F I C T I O N A L H E R O E S A N A L Y S I S

IT606 – Programming Lab (Python)

Group Members:

1. Dhvani Golani 202118020
2. Vashishth Raval 202118026

Table of Contents

Project Description

Since childhood, we always wanted to possess the powers of our superheroes whom we looked up to. Super strength and blazing laser from eyes like Superman, Spidey sense like that of Spiderman, super speed of the Flash, magic like that of Harry Potter and many more to count. Such strong faith we had that we thought nothing can go wrong until such superheroes exist in this world. As we grew older, we began to realize that superheroes are works of fiction and are not real, though they still have a large impact on each and everybody's lives. Many publishing houses like Marvel, DC Comics, NBC, J.K. Rowling Publishing house, etc. are still working to modernize superhero characters so as to stimulate the crazy kid within the young and the old.

Exploratory Data Analysis and Statistical Analysis of all the Marvel, DC, Anime, and Indian superheroes and supervillains datasets and you can battle between any two of them and see that who wins the battle.

In this project, we have taken a dataset comprising of various attributes of all the marvel, DC, and many other fictional superheroes and performed exploratory data analysis on top of this data. We have used pandas and matplotlib libraries in Python to create a prediction model for superhero characters alignment.

Analysis Performed

IMPORT LIBRARIES

```
# Loading the required libraries
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
# Libraries for ploting
import numpy as np
import os
import plotly.offline as pyo
import plotly.graph_objs as go
import plotly.offline as py
from plotly import tools
import numpy as np
import matplotlib
from plotly.offline import iplot
import warnings
warnings.filterwarnings("ignore")
import cufflinks as cf
cf.go_offline()
import plotly.io as pio
pio.renderers.default='colab'
```

- Pandas Library for data manipulation and analysis.
- Pyplot is a module of Matplotlib for plotting 2D graphics.
- Seaborn Library for data visualization and exploratory data analysis.
- Cufflinks is another library that connects the Pandas data frame with Plotly enabling users to create visualizations directly from Pandas.
- Also, iplot is imported for interactive plots

READING DATASET

- Read_csv is an important pandas function to read csv files and do operations on it.

```
[ ]  # Importing CSV file to DataFrame format
     marvel=pd.read_csv('/content/charcters_stats.csv')
```

- The shape property returns a tuple representing the dimensionality of the DataFrame.

```
# Number of rows and columns
marvel.shape
```

```
(420, 9)
```

- The head() function is used to get the first n rows. By default, it returns first 5 rows of the DataFrame.

```
# Top 5 rows of DataFrame
marvel.head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
0	Alan Scott	good	63	80	23	90	98	32	386
1	Amazo	bad	75	100	100	100	100	100	575
2	Angel	good	63	13	46	64	17	42	245
3	Angel Salvadore	good	38	10	28	28	46	60	210
4	Animal Man	good	56	48	47	85	73	80	389

- The tail() function is used to get the last n rows. By default, it returns last 5 rows of the DataFrame.

```
[ ] # Last 5 rows of DataFrame
    marvel.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
415	Krish	good	60	66	70	51	66	55	368
416	Ra-One	bad	50	55	40	22	60	50	277
417	G-One	good	60	50	40	22	50	50	272
418	Bahubali	good	70	82	88	42	66	70	418
419	Hero	good	35	50	26	14	56	33	214

- Pandas describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values.

```
[ ]  # statistical data of all the characters
     marvel.describe()
```

	Intelligence	Strength	Speed	Durability	Power	Combat	Total
count	420.000000	420.000000	420.000000	420.000000	420.000000	420.000000	420.000000
mean	62.388095	41.052381	38.035714	58.102381	57.421429	60.521429	317.611905
std	20.839524	32.189877	22.577074	30.208764	26.865901	23.131708	105.245409
min	6.000000	1.000000	1.000000	1.000000	0.000000	6.000000	60.000000
25%	50.000000	10.000000	23.000000	28.000000	35.000000	42.000000	239.000000
50%	63.000000	32.000000	33.000000	56.000000	58.000000	64.000000	303.500000
75%	75.000000	73.000000	50.000000	85.000000	75.250000	80.000000	381.750000
max	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	600.000000

- Count function is used to get the count of non-NA cells for each column or row.

```
[ ]  # count not null cells for each column or row
     marvel.count()
```

```
Name          420
Alignment     420
Intelligence  420
Strength      420
Speed         420
Durability    420
Power         420
Combat        420
Total         420
dtype: int64
```

- Extracting two columns namely, Name and Strength, from Marvel DataFrame.

```
[ ]  # Extract two columns form DataFrame
     marvel[['Name','Strength']].head(12)
```

	Name	Strength
0	Alan Scott	80
1	Amazo	100
2	Angel	13
3	Angel Salvadore	10
4	Animal Man	48
5	Annihilus	80
6	Ant-Man	10
7	Ant-Man II	10
8	Anti-Monitor	90
9	Apocalypse	100
10	Aquababy	16
11	Aqualad	44

- value_counts() function returns object containing counts of unique values.
- Here, we've extracted counts of unique values for the column named Alignment.

```
[ ]  # Counts of unique values for Alignment
     alignment=marvel['Alignment'].value_counts()
```

```
[ ]  # Display Counts of unique values for Alignment
     alignment
```

```
good       292
bad        117
neutral     11
Name: Alignment, dtype: int64
```

- We created a DataFrame for the counts of unique values for the column named Alignment.

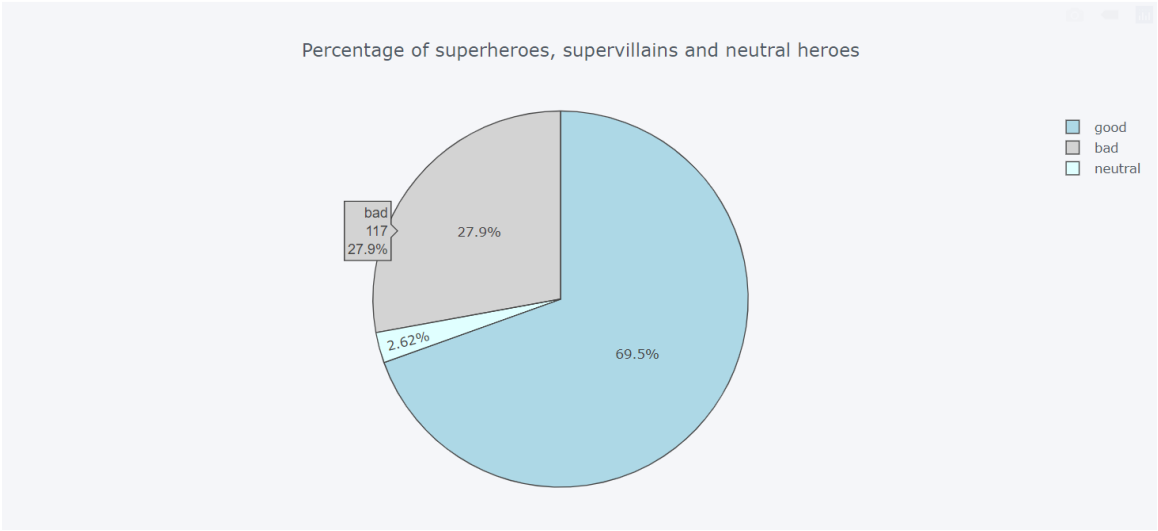
```
[ ]  # Create DataFrame for Counts of unique values for Alignment
     data_plot=pd.DataFrame({'labels':alignment.index,'counts':alignment.values})
```

```
[ ]  # Display DataFrame for Counts of unique values for Alignment
     data_plot
```

	labels	counts
0	good	292
1	bad	117
2	neutral	11

- Pie Chart for the counts of unique values for the column named Alignment.

```
#Pie Chart for Counts of unique values for Alignment
data_plot.iplot(kind='pie',labels='labels',values='counts',colors=['lightblue','lightgrey','lightcyan'],
                width=1,title='Percentage of superheroes, supervillains and neutral heroes')
```



- The pie chart is considered as one of the important types of data representation. Each segment or sector forms a certain portion of the total percentage.

- Thus, from the pie chart we can conclude that, there are 69.5% Superheroes, 27.5% Supervillains and 2.62% neutral heroes.

SUPERHERO ANALYSIS

- For Superhero Analysis, we have created a DataFrame where the value for Alignment is good.

```
[ ]  # DataFrame for Superheroes (Alignment = good)
     good=marvel[marvel['Alignment']=="good"]
```

- The head() function is used to get the first n rows. By default, it returns first 5 rows of the DataFrame.

- The tail() function is used to get the last n rows. By default, it returns last 5 rows of the DataFrame.

```
[ ]  # Top 5 rows of DataFrame
     good.head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
0	Alan Scott	good	63	80	23	90	98	32	386
2	Angel	good	63	13	46	64	17	42	245
3	Angel Salvadore	good	38	10	28	28	46	60	210
4	Animal Man	good	56	48	47	85	73	80	389
6	Ant-Man	good	100	10	23	28	32	32	225

```
[ ]  # Last 5 rows of DataFrame
     good.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
414	Shaktiman	good	60	56	55	33	56	25	285
415	Krish	good	60	66	70	51	66	55	368
417	G-One	good	60	50	40	22	50	50	272
418	Bahubali	good	70	82	88	42	66	70	418
419	Hero	good	35	50	26	14	56	33	214

- Pandas sort_values() function sorts a data frame in Ascending or Descending order of passed Column.

- So here we have sorted the data frame according to the speed in descending order.

```
[ ]  # Sort Superheroes by their Speed
     good.sort_values(by=['Speed'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
269	Nova	good	38	60	100	100	100	25	423
172	Impulse	good	50	10	100	60	63	60	343
43	Black Bolt	good	75	67	100	84	100	56	482
203	Krypto	good	9	80	100	90	72	40	391
347	Stardust	good	88	85	100	100	100	85	558

- The following code helps extracting Superheroes with Speed greater than equal to 90.

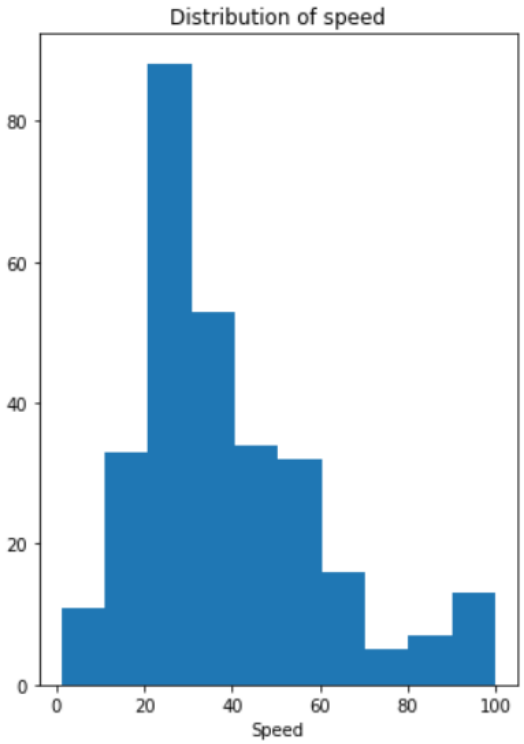
```
[ ]  # Superheroes with speed greater than equal to 90
     good[good['Speed']>=90]
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
43	Black Bolt	good	75	67	100	84	100	56	482
135	Flash	good	63	10	100	60	83	32	348
172	Impulse	good	50	10	100	60	63	60	343
179	Jack of Hearts	good	63	55	100	30	70	30	348
203	Krypto	good	9	80	100	90	72	40	391
226	Martian Manhunter	good	100	100	96	100	100	85	581
269	Nova	good	38	60	100	100	100	25	423
288	Quicksilver	good	63	28	100	60	57	56	364
312	Saitama	good	100	100	100	100	100	100	600
347	Stardust	good	88	85	100	100	100	85	558
358	Supergirl	good	94	98	92	100	85	75	544
359	Superman	good	100	100	100	100	94	85	579
369	Thor	good	69	100	92	100	100	85	546

- A histogram is a graphical representation that organizes a group of data points into user-specified ranges.

- By plotting this Histogram, we can conclude that there are maximum number of superheroes having speed in the range of 20 to 30 and least number of superheroes lie in the speed range of 70 to 80.

```
[ ]  # Histogram for speed for good superheroes
     plt.figure(figsize=(5,7))
     plt.hist(good['Speed'])
     plt.title("Distribution of speed")
     plt.xlabel("Speed")
     plt.show()
```



- So here we have sorted the data frame according to the power in descending order.

