Experiment 1.2

Student Name: Pushpraj Roy
UID: 20BCS9866
Branch: CSE
Section/Group: WM_617-A
Semester: 5th
Subject Code: 20CSP-317
Subject Name: Machine Learning Lab

Aim: Performing Data visualization on any dataset.

Dataset taken: - Pokemon dataset

Code for uploading the dataset:

```
import pandas as pd
data = pd.read_csv("pokemon.csv")
data.head()
```

Out[2]:

	id	species	generation_id	height	weight	base_experience	type_1	type_2	hp	attack	defense	speed	special-attack	special-defense
0	1	bulbasaur	1	0.7	6.9	64	grass	poison	45	49	49	45	65	65
1	2	ivysaur	1	1.0	13.0	142	grass	poison	60	62	63	60	80	80
2	3	venusaur	1	2.0	100.0	236	grass	poison	80	82	83	80	100	100
3	4	charmander	1	0.6	8.5	62	fire	NaN	39	52	43	65	60	50
4	5	charmeleon	1	1.1	19.0	142	fire	NaN	58	64	58	80	80	65

```
data.tail()
```

```
In [3]: data.tail()
```

Out[3]:

	id	species	generation_id	height	weight	base_experience	type_1	type_2	hp	attack	defense	speed	special-attack	special-defense
802	803	poipole	7	0.6	1.8	189	poison	NaN	67	73	67	73	73	67
803	804	naganadel	7	3.6	150.0	243	poison	dragon	73	73	73	121	127	73
804	805	stakataka	7	5.5	820.0	257	rock	steel	61	131	211	13	53	101
805	806	blacephalon	7	1.8	13.0	257	fire	ghost	53	127	53	107	151	79
806	807	zeraora	7	1.5	44.5	270	electric	NaN	88	112	75	143	102	80

Libraries

Python provides various libraries that come with different features for visualizing data. All these libraries comes with different features and can support various types of graphs. The four main libraries are:

1. matplotlib
2. Seaborn
3. Bokeh
4. Plotly

Matplotlib

It is an easy to use low level data visualization library that is build on numpy arrays it consist of various plots like scatter plot, line graph/plot, histogram etc. It provides lots of flexibility

pip install matplotlib

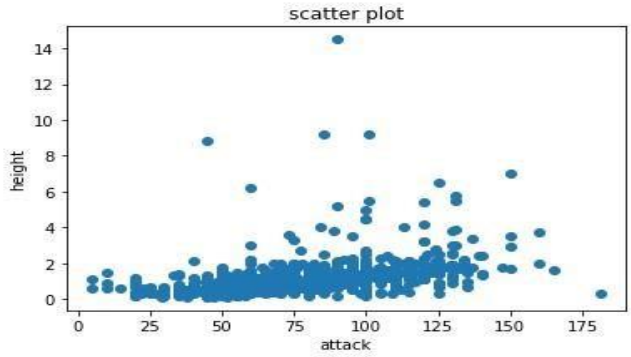
import matplotlib.pyplot as plt

Data Visualization on the dataset:

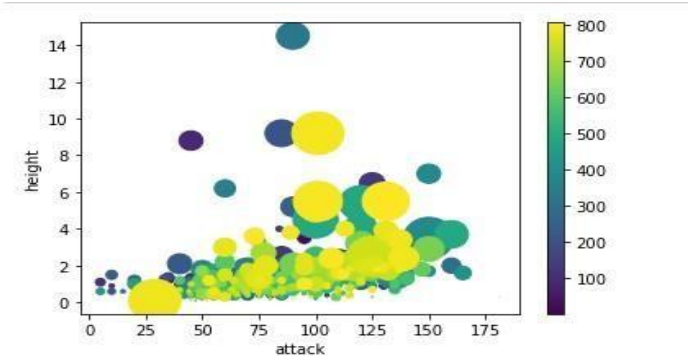
Matplotlib

1. Scatter Plot

```
plt.scatter(data['attack'], data['height'])
plt.title('scatter plot')
plt.xlabel('attack')
plt.ylabel('height')
plt.show()
```

