# Abstract

The goal of this project was to create a model that can predict the success of a Valorant team based on the agent role and stats of a player. Utilizing web scraping techniques data was taken from <vlr.gg> to create a database of player statistics and a team’s ELO rating. Ultimately, I was not able to create a model that is able to accurately predict the success of a team, however, this project does serve as a great starting point for further modeling and analysis into the topic.

# Design

Valorant has an emerging e-sports scene with large and small organizations constantly scouting for the next superstar player. Potential clients could be e-sport organizations looking to replace current members of their team. A predictive model would create an initial starting point to make decisions such as how much money to invest into the development of a player or which players would have the most impact on immediate success.

# Data

Player stats is scrapped from vlr.gg, a website that tracks pro-Valorant stats in tournaments. Each row represents the stats of a player on the agents (the role on a team) they have played in the last 60 days. Each of these players have played at least 50 rounds and a rating of 1550. Also, from vlr.gg the Elo rating of professional teams were scrapped. Elo is a measure of a ranking based on normal distribution. A higher Elo rating indicates that the team has won a lot and is ranked higher than a team with a lower rating.

A total of 2019 player’s stats were scrapped. Most having multiple agents played. Players that were on teams that have not yet been ranked were dropped (22 teams without ratings)

# Algorithms

Dummy variables were added to so that machine learning could determine what role a player played. This would be represented as a 1 in ‘is\_duelist’,’is\_controller’,’is\_initiator’,’is\_sentinel’ columns. Deaths and First Deaths were given a negative connotation by making them negative and interaction terms for kills and deaths, and first kills and first deaths were added to denote the relationship between these 2 stats.

Multiple models along with multiple data transformations were tried. Polynomial Modeling had the best results after validation.

Data was split into a 80/20 train vs test and evaluated based on mean R2 scores and Mean Absolute Error (MAE). MAE provided the difference of predicted vs. actual ELO rating.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Name** | **R2 Score** | **MAE** | **Test R2** | **Test MAE** |
| unfiltered initial model | 0.033 | 154.35 | -0.0002 | 146.40 |
| ---------- | ---------- | ---------- | ---------- | ---------- |
| filtered by rank | 0.033 | 121.14 | 0.016 | 118.21 |
| ---------- | ---------- | ---------- | ---------- | ---------- |
| dueliest only | 0.043 | 121.07 | -0.006 | 122.78 |
| controller only | 0.027 | 121.17 | 0.043 | 119.21 |
| initiator only | 0.027 | 121.40 | 0.024 | 115.76 |
| sentinel only | 0.038 | 120.66 | -0.057 | 120.52 |
| ---------- | ---------- | ---------- | ---------- | ---------- |
| log y | 0.034 | 0.076 | 0.017 | 0.075 |
| log duelist | 0.043 | 0.076 | -0.005 | 0.078 |
| log controller | 0.028 | 0.076 | 0.045 | 0.075 |
| log initiator | 0.028 | 0.076 | 0.025 | 0.073 |
| log sentinel | 0.038 | 0.076 | -0.054 | 0.076 |
| ---------- | ---------- | ---------- | ---------- | ---------- |
| poly y | 0.048 | 119.57 | 0.092 | 111.89 |
| poly duelist | 0.047 | 119.52 | 0.140 | 111.72 |
| poly controller | 0.040 | 119.18 | 0.120 | 110.44 |
| poly initiator | 0.041 | 120.20 | 0.104 | 108.90 |
| poly sentinel | 0.074 | 117.59 | 0.109 | 108.57 |

# Tools

* Numpy and Pandas for data manipulation
* Scikit-learn and Statsmodels for modeling
* Matplotlib and Seaborn for plotting
* BeautifulSoup for web-scrapping

# Communication

Findings were presented along with slides. Additionally the full results can be found on my Github.