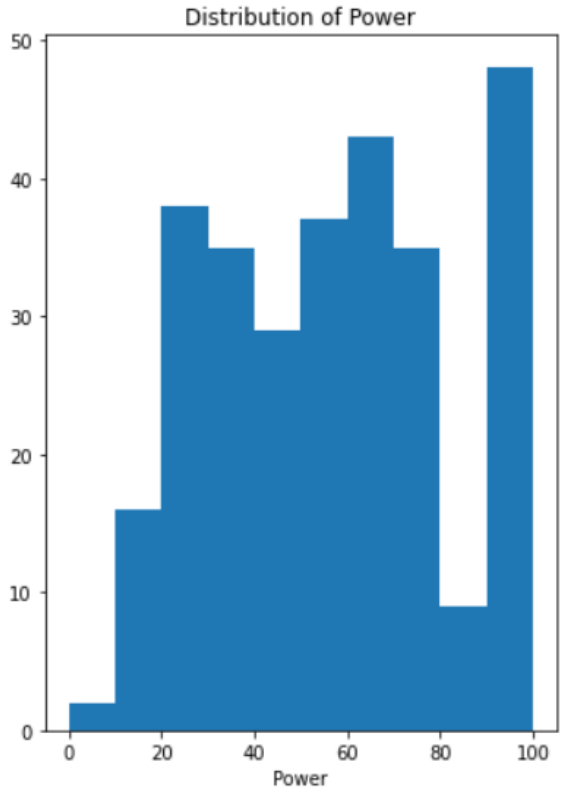
```
[ ]  # Sort Superheroes by their Power
     good.sort_values(by=['Power'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
99	Deadman	good	50	10	33	100	100	42	335
323	Shadow King	good	75	12	27	100	100	75	389
116	Dr Manhattan	good	88	32	42	95	100	42	399
396	Watcher	good	100	80	67	89	100	56	492
111	Doctor Strange	good	100	10	12	84	100	60	366

- A histogram is a graphical representation that organizes a group of data points into user-specified ranges.

- By plotting this Histogram, we can conclude that there are maximum number of superheroes having power in the range of 90 to 100 and least number of superheroes lie in the power range of 0 to 10.

```
[ ]  # Histogram for power for good superheroes
     plt.figure(figsize=(5,7))
     plt.hist(good['Power'])
     plt.title("Distribution of Power")
     plt.xlabel("Power")
     plt.show()
```



- Here we have sorted the data frame according to the Total of Superheroes in descending order and stored. Also using the head() and tail() function to view first and last 5 rows of the DataFrame.

```
[ ] # Sort Superheroes by their Total and store them
    good_max_power=good.sort_values(by=['Total'],ascending=False)
```

```
[ ] # Display Top 5 rows
    good_max_power.head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
312	Saitama	good	100	100	100	100	100	100	600
226	Martian Manhunter	good	100	100	96	100	100	85	581
359	Superman	good	100	100	100	100	94	85	579
347	Stardust	good	88	85	100	100	100	85	558
369	Thor	good	69	100	92	100	100	85	546

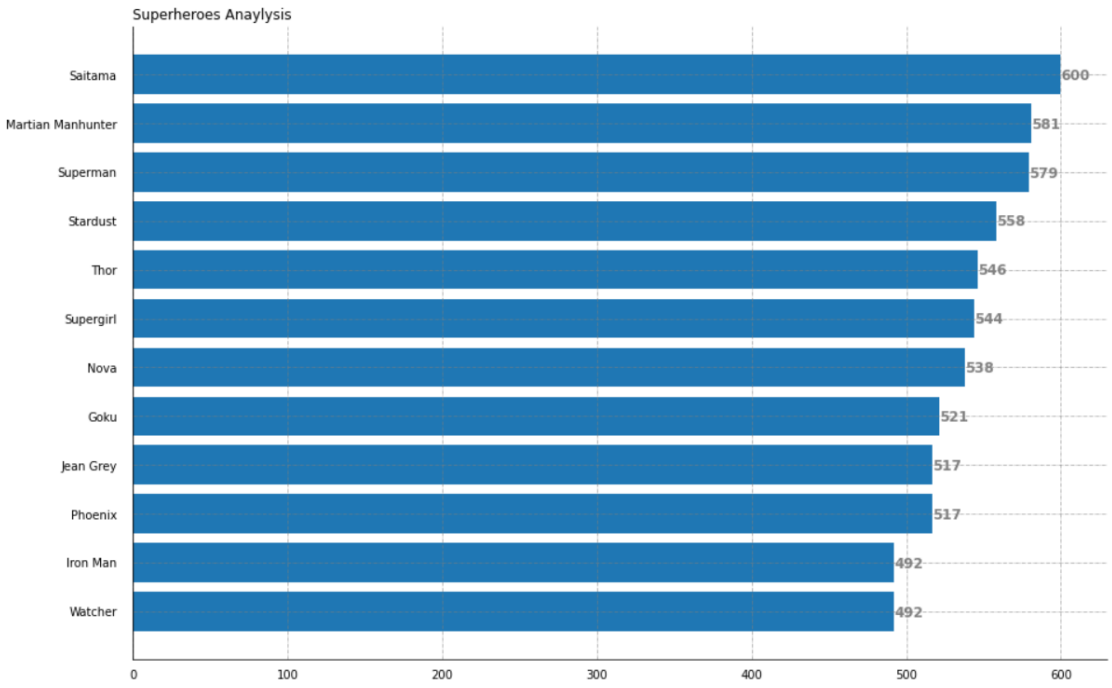
```
[ ] # Display Last 5 rows
    good_max_power.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
289	Quill	good	38	10	12	14	23	14	111
10	Aquababy	good	10	16	12	14	37	14	103
405	Wyatt Wingfoot	good	10	10	12	1	1	56	90
200	Kool-Aid Man	good	25	10	12	14	10	14	85
137	Franklin Storm	good	38	10	1	1	0	10	60

- The pictorial representation of a grouped data, in the form of vertical or horizontal rectangular bars, where the lengths of the bars are equivalent to the measure of data, are known as bar graphs or bar charts.

- The following is the code for Horizontal Bar Chart for Superheroes.

