A Damage Output Analysis of Dungeons and Dragons

Robert L. Kline

Western Governors University

Table of Contents

Project Overview

A1. Research Question

The project was undertaken to identify if increased complexity, by way of adding additional modifiers to combat capabilities, would result in a correlating increase in damage output. The purpose of this analysis was to identify the optimal damage routes and if expended time and effort in keeping track of the additional variables provided a worthwhile benefit.

A2. Project Scope

The project included analysis of core weapons, abilities, spells, and modifiers available to the Barbarian, Ranger, and Warlock classes. The project utilized recommended player ability scores and monster challenge rating for level 10 characters to determine a calculation regarding damage consistency. The project did not include analysis of any non-damage related abilities, cross-class calculations, additional classes, or role-playing-centric decisions.

A3. Solution overview

The analytics were performed in Python utilizing the "Pandas" and "NumPy" modules for calculations and data importing, utilizing the "MatPlotLib", "StatsModels", and "Seaborn" modules for visual representation, and Excel for capturing base ability info and CSV generation. Data regarding the abilities, interactions, and required calculations was taken directly from the D&D 5e Player's Handbook and summarized into Excel. Calculations for dice rolls were performed using the probability method for a dice roll (i.e., $n(n+1) / 2n$) and the modifications to this output were handled via the creation of various Python objects and classes.

Project Plan

B1. Project Plan

The deliverables and methodology did not deviate from the initial plan.

The goal of the project was to provide two major deliverables:

1. A ranking of weapons, abilities, spells, and modifiers as they pertain to the average damage output for each of the Barbarian, Ranger, and Warlock classes; to be provided as a descending list of damage ranking with a corresponding visual representation.
2. Identify the linear relationship between damage and complexity; to be provided as a scatterplot comparing the damage output with varying levels of complexity and a regression line to indicate the associated relationship, if any.

B2. Planning Methodology

As there were no iterative deliverables, the project followed a Waterfall methodology.

- Requirements: This phase consisted of gathering the high-level requirements of the data required to make the appropriate calculations.
- Design: This phase consisted of identifying the best tools and methods to utilize with the Python architecture to properly analyze the data.
- Implementation: This phase was the actual data analysis utilizing Python and the data collected during requirements gathering.
- Verification: This phase was the validation that the analysis methodology utilized were performing appropriately and to fine-tune any of the code that was developed.
- Maintenance: This phase consisted of final delivery of the data analysis with options for continual optimization or addition by the way of re-usable classes and updateable parameters for Attribute and Proficiency (which increase based on character level).

B4. Timeline and Milestones

The following Table shows the projected and actual dates and durations for each Milestone:

Milestone	Proposed			Actual		
	Start Date	End Date	Duration (Days)	Start Date	End Date	Duration (Days)
Gather all raw data	7/2/2023	7/5/2023	3	7/2/2023	7/5/2023	3
Investigate Python code	7/6/2023	7/7/2023	1	7/6/2023	7/7/2023	1
Build Python code	7/8/2023	7/12/2023	5	7/8/2023	7/13/2023	6
Fine-tune Python code	7/13/2023	7/16/2023	4	7/14/2023	7/14/2023	< 1
Clean up Jupyter Notebook	7/17/2023	7/18/2023	2	7/14/2023	7/15/2023	1
Project Delivery	7/19/2023	7/20/2023	1	7/15/2023	7/15/2023	< 1

Methodology

C1. Data Selection and Collection Differences

There was no major difference regarding the data selection and collection process as initially outlined in the project plan. The data selected was for the three proposed classes, contained all damage-related weapons, spells, abilities, and modifiers, and was drawn directly from Wizards of the Coast source material. The data was then collected into an Excel spreadsheet for consolidation and conversion to a CSV file, then imported into Jupyter notebook via Python.

C2. Handling Obstacles

There were no major obstacles during data gathering outside of the amount of time it took to read through and gather the appropriate materials. To ensure that no data was missed, the source material was reviewed several times and compared against captured data to ensure completeness and accuracy.

C3. Unplanned Data Governance

All data utilized within this project was covered and approved for use under the Open-Gaming License (OGL) or Creative Commons agreement that Wizards of the Coast has created with their community. As such, there was no concern regarding data governance.

C3a. Advantages and Limitations of the Data Set

Because the data was all source material data, there was an advantage to knowing that all the described calculations and functionalities of weapons, spells, abilities, and modifiers were accurate and available.

However, because the data was largely provided in a printed book and spread out amongst the various classes, extracting the data to be entered into Excel was a time-consuming and tedious process. Additionally, the interactions between weapons, spells, abilities, and modifiers were only provided on a high-level, not in a detailed case-by-case basis. As such, the interaction had to be manually calculated.

D1. Data Extraction and Data Preparation Process

Weapons function in a consistent and predictable way across all classes. As such, the data for weapons was gathered for storage in a flat CSV file in the following process:

1. Identify all weapons available to the three different Classes and include them in an Excel spreadsheet.
2. Identify and note if there are 1handed or 2handed variants of how to utilize the weapons.
3. Identify the Weapon Type to be able to limit which classes can use each weapon.
4. Identify the Difficulty to be able to limit which classes can use each weapon.
5. Identify the Weight to be able to limit which classes can use each weapon.
6. Identify the ability to Dual Wield a weapon to make them available for the Dual Wield calculations.
7. Identify the Dice Type to be able to calculate the average dice roll value.
8. Identify the Number of Dice to be able to calculate the total average roll value.
9. Store all values in Excel.
10. Validate data.
11. Convert to CSV file format.
12. Import into Jupyter Notebooks.

Spells, abilities, and modifiers all have much more varied functionality and are not easily imported as a flat file. Data for these aspects was gathered in the following way:

1. Identify all spells, abilities, and modifiers available for each class.
2. Identify if there is a mutual exclusivity.
3. Identify what modification the item will have on attack:
    a. Attack Damage indicates that it provides a simple damage addition.
    b. Attack Chance indicates that it improves the chance to land an attack.
    c. Additional Attack indicates that an additional attack can be made.
    d. Replace Attack indicates that it supersedes a normal attack and fully replaces it.
4. Identify the dice type used, if any, to be able to calculate the average dice roll value.
5. Identify the number of dice, if any, to be able to calculate the total average roll value.

6. Identify a general description of mechanical function.
7. Store all values in Excel.
8. Create a Python Object or Class to store the functionality for use in Jupyter Notebooks.

As Python is the main method of analyzing and calculation, preparing the data for direct import was relevant. As well, because the inconsistent nature how modifications were calculated, consolidating data made it extremely easy for Python Class and Object creation.

E1. Methods used to Analyze Data

Python was the predominant tool utilized for analysis of the data. “Pandas” was utilized for data extraction and organization, “NumPy” was used for calculations, and “MatPlotLib”, “Seaborn”, and “StatsModels” were utilized for data visualization.

Data was evaluated based on level 10 characters, which is the max level for characters in the Curse of Strahd beginner’s campaign (Wizards of the Coast 2016). Data values were prepared based on the Adventurer’s League recommended standard array for attribute distribution (15, 14, 13, 12, 10, 8) and fighting against the Adventurer’s League recommended monster difficulty rating (Wizards of the Coast, 2023). These recommendations were used as a baseline for all characters to provide consistent parameters for damage output. The resulting values, the values for chance to hit (missing results in no or decreased damage) and chance to critically hit (which results in double-damage) were calculated as Python Objects.

Base calculations for weapon damage were developed based on the chance for the average value of a dice roll and multiplied by the number of dice used. Python classes and objects were developed for each spell, ability, and modifier to be able to calculate their overall effect on weapons, or on each other. The correlation of damage output and complexity was calculated via a linear regression model with a correlation coefficient output, and the regression line on the visual output was calculated using the least-squares method.

E2. Advantages and Limitations of Tools and Techniques

Python is an extremely versatile tool, and the Jupyter Notebooks functionality allows for ease of entry, validation, and revision of data. The ability to import flat data, add columns based on calculations, and develop Objects and Classes meant that the analysis of the data was consistent and easily updated if errors were found at any point.

However, there is the possibility of data being incorrectly because Python will only identify if there is an error in syntax. Python will not identify if there is an error in ordering of abilities that would result in an incorrect calculation. Extra care was required to ensure that calculations fell in line with the mechanical functionality of the game itself.

E3. Step-by-Step Explanation

The analysis was performed in the following steps:

1. Import necessary Python modules.
2. Import weapon data as a data frame.
3. Create an object for the Proficiency, Attribute, and monster AC rating.
4. Create an object for Hit Chance calculations, based on a d20 roll.
5. Create an object for Advantage calculations, based on two d20 rolls and selecting the greater.
6. Create an object for the Critical damage multiplier.
7. Create a new column in the weapon data frame to calculate the average weapon damage, based on the previously created objects (not including Hit/Advantage calculations)
8. Create a correlation data frame for each character class.

9. Create a Python class for each spell, ability, and modifier, for each character class.
10. Create a “Complexity Level 0” data frame for each character class, which identifies which weapons can be utilized and how much average damage they will result in.
11. Store the “Complexity Level 0” data as a new data frame consisting only of the Weapon and the Damage Output, and rank the results based on descending order.
12. Import the results into the class correlation data frame.
13. Create a “Complexity Level 1” data frame for each character class, which identifies the effect that each applicable modifier has on attack damage.
14. Create a new column in the “Complexity Level 1” data frame based on the damage calculations of the applied modifiers.
15. Store the “Complexity Level 1” data as a new data frame consisting only of the Modifier columns and the Damage Output, and rank the results based on descending order.
16. Import the results into the class correlation data frame.
17. Create a “Complexity Level 2” data frame for each character class, using the Modifier with the highest damage output as the base and apply each additional applicable modifier to it, until all combinations of the modifier identified “Complexity level 1” and applicable modifiers have been included.
18. Store the “Complexity Level 2” data as a new data frame consisting only of the Modifier columns and the Damage Output, and rank the results based on descending order.
19. Import the results into the class correlation data frame.
20. Repeat steps 17 through 19 for the previous Complexity Level, until all combinations of modifiers have been exhausted for each individual class.
21. Fit a regression line for the correlation data based on the least-squares method for each individual class.
22. Fit a scatterplot and the regression line together in a graph for each individual class and report the correlation coefficient for each individual class.

23. Create a new combined correlation data frame which consists of data from all three classes.
24. Fit a regression line for the combined correlation data using the least-squares method.
25. Fit a scatterplot and the regression line together in a graph and identify the correlation coefficient for the combined data.

