Saad Najam

Assignment #2

In [2]:
```
import pandas as  pd
```

In [3]:
```
data = pd.read_csv("C:\\Users\\User\\Desktop\\python_Assignments\\kaggleFolder\\Pokemon.csv
```

In [5]:
```
data.head(n=10)
```

Out[5]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legen
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	
5	5	Charmeleon	Fire	NaN	405	58	64	58	80	65	80	1	
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	
7	6	CharizardMega Charizard X	Fire	Dragon	634	78	130	111	130	85	100	1	
8	6	CharizardMega Charizard Y	Fire	Flying	634	78	104	78	159	115	100	1	
9	7	Squirtle	Water	NaN	314	44	48	65	50	64	43	1	

In [6]:
```
data.tail(n=10)
```

Out[6]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
790	714	Noibat	Flying	Dragon	245	40	30	35	45	40	55	6	
791	715	Noivern	Flying	Dragon	535	85	70	80	97	80	123	6	
792	716	Xerneas	Fairy	NaN	680	126	131	95	131	98	99	6	
793	717	Yveltal	Dark	Flying	680	126	131	95	131	98	99	6	

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
794	718	Zygarde50% Forme	Dragon	Ground	600	108	100	121	81	95	95	6	
795	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	6	
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	6	
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	6	
798	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160	60	170	130	80	6	
799	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 800 entries, 0 to 799
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   #           800 non-null    int64 
 1   Name        800 non-null    object
 2   Type 1      800 non-null    object
 3   Type 2      414 non-null    object
 4   Total       800 non-null    int64 
 5   HP          800 non-null    int64 
 6   Attack      800 non-null    int64 
 7   Defense     800 non-null    int64 
 8   Sp. Atk     800 non-null    int64 
 9   Sp. Def     800 non-null    int64 
 10  Speed       800 non-null    int64 
 11  Generation  800 non-null    int64 
 12  Legendary   800 non-null    bool  
dtypes: bool(1), int64(9), object(3)
memory usage: 75.9+ KB
```

In [8]:
```
data.describe()
```

Out[8]:

	#	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed
count	800.000000	800.00000	800.000000	800.000000	800.000000	800.000000	800.000000	800.000000
mean	362.813750	435.10250	69.258750	79.001250	73.842500	72.820000	71.902500	68.277500
std	208.343798	119.96304	25.534669	32.457366	31.183501	32.722294	27.828916	29.060474
min	1.000000	180.00000	1.000000	5.000000	5.000000	10.000000	20.000000	5.000000
25%	184.750000	330.00000	50.000000	55.000000	50.000000	49.750000	50.000000	45.000000

	#	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed
50%	364.500000	450.00000	65.000000	75.000000	70.000000	65.000000	70.000000	65.000000
75%	539.250000	515.00000	80.000000	100.000000	90.000000	95.000000	90.000000	90.000000
max	721.000000	780.00000	255.000000	190.000000	230.000000	194.000000	230.000000	180.000000

In [9]:
```
data.isnull().sum()
```

Out[9]:
```
#             0
Name          0
Type 1        0
Type 2      386
Total         0
HP            0
Attack        0
Defense       0
Sp. Atk       0
Sp. Def       0
Speed         0
Generation    0
Legendary     0
dtype: int64
```

In [10]:
```
data.notnull().sum()
```

Out[10]:
```
#           800
Name        800
Type 1      800
Type 2      414
Total       800
HP          800
Attack      800
Defense     800
Sp. Atk     800
Sp. Def     800
Speed       800
Generation  800
Legendary   800
dtype: int64
```

In [11]:
```
data["Type 1"].value_counts()
```

Out[11]:
```
Water       112
Normal       98
Grass        70
Bug          69
Psychic      57
Fire         52
Electric     44
Rock         44
Dragon       32
Ghost        32
Ground       32
```