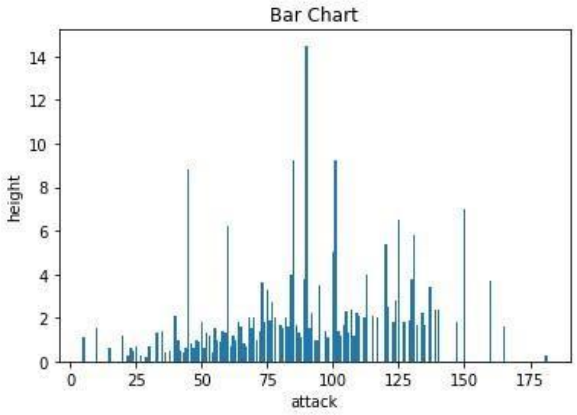



```
plt.scatter(data['attack'], data['height'], c = data['id'],
s= data['weight']) plt.xlabel('attack')
plt.ylabel('height')

plt.colorbar()
plt.show()
```



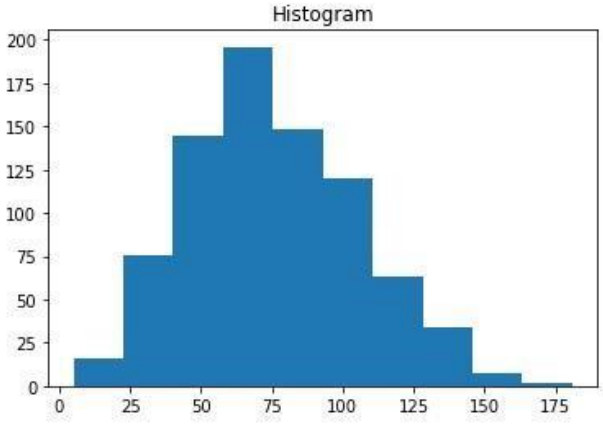
2. Bar Chart

```
plt.bar(data['attack'], data['height'])
plt.title("Bar Chart")
plt.xlabel('attack')
plt.ylabel('height')
plt.show()
```



3. Histogram

```
plt.hist(data['attack'])

plt.title("Histogram")

plt.show()
```



Seaborn

Seaborn

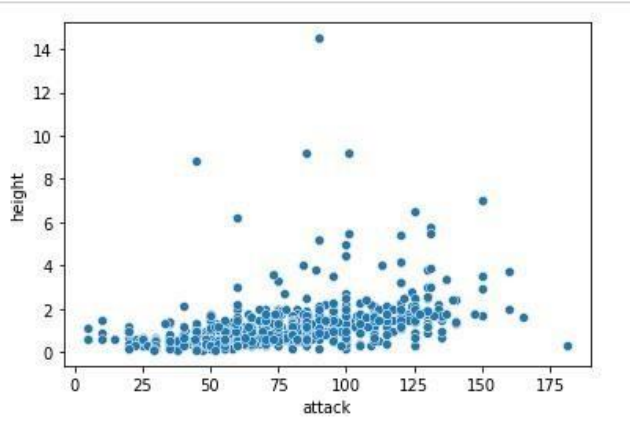
It is a high level interface build on top of the matplotlib. It provides beautiful design styles and color palettes to make more attractive graphs

Scatter plot

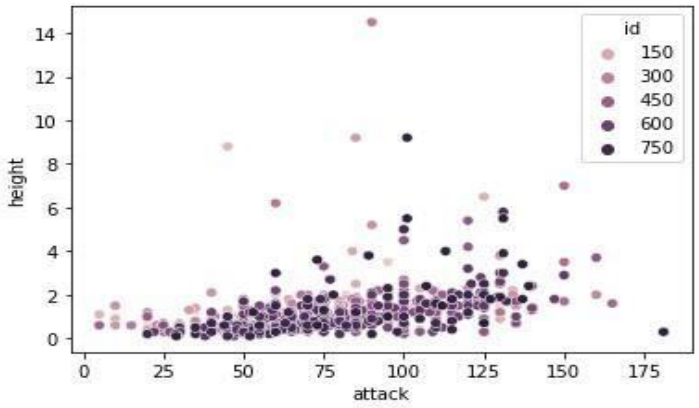
it is plotted using the scatterplot() method. This is similar to matplotlib, but additional argument data is required