The ability to generate a ranking of damage output at each level of complexity verified that the first deliverable could be met. The ability to generate the correlation coefficient and subsequent visual representation verified that the second deliverable could be met.

Results

F1. Statistical Significance

For the Barbarian, the following data was discovered:

1. At Complexity Level 0, two-handed heavy weapons such as the “Greatsword” and “Maul” were identified as the best weapon for damage output, averaging 5.82 damage.
2. At Complexity Level 1, the Great Weapon Master ability was identified as the greatest damage multiplier, providing an average damage output of 13.39. This solidified that the use of two-handed heavy weapons, as the ability is only usable with weapons of that variety.
3. At Complexity Level 2, the Great Weapon Master ability in combination with Extra Attack was identified as the greatest damage multiplier, providing an average damage output of 26.79 damage.
4. At Complexity Level 3, the Great Weapon Master ability in combination with Extra Attack and Brutal Critical was identified as the greatest damage multiplier, providing an average damage output of 47.07 damage.

5. At Complexity Level 4, the Great Weapon Master ability in combination with Extra Attack, Brutal Critical, and Rage was identified as the greatest damage multiplier, providing an average damage output of 57.98 damage.
6. At Complexity Level 5, the Great Weapon Master ability in combination with Extra Attack, Brutal Critical, Rage, and Frenzy was identified as the greatest damage multiplier, providing an average damage output of 77.15 damage.
7. At Complexity Level 6, the Great Weapon Master ability in combination with Extra Attack, Brutal Critical, Rage, Frenzy, and Savage Attacker was identified as the greatest damage multiplier, providing an average damage output of 90.78 damage.
8. At Complexity Level 7, the Great Weapon Master ability in combination with Extra Attack, Brutal Critical, Rage, Frenzy, Savage Attacker, and any ability to provides Advantage was identified as the optimal damage combination, providing an average damage output of 95.38 damage. This was the highest level of complexity possible with the Barbarian character class.
9. The correlation coefficient was identified as 0.97, indicating a strong positive correlation between Damage Output and Complexity (Strong being a value of 0.6 or greater).

For the Ranger, the following data was discovered:

1. At Complexity Level 0, two-handed heavy weapons such as the “Greatsword” and “Maul” were identified as the best weapon for damage output, averaging 5.82 damage.
2. At Complexity Level 1, Lightning Arrow was identified as the greatest damage multiplier, providing an average damage output of 24.75. Because Lightning Arrow is only usable with ranged weapons, only calculations of ranged weapons were considered, as they opened the route to the highest damage output.
3. At Complexity Level 2, the Lightning Arrow ability in combination with Wolf was identified as the greatest damage multiplier, providing an average damage output of 31.05 damage.

4. At Complexity Level 3, the Lightning Arrow ability in combination with Wolf and Extra Attack was identified as the greatest damage multiplier, providing an average damage output of 35.88 damage.
5. At Complexity Level 4, the Lightning Arrow ability in combination with Wolf, Extra Attack, and Fighting Style: Archery was identified as the greatest damage multiplier, providing an average damage output of 40.81 damage.
6. At Complexity Level 5, the Lightning Arrow ability in combination with Wolf, Extra Attack, Fighting Style: Archery, and Horde Breaker was identified as the greatest damage multiplier, providing an average damage output of 46.45 damage. This was the highest level of complexity possible with the Ranger character class.
7. The correlation coefficient was identified as 0.97, indicating a strong positive correlation between Damage Output and Complexity (Strong being a value of 0.6 or greater).

For the Warlock, the following data was discovered:

1. At Complexity Level 0, “Light Crossbow”, “Quarterstaff 2h”, and “Greatclub” were identified as the best weapons for damage output, averaging 4.17 damage.
2. At Complexity Level 1, the Witch Bolt ability and Eldritch Blast Cantrip were identified as having the greatest damage outputs, with 51.48 and 7.26 damage respectively. Warlocks may only use a non-cantrip spell twice during combat and would utilize their weapon or cantrip spell for all other consistent damage, which is why both values were noted.
3. At Complexity Level 2, the Witch Bolt ability with True Strike and Eldritch Blast Cantrip with Agonizing Blast were identified as having the greatest damage outputs, with 72.072 and 11.26 damage respectively. or cantrip spell for all other consistent damage, which is why both values were noted.

4. At Complexity Level 3, the Witch Bolt ability with True Strike and Eldritch Blast Cantrip with Agonizing Blast and True Strike were identified as having the greatest damage outputs, with 72.072 and 15.76 damage respectively.
5. The correlation coefficient was identified as 0.61, indicating a strong positive correlation between Damage Output and Complexity (Strong being a value of 0.6 or greater), though just barely.

For all classes combined, he correlation coefficient was identified as 0.91, indicating a strong positive correlation between Damage Output and Complexity (Strong being a value of 0.6 or greater).

F2. Practical Significance

Practically, two things can be noted:

1. There is an optimal route for Damage Output at each level of complexity, which provides players with a strong basis for how to build a combat-optimized character.
2. Damage output scales with complexity, meaning that players should strive for the highest level of complexity that they are comfortable with handling, to achieve maximum combat effectiveness.

F3. Overall Success and Effectiveness

The initial hypothesis stated the following:

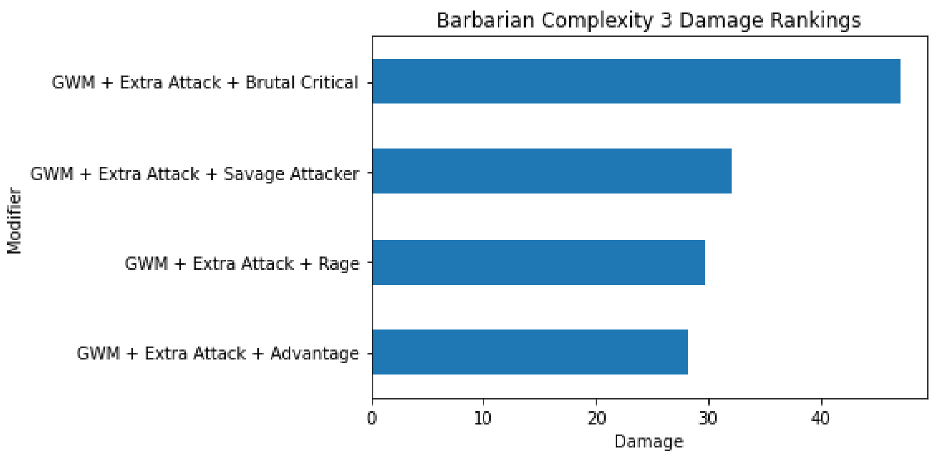
> "Weapons, abilities, spells, and modifiers will have an identifiable ranking regarding damage output or relative effect on damage output. The correlation between damage output and complexity will be measurable as a strong linear relationship."

Because we were able to provide results for each of the deliverables, the project would be considered successful. Furthermore, because the data was extremely clear as it pertained to optimization and the correlation, the execution proved to be effective.

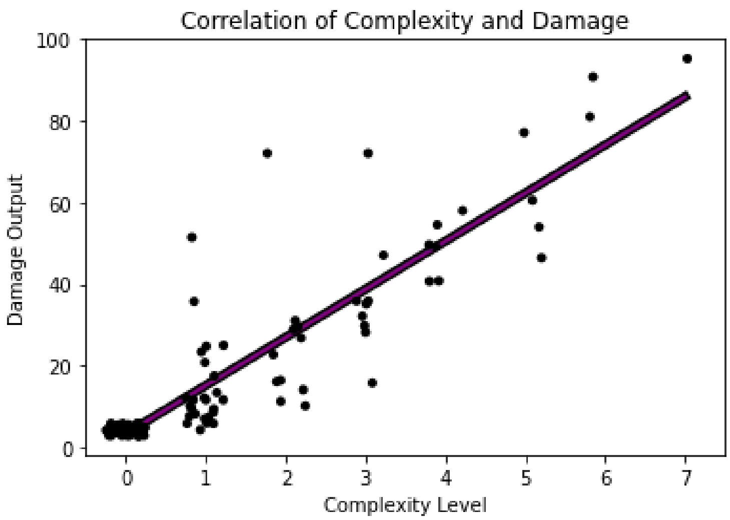
G1. Conclusions

Based on the data, there are clear and definable optimal combinations of weapons, spells, abilities, and modifiers for each class at each level of complexity. Additionally, there is a strong and positive correlation between increasing complexity and overall damage output.

Below is an example of the Barbarian Complexity Level 3 Damage Rankings:



Below is the Correlation of Complexity and Damage graph, combined for all classes:



Additional graphs for individual class correlations are available in the Appendices, along with the graphs for optimal damage output at each complexity level.

G2. Reasoning for Visual Representation

For the ranking charts, a bar graph was utilized as it provided an obvious representation regarding which Weapon, or combination of Spells, Abilities, and Modifiers provided the best damage output. They were separated at each level of complexity to minimize the total space that would be taken up by a given graph.

For the complexity and damage correlation, the use of a scatterplot and regression line were considered an excellent option for visual representation, as they provide a clear indicator of the range of damage available at each complexity level and that there is a strong correlation. The combined graph was provided to encompass all of the work in the project and the individual class graphs were provided to showcase differences between the classes.

G3. Recommended Courses of Action

There are two courses of action recommended for players, in consideration of character creation:

1. If a player is looking to provide maximum damage output, it is recommended to utilize the Barbarian class, focusing on two-handed heavy weapons.
2. If a player is looking to provide optimal damage output with minimized complexity, it is recommended to utilize the Warlock Class, focusing on Witch Bolt and Eldritch Blast.

However, these recommendations should only be viewed as a starting point. The most important aspect about D&D is to have fun with your friends!

H1. Panopto Recording

The Panopto recording is available at the following URL:

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=9ef7b62c-ab34-432b-b540-b04100517ce5

Appendices

I1. Code Supported

The code is present in the ‘D195 Capstone Robert Kline - A Damage Analysis of Dungeons and Dragons.html’ file attached to the submission.