```
# Create DataFrame of superheroes
df = pd.DataFrame(good_max_power)

# taking value of top 12 superheroes and their total
name = df['Name'].head(12)
total = df['Total'].head(12)

# Figure Size
fig, ax = plt.subplots(figsize =(15, 10))

# Horizontal Bar Plot
ax.barh(name, total)
# Remove axes splines
for s in ['top', 'right']:
    ax.spines[s].set_visible(False)
# Remove x, y Ticks
ax.xaxis.set_ticks_position('none')
ax.yaxis.set_ticks_position('none')
ax.xaxis.set_tick_params(pad = 5)
ax.yaxis.set_tick_params(pad = 10)
# Add x, y gridlines
ax.grid(color ='grey',linestyle ='-.', linewidth = 0.5,)
# Show top values
ax.invert_yaxis()

# Add annotation to bars
for i in ax.patches:
    plt.text(i.get_width()+0.2, i.get_y()+0.5,
             str(round((i.get_width()), 2)),
             fontsize = 12, fontweight ='bold',
             color ='grey')

# Add Plot Title
ax.set_title('Superheroes Anaylysis',loc ='left')

# Show Plot
plt.show()
```



SUPERVILLAIN ANALYSIS

- For Supervillain Analysis, we have created a DataFrame where the value for Alignment is bad.
- The head() function is used to get the first n rows. By default, it returns first 5 rows of the DataFrame.

- The tail() function is used to get the last n rows. By default, it returns last 5 rows of the DataFrame.

```
[ ] # DataFrame for Supervillains (Alignment = bad)
    bad=marvel[marvel['Alignment']=="bad"]
```

```
[ ] # Display Top 5 rows
    bad.head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
1	Amazo	bad	75	100	100	100	100	100	575
5	Annihilus	bad	75	80	47	56	59	64	381
8	Anti-Monitor	bad	88	90	38	90	100	90	496
9	Apocalypse	bad	100	100	33	100	100	60	493
14	Arclight	bad	38	63	23	42	52	70	288

```
[ ] # Display Last 5 rows
    bad.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
390	Walrus	bad	50	28	8	50	11	20	167
393	Warp	bad	38	10	23	28	63	50	212
398	Willis Stryker	bad	38	16	23	28	41	60	206
412	Zoom	bad	50	10	100	28	72	28	288
416	Ra-One	bad	50	55	40	22	60	50	277

- Pandas sort_values() function sorts a data frame in Ascending or Descending order of passed Column.

- So here we have sorted the data frame according to the strength in descending order.

```
# Sort Supervillains by their Strength
bad.sort_values(by=['Strength'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
1	Amazo	bad	75	100	100	100	100	100	575
142	General Zod	bad	94	100	96	100	94	95	579
9	Apocalypse	bad	100	100	33	100	100	60	493
221	Magus	bad	88	100	70	99	100	74	531
42	Black Adam	bad	88	100	92	100	89	56	525

- The following code helps extracting Supervillain with strength greater than equal to 90.

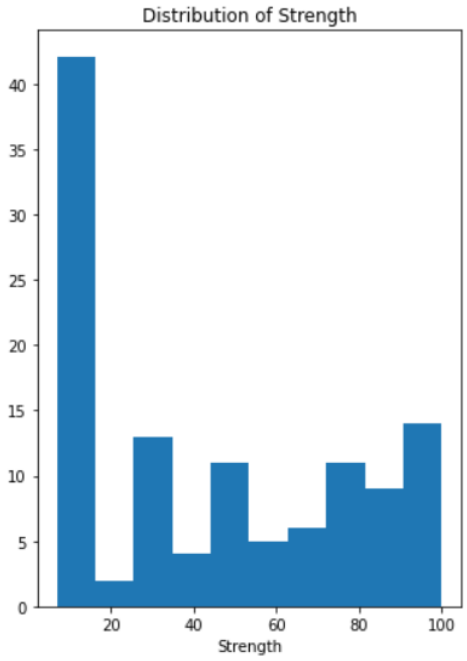
```
[ ] # Supervillains with strength greater than equal to 90
    bad[bad['Strength']>=90]
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
1	Amazo	bad	75	100	100	100	100	100	575
8	Anti-Monitor	bad	88	90	38	90	100	90	496
9	Apocalypse	bad	100	100	33	100	100	60	493
42	Black Adam	bad	88	100	92	100	89	56	525
88	Cyborg Superman	bad	75	93	92	100	100	80	540
95	Darkseid	bad	88	100	23	100	100	95	506
105	Destroyer	bad	50	95	58	98	90	70	461
115	Dormammu	bad	88	95	83	100	100	80	546
128	Faora	bad	88	95	75	100	87	90	535
142	General Zod	bad	94	100	96	100	94	95	579
221	Magus	bad	88	100	70	99	100	74	531
229	Match	bad	75	95	83	85	90	70	498
232	Maxima	bad	75	90	35	56	67	75	398
339	Solomon Grundy	bad	9	93	13	100	78	30	323
357	Superboy-Prime	bad	94	100	100	100	100	85	579
365	Thanos	bad	88	100	17	100	100	80	485

- A histogram is a graphical representation that organizes a group of data points into user-specified ranges.

- By plotting this Histogram, we can conclude that there are maximum number of Supervillain having strength in the range of 5 to 15 and least number of supervillains lie in the strength range of 15 to 25.

```
[ ]  # Histogram for Strength for bad Supervillains
     plt.figure(figsize=(5,7))
     plt.hist(bad['Strength'])
     plt.title("Distribution of Strength")
     plt.xlabel("Strength")
     plt.show()
```



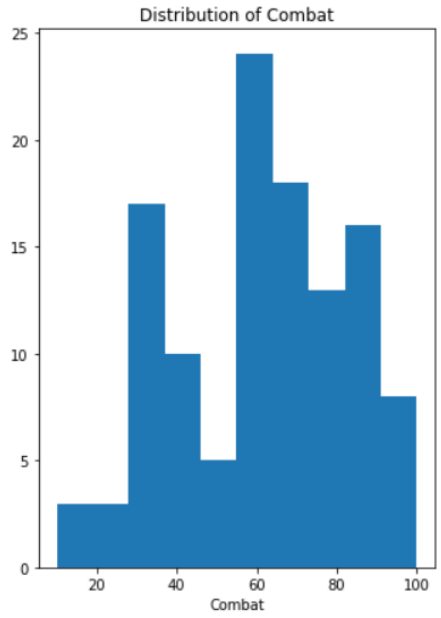
- So here we have sorted the data frame according to the combat in descending order.