1. Scatter Plot import seaborn as sns import matplotlib.pyplot as plt import pandas as pd

```
sns.scatterplot(x='attack', y='height', data= data) plt.show()
```



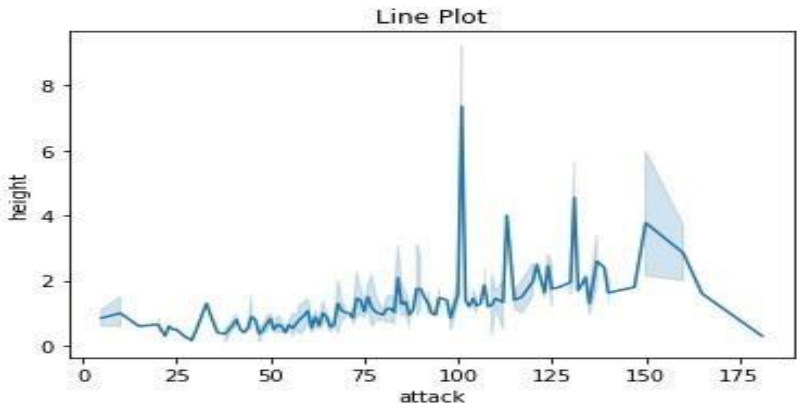

```
sns.scatterplot(x='attack', y='height', data= data, hue='id')
plt.show()
```



2. Line Plot

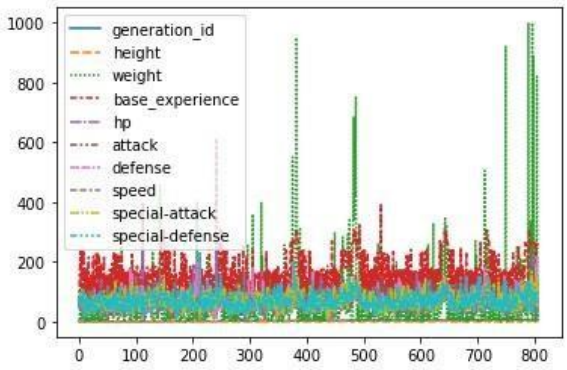
sns.lineplot(x='attack', y='height', data=

data) plt.title("Line Plot") plt.show()



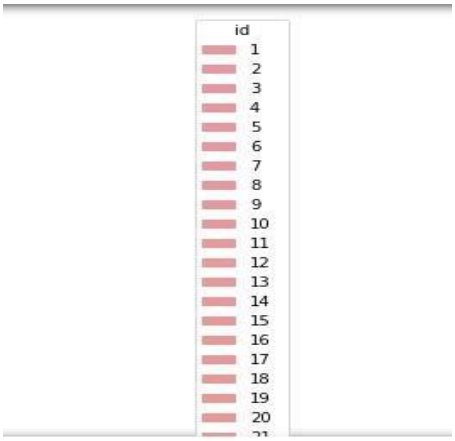
sns.lineplot(data=data.drop(['id'],axis=1))

```
Out[15]: <AxesSubplot:>
```