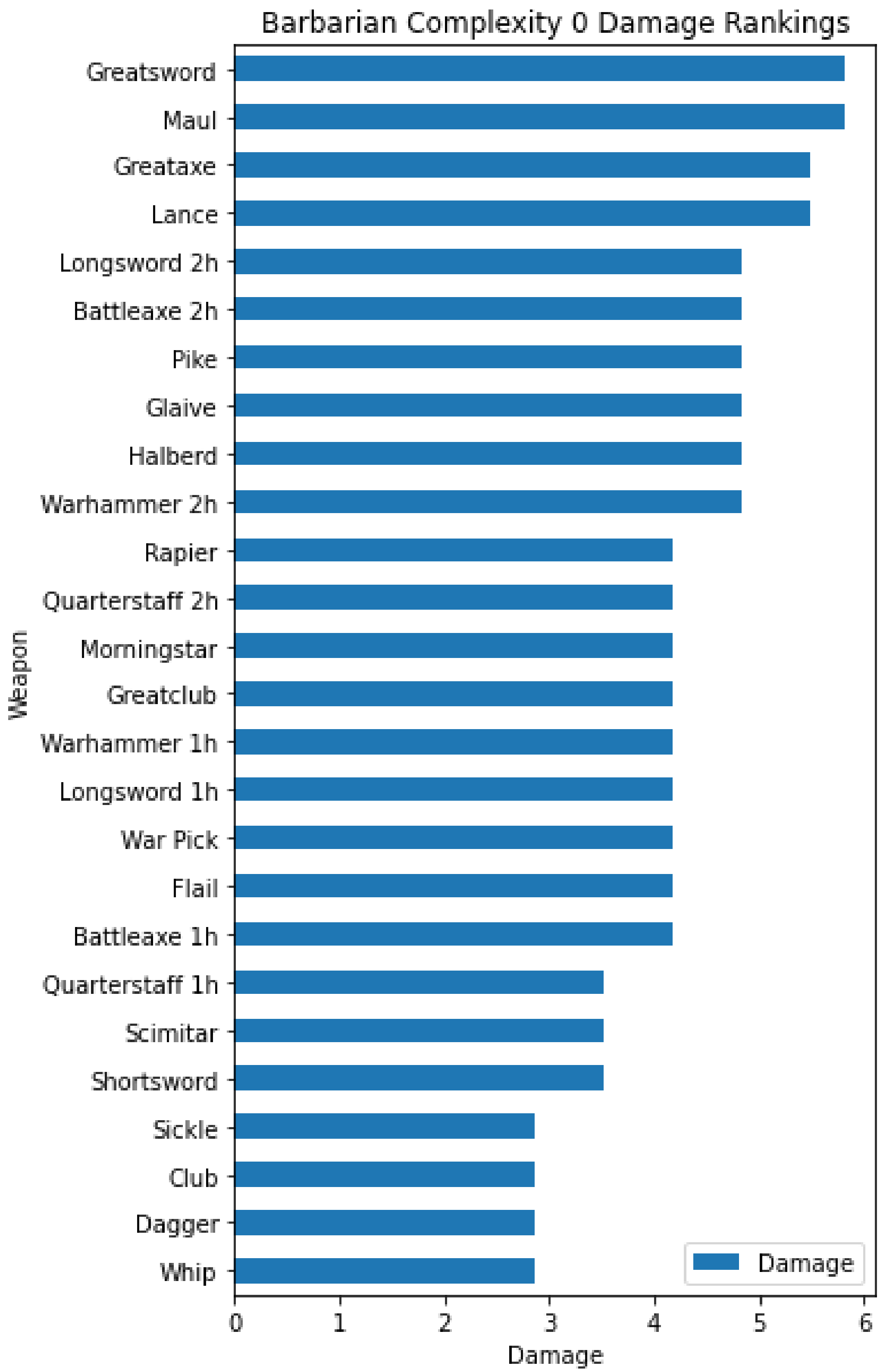
I2. Data used for Analytics Solution

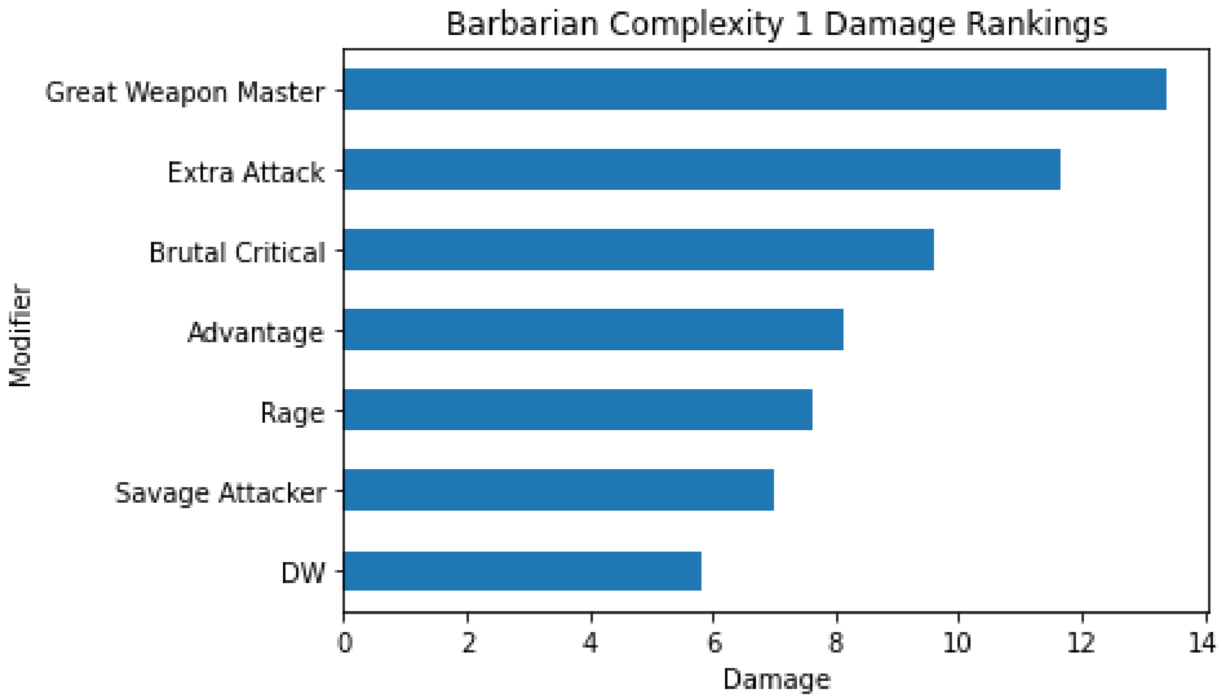
The data is present in the ‘Data Master File.xlsx’ file attached to the submission.

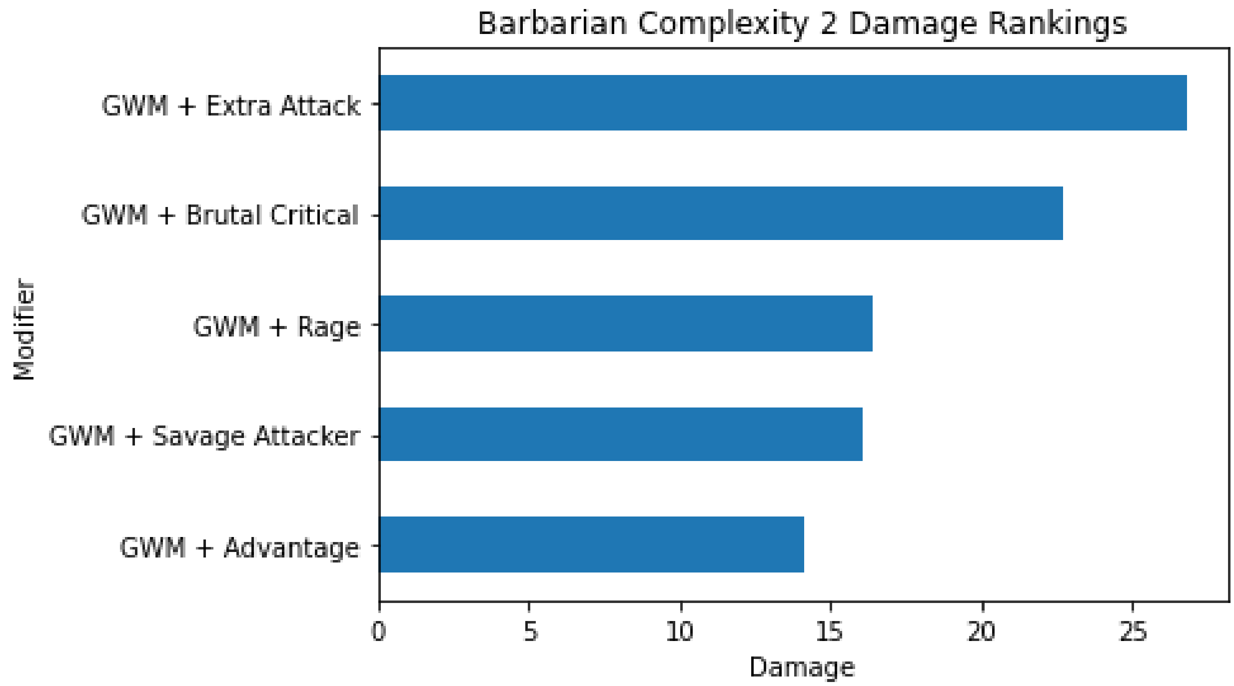
I3. Web Sources

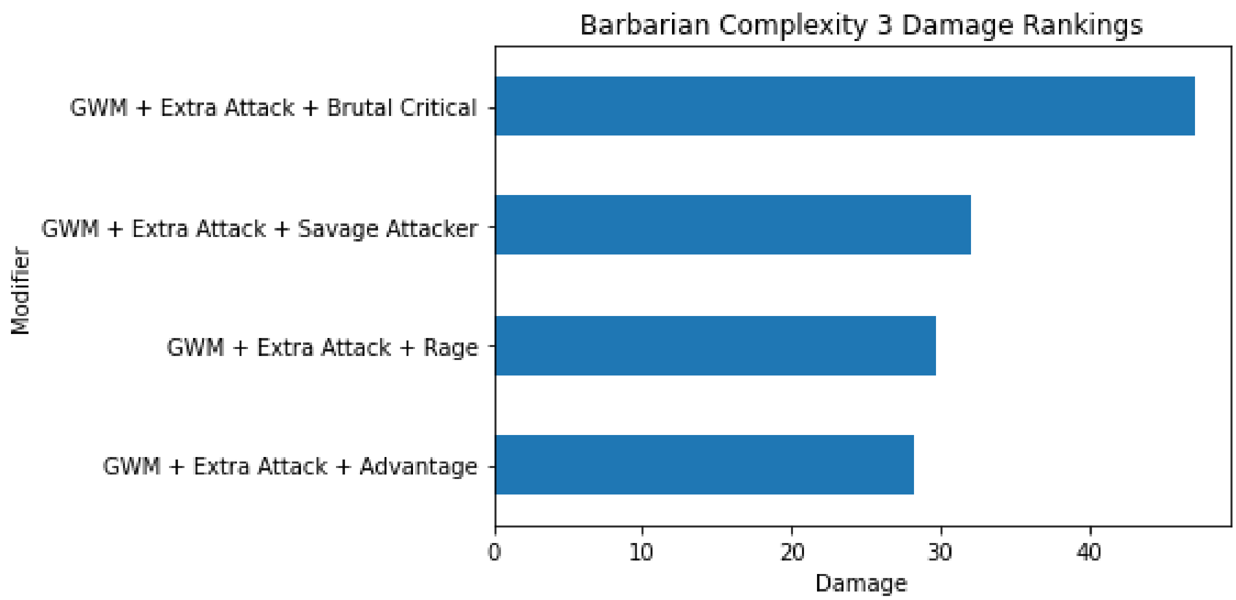
The ‘anydice.com’ utility was utilized to generate the average value multiplier for the “Savage Attacker” modifier, which had an average damage output multiplicative value of “1.2”. (Flick 2023).