```
[ ]  # Sort Supervillains by their Combat
     bad.sort_values(by=['Combat'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
1	Amazo	bad	75	100	100	100	100	100	575
290	Ra's Al Ghul	bad	100	28	32	42	27	100	329
310	Sabretooth	bad	55	53	25	90	28	100	351
95	Darkseid	bad	88	100	23	100	100	95	506
301	Rick Flag	bad	88	11	23	28	19	95	264

- By plotting this Histogram, we can conclude that there are maximum number of Supervillain having combat in the range of 60 to 70 and least number of supervillains lie in the combat range of 50 to 60.

```
[ ]  # Histogram for Combat for bad Supervillains
     plt.figure(figsize=(5,7))
     plt.hist(bad['Combat'])
     plt.title("Distribution of Combat")
     plt.xlabel("Combat")
     plt.show()
```



- Here we have sorted the data frame according to the Total of supervillains in descending order and stored. Also using the head() and tail() function to view first and last 5 rows of the DataFrame.

```
[ ]  # Sort Supervillains by their Total and store them
     bad_max_power=bad.sort_values(by=['Total'],ascending=False)
```

```
[ ]  # Display Top 5 rows
     bad_max_power.head()
```

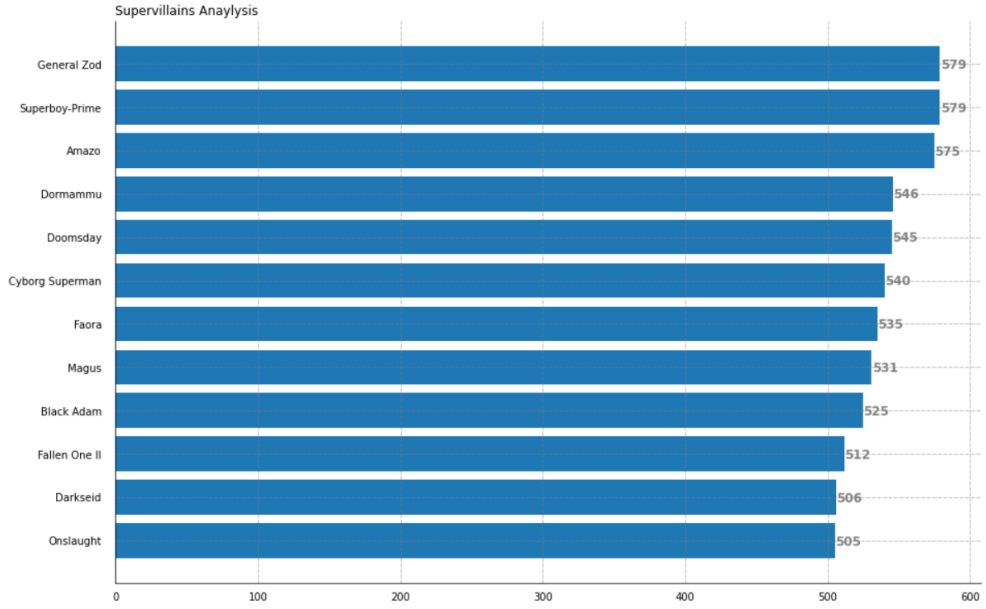
	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
142	General Zod	bad	94	100	96	100	94	95	579
357	Superboy-Prime	bad	94	100	100	100	100	85	579
1	Amazo	bad	75	100	100	100	100	100	575
115	Dormammu	bad	88	95	83	100	100	80	546
113	Doomsday	bad	88	80	67	100	100	90	545

```
[ ]  # Display Last 5 rows
     bad_max_power.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
380	Two-Face	bad	88	10	12	14	9	28	161
302	Riddler	bad	100	10	12	14	10	14	160
334	Siren	bad	38	10	12	14	55	28	157
286	Pyro	bad	38	10	12	14	50	28	152
296	Red Mist	bad	25	10	23	14	20	56	148

- The pictorial representation of a grouped data, in the form of vertical or horizontal rectangular bars, where the lengths of the bars are equivalent to the measure of data, are known as bar graphs or bar charts.
- The following is the code for Horizontal Bar Chart for supervillains.

```
# Create DataFrame of Supervillains
df = pd.DataFrame(bad_max_power)

# taking value of top 12 supervillains and their total
name = df['Name'].head(12)
total = df['Total'].head(12)

# Figure Size
fig, ax = plt.subplots(figsize =(15, 10))
# Horizontal Bar Plot
ax.barh(name, total)
# Remove axes splines
for s in ['top', 'right']:
    ax.spines[s].set_visible(False)
# Remove x, y Ticks
ax.xaxis.set_ticks_position('none')
ax.yaxis.set_ticks_position('none')
# Add padding between axes and labels
ax.xaxis.set_tick_params(pad = 5)
ax.yaxis.set_tick_params(pad = 10)
# Add x, y gridlines
ax.grid(color ='grey',linestyle ='-.', linewidth = 0.5,)
# Show top values
ax.invert_yaxis()
# Add annotation to bars
for i in ax.patches:
    plt.text(i.get_width()+0.2, i.get_y()+0.5,
             str(round((i.get_width()), 2)),
             fontsize = 12, fontweight ='bold',
             color ='grey')
# Add Plot Title
ax.set_title('Supervillains Anaylysis',loc ='left')
# Show Plot
plt.show()
```


Supervillains Anaylysis
General Zod
Superboy-Prime
Amazo
Dormammu
Doomsday
Cyborg Superman
Faora
Magus
Black Adam
Fallen One II
Darkseid
Onslaught
579
579
575
546
545
540
535
531
525
512
506
505
0
100
200
300
400
500
600

NEUTRAL HEROES ANALYSIS

- For Neutral Heroes Analysis, we have created a DataFrame where the value for Alignment is neutral.
- The head() function is used to get the first n rows. By default, it returns first 5 rows of the DataFrame.

- The tail() function is used to get the last n rows. By default, it returns last 5 rows of the DataFrame.