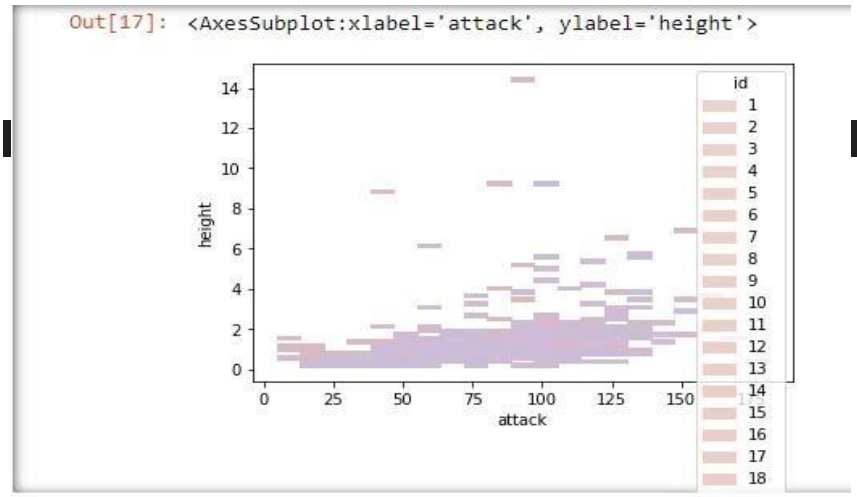
3. Bar Plot

```
sns.barplot(x='attack', y='height', data=data, hue= 'id')
plt.show()
```



4. Histogram

```
sns.histplot(x='attack', y='height', data=data, kde = True ,hue='id')
```



Plotly

pip install plotly.express import
plotly.express as px
import pandas as pd

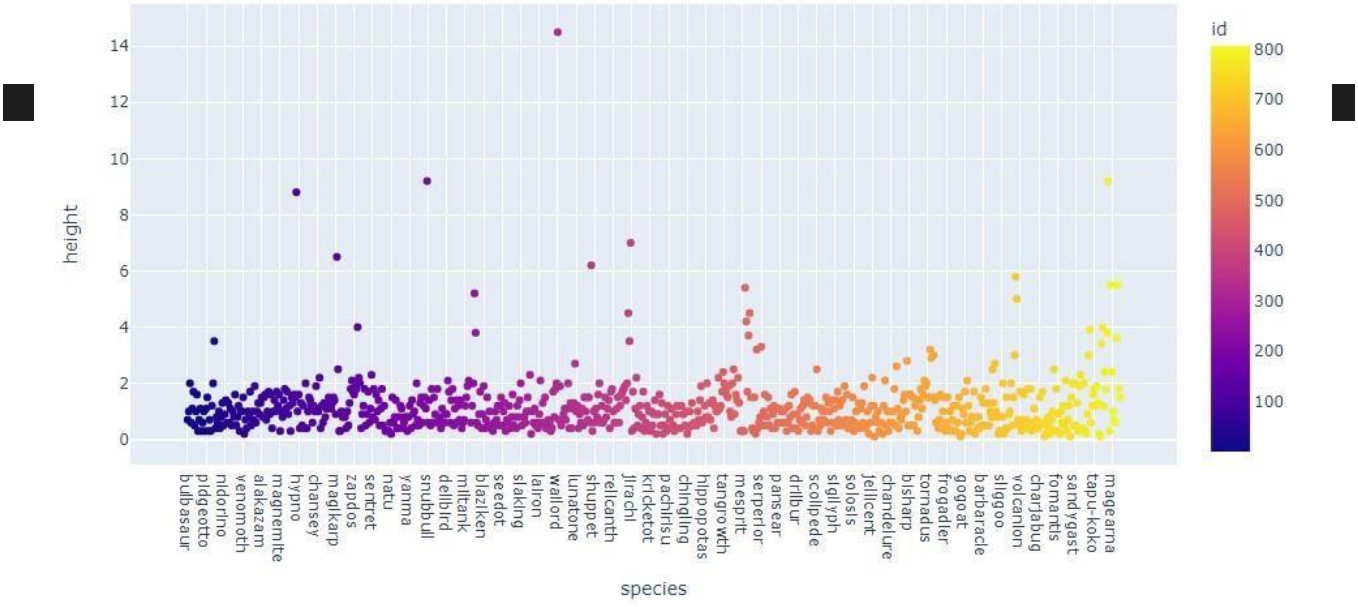
Plotly

Plotly has hover tool capabilities that allow us to detect any outliers or anomalies in numerous data points. It allows more customization. It makes the graph visually more attractive.