```
Dark        31
Poison      28
Steel       27
Fighting    27
Ice         24
Fairy       17
Flying       4
Name: Type  1, dtype: int64
```

In [12]:
```
data.sort_values("Defense")
```

Out[12]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
121	113	Chansey	Normal	NaN	450	250	5	5	35	105	50	1	
488	440	Happiny	Normal	NaN	220	100	5	5	15	65	30	4	
261	242	Blissey	Normal	NaN	540	255	10	10	75	135	55	2	
186	172	Pichu	Electric	NaN	205	20	40	15	35	35	60	2	
188	174	Igglybuff	Normal	Fairy	210	90	30	15	40	20	15	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
414	377	Regirock	Rock	NaN	580	80	100	200	50	100	50	3	
223	208	Steelix	Steel	Ground	510	75	85	200	55	65	30	2	
333	306	AggronMega Aggron	Steel	NaN	630	70	140	230	60	80	50	3	
230	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
224	208	SteelixMega Steelix	Steel	Ground	610	75	125	230	55	95	30	2	

800 rows × 13 columns

In [13]:
```
data.sort_values(["Defense","Attack","HP"])
```

Out[13]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
488	440	Happiny	Normal	NaN	220	100	5	5	15	65	30	4	
121	113	Chansey	Normal	NaN	450	250	5	5	35	105	50	1	
261	242	Blissey	Normal	NaN	540	255	10	10	75	135	55	2	
68	63	Abra	Psychic	NaN	310	25	20	15	105	55	90	1	
257	238	Smoochum	Ice	Psychic	305	45	30	15	85	65	65	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
223	208	Steelix	Steel	Ground	510	75	85	200	55	65	30	2	
414	377	Regirock	Rock	NaN	580	80	100	200	50	100	50	3	
230	213	Shuckle	Bug	Rock	505	20	10	230	10	230	5	2	
224	208	SteelixMega Steelix	Steel	Ground	610	75	125	230	55	95	30	2	
333	306	AggronMega Aggron	Steel	NaN	630	70	140	230	60	80	50	3	

800 rows × 13 columns

In [14]:
```
attack_mean = data["Attack"].mean()
```

In [18]:
```
def set_attack(value):
    if(value > attack_mean ):
        return "High_Attack"
    elif(value == attack_mean):
        return  "Attact_Neutral"
    else:
        return "Low_Attack"
```

In [19]:
```
data["Attack_high_low"] = data["Attack"].apply(set_attack)
data
```

Out[19]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
795	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	6	
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	6	
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	6	

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	L
798	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160	60	170	130	80	6	
799	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	

800 rows × 14 columns

Find Speed_low and Speed_High

In [25]:
```
Speed_mean = data["Speed"].mean()
```

In [26]:
```
def Check_speed(value):
    if(value > Speed_mean):
        return "High_Speed"
    elif(value == Speed_mean):
        return "Neutral_Speed"
    else:
        return "Low_Speed"
```

In [27]:
```
data["Speed_High_Low"] = data["Speed"].apply(Check_speed)
```

In [28]:
```
data.head(n=10)
```

Out[28]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legen
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	
5	5	Charmeleon	Fire	NaN	405	58	64	58	80	65	80	1	
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	
7	6	CharizardMega Charizard X	Fire	Dragon	634	78	130	111	130	85	100	1	
8	6	CharizardMega Charizard Y	Fire	Flying	634	78	104	78	159	115	100	1	
9	7	Squirtle	Water	NaN	314	44	48	65	50	64	43	1	

Find low_HP And High_HP

In [29]:
```
HP_mean = data["HP"].mean()
```

In [30]:
```
def Check_HP(value):
    if(value > HP_mean):
        return "High_HP"
    elif(value == HP_mean):
        return "Neutral_HP"
    else:
        return "Low_HP"
```

In [31]:
```
data["HP_High_Low"] = data["HP"].apply(Check_HP)
```

In [32]:
```
data.head(n=10)
```

Out[32]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legen
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	
5	5	Charmeleon	Fire	NaN	405	58	64	58	80	65	80	1	
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	
7	6	CharizardMega Charizard X	Fire	Dragon	634	78	130	111	130	85	100	1	
8	6	CharizardMega Charizard Y	Fire	Flying	634	78	104	78	159	115	100	1	
9	7	Squirtle	Water	NaN	314	44	48	65	50	64	43	1	

In [ ]: