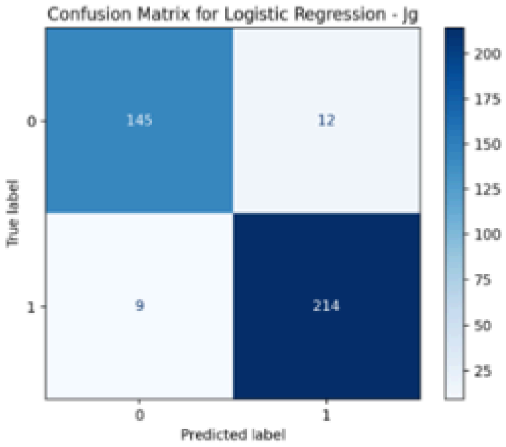
| | Role | Accuracy | Precision | Recall | F1 Score |
|---|---|---:|---:|---:|---:|
| 0 | Top | 0.9536 | 0.9536 | 0.9536 | 0.9536 |
| 1 | Jungle | 0.9447 | 0.9447 | 0.9447 | 0.9447 |
| 2 | Mid | 0.9316 | 0.9326 | 0.9316 | 0.9311 |
| 3 | Bot | 0.9155 | 0.9158 | 0.9155 | 0.9148 |
| 4 | Sup | 0.8941 | 0.8937 | 0.8941 | 0.8931 |

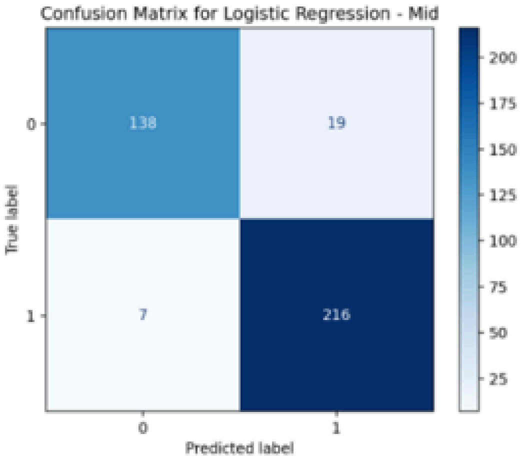
As our model was run off of a random sample of 80% of the data and tested on the remaining 20% (380 games), we were able to produce accuracy calculations based off of predicting the outcome of the games. For the toplane, the accuracy of our logistic regression model was 0.9536, i.e. it was able to predict the outcome of the game correctly 95.36% of the time. For the jungle, this statistic was 0.9447; for midlane, 0.9316; for ADC, 0.9155; and for support, 0.8941. These are relatively high statistics overall–the combined botlane of ADC and support suffer a bit more in terms of accuracy due to some unique characteristics of the two roles. Supports, especially, are difficult to pin down because of how different things are prioritized, depending on the champion of the support.

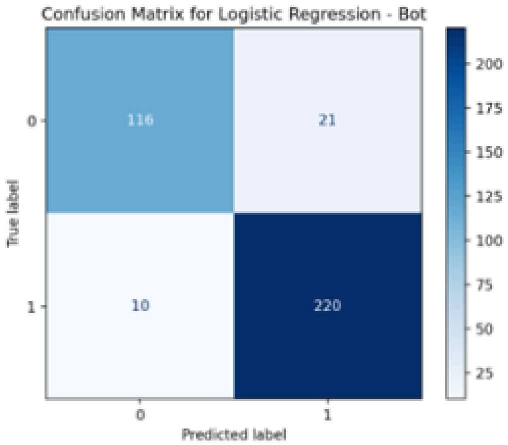
From sk-learn, we also have precision, recall, and F1 scores. They are all relatively similar, indicating that there is a healthy balance between precision and recall, so our model should be finding the true positives with minimal error.

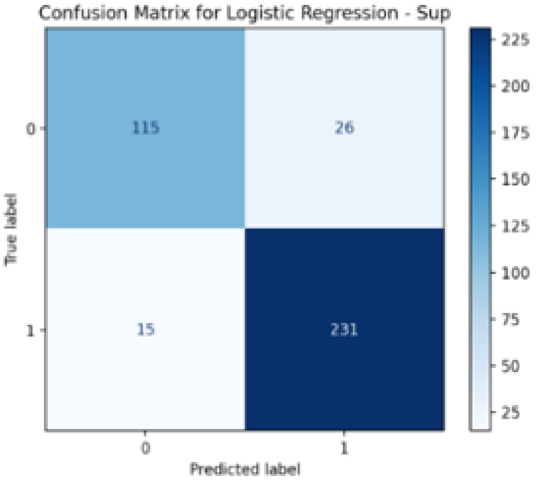
Here are our associated confusion matrices for each role.







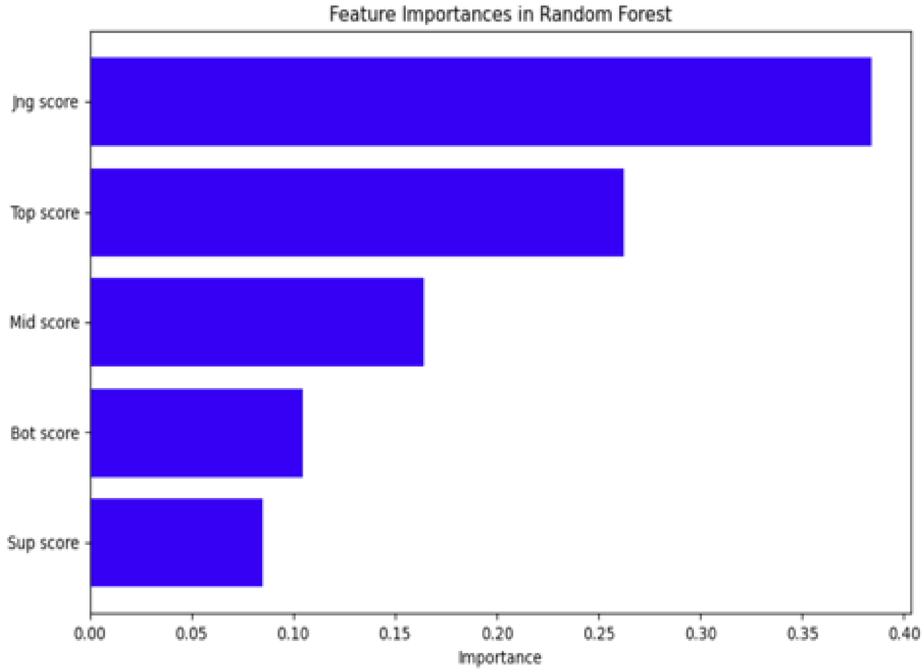




Overall, our model is strong and runs with minimal errors. We were able to garner the features that best correlate to strong performance per position with high accuracy, precision, recall, and F1 score. The model likely performed well due to the sheer number of games we were able to train the model on (almost 2000 games), with each game having detailed statistics. As such, we were able to find which statistics would lead to a win or a loss, per position. Our model did decrease in terms of accuracy, etc. especially in the case of supports, as mentioned earlier, due to the inconsistent nature of what they prioritize. There are a few different “classes” of supports in pro-play that, depending on what they are, will result in different stats. For instance, enchanters are champions that primarily focus on healing or shielding other characters, but are usually very fragile themselves. Meanwhile, engage-supports/tanks are played to soak up damage. As of right now, our model does not differentiate between the types of supports, so it assumes that the champions themselves are monolithic and should prioritize the same features, which creates volatility in the results.

# Model 2: Random Forest

## Data Visualizations



## Discussion

Our model clearly indicates that jungle is the most important role, followed by top, mid, bot, then support. Jungle has by-far the highest weight, being 0.385, whereas top is 0.251, mid is 0.181, bot is 0.110, and support is 0.072. This is rather reasonable and can be believed to be accurate–junglers set the tempo for the game and are expected to impact every other role’s performance, either through ganking (essentially ambushing the enemy laner), or gaining objectives, such as defeating void grubs or dragons, which can be a great boon to the entire team (for instance, void grubs allow everyone on the team to essentially defeat towers easier and quicker, while defeating enough dragons will produce a permanent buff for the team, aka make the team stronger). Junglers have consistently had a large role in the game and many Worlds Finals were decided by outstanding Jungle performances, such as the championships from 2018-2021.

Meanwhile, this year has, for the most part, had a very top-sided meta, due to the nature of toplane champions being very strong as well as the existence and importance of void grubs–as void grubs exist on the topside of the map, if a toplaner is doing better than their opponent, they can disrupt the enemy jungler from being able to take the void grubs and potentially help their own jungler secure them instead. Good toplane players have a notable impact on the game, and there is generally a lot of hype and excitement around the best toplane players due to the role's potential for flashy and critical plays.

Midlaners, on the other hand, are also a crucial entity to a team, but perhaps less so than a toplaner or jungler. A midlaner also has the potential to "roam," or visit another lane, potentially making an impact across the map and outside of their lane. Since the lane is in the middle of the map, they are able to easily access and join fights over objectives. Botlaners, on the other hand, have noticeably had minimal impact in the metas leading up to worlds. One of the top botlane players in recent years, Gumyausi of T1, likened the role to an egg in instant noodles–without water, noodles, and soup base (the other roles), a botlaner cannot make an impact. However, it can be noted that in hindsight, Worlds featured some crucial performances from botlaners that effectively decided the entire game; as it was impossible for our dataset to take this into account, this could potentially introduce a level of error in our results.

It makes sense that the model would output support as the lowest impact role, due to how difficult it is to quantify the contributions of a support–although there are some metrics that can point to the makings of a good support, such as high vision score, many times supports are responsible for starting teamfights and protecting the glass-cannon damage dealers during the fight, which is harder to quantify. Additionally, supports by themselves, in professional play, do not have carry-potential, i.e. a support player cannot single-handedly win a game by themselves. However, they can change the entire course of a game by starting a crucial teamfight for their team that they win, but it is very difficult to quantify this for it is truly a single moment in time that can change the course of a game. Additionally, since our model is based off of logistic regression and it was mentioned earlier how the value of supports may not be reflected accurately in that model, our random forest would be affected a bit as well. Still, overall, support players as individuals likely have the lowest impact on the game, which our model reflects.

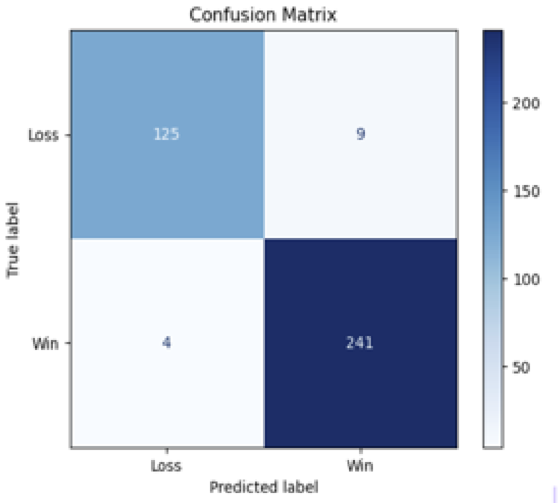
## Quantitative Metrics and Analysis

| | Metric | Value |
|---|---|---|
| 0 | Accuracy | 0.9700 |
| 1 | Precision | 0.9700 |

| | Metric | Value |
|---|---|---|
| 2 | Recall | 0.9700 |
| 3 | F1-score | 0.9700 |

As with logistic regression, we were able to produce accuracy, precision, recall, and f-1 score calculations based on training on 80% of the data and testing on 20%. With the feature weights generated by Random Forest, we were able to accurately predict the result of games from player performances 97% of the time. The high accuracy, along with the varied estimated feature weights, reveal that player importance definitely does exist in League of Legends. Specific roles have more impact on the game than others.The precision, recall and F1-score values are all once again very similar (identical to 2 significant figures), meaning our model should be finding true positives with minimal error.

Below the confusion matrix for our model’s predictions:



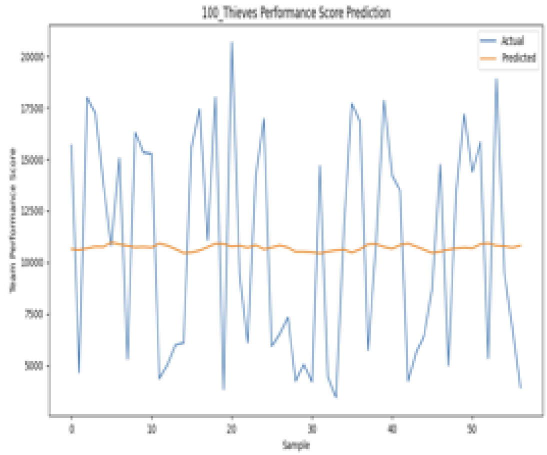
Overall, our Random Forest model performs well with minimal error, similar to logistic regression, likely due to a large number of games and how accurate our logistic regression model was (since we used scores derived from the output of the first model). Our model may not necessarily accurately reflect the impact of supports as explained prior, but overall and in terms of relative impact, our model does a good job of quantifying the impact of a role through the feature coefficients. We have high precision and accuracy values, indicating the high quality of our model.

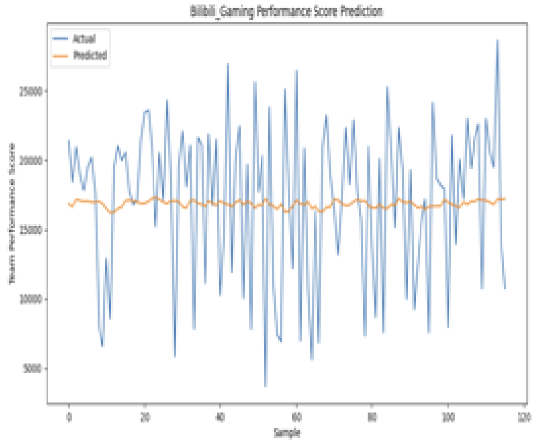
# Model 3: LSTM

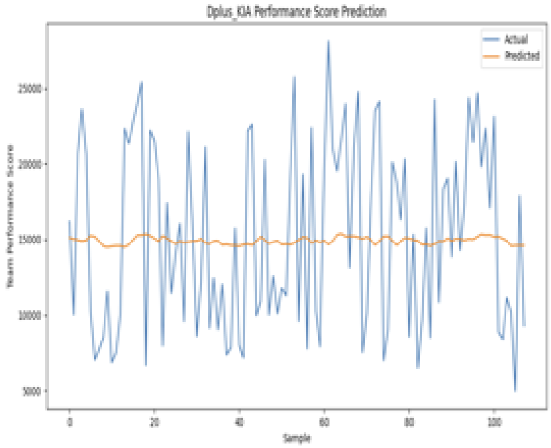
## Visualizations

>

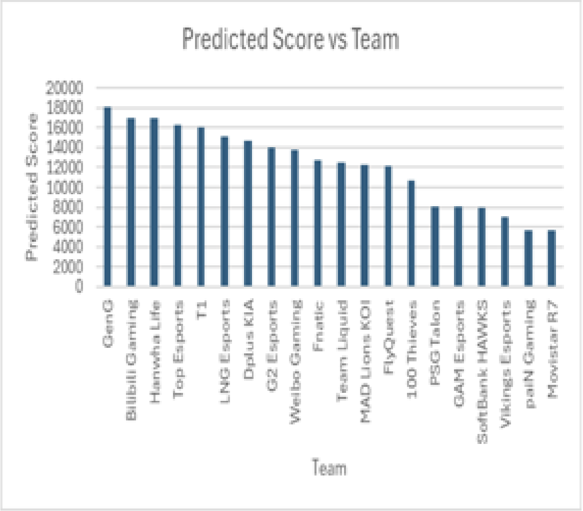
Sample of end results for 3 of the 20 teams

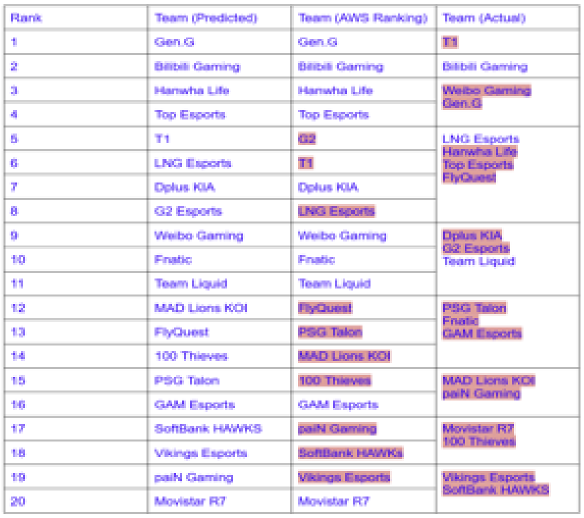






Visualization of outputted predicted scores, corresponding to the teams accordingly.





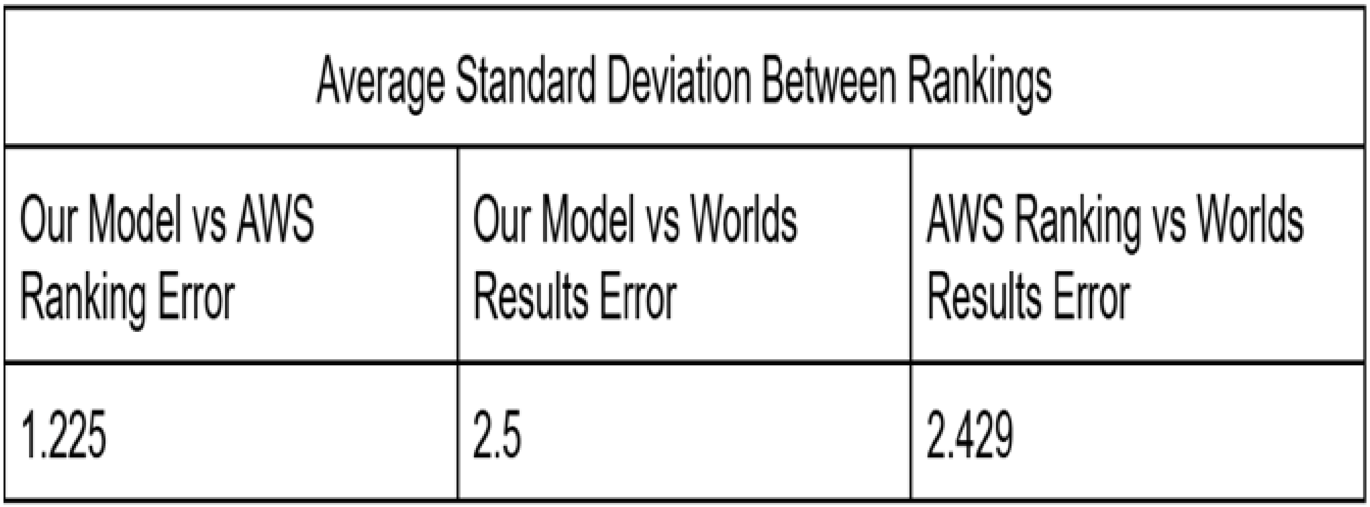
| Rank | Team (Predicted) | Team (AWS Ranking) | Team (Actual) |
|---|---|---|---|
| 1 | Gen.G | Gen.G | T1 |
| 2 | Bilibili Gaming | Bilibili Gaming | Bilibili Gaming |
| 3 | Hanwha Life | Hanwha Life | Weibo Gaming<br>Gen.G |
| 4 | Top Esports | Top Esports | |
| 5 | T1 | G2 | LNG Esports<br>Hanwha Life<br>Top Esports<br>FlyQuest |
| 6 | LNG Esports | T1 | |
| 7 | Dplus KIA | Dplus KIA | |
| 8 | G2 Esports | LNG Esports | |
| 9 | Weibo Gaming | Weibo Gaming | Dplus KIA<br>G2 Esports<br>Team Liquid |
| 10 | Fnatic | Fnatic | |
| 11 | Team Liquid | Team Liquid | |
| 12 | MAD Lions KOI | FlyQuest | PSG Talon<br>Fnatic<br>GAM Esports |
| 13 | FlyQuest | PSG Talon | |
| 14 | 100 Thieves | MAD Lions KOI | |
| 15 | PSG Talon | 100 Thieves | MAD Lions KOI<br>paiN Gaming |
| 16 | GAM Esports | GAM Esports | |
| 17 | SoftBank HAWKS | paiN Gaming | Movistar R7<br>100 Thieves |
| 18 | Vikings Esports | SoftBank HAWKs | |
| 19 | paiN Gaming | Vikings Esports | Vikings Esports<br>SoftBank HAWKS |
| 20 | Movistar R7 | Movistar R7 | |

## Discussion

>

The LSTM produced the final results: a ranking of each team at Worlds. In the visualizations section, there is a table with the predicted ranks of each team, as well as the LoL Esports Global Power Rankings powered by AWS produced right before Worlds, and the actual results in the last column. Note that for the AWS Ranking, teams that did not make it to worlds were removed from the ranking to ensure consistency across the table. Additionally, for the actual results, due to the formatting of the tournament, some teams share rankings–for instance, Weibo Gaming and Gen.G share 3rd-4th place since they both lost in the semifinals (to Bilibili Gaming and T1, respectively). As such, those cells have been merged since there is no way of telling which team is 3rd or 4th.

There are clear discrepancies between the actual results and the predicted results, but there is a less drastic difference between the predicted results and the AWS rankings. The AWS rankings were not able to match up very well to the actual results either, which proves that a ranking of teams may be inherently flawed from the beginning–this logic holds in traditional sports as well, where it is beyond extremely rare for someone to predict a perfect March Madness bracket, even with the best minds and technologies at work.



| Average Standard Deviation Between Rankings | | |
|---|---|---|
| Our Model vs AWS Ranking Error | Our Model vs Worlds Results Error | AWS Ranking vs Worlds Results Error |
| 1.225 | 2.5 | 2.429 |

We were able to come within 1.2 ranks in comparison to the AWS Rankings, and 2.5 in comparison to Worlds. Although we are unable to ascertain how exactly the AWS Rankings are calculated, our rankings come fairly close to the AWS rankings, implying that either our methodologies are similar or they are different but come to similar conclusions, validating our process. Meanwhile, we have a 2.5 average standard deviation when it comes to the actual Worlds results, which can likely be attributed to the factors discussed above (oversimplification and the variable nature of sports). This means that our model performed a little worse than the AWS Rankings, for the AWS Rankings produced an average standard deviation of 2.429, while our model is 2.5, relative to the actual results of Worlds, so the slight discrepancies between our models results in more inaccuracy for us.

Our model was able to accurately predict the placement of 3 teams–Bilibili Gaming, LNG Esports, and Team Liquid but also had some large mispredictions worth noting. For instance, our model ranked G2 at 8, whereas the AWS ranking put G2 at 5–in all actuality, our model came much closer to the true result, as G2 placed 9th-11th at the actual tournament. Though we are unsure of the actual AWS methodology, they say they value strength of opponent, margin of victory, and context of play (valuing say, international tournaments over regular season games). G2 performed well domestically and had a high finish at the Mid-Season Invitational, the other international event held in May; as such, it makes sense that AWS would rate them highly, whereas our model focused more on actual game performance and did not differentiate between games. While our model worked better in this instance, the average standard deviation in comparison to the actual results for the AWS rankings was 2.429, whereas our model was 2.5, so the AWS model is generally more accurate–it could be that looking at game performance is less accurate than the amalgamation of features used by AWS to determine their power rankings.