```
[ ]  # DataFrame for neutral heroes (Alignment = neutral)
     neutral=marvel[marvel['Alignment']=="neutral"]
```

```
[ ]  # Display Top 5 rows
     neutral.head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
41	Bizarro	neutral	75	95	100	100	95	85	550
100	Deadpool	neutral	50	15	30	100	100	100	395
103	Deathstroke	neutral	75	30	35	100	36	90	366
139	Galactus	neutral	100	100	83	100	100	70	553
190	Juggernaut	neutral	44	100	42	100	74	70	430

```
[ ]  # Display Last 5 rows
     neutral.tail()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
292	Raven	neutral	50	10	29	70	62	40	261
294	Red Hood	neutral	75	12	23	28	24	95	257
313	Sandman	neutral	50	75	47	97	62	56	387
367	The Comedian	neutral	63	14	17	10	12	80	196
377	Toad	neutral	50	34	58	56	49	28	275

- Pandas sort_values() function sorts a data frame in Ascending or Descending order of passed Column.

- So here we have sorted the data frame according to the durability in descending order.

```
[ ] # Sort neutral heroes by their Durability
    neutral.sort_values(by=['Durability'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
41	Bizarro	neutral	75	95	100	100	95	85	550
100	Deadpool	neutral	50	15	30	100	100	100	395
103	Deathstroke	neutral	75	30	35	100	36	90	366
139	Galactus	neutral	100	100	83	100	100	70	553
190	Juggernaut	neutral	44	100	42	100	74	70	430

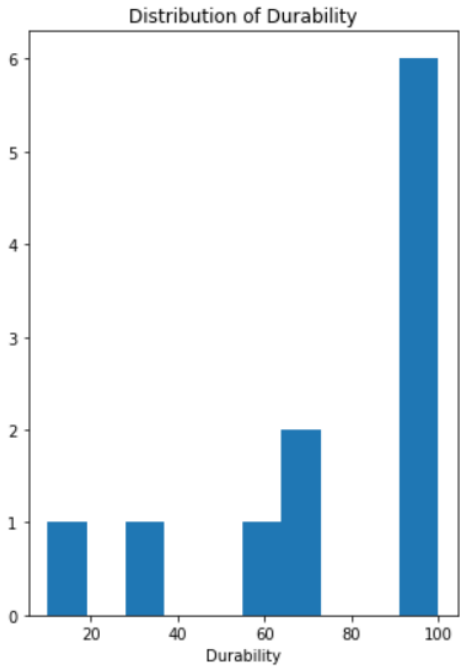
- The following code helps extracting neutrals heroes with durability greater than equal to 90.

```
[ ] # neutral heroes with durability greater than equal to 90
    neutral[neutral['Durability']>=90]
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
41	Bizarro	neutral	75	95	100	100	95	85	550
100	Deadpool	neutral	50	15	30	100	100	100	395
103	Deathstroke	neutral	75	30	35	100	36	90	366
139	Galactus	neutral	100	100	83	100	100	70	553
190	Juggernaut	neutral	44	100	42	100	74	70	430
313	Sandman	neutral	50	75	47	97	62	56	387

- By plotting this Histogram, we can conclude that there are maximum number of neutrals heroes having durability in the range of 90 to 100 and least number of neutrals heroes lie in the durability range of 70 to 80.

```
# Histogram for Durability for neutral heroes
plt.figure(figsize=(5,7))
plt.hist(neutral['Durability'])
plt.title("Distribution of Durability")
plt.xlabel("Durability")
plt.show()
```



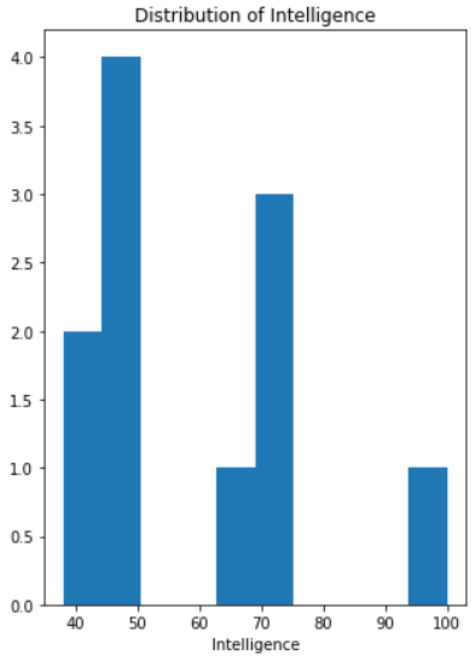
- So here we have sorted the data frame according to the intelligence in descending order.

```
[ ]  # Sort neutral heroes by their Intelligence
     neutral.sort_values(by=['Intelligence'],ascending=False).head()
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
139	Galactus	neutral	100	100	83	100	100	70	553
41	Bizarro	neutral	75	95	100	100	95	85	550
103	Deathstroke	neutral	75	30	35	100	36	90	366
294	Red Hood	neutral	75	12	23	28	24	95	257
367	The Comedian	neutral	63	14	17	10	12	80	196

- By plotting this Histogram, we can conclude that there are maximum number of neutrals heroes having intelligence in the range of 45 to 50 and least number of neutrals heroes lie in the intelligence range of 95 to 100.

```
[ ]  # Histogram for Intelligence for neutral heroes
     plt.figure(figsize=(5,7))
     plt.hist(neutral['Intelligence'])
     plt.title("Distribution of Intelligence")
     plt.xlabel("Intelligence")
     plt.show()
```



- Here we have sorted the data frame according to the Total of neutrals heroes in descending order and stored. Also using the head() and tail() function to view first 10 rows of the DataFrame.