Scatterplot

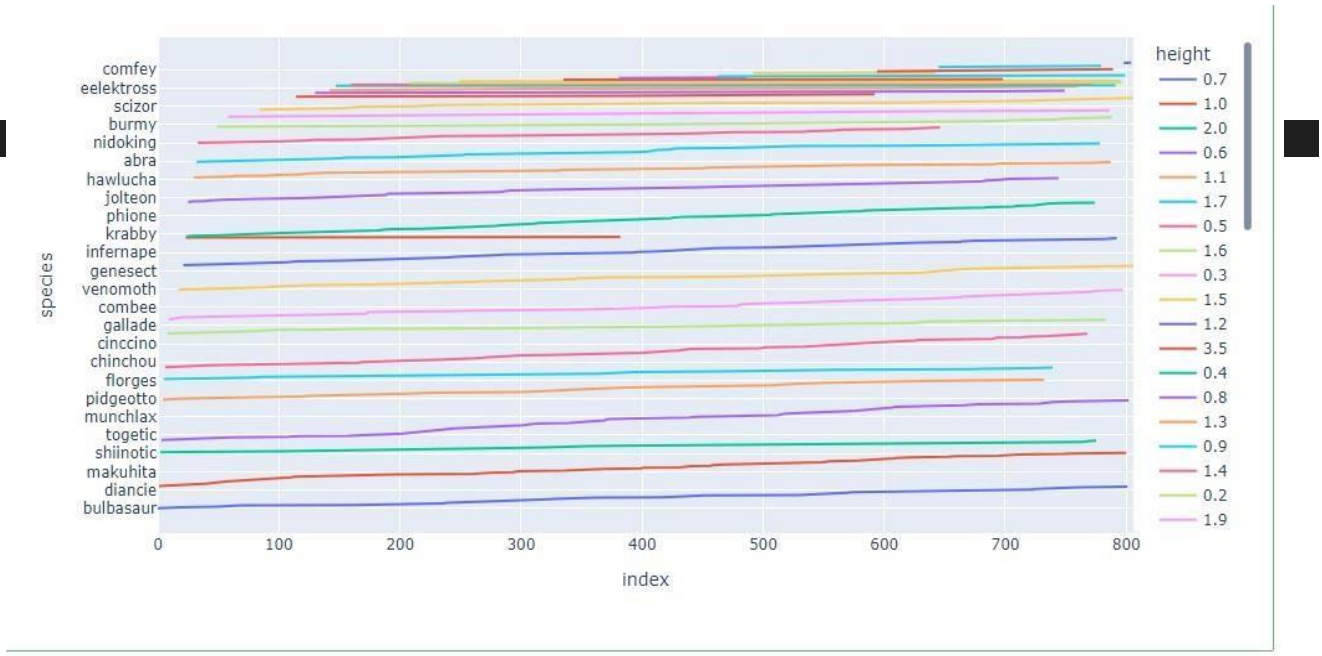
in plotily can be created using the scatter() method of plotly.express. like seaborn, an extra data argument is also required here.