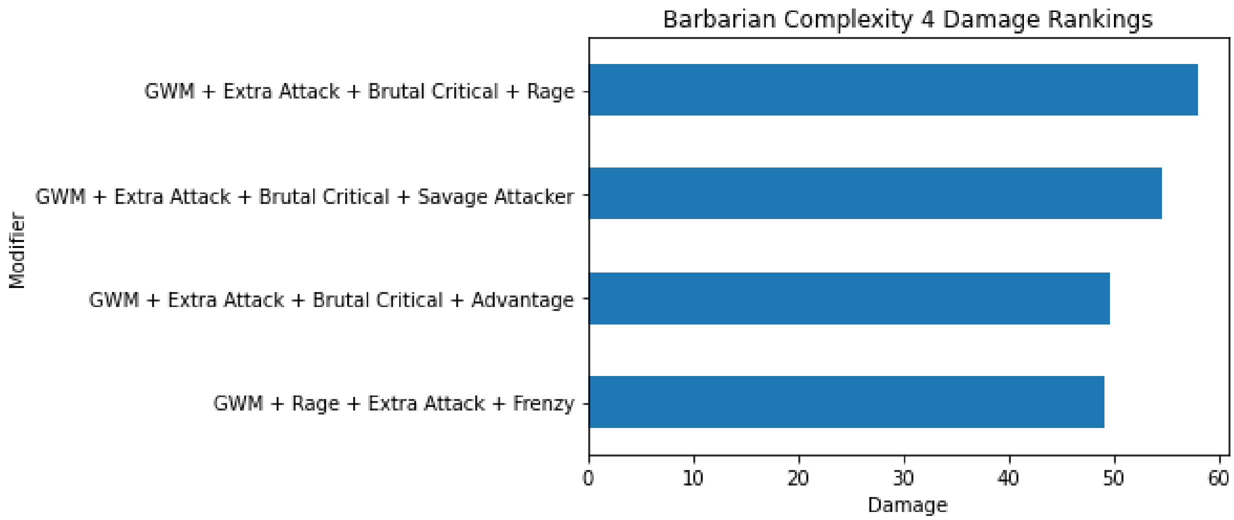
I4. Relevant Deliverables

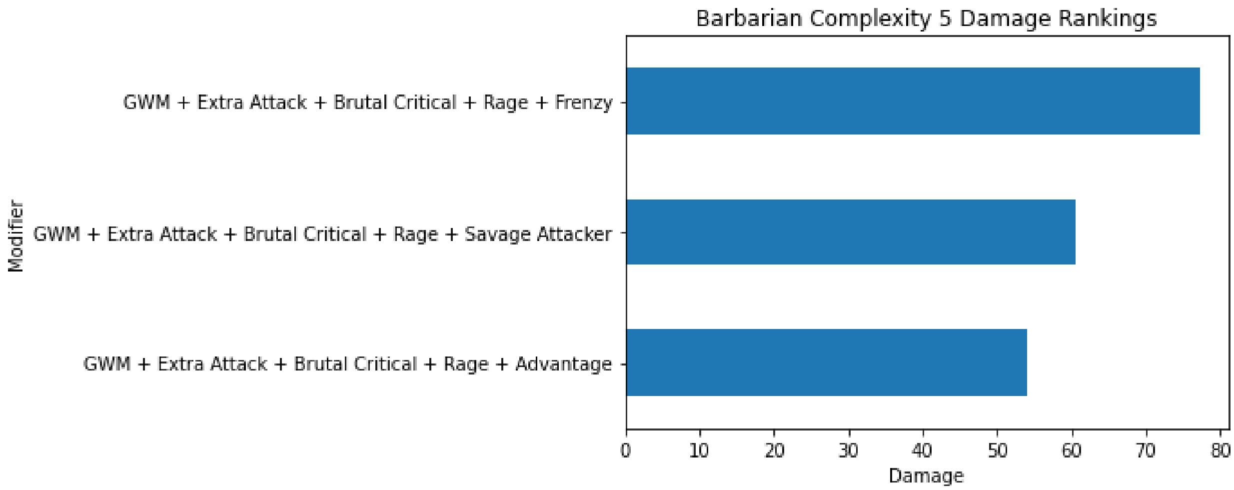


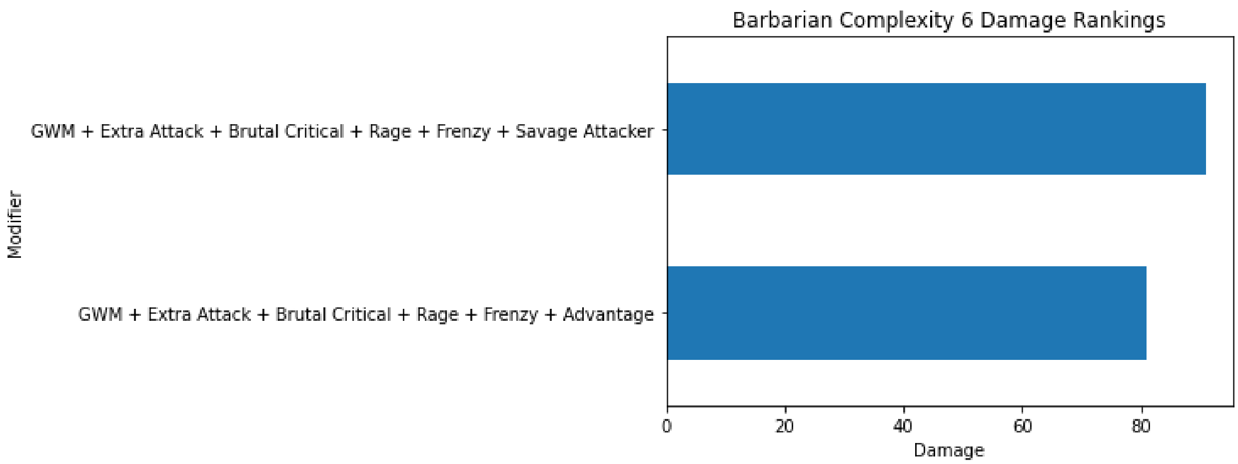

Barbarian Complexity 1 Damage Rankings
Great Weapon Master
Extra Attack
Brutal Critical
Advantage
Rage
Savage Attacker
DW
Modifier
0
2
4
6
8
10
12
14
Damage

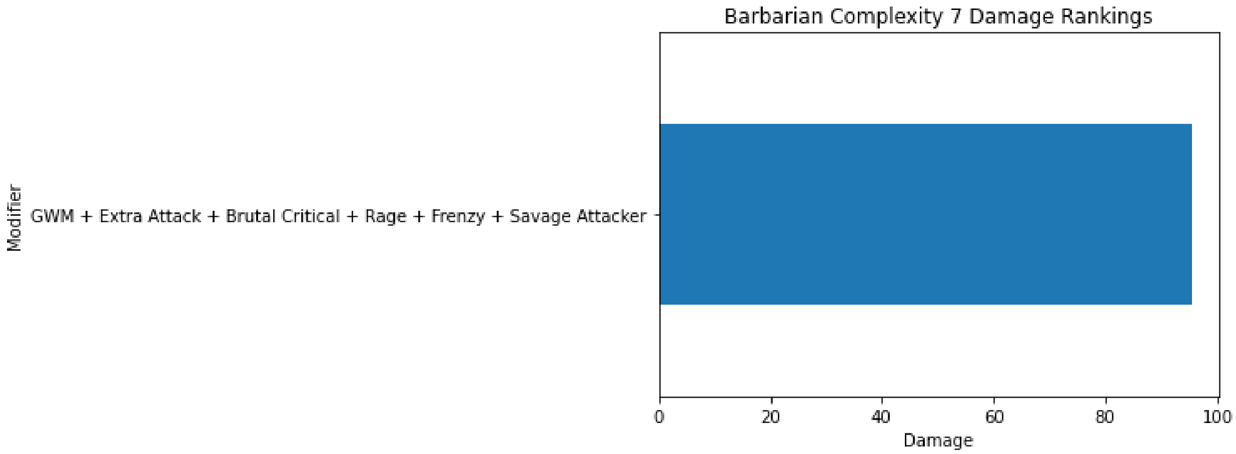

Barbarian Complexity 2 Damage Rankings
GWM + Extra Attack
GWM + Brutal Critical
GWM + Rage
GWM + Savage Attacker
GWM + Advantage
Modifier
0
5
10
15
20
25
Damage