Clearly, our model was unable to predict the winner of Worlds accurately. T1 barely made it to Worlds, entering as the last seed from the Korean region and winning their Worlds qualification game in an extremely close best-of-5 that took all 5 games to decide the winner. The team underperformed heavily during the summer, which likely caused our model to rank them lower. In this sense, the AWS rankings agreed, as both our model and AWS did not have T1 in the top 4. Based on pure results throughout the season, Gen.G were favorites to win Worlds, having won the Mid-Season Invitational as well as the spring season, and came 2nd place in the summer season in the Korean league. However, there is superstition that some players on Gen.G underperform at Worlds, perhaps due to the immense amount of pressure or another factor. Meanwhile, there is also a superstition that T1–the team of the most renowned esports player of all time, Faker–steps up and overperforms at Worlds. These are, for the most part, claims made by fans and haters, but there may be a kernel of truth to the superstitions that cannot be reflected by our current models.

While our model predicted FlyQuest to rank around 13, they actually placed 5-8th, getting eliminated in the quarterfinals (having made it to the playoff stage of the competition). Some argue that FlyQuest got “easy” matchups in the round before playoffs, only needing to beat minor region teams and another team from their own region (North America) to make it to playoffs. Additionally, FlyQuest had a rather poor showing at the Mid-Season Invitational, getting knocked out before even making it to the main stage of the competition. To FlyQuest’s credit, they had a strong summer season in North America and finished as the first seed in the region; however, North America has generally been a weaker region, which is reflected in the Regional Strength Score that was used as a scaling factor, as denoted in Data Preprocessing 3. Although they are the 1st seed from North America, due to Western teams being generally weaker than Eastern teams for the past decade and generally not performing well at worlds, FlyQuest making it to playoffs was a shock to many and defied many analyst–and evidently model–predictions.

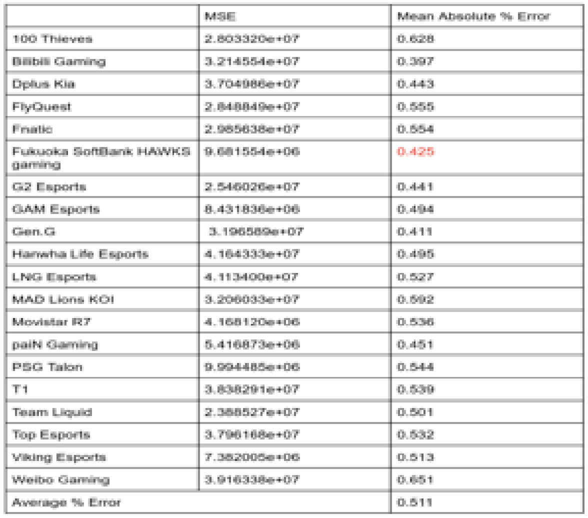
Another large discrepancy was Weibo Gaming. Though they performed poorly in the spring season in China, they were able to pick it up in the summer, placing 2nd in the Chinese summer playoffs; still, they barely made it to Worlds, as they lost to LNG Esports in the Chinese Regional Gauntlet that decides the 3rd and 4th seeds who make it to Worlds from China. As such, they had to beat JD Gaming and qualified to Worlds as the 4th seed from China. Although both our model and AWS predicted Weibo Gaming to barely miss playoffs (placing 9th), they made it and were eliminated in semifinals, which could potentially be attributed to the team stepping up to the occasion–they even beat LNG Esports to make it to the semifinals, the very team they had lost to previously for the 3rd seed.

Overall, although our model was not able to correctly predict who would win Worlds 2024, it came relatively close to the AWS rankings, indicating that there is some validity to our model’s results and that the problems may come inherently from a ranking of teams in such a way.

## Quantitative Metrics and Analysis

>

Unfortunately, our LSTM model was not as accurate as the other models. This is indicated by both our individual team results/graphs and the final results (a prediction of who will win at worlds). In regards to the former, our model was able to produce MSE and Mean Absolute % Error numbers, shown in the table below.



| | MSE | Mean Absolute % Error |
|---|---|---|
| 100 Thieves | 2.803320e+07 | 0.628 |
| Bilibili Gaming | 3.214554e+07 | 0.397 |
| Dplus Kia | 3.704986e+07 | 0.443 |
| FlyQuest | 2.848849e+07 | 0.555 |
| Fnatic | 2.985638e+07 | 0.554 |
| Fukuoka SoftBank HAWKS gaming | 9.681554e+06 | 0.425 |
| G2 Esports | 2.546026e+07 | 0.441 |
| GAM Esports | 8.431836e+06 | 0.494 |
| Gen.G | 3.196589e+07 | 0.411 |
| Hanwha Life Esports | 4.164333e+07 | 0.495 |
| LNG Esports | 4.113400e+07 | 0.527 |
| MAD Lions KOI | 3.206033e+07 | 0.592 |
| Movistar R7 | 4.168120e+06 | 0.536 |
| paiN Gaming | 5.416873e+06 | 0.451 |
| PSG Talon | 9.994485e+06 | 0.544 |
| T1 | 3.838291e+07 | 0.539 |
| Team Liquid | 2.388527e+07 | 0.501 |
| Top Esports | 3.796168e+07 | 0.532 |
| Viking Esports | 7.382005e+06 | 0.513 |
| Weibo Gaming | 3.916338e+07 | 0.651 |
| Average % Error | | 0.511 |

Our models were only capable of predicting future values with an error of about 50%, meaning that the prediction of the next value would likely be within plus or minus of 50% of the true value–a very large error margin, especially in comparison to our logistic regression and random forest models.

There are a number of potential explanatory factors. For one, the performance of a team can vary greatly throughout a season. A team that starts off performing very well can experience some troubles later on in the year, or vice versa. As such, our model can become confused, not being able to capture any trends due to the variability of performances game-to-game. Additionally, we are undoubtedly simplifying down League of Legends immensely through assigning scores to each role/player; it is possible that our metric of determining what makes a player good is inherently flawed due to a misunderstanding of the game or perhaps the game's innate complexity, which would lead to difficulties in predicting performance.