1. ScatterPlot fig= px.scatter(data, x= "species", y= "height", color="id")
fig.show()



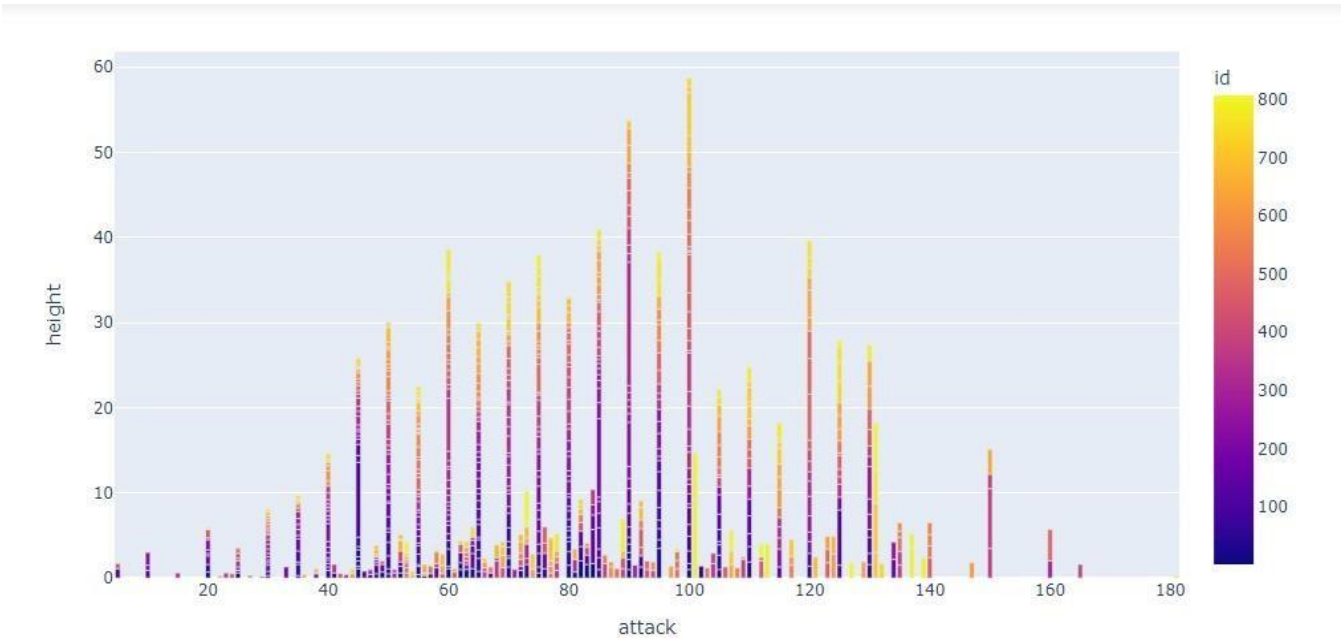
2. Line Chart

```
fig= px.line(data,y='species', color='height')

# Showing the plot

fig.show()
```



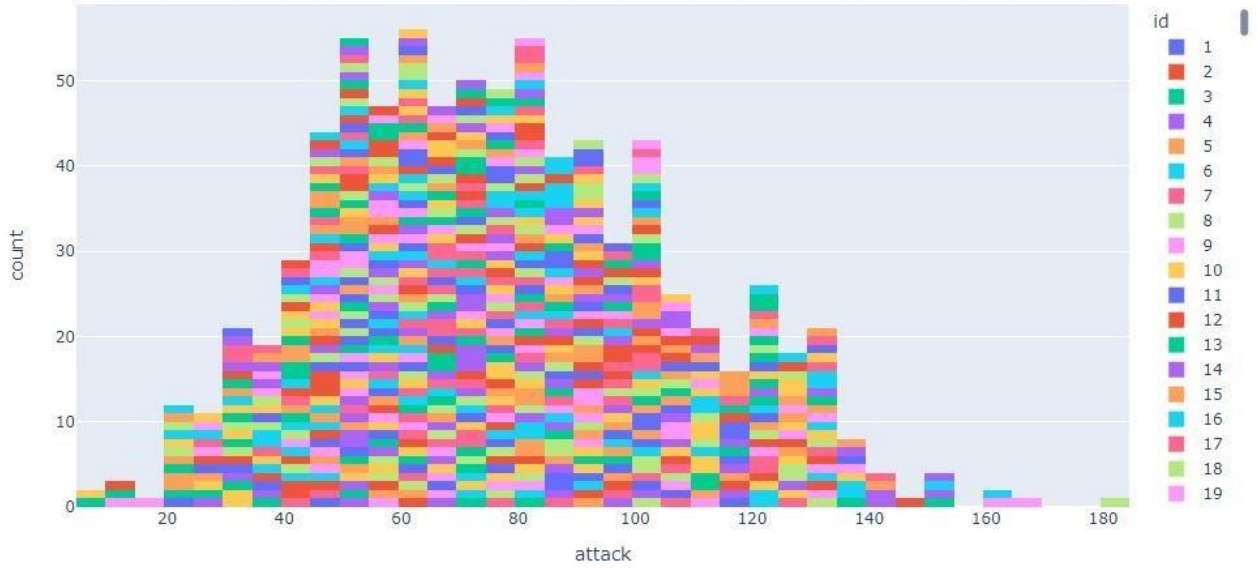
3. Bar Chart

```
fig = px.bar(data, x='attack', y='height', color= 'id')

fig.show()
```