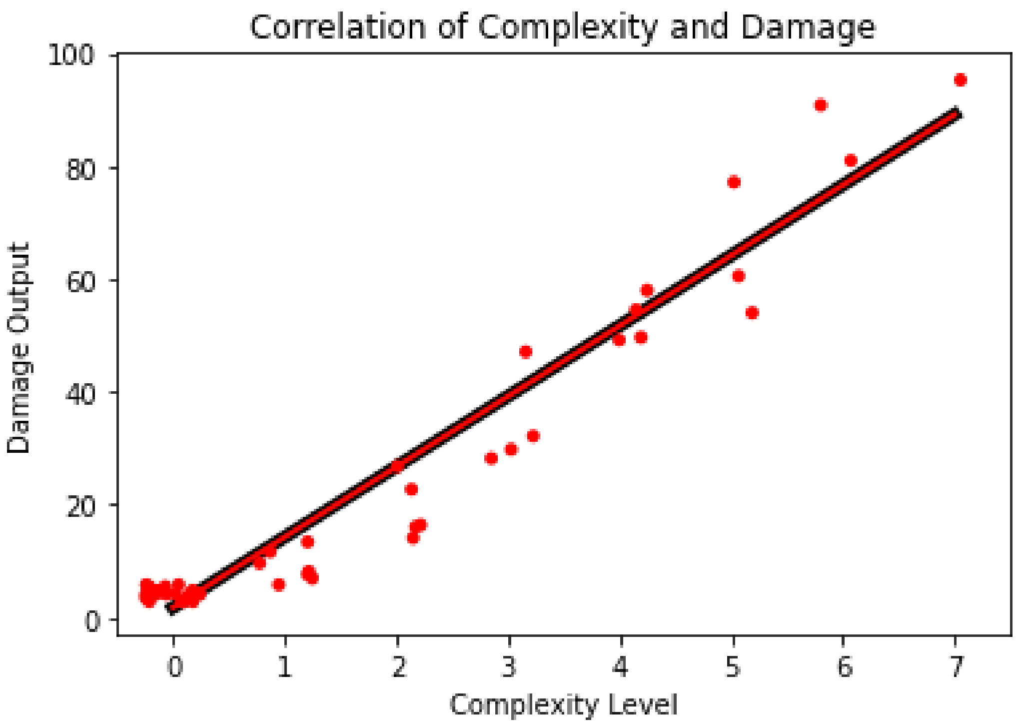

Barbarian Complexity 3 Damage Rankings
GWM + Extra Attack + Brutal Critical
GWM + Extra Attack + Savage Attacker
GWM + Extra Attack + Rage
GWM + Extra Attack + Advantage
Modifier
0
10
20
30
40
Damage

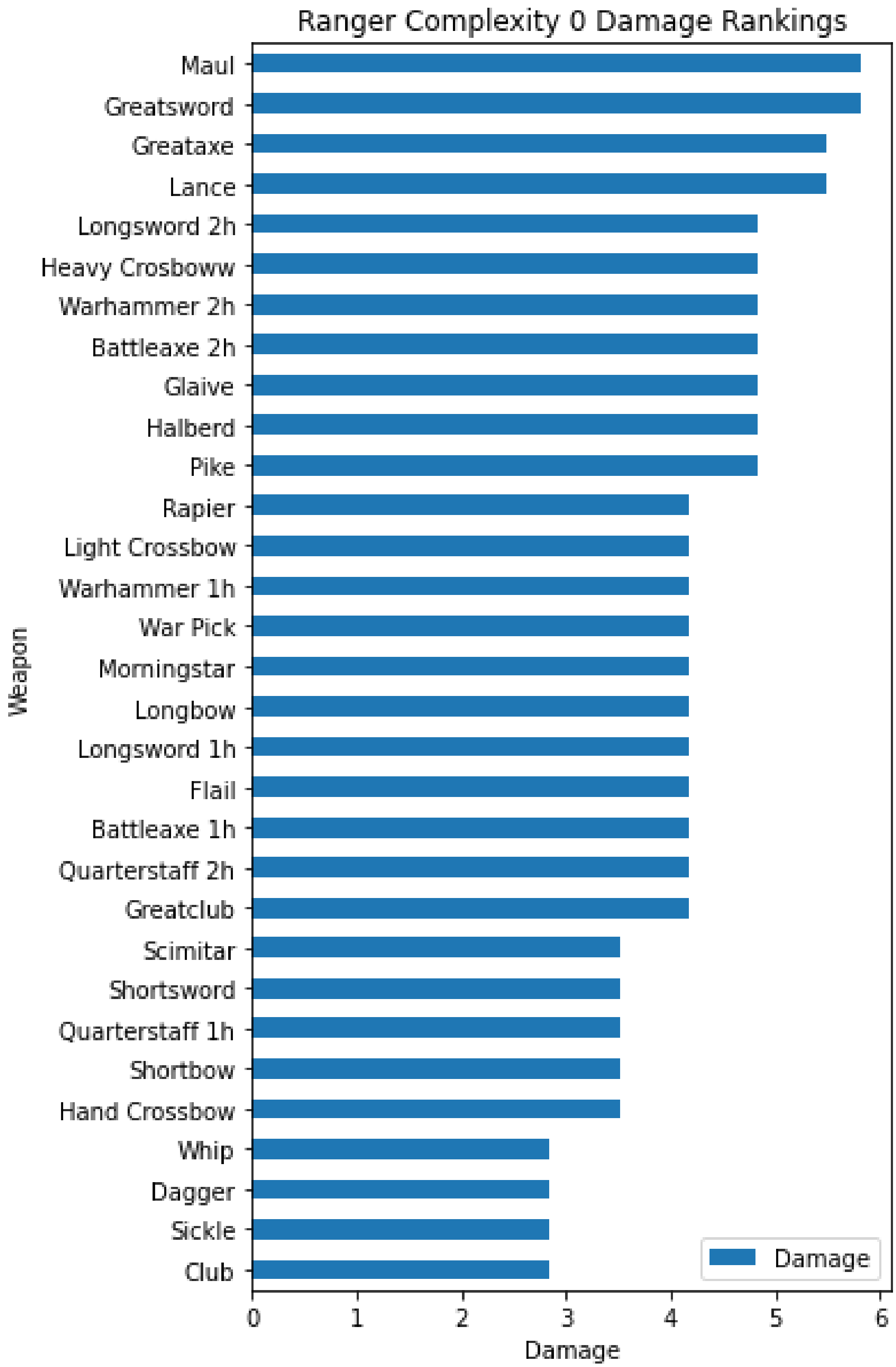

Barbarian Complexity 4 Damage Rankings
GWM + Extra Attack + Brutal Critical + Rage
GWM + Extra Attack + Brutal Critical + Savage Attacker
GWM + Extra Attack + Brutal Critical + Advantage
GWM + Rage + Extra Attack + Frenzy
Modifier
0
10
20
30
40
50
60
Damage

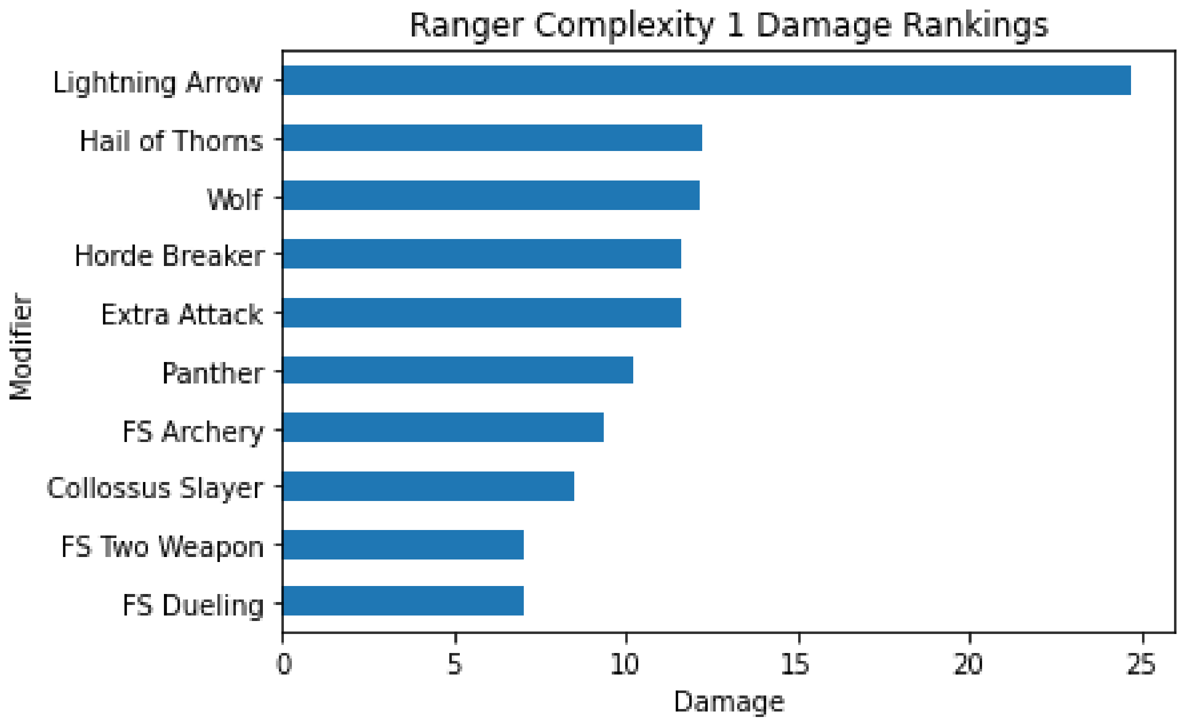

Barbarian Complexity 5 Damage Rankings
GWM + Extra Attack + Brutal Critical + Rage + Frenzy
GWM + Extra Attack + Brutal Critical + Rage + Savage Attacker
GWM + Extra Attack + Brutal Critical + Rage + Advantage
Modifier
0
10
20
30
40
50
60
70
80
Damage