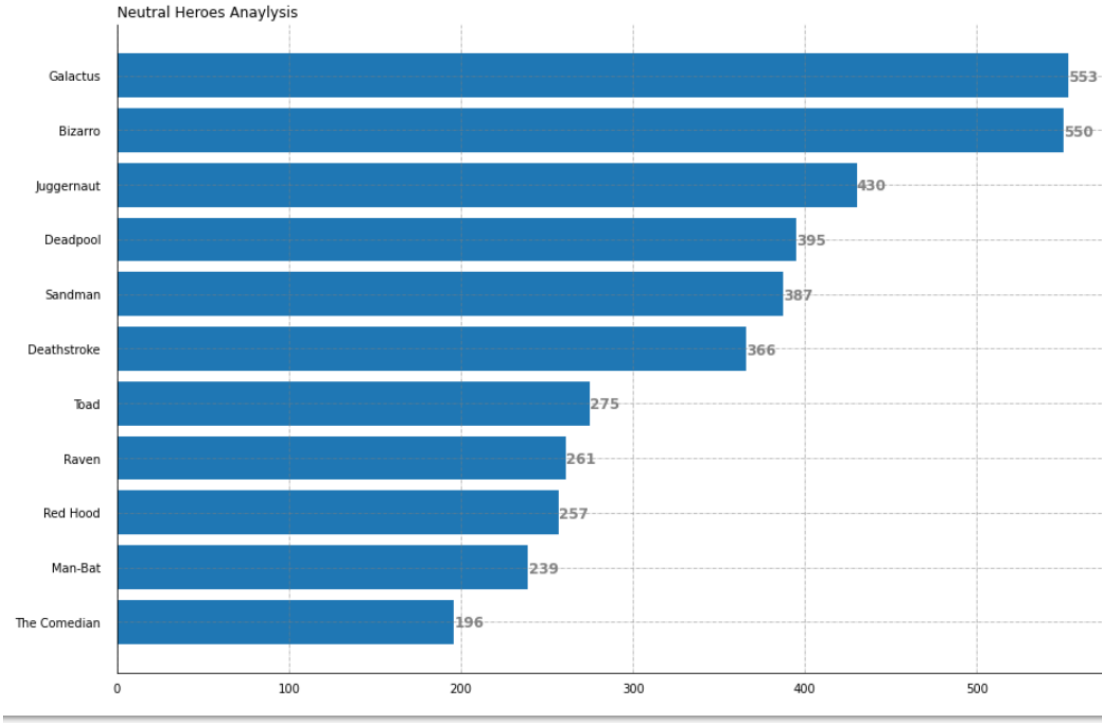
```
[ ]  # Sort neutral heroes by their Total and store them
     neutral_max_power=neutral.sort_values(by=['Total'],ascending=False)
```

```
[ ]  # Display all rows
     neutral_max_power
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
139	Galactus	neutral	100	100	83	100	100	70	553
41	Bizarro	neutral	75	95	100	100	95	85	550
190	Juggernaut	neutral	44	100	42	100	74	70	430
100	Deadpool	neutral	50	15	30	100	100	100	395
313	Sandman	neutral	50	75	47	97	62	56	387
103	Deathstroke	neutral	75	30	35	100	36	90	366
377	Toad	neutral	50	34	58	56	49	28	275
292	Raven	neutral	50	10	29	70	62	40	261
294	Red Hood	neutral	75	12	23	28	24	95	257
222	Man-Bat	neutral	38	18	50	70	33	30	239
367	The Comedian	neutral	63	14	17	10	12	80	196

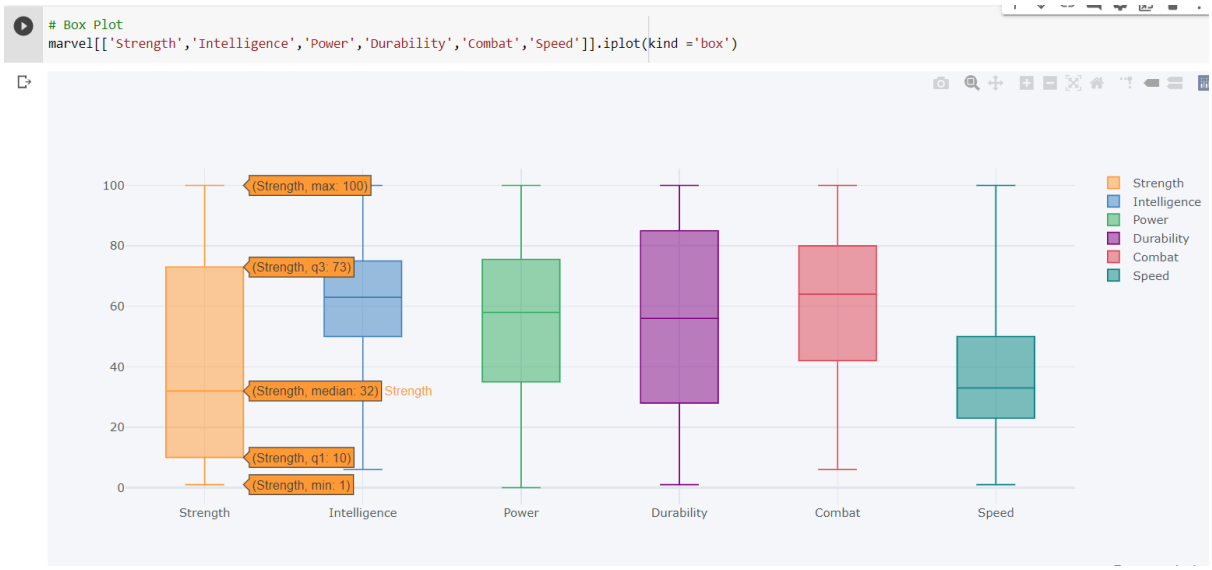
- The pictorial representation of a grouped data, in the form of vertical or horizontal rectangular bars, where the lengths of the bars are equivalent to the measure of data, are known as bar graphs or bar charts.
- The following is the code for Horizontal Bar Chart for neutrals heroes.

