Data Science and Analytics for Esports

Arjun Agrawal[1], Dr. Claudio Silva[2], Peter Xenopoulos[2]

[1]The Peddie School, Hightstown, New Jersey, [2]The New York University Visualization, Imaging, and Data Analysis Center, New York City, New York

Introduction

The use of analytics in professional sports is widespread and rapidly increasing. Similarly, there is a need for analytics in the emerging area of esports, or professional video gaming. Counter-Strike: Global Offensive, also known as CS: GO, is one of the most popular esports with over forty million copies sold, yet it has lacking analytics.[1] CS: GO is a multiplayer first person shooter video game that involves two teams of five players. Each team is designated as either the Terrorists (T) or the Counter-Terrorists (CT) and plays a total of thirty one-minute-fifty-five-second rounds, with the teams switching sides after the first fifteen rounds. There are two bombsites on each map (Figure 1). The objective for the T side is to detonate a bomb at either of the bombsites or eliminate all of the players on the CT side. The objective for the CT side is to defuse a planted bomb at either of the bombsites, eliminate all of the players on the T side, or survive until the round ends without a bomb detonating. The first team to win sixteen rounds wins the match.

Figure 1. Dust II, a Bomb Defusal map in CS: GO. The lower green site represents the T side spawn, the upper green site represents the CT side spawn, and the two red sites represent the two bombsites.[2]

The data for each match, which includes both player actions, such as movement, and non-player events, such as round starts, is stored in a demofile. A demofile is a recording of the match generated by CS: GO that stores the data as a text of sequential sets of events with no information contextualization. In order to perform analytics on the stored data, it must be modeled into an organized data structure. The data parser developed by Dr. Claudio Silva and Peter Xenopoulos parses the data into Pandas DataFrames. To this end, we introduce an analytics package consisting of (1) generalized functions to allow for the efficient filtering and aggregation of data; and (2) specialized functions to allow for the efficient calculation of CS: GO match statistics.

Results

The analytics functions for the package were successfully developed, with one core function and four subfunctions that allow for filtering, grouping, and aggregation of data with full autonomy, and thirteen supplementary functions derived from the core function that allow for the calculation of CS: GO match statistics. The following is a list of all of the functions and their functionality or output:

Core Function:
- **calc_stats()**: calls the subfunctions, and filters, groups, and aggregates a given DataFrame.

Subfunctions:
- **check_filters()**: checks given data filters for a given DataFrame for validity.
- **extract_num_filters()**: returns a tuple of two lists that contains the logical operators and numeric values from given data filters of integer or float type.
- **num_filter_df**: filters a given DataFrame by given filters of integer or float type.
- **filter_df**: filters a given DataFrame by given filters of string or boolean type.

Supplementary Functions:
- **accuracy()**: returns a DataFrame with weapon fires, strafe percentage, accuracy percentage, and headshot accuracy percentage, by player or team.
- **kast()**: returns a DataFrame with KAST percentage and statistics, by player.
- **kill_stats()**: returns a DataFrame with kills, deaths, assists, flash assists, plus-minus, first kills, first kills plus-minus, trades, headshots, headshot percentage, accuracy percentage, headshot accuracy percentage, kill-death ratio, kills per round, and KAST percentage, by player or team.
- **adr()**: returns a DataFrame with normalized and raw ADR, by player or team.
- **util_dmg()**: returns a DataFrame with given utility damage, utility damage, grenades thrown, given utility damage per grenade, and utility damage per grenade, by player or team.
- **flash_stats()**: returns a DataFrame with enemy flashes, flash assists, enemy blind time, team flashes, flashes thrown, enemy flashes per throw, and enemy blind time per enemy, by player or team.
- **bomb_stats()**: returns a DataFrame with bomb plants, defuses, and defuse percentage, by side and bombsite.
- **econ_stats()**: returns a DataFrame with buy type, average equipment value, average cash, and average spend, by side.
- **kill_breakdown()**: returns a DataFrame with kills by weapon type and player or team.
- **util_dmg_breakdown()**: returns a DataFrame with given utility damage, utility damage, grenades thrown, given utility damage per grenade, and utility damage per grenade, by grenade type and player or team.
- **win_breakdown()**: returns a DataFrame with win type by team.
- **player_box_score()**: returns a player box score DataFrame with summary statistics, by player (Figure 2).
- **team_box_score()**: returns a team box score DataFrame with summary statistics, by team (Figure 3).