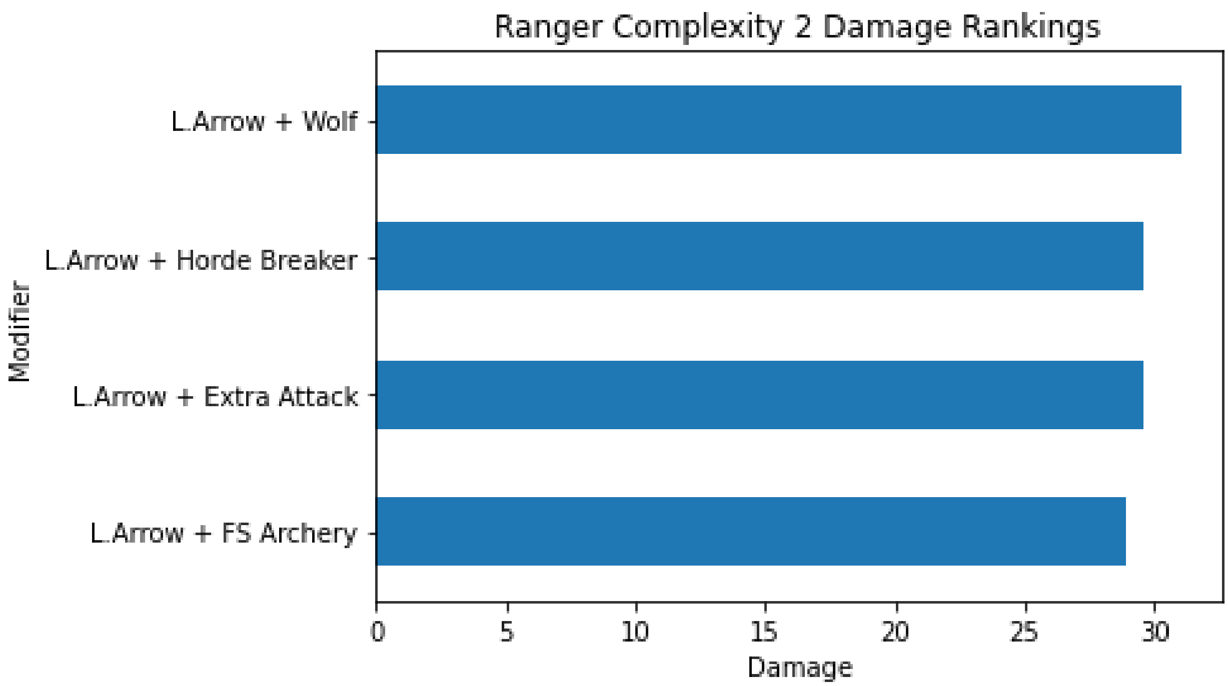

Barbarian Complexity 6 Damage Rankings
GWM + Extra Attack + Brutal Critical + Rage + Frenzy + Savage Attacker
GWM + Extra Attack + Brutal Critical + Rage + Frenzy + Advantage
Modifier
0
20
40
60
80
Damage

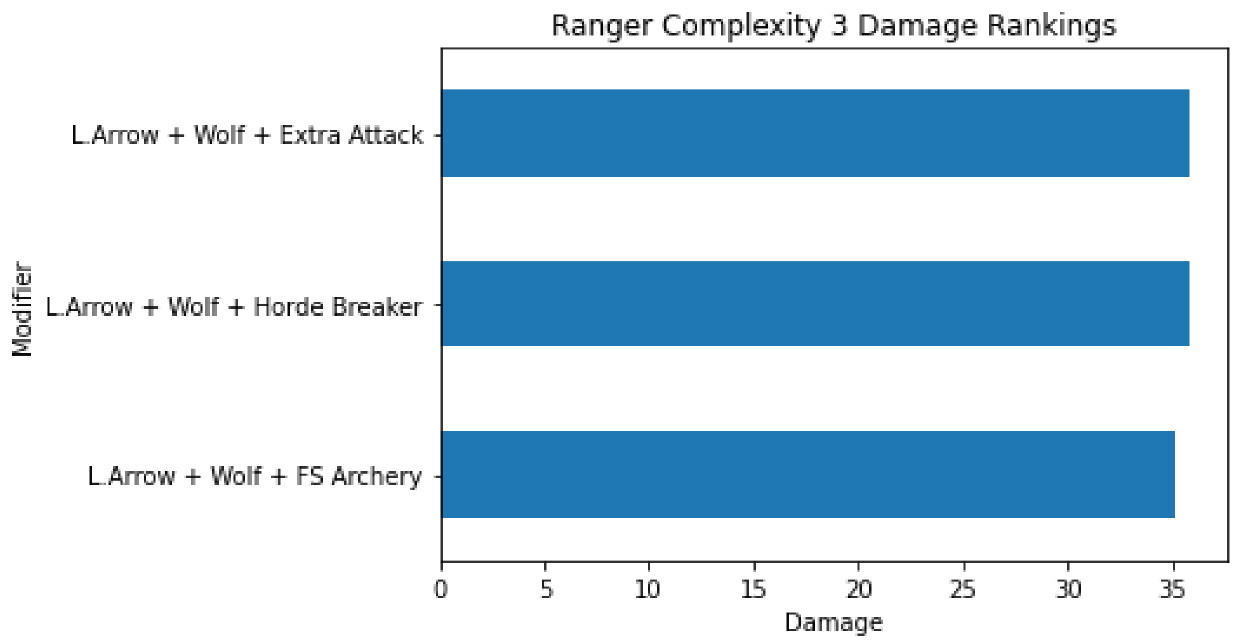

Barbarian Complexity 7 Damage Rankings
Modifier
GWM + Extra Attack + Brutal Critical + Rage + Frenzy + Savage Attacker
0
20
40
60
80
100
Damage

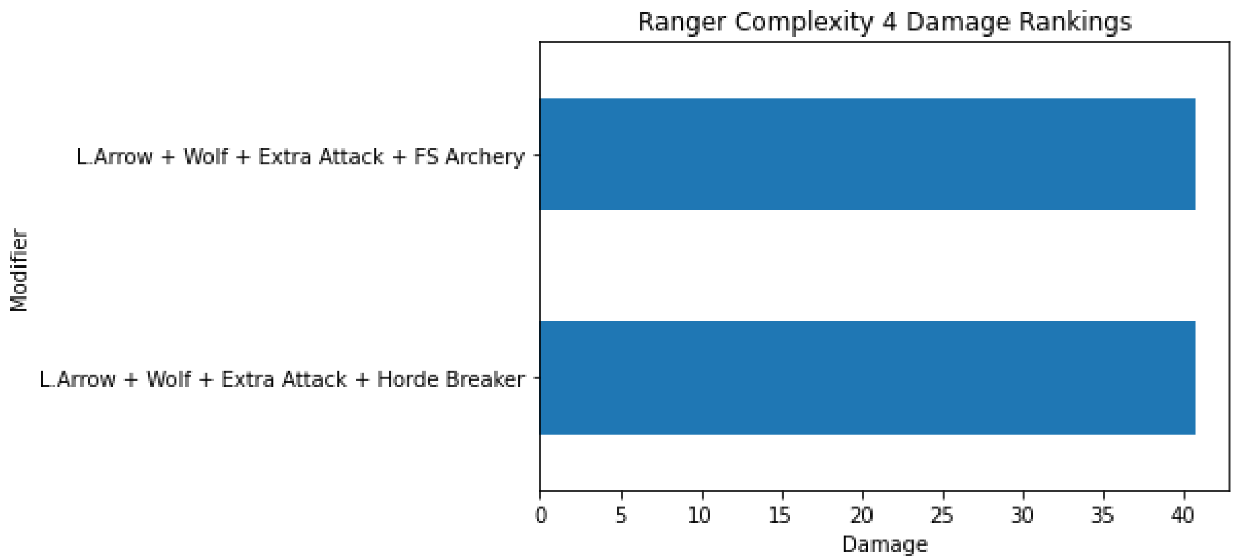

Correlation of Complexity and Damage
Damage Output
100
80
60
40
20
0
0
1
2
3
4
5
6
7
Complexity Level

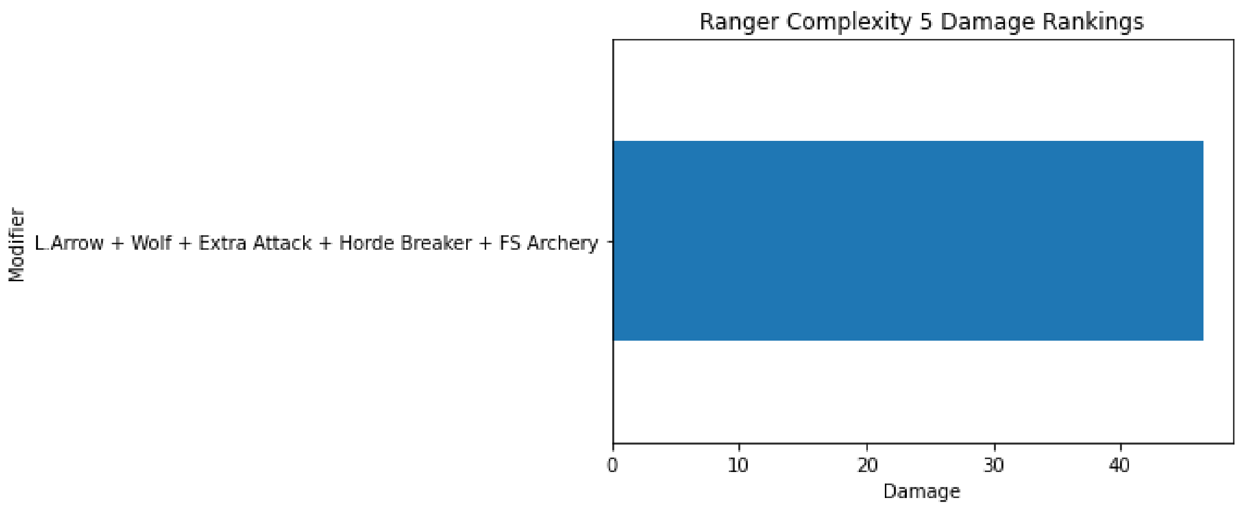

Ranger Complexity 0 Damage Rankings
Maul
Greatsword
Greataxe
Lance
Longsword 2h
Heavy Crosboww
Warhammer 2h
Battleaxe 2h
Glaive
Halberd
Pike
Rapier
Light Crossbow
Warhammer 1h
War Pick
Morningstar
Longbow
Longsword 1h
Flail
Battleaxe 1h
Quarterstaff 2h
Greatclub
Scimitar
Shortsword
Quarterstaff 1h
Shortbow
Hand Crossbow
Whip
Dagger
Sickle
Club
Weapon
Damage
0
1
2
3
4
5
6
Damage

