Competitive E-Sports Data Analysis - Setting up a Dashboard for Efficient Team, Player, and Weapon Data Analysis

Sai Poosarla

1 Introduction

Apex Legends is a competitive BR (”Battle Royale”) game that involves 20 teams of 3 that are meant to survive on a map by fighting against each other until only 1 team remains. It is of the same genre as the famous ”Fortnite”.

Over the last 5 years, Apex Legends has developed a highly competitive e-sports scene, with the best teams competing for upto a million dollars every year. With the ever increasing stakes, teams have hired coaches and analysts to assist teams in choosing the best compositions, the best high-level plans (macro plans) and staying ahead of frequent game changes to exploit advantages.

One variable among the many is weapons. Players can hold upto two weapons at a time, a ”Primary” and a ”Secondary” weapon, the primary being a longer range weapon usually, and the secondary being more effective at short ranges. Throughout the game’s lifespan, there have been frequent updates to weapons and their stats, and finding the best weapon combos after an update has always been of utmost importance in the e-sports scene.

During tournaments, weapon usage stats for each player is recorded, along with a slew of other useful stats. This project seeks to make it easier to see trends in weapon effectiveness relating to player and team performance over a tournament. If I were a coach, I’d like to answer questions like ”Which weapon was the best kill-rate wise?”, ”What about damage-wise?”, ”What about assist wise?”, ”Survivability wise?”.

Eventually, I’d like to perform a more focused hypothesis test approach to answer some question based on the insights I gathered from the previous data exploration.

2 Data Collection

The main source of the data was apexlegends.com/algs which is where all of the Apex Legends Global Series (ALGS) tournament data can be found. Since I was mainly concerned with the biggest international events over the last 2 years, my data was limited to the 5 ALGS Playoffs and Championship that occurred over the last two years.

I used Python to scrape three main data tables, a ”Player Stats” table, ”Team Stats” table and a ”Weapon Stats” table. The main libraries used were Selenium and BeautifulSoup. The ”Player Stats” and ”Team Stats” tables were quite straight forward to extract, since they were stored in the html and could easily be read into pandas using the ”read html” function.

The ”Weapon Stats” table was a bit more complicated to scrape, since the table with weapon stats was generated via a js script. I used Selenium to go through each player in the weapon stats dropdown, with ample waits and sleeps to make sure there was no chance of the site being overloaded.

For both the weapon and team/player data, the table scraper code was put into a loop that ran through each tournament. The final result was three pandas dataframes with all relevant player, team and weapon stats respectively.

There were some challenges and considerations to make along the way. One was to devise a way to create a connection between the Player and Team tables. I ended up using the image source for each player as the key since each player had the correct team image next to their name.

3 Data Cleaning and Transformation

I used Microsoft SQL Server as the RDBMS for this project. I set up a local SQL server instance on my computer using SQLExpress. I used SQLAlchemy in Python to load the three dataframes in Python to MSSQL. In MSSQL, I first made sure to create three views for the tables to create a layer where I could add calculated columns or perform other dynamic changes without having to affect the source loads.

There were quite a few columns that needed to be cleaned up, for example the ”Accuracy” column had ”NaN%” entries. ”REPLACE” and ”NULLIF” was used here to first replace the ”%” signs and then replace the ”NaN”s with nulls, before being cast as floating point integers.

I also converted a ”Playtime” column which was in the format ”ddh ddm dds” into seconds using the ”SUBSTRING” function to separate each unit and multiply by the correct amount, before casting as INT. ”COALESCE” was used in case ”hours” was missing to replace nulls with 0.

Finally, to make sure the PowerBI model that I create with these tables are straightforward and follow a clean Star-Schema, I added dimension tables for the team-player relationship and a competition list table, along with necessary keys in each table. The goal was to not have any instances of ”Many-Many” relationships in PowerBI, and have as much calculation and transformation as upstream as possible so that the dashboard performance is optimal.