4. Histogram

```
fig = px.histogram(data, x='attack', color= 'id')
fig.show()
```



Bokeh

Bokeh library

KDE

KDE Plot described as Kernel Density Estimate is used for visualizing the Probability Density of a continuous variable. It depicts the probability density at different values in a continuous variable.

Bokeh

Bokeh is mainly famous for its interactive charts visualization. Bokeh renders its plots using HTML and JavaScript that uses modern web browsers for presenting elegant, concise construction of novel graphics with high-level interactivity.

Scatter Plot

Scatter Plot in Bokeh can be plotted using the scatter() method of the plotting module. Here pass the x and y coordinates respectively.

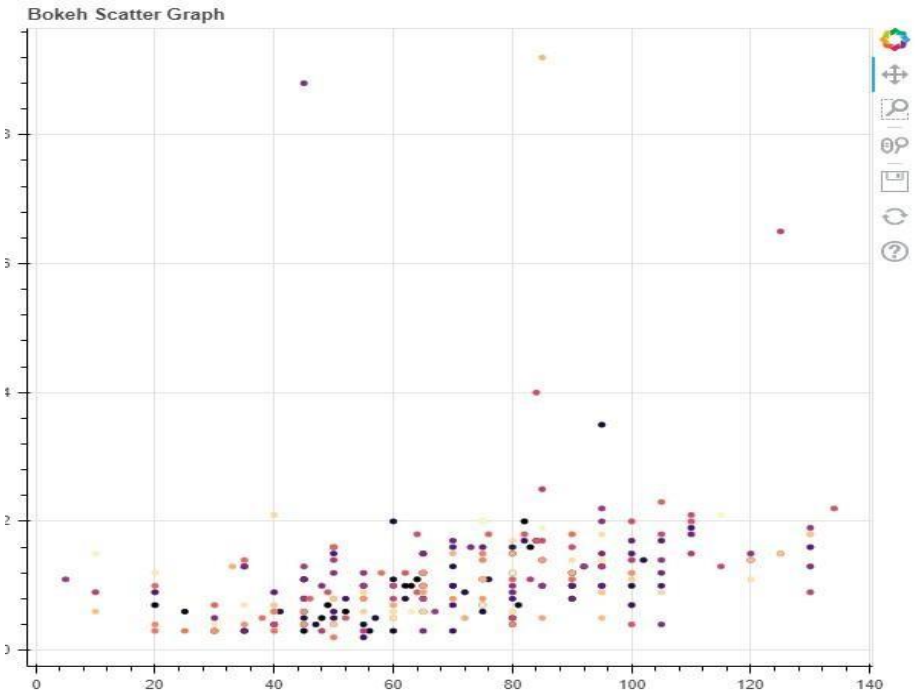
pip install bokeh from bokeh.plotting import figure,