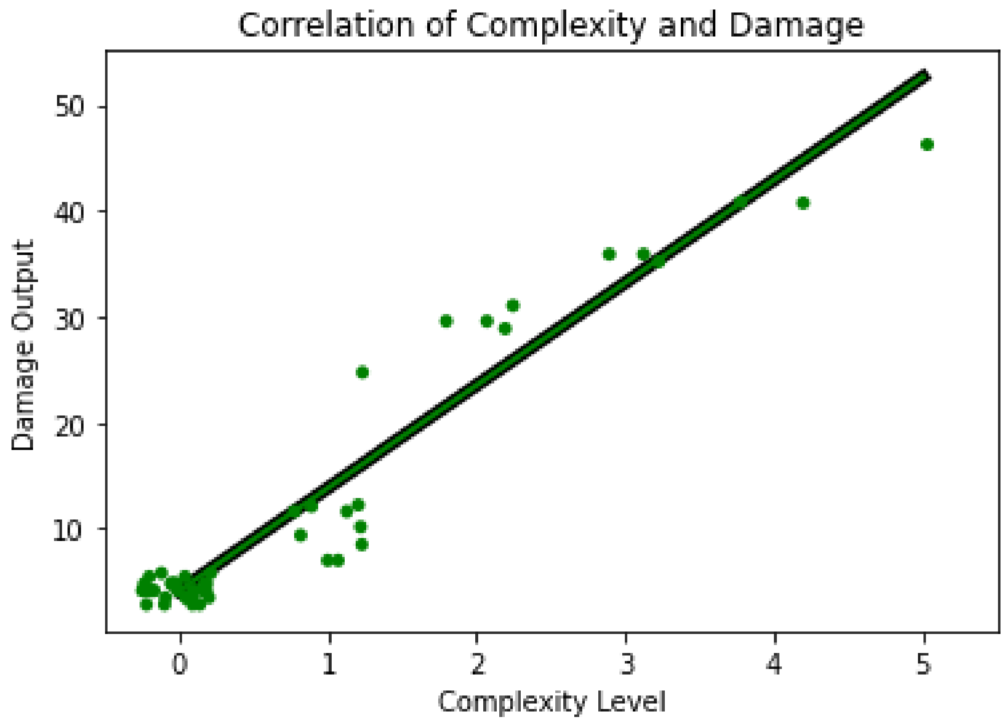

Ranger Complexity 1 Damage Rankings
Lightning Arrow
Hail of Thorns
Wolf
Horde Breaker
Extra Attack
Panther
FS Archery
Collossus Slayer
FS Two Weapon
FS Dueling
Modifier
0
5
10
15
20
25
Damage

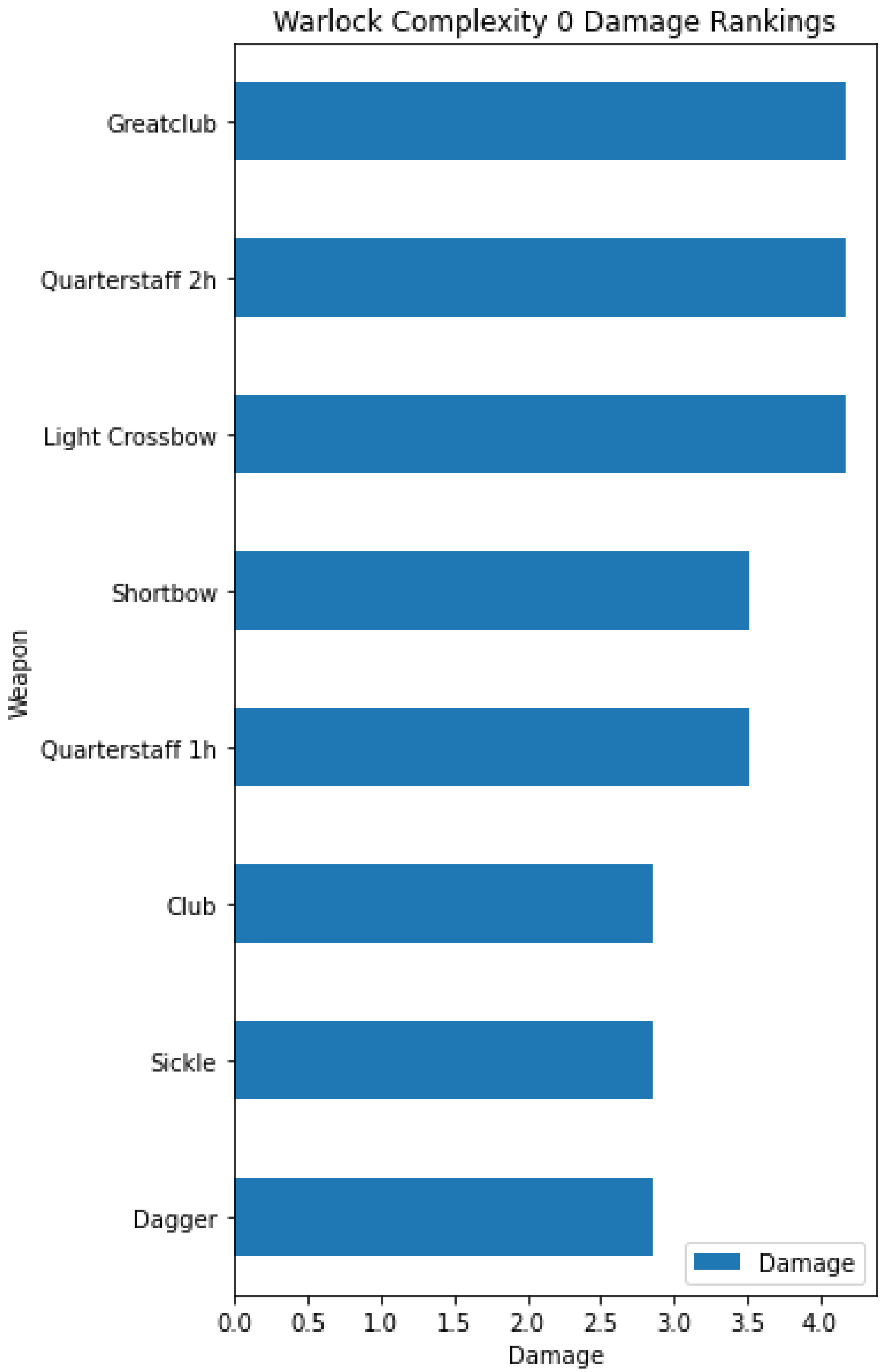

Ranger Complexity 2 Damage Rankings
L.Arrow + Wolf
L.Arrow + Horde Breaker
L.Arrow + Extra Attack
L.Arrow + FS Archery
Modifier
0
5
10
15
20
25
30
Damage

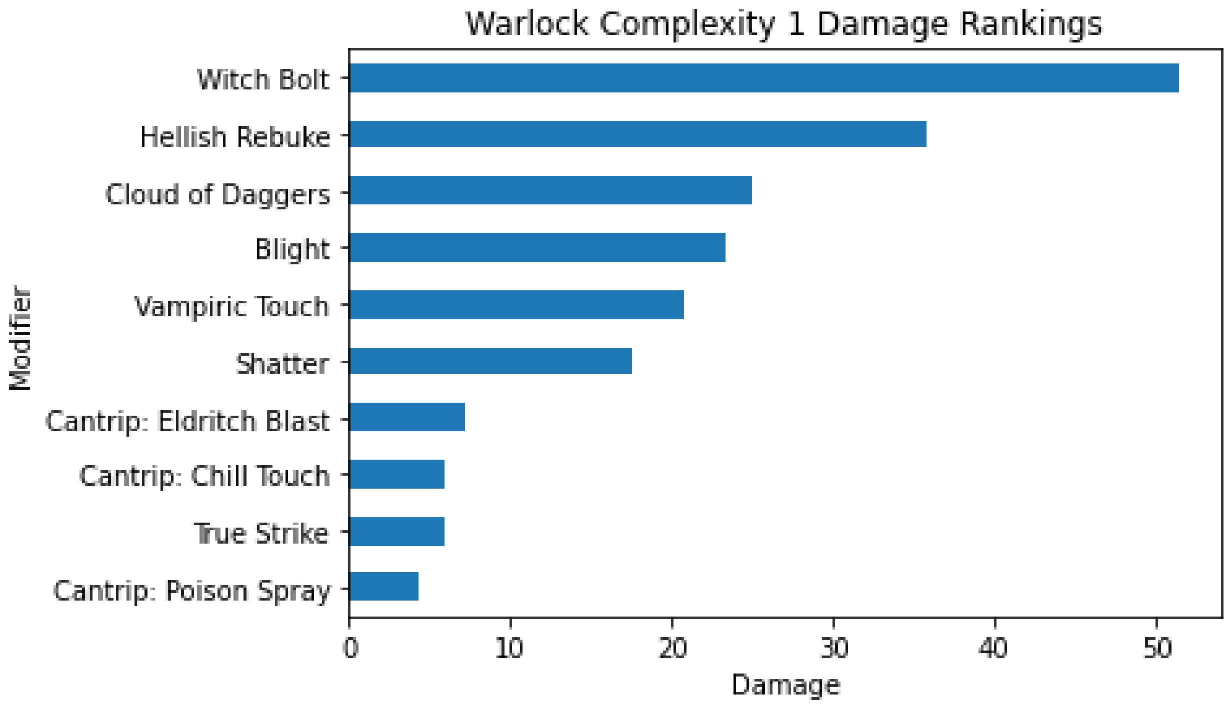

Ranger Complexity 3 Damage Rankings
L.Arrow + Wolf + Extra Attack
L.Arrow + Wolf + Horde Breaker
L.Arrow + Wolf + FS Archery
Modifier
0
5
10
15
20
25
30
35
Damage