```
# Create DataFrame of neutral heroes
df = pd.DataFrame(neutral_max_power)

# taking value of top 12 neutral heroes and their total
name = df['Name']
total = df['Total']

# Figure Size
fig, ax = plt.subplots(figsize =(15, 10))
# Horizontal Bar Plot
ax.barh(name, total)
# Remove axes splines
for s in ['top', 'right']:
    ax.spines[s].set_visible(False)
# Remove x, y Ticks
ax.xaxis.set_ticks_position('none')
ax.yaxis.set_ticks_position('none')
# Add padding between axes and labels
ax.xaxis.set_tick_params(pad = 5)
ax.yaxis.set_tick_params(pad = 10)
# Add x, y gridlines
ax.grid(color ='grey',linestyle ='-.', linewidth = 0.5,)

# Show top values
ax.invert_yaxis()

# Add annotation to bars
for i in ax.patches:
    plt.text(i.get_width()+0.2, i.get_y()+0.5,
             str(round((i.get_width()), 2)),
             fontsize = 12, fontweight ='bold',
             color ='grey')

# Add Plot Title
ax.set_title('Neutral Heroes Anaylysis',loc ='left')
# Show Plot
plt.show()
```


Neutral Heroes Anaylysis
Galactus
Bizarro
Juggernaut
Deadpool
Sandman
Deathstroke
Toad
Raven
Red Hood
Man-Bat
The Comedian
553
550
430
395
387
366
275
261
257
239
196
0
100
200
300
400
500

Box Plot

- Boxplots are a standardized way of displaying the distribution of data based on a five number summary (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”).



- By plotting the box plot of all the characters of dataset we can conclude the min, max, median, q1, q3 of all the characteristic of database.

	Strength	Intelligence	Power	Durability	Combat	Speed
Minimum	1	6	0	1	6	1
First quartile (Q1)	10	50	35	28	42	23
Median	32	63	58	56	64	33
Third quartile (Q3)	73	75	75.5	85	80	50
Maximum	100	100	100	100	100	100

- BATTLE

```
[ ]  # taking input from the user
     sp1=input("Enter character name 1: ")
     sp2=input("Enter character name 2: ")

     # Create list of all Names from DataFrame
     sp=list(marvel['Name'])

     # Loop for Lowercase all Names of the characters
     for i in range(len(sp)):
       sp[i] = sp[i]. lower()

     # Loop for validation of names entered by the user
     if (sp1.lower() in sp) and (sp2.lower() in sp):

       # Extract data of both characters and create a dataframe
       superhero1=marvel[marvel['Name']==sp1.title()]
       df1=pd.DataFrame(superhero1)
       superhero2=marvel[marvel['Name']==sp2.title()]
       df2=pd.DataFrame(superhero2)

       # Merge DataFrames of both the characters
       c=[df1,df2]
       compare=pd.concat(c)

       # Display DataFrame of both the characters
       print('\n')
       display(compare)
       print('\n')

       # Plot Graph of both the characters
       plt.figure(figsize=(4,5))
       plt.bar(list(compare['Name']),list(compare['Total']),color=["black","red"])
       plt.show()
       print('\n')

       # Comparing the value of both the characters for the Battle!!
       sp1_total=superhero1.Total.item()
       sp2_total=superhero2.Total.item()
       if sp1_total>sp2_total:
         print(superhero1.Name.item() , "Won the Battle!!!")
       elif sp2_total>sp1_total:
         print(superhero2.Name.item() , "Won the Battle!!!")
       else:
         print("No one wins the Battle!!!")

     else:
       print("Enter valid details!")
```

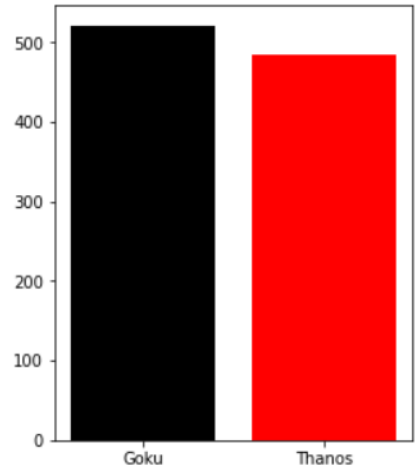
- Here we write a program to a battle between any two characters that the user wants and they can also see the bar plots as well as all the characteristics of their selected two-character

- In the end, they can see who wins the battle between them.

- Here we start the battle between Goku Vs Thanos. By the bar plot and given data describe that Goku has more ability than Thanos. Goku's Total > Thanos Total. So, then Goku wins the battle.

```
Enter character name 1: GoKu
Enter character name 2: ThaNos
```

	Name	Alignment	Intelligence	Strength	Speed	Durability	Power	Combat	Total
147	Goku	good	56	100	75	90	100	100	521
365	Thanos	bad	88	100	17	100	100	80	485




```
Goku Won the Battle!!!
```

Conclusion and future work

- Super Heroes have been in popular culture for a long time and now more than ever. Since its creation, superheroes have not been very diverse, but that is changing rapidly. This fictional hero’s analysis aims to provide an overview of all the characters and their physical as well as superpower characteristics, helping researchers and curious minds identify trends and patterns. Finding out the overall characteristics of all the fictional character's data and from that, we can battle between them and see in the output who wins a battle whether is there a tie between them or not.

- In future work we can add User interface (UI) in this and make this project more interesting. It can also be created like a card game. By using Artificial intelligence (AI) and machine learning (ML) we can take this project into next level.

- by collecting, analysing, and interpreting large volumes of data, in many cases, to improve a company's operations. we can develop statistical models that analyse data and detect patterns, trends, and relationships in data sets. This information can be used to predict behaviour or to identify business and operational risks. We can present data insights to decision-makers in a way that is understandable and applicable to problem-solving.

Learning from the project

- We learned that numpy, seaborn, matplotlib, and pandas forms a solid foundation for a large set of Python packages that provide higher-level functions related to data analytics, machine learning, and AI algorithms. These packages are widely deployed to gain insights that help in data analysis of all the characters in datasets. Finding out the overall characteristics of all the character's data and from that, we can start a battle between them and see in the output who wins a battle whether is there a tie between them or not.

- We also learned that numpy is the first step towards Data Analysis using Python, in which we learned about numpy arrays – creation, methods, and attributes, Basic math with arrays manipulation with arrays and using numpy for simulations. Data Analysis with pandas Series, dataframes & all operations with it. We also learnt about matplotlib which is needed for visualizing data. Also, we applied applications of Higher-level libraries for plotting: seaborn and pandas. However, since both of these libraries are built on top of matplotlib we need to acquire the basic terminology and concepts of matplotlib because frequently we were supposed to make modifications to the objects and plots produced by those higher-level libraries.

- We found that Exploratory data analysis is an approach to analysing data sets to summarize their main characteristics, often with visual methods. It is used to understand the data, get context about it, understand the variables and the relationship between them, and formulate hypothesis that could be useful when building predictive models.