output_file,show from bokeh.palettes import magma

import pandas as pd

1. Scatter Plot graph = figure(title = "Bokeh Scatter Graph") color = magma(245)

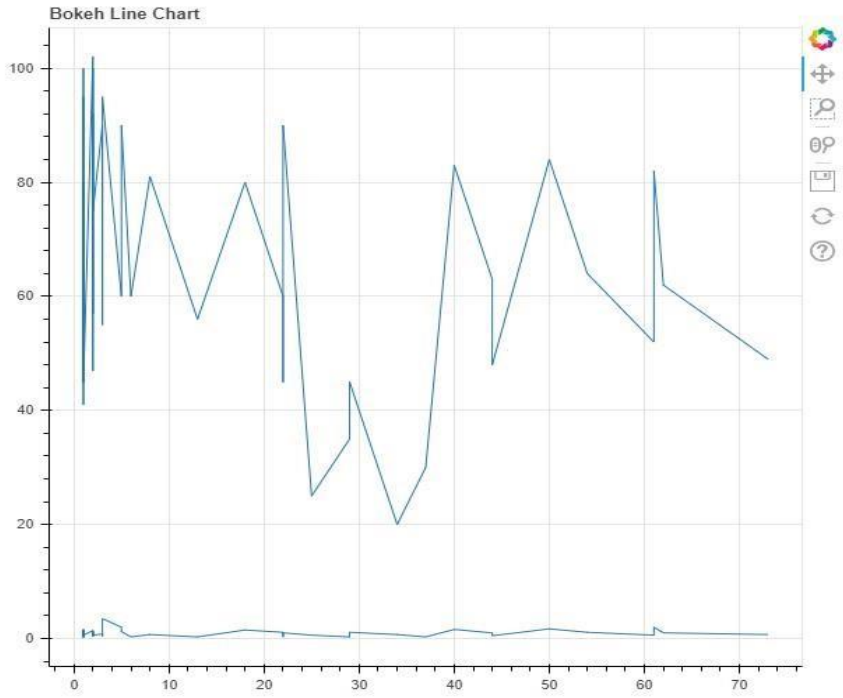
```
graph.scatter(data['attack'], data['height'], color=color)
show(graph)
```



2. Line Chart

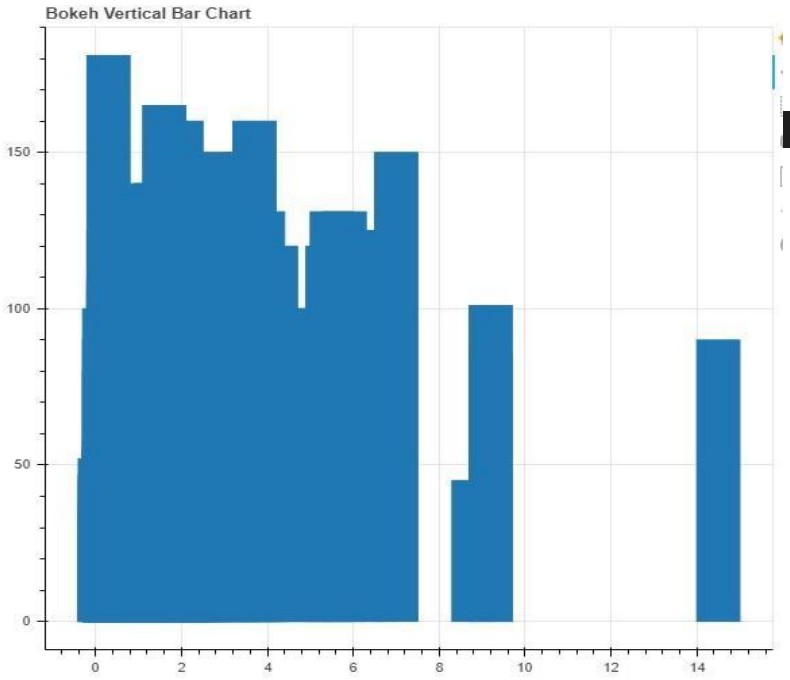
```
# instantiating the figure object graph = figure(title

= "Bokeh Line Chart")

# Count of each unique value of  #

attack column df =

data['attack'].value_counts() df =
```

```
data['height'].value_counts() #
plotting the graph graph.line(df,
data['attack']) graph.line(df,
data['height']) # displaying the
model show(graph)
```



3. Bar Chart

```
#vertical bar graph = figure(title = "Bokeh
Vertical Bar Chart") graph.vbar(data['height'],
top=data['attack']) show(graph)
```



Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			