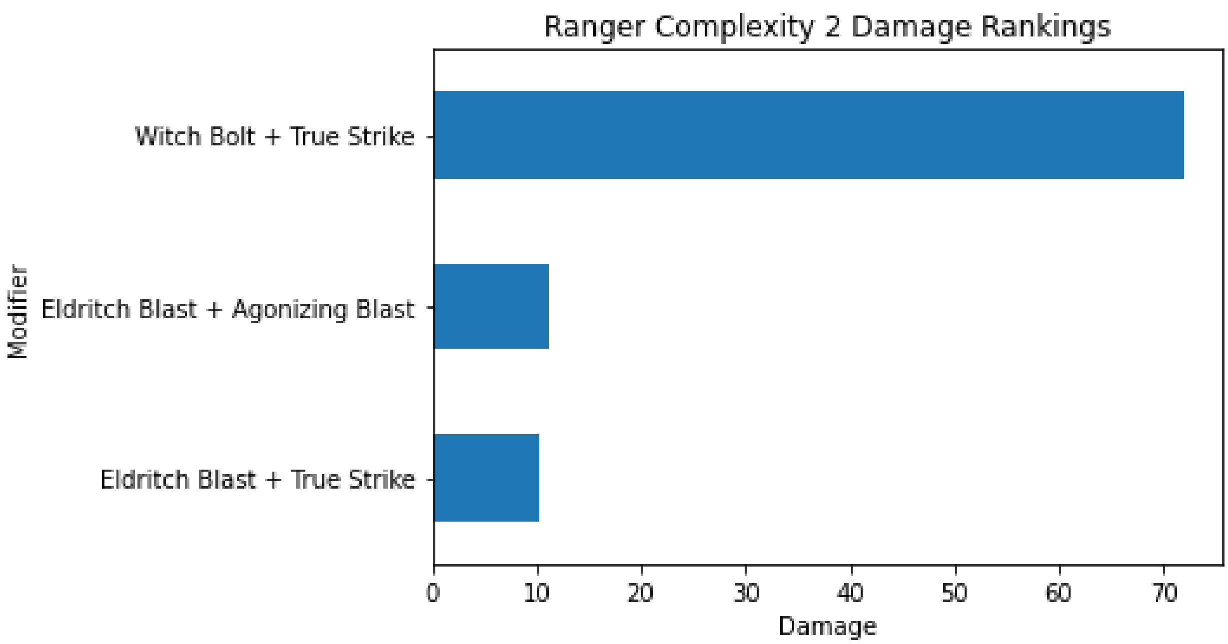

Ranger Complexity 4 Damage Rankings
L.Arrow + Wolf + Extra Attack + FS Archery
L.Arrow + Wolf + Extra Attack + Horde Breaker
Modifier
0
5
10
15
20
25
30
35
40
Damage

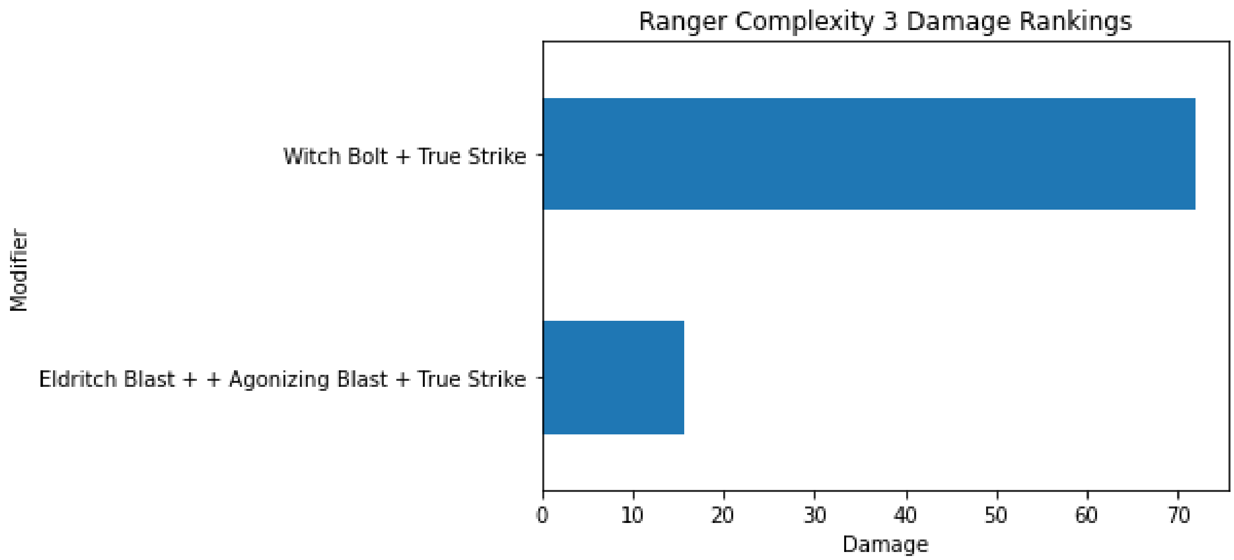

Ranger Complexity 5 Damage Rankings
L.Arrow + Wolf + Extra Attack + Horde Breaker + FS Archery
Modifier
0
10
20
30
40
Damage

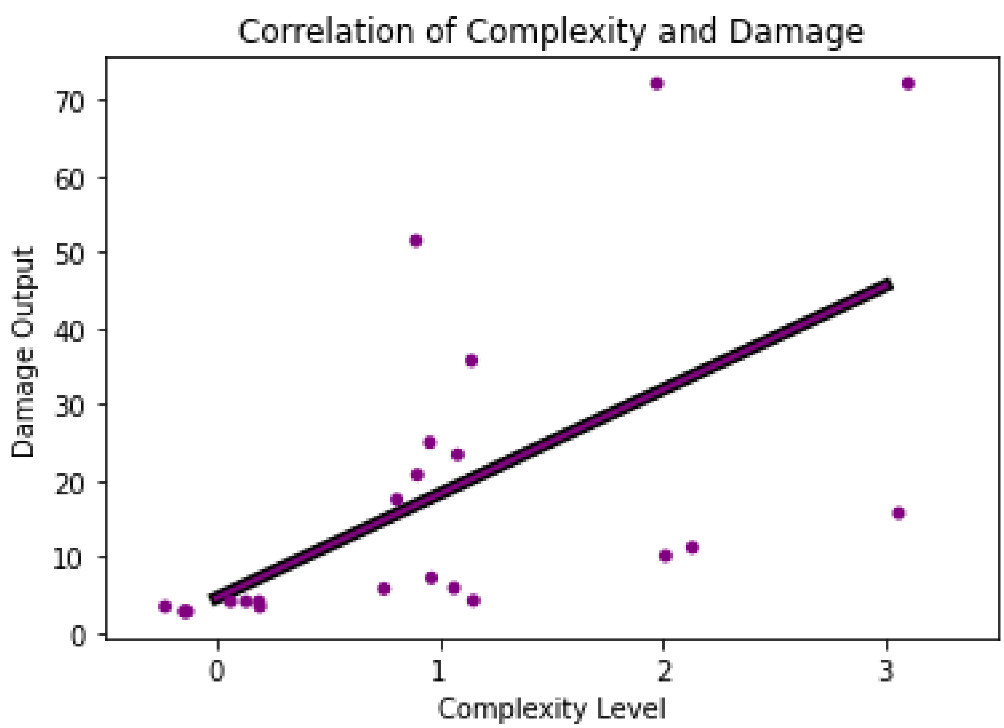

Correlation of Complexity and Damage
Damage Output
Complexity Level
50
40
30
20
10
0
1
2
3
4
5

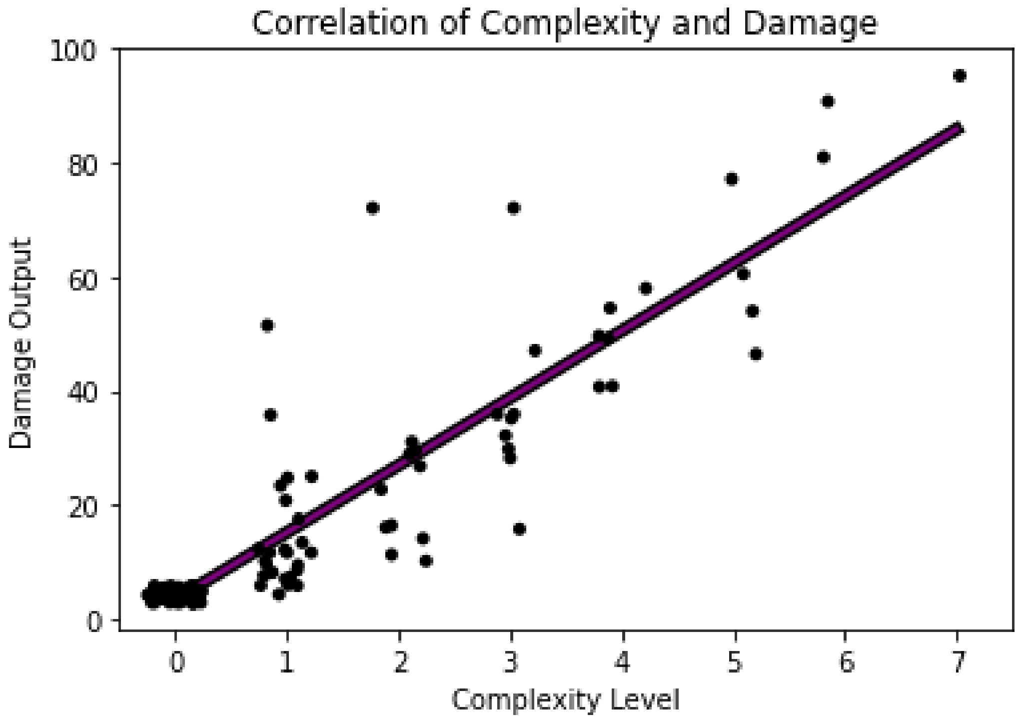

Warlock Complexity 0 Damage Rankings
Greatclub
Quarterstaff 2h
Light Crossbow
Shortbow
Quarterstaff 1h
Club
Sickle
Dagger
Weapon
Damage
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Damage


Warlock Complexity 1 Damage Rankings
Modifier
Witch Bolt
Hellish Rebuke
Cloud of Daggers
Blight
Vampiric Touch
Shatter
Cantrip: Eldritch Blast
Cantrip: Chill Touch
True Strike
Cantrip: Poison Spray
0
10
20
30
40
50
Damage


Ranger Complexity 2 Damage Rankings
Modifier
Witch Bolt + True Strike
Eldritch Blast + Agonizing Blast
Eldritch Blast + True Strike
0
10
20
30
40
50
60
70
Damage


Ranger Complexity 3 Damage Rankings
Modifier
Witch Bolt + True Strike
Eldritch Blast + + Agonizing Blast + True Strike
0
10
20
30
40
50
60
70
Damage


Correlation of Complexity and Damage
Damage Output
0
10
20
30
40
50
60
70
0
1
2
3
Complexity Level



J1. Sources

Flick, J. (2023). *Anydice Savage Attacker*. Anydice. Retrieved July 12, 2023, from https://anydice.com/

Dungeons and Dragons Fifth Edition Player's Handbook. Wizards of the Coast, 2014.

The Curse of Strahd. Wizards of the Coast, 2016.

Wizards of the Coast. (2023). *Adventurers League: Organized Play: Dungeons & Dragons.* D&D Official | Dungeons & Dragons. https://dnd.wizards.com/adventurers-league