Still, the largest determinant of such a large accuracy error is likely due to the fact that our model relies on an element of consistency or induction; unfortunately, just because a team plays well one day does not mean they will play well on the next day. Just like with any sports predictor, the true results are sometimes very surprising; though strong teams will generally beat weak teams, sometimes the underdog surprises us.

## Model Comparisons

The first stage is right after our logistic regression model, meaning we only have weights indicating how important each statistic is to winning the game for each role. As such, we multiply each weight by the statistics for each game, creating a total team score. Averaging the team scores across every match produces the final performance rating.

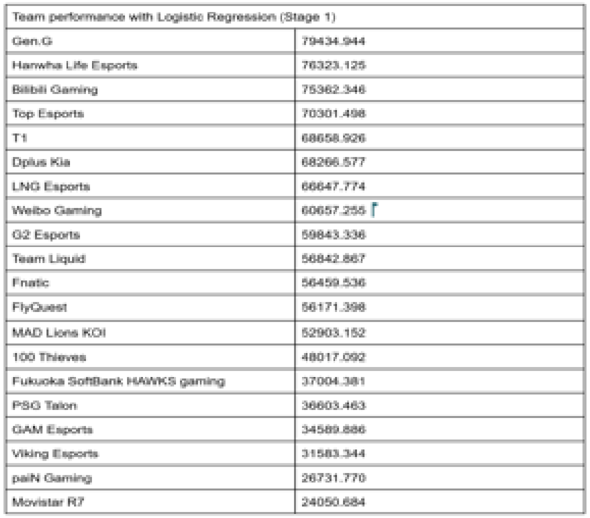
The second stage is right after our random forest model. Now, our team score is calculated by multiplying a player's score (calculated by the weights from the logistic regression model) by the role impact weight, and is calculated for every game and then averaged, as it was in the first stage. The final stage is the result of our project, with the predictions from LSTM.

This provided us with team performance ratings:

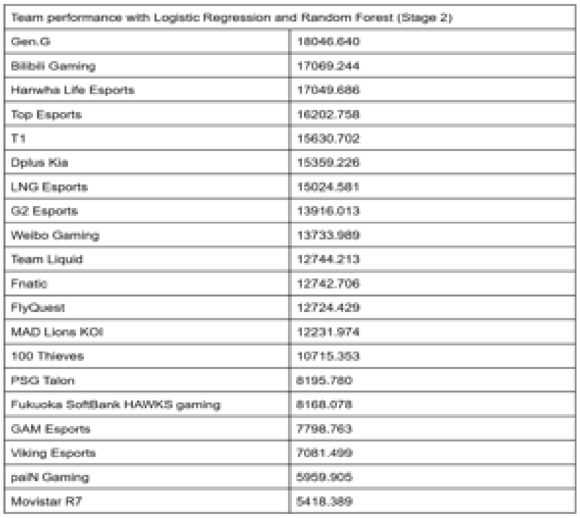
1. Only taking player performance into account
2. Taking player performance and role impact into account
3. Taking player performance, role impact, and change over time into account

The team rankings from each stage are as follows.

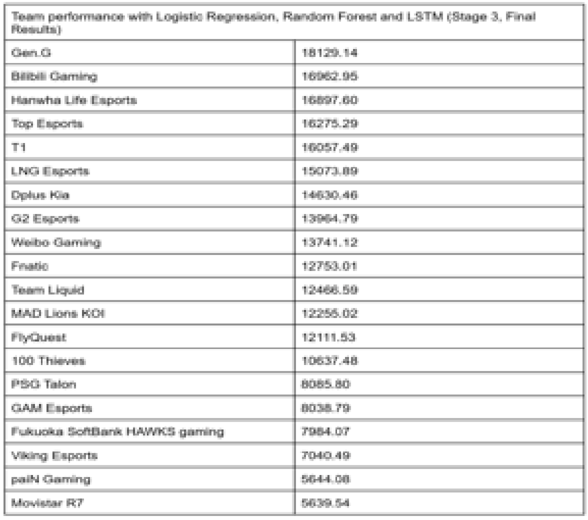
Our project was built off of models working off of each other. We used logistic regression to find the factors that make a player good at their role, and used random forest to find how impactful each role is to the outcome of a game. Finally, LSTM worked to predict the strength of teams during the midpoint of when Worlds 2024 is (October 14th), based on its memory of all the games across the season. While our first two models were rather accurate, with high accuracy and precision across the board, the LSTM model was not, with high MSE and Mean Absolute Percentage Error (MAPE). To illustrate the effectiveness of each model, the Regional Strength Score was applied to our summed team performance ratings from each stage in our model.



| Team performance with Logistic Regression (Stage 1) | |
|---|---|
| Gen.G | 79434.944 |
| Hanwha Life Esports | 76323.125 |
| Bilibili Gaming | 75362.346 |
| Top Esports | 70301.498 |
| T1 | 68658.926 |
| Dplus Kia | 68266.577 |
| LNG Esports | 66647.774 |
| Weibo Gaming | 60657.255 |
| G2 Esports | 59843.336 |
| Team Liquid | 56842.867 |
| Fnatic | 56459.536 |
| FlyQuest | 56171.398 |
| MAD Lions KOI | 52903.152 |
| 100 Thieves | 48017.092 |
| Fukuoka SoftBank HAWKS gaming | 37004.381 |
| PSG Talon | 36603.463 |
| GAM Esports | 34589.886 |
| Viking Esports | 31583.344 |
| paiN Gaming | 26731.770 |
| Movistar R7 | 24050.684 |