	Player	K	D	A	FA	HS%	ACC%	HS ACC%	KDR	KAST%	ADR	UD	UD Per Nade	EF	EF Per Throw
0	device	23	17	2	1	0.304348	0.267062	0.023739	1.352941	0.678571	73.535714	171	8.142857	13	0.928571
1	Xyp9x	22	16	3	1	0.318182	0.200772	0.021236	1.375000	0.678571	90.000000	153	4.500000	17	0.894737
2	nitr0	19	17	1	0	0.842105	0.127907	0.034884	1.117647	0.678571	63.571429	83	4.611111	8	0.666667
3	EliGE	18	21	3	0	0.500000	0.185031	0.022869	0.857143	0.500000	85.500000	119	4.576923	3	0.500000
4	NAF	17	19	0	0	0.294118	0.160896	0.024440	0.894737	0.642857	77.035714	76	3.619048	7	1.166667
5	Stewie2K	17	20	6	0	0.588235	0.242958	0.035211	0.850000	0.642857	83.071429	118	6.555556	7	1.000000
6	dupreeh	17	16	1	0	0.529412	0.168539	0.022472	1.062500	0.678571	61.785714	70	3.684211	14	1.166667
7	gla1ve	17	16	8	0	0.529412	0.162069	0.020690	1.062500	0.642857	86.285714	175	7.608696	19	1.000000
8	Magisk	16	19	5	2	0.375000	0.174393	0.017660	0.842105	0.607143	60.000000	161	5.031250	16	1.066667
9	Twistzz	13	19	2	0	0.461538	0.143243	0.021622	0.684211	0.642857	48.285714	2	0.068966	10	0.909091

Figure 2. The DataFrame output of the function player_box_score() that contains summary statistics for each player for a sample match. This function allows players to evaluate their individual performance in a given match. For example, Twistzz can evaluate his ADR of 48.3, which was significantly lower than the other players' ADR.

	Astralis	Team Liquid
Score	16	12
CT Wins	9	8
T Wins	7	4
K	95	84
D	84	96
A	19	12
FA	4	0
+/-	11	-12
FK	11	17
HS	38	46
HS%	0.4	0.547619
Strafe%	0.03129	0.021008
ACC%	0.189456	0.1662
HS ACC%	0.021003	0.027544
ADR	371.607143	357.464286
UD	730	398
Nades Thrown	129	112
UD Per Nade	5.658915	3.553571
EF	79	35
Flashes Thrown	79	42
EF Per Throw	1.0	0.833333
EBT Per Enemy	2.511744	1.861848
Full Buy	20	21
Full Eco	1	2
Pistol	2	2
Semi Buy	5	0
Semi Eco	0	3
Avg EQ Value	22292	18725
Avg Cash	28921	17148
Avg Spend	12919	12288
CT Bomb Defusal Wins	4	1
CT T Elim Wins	5	6
CT Time Expired Wins	0	1
T Bomb Detonation Wins	2	3
T CT Elim Wins	5	1

Figure 3. The DataFrame output of the function team_box_score() that contains summary statistics for each team for a sample match. This function allows teams to evaluate their overall performance in a given match. For example, Astralis can evaluate their HS% of 40%, which was significantly lower than Team Liquid's HS% of 54.8%.

Future Work

The project is ongoing, with the next steps being to develop visualization functions to visualize spatiotemporal data, such as kill locations, and a neural-network-based algorithm to predict match outcomes based on match states and spatiotemporal data.

Acknowledgements

I would like to thank Dr. Claudio Silva for giving me the opportunity to research at his lab and his PhD student, Peter Xenopoulos, for mentoring and guiding me throughout the project. I would also like to thank Mrs. Joy Wolfe, Mr. David Luo, Ms. Madeleine Cozine, Dr. Sarah Venanzi, and Dr. Shani Peretz for helping me throughout the research process.

References

1. Rizani, M.N., & Iida, H. (2018). Analysis of Counter-Strike: Global Offensive. *2018 International Conference on Electrical Engineering and Computer Science (ICECOS)*, 373-378.
2. Valve Corporation. (n.d.). *Dust II Radar* [Map]. https://en.wikipedia.org/wiki/File:Dust_II_radar.png.