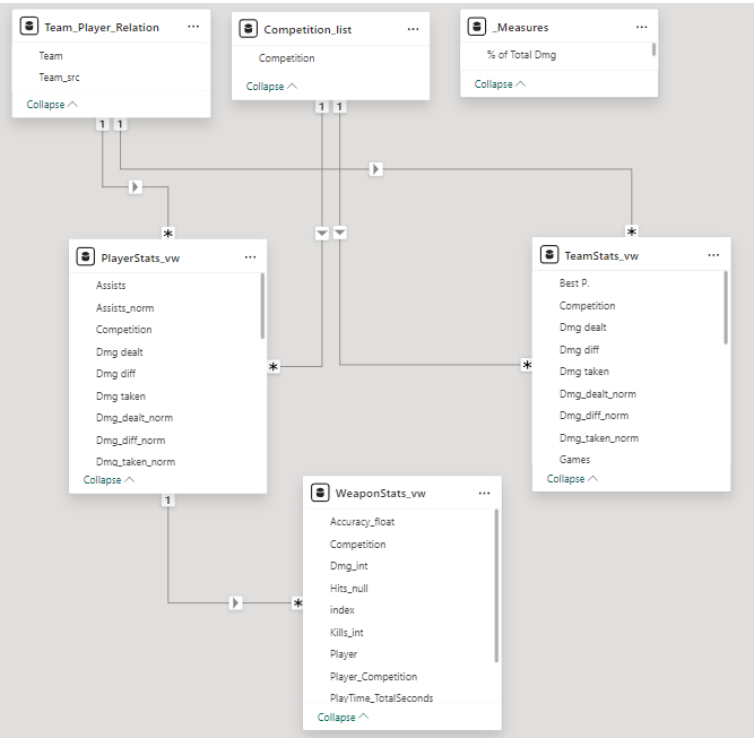

Figure 1: PowerBI model after cleaning and transformation in MSSQL

4 Dashboard Design and Insights

The PowerBI dashboard has 2 views, one focusing on the Team and Player stats, and the other focusing on the Weapon stats. The views are organized by having two scatter charts that show at a glance any outliers, accompanied by tables that show corresponding data that can be filtered by highlighting any of the scatter points.

This format works for both views and since they are consistent, it allows for a familiar analysis flow of investigating outliers using scatter plots and highlighting the chart to filter the corresponding tables to dig deeper.

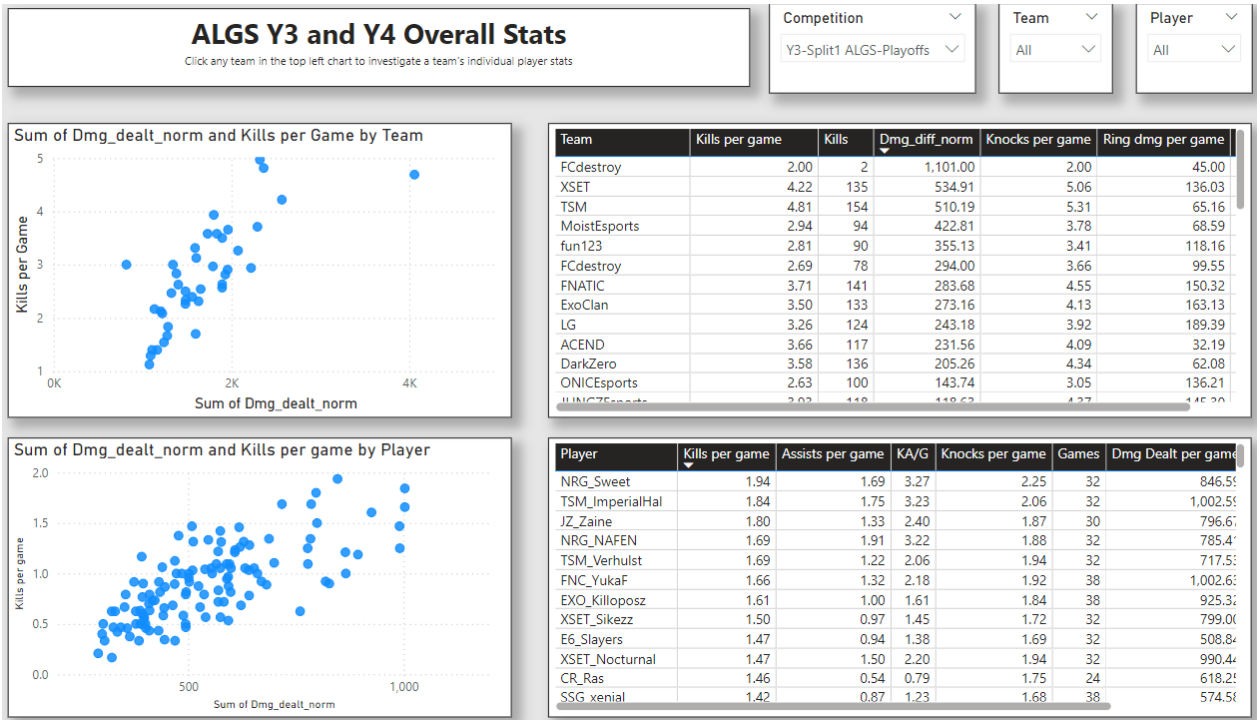

Figure 2: PowerBI view with two scatter charts and corresponding tables

Some preliminary insights were related to weapon usage and corresponding kill rates and damage rates. Specifically, in the Y4 Split 1 Playoffs, even though the most powerful primary weapon was agreed to be the Hemlock Burst Rifle, with a majority of the playtime share being on that weapon, there were players who favored other weapons heavily. Based on the two metrics mentioned above, their choices were not justified for this competition.

The goal is to quantify exactly how bad those picks were in relation to standard deviation, which is the next step for this project.