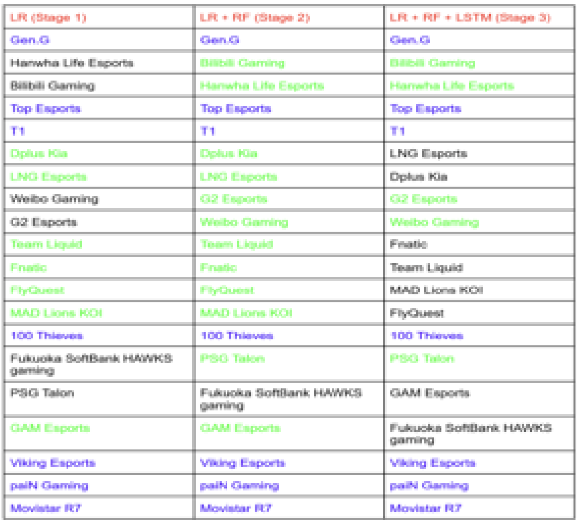


| Team performance with Logistic Regression and Random Forest (Stage 2) | |
|---|---|
| Gen.G | 18046.640 |
| Bilibili Gaming | 17069.244 |
| Hanwha Life Esports | 17049.686 |
| Top Esports | 16202.758 |
| T1 | 15630.702 |
| Dplus Kia | 15359.226 |
| LNG Esports | 15024.581 |
| G2 Esports | 13916.013 |
| Weibo Gaming | 13733.989 |
| Team Liquid | 12744.213 |
| Fnatic | 12742.706 |
| FlyQuest | 12724.429 |
| MAD Lions KOI | 12231.974 |
| 100 Thieves | 10715.353 |
| PSG Talon | 8195.780 |
| Fukuoka SoftBank HAWKS gaming | 8168.078 |
| GAM Esports | 7798.763 |
| Viking Esports | 7081.499 |
| paiN Gaming | 5959.905 |
| Movistar R7 | 5418.389 |



| Team performance with Logistic Regression, Random Forest and LSTM (Stage 3, Final Results) | |
|---|---|
| Gen.G | 18129.14 |
| Bilibili Gaming | 16962.95 |
| Hanwha Life Esports | 16897.60 |
| Top Esports | 16275.29 |
| T1 | 16057.49 |
| LNG Esports | 15073.89 |
| Dplus Kia | 14630.46 |
| G2 Esports | 13964.79 |
| Weibo Gaming | 13741.12 |
| Fnatic | 12753.01 |
| Team Liquid | 12466.59 |
| MAD Lions KOI | 12255.02 |
| FlyQuest | 12111.53 |
| 100 Thieves | 10637.48 |
| PSG Talon | 8085.80 |
| GAM Esports | 8038.79 |
| Fukuoka SoftBank HAWKS gaming | 7984.07 |
| Viking Esports | 7040.49 |
| paiN Gaming | 5644.08 |
| Movistar R7 | 5639.54 |

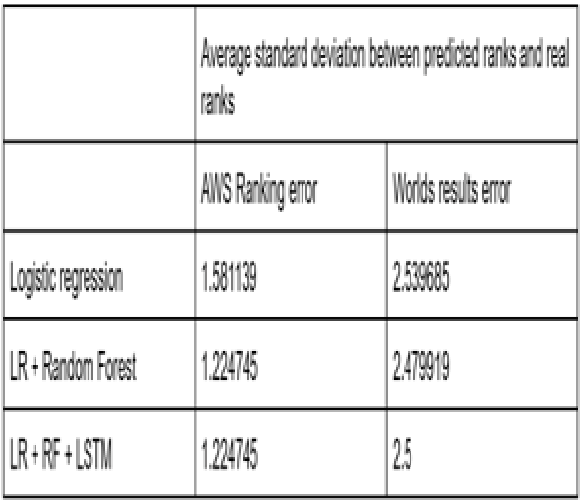
A cross-comparison is as follows.



| LR (Stage 1) | LR + RF (Stage 2) | LR + RF + LSTM (Stage 3) |
|---|---|---|
| Gen.G | Gen.G | Gen.G |
| Hanwha Life Esports | Bilibili Gaming | Bilibili Gaming |
| Bilibili Gaming | Hanwha Life Esports | Hanwha Life Esports |
| Top Esports | Top Esports | Top Esports |
| T1 | T1 | T1 |
| Dplus Kia | Dplus Kia | LNG Esports |
| LNG Esports | LNG Esports | Dplus Kia |
| Weibo Gaming | G2 Esports | G2 Esports |
| G2 Esports | Weibo Gaming | Weibo Gaming |
| Team Liquid | Team Liquid | Fnatic |
| Fnatic | Fnatic | Team Liquid |
| FlyQuest | FlyQuest | MAD Lions KOI |
| MAD Lions KOI | MAD Lions KOI | FlyQuest |
| 100 Thieves | 100 Thieves | 100 Thieves |
| Fukuoka SoftBank HAWKS gaming | PSG Talon | PSG Talon |
| PSG Talon | Fukuoka SoftBank HAWKS gaming | GAM Esports |
| GAM Esports | GAM Esports | Fukuoka SoftBank HAWKS gaming |
| Viking Esports | Viking Esports | Viking Esports |
| paiN Gaming | paiN Gaming | paiN Gaming |
| Movistar R7 | Movistar R7 | Movistar R7 |

From the rankings from the models above, we can clearly see a difference in the ordering across all 3 models. This means that there is definitely an impact to factoring in role impact and performance over time.

To check for the error of each model, we took the average standard deviation between predicted and real ranks for each of the models regarding both the AWS Rankings and the True results:



| | Average standard deviation between predicted ranks and real ranks | |
|---|---|---|
| | AWS Ranking error | Worlds results error |
| Logistic regression | 1.581139 | 2.539685 |
| LR + Random Forest | 1.224745 | 2.479919 |
| LR + RF + LSTM | 1.224745 | 2.5 |

From our error table, it is evident that applying the role impact from random forest made significant improvements to our model whereas the LSTM with the high error rates could not track patterns well enough, and did not change our results in regards to AWS and had a slight increase in inaccuracy when it came to the actual results.

## Next Steps

Moving forward, there are several ways to improve upon this project. It is clear that there may be an inherent problem with a ranking system in order to predict sports results. As such, it may be more beneficial to pursue a different prediction system, such as predicting head-to-head results between two teams. This could involve other ML models, such as SVMs, decision trees, or neural networks. Additionally, we could work on simulating drafts and taking into account the champions that are picked and banned, as well as finding the champion meta. We could also potentially factor in historical results; for instance, T1 has historically performed well at Worlds, so we could try to take that into account (perhaps through neural network layers or similar). Overall, there are several future steps that could be taken to better predict the winner of Worlds.

# Contributions

| Name | Proposal Contributions |
|---|---|
| Vansh | - Contributed to the proposal topic brainstorming<br>- Came up with potential statistics such as KDA to analyze which would help us make our final predictions<br>- Found the csv files to use as data<br>- Brainstormed the methods to be used for the project<br>- Wrote the methods section of the report<br>- Created the slides for the presentation video |
| Eric | - Brainstormed the methods to be used for the project<br>- Came up with potential statistics such as KDA to analyze which would help us make our final predictions<br>- Worked on setting up Streamlit |
| Joshua | - Created the Gantt chart and contribution table<br>- Brainstormed the methods to be used for the project<br>- Came up with potential statistics such as KDA to analyze which would help us make our final predictions<br>- Proofreading<br>- Made the presentation slides and script<br>- Did the presentation video |
| Derek | - Wrote the references section of the proposal<br>- Contributed to the proposal topic brainstorming<br>- Proofreading<br>- Brainstormed the methods to be used for the project<br>- Wrote the literature reviews |
| Enya | - Came up with potential statistics such as KDA to analyze which would help us make our final predictions<br>- Wrote problem definition section<br>- Wrote the potential results and discussion section<br>- Brainstormed the methods to be used for the project<br>- Arranged meetings with the TA advisor<br>- Worked on setting up the Github |

| Name | Midterm Contributions |
|---|---|
| Vansh | - Created a linear regression model to understand what stats contribute the most to winning per role. |

| Name | Midterm Contributions |
|---|---|
| | - Cleaned data by removing unnecessary columns using scripting.<br>- Brainstormed the methods to be used for the project<br>- Created the contribution table for the midterm checkpoint<br>- Formulated our inputs and outputs for the models and potential insights. |
| Eric | - Wrote the midterm report.<br>- Tabulated Summer and Spring split LPL data.<br>- Assisted in verifying the insights generated by the models and adjusting them.<br>- Brainstormed inputs and outputs for the models and useful features. |
| Joshua | - Tabulated Summer and Spring split LPL data.<br>- Helped with writing midterm report<br>- Cleaned data by removing unnecessary columns using Excel.<br>- Formulated our inputs and outputs for the models and potential insights.<br>- Assisted in verifying the insights generated by the models and adjusting them.<br>- Created the data visualizations |
| Derek | - Created a logistic regression model to understand what stats contribute the most to winning per role.<br>- Assisting in verifying the insights generated by the models and adjusting them.<br>- Identified the ML techniques to be used in this phase.<br>- Formulated our inputs and outputs for the models and potential insights. |
| Enya | - Contributed towards creating the Logistic regression model.<br>- Assisted in verifying the insights generated by the models and adjusting the model.<br>- Updating the Streamlit with the midterm checkpoint.<br>- Formulated our inputs and outputs for the models and potential insights. |

| Name | Final Contributions |
|---|---|
| Vansh | - Data processing for the LSTM model.<br>- Cleaned data by removing unnecessary columns using scripting.<br>- Defining LSTM inputs and outputs.<br>- Developed the final presentation.<br>- Developed video script. |
| Joshua | - Data processing for the LSTM model<br>- Defining LSTM inputs and outputs<br>- Developed the final presentation.<br>- Developed script |

| Name | Final Contributions |
|---|---|
| Eric | - Developed the Random Forest model<br>- Extracted team scores from Random Forest.<br>- Developed the final presentation.<br>- Streamlit page modifications. |
| Derek | - Developing the LSTM model.<br>- Developing the final presentation.<br>- Data processing for the LSTM model.<br>- Extracting team rankings from LSTM for final results. |
| Enya | - Developed the Random Forest model.<br>- Developed the final presentation.<br>- Writing final report sections.<br>- Defined model goals and next steps. |

# Click here to view Gantt Chart

# References

[1] K. Arık, “Machine learning models on MOBA Gaming: League of Legends Winner Prediction,” Acta Infologica, vol. 7, no. 1, pp. 139–151, Jun. 2023. doi:10.26650/acin.1180583

[2] F. Bahrololloomi, S. Sauer, F. Klonowski, R. Horst, and R. Dörner, “A machine learning based analysis of e-sports player performances in League of Legends for winning prediction based on player roles and performances,” Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2022. doi:10.5220/0010895900003124

[3] Q. Shen, “A machine learning approach to predict the result of League of Legends,” 2022 International Conference on Machine Learning and Knowledge Engineering (MLKE), pp. 38–45, Feb. 2022. doi:10.1109/mlke55170.2022.00013

[4] A. Walker, “More people watched League of Legends than the NBA finals,” Kotaku Australia, https://www.kotaku.com.au/2016/06/more-people-watched-league-of-legends-than-the-nba-finals/ (accessed Oct. 4, 2024).

[5] L. Cabreros, “Patch 14.18 notes,” leagueoflegends.com, https://www.leagueoflegends.com/en-us/news/game-updates/patch-14-18-notes/ (accessed Oct. 4, 2024).

[6] "Global Power Rankings," LoL Esports, Web Archive, Sep. 19, 2024. [Online]. Available: https://web.archive.org/web/20240919192541/https://lolesports.com/en-US/gpr. [Accessed: Dec. 3, 2024].

[7] A. Singh, "T1 Gumayusi's ADC analogy: ‘It’s like eating noodles,’" ONE Esports, Oct. 30, 2023. [Online]. Available: https://www.oneesports.gg/league-of-legends/t1-gumayusi-adc-analogy-noodle/. [Accessed: Dec. 3, 2024].