In [4]:
```
import pandas as pd
```

In [8]:
```
data = pd.read_csv("Pokemon.csv")
data.head(n=10)
```

Out[8]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defer
0	1	Bulbasaur	Grass	Poison	318	45	49	
1	2	Ivysaur	Grass	Poison	405	60	62	
2	3	Venusaur	Grass	Poison	525	80	82	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	1
4	4	Charmander	Fire	NaN	309	39	52	
5	5	Charmeleon	Fire	NaN	405	58	64	
6	6	Charizard	Fire	Flying	534	78	84	
7	6	CharizardMega Charizard X	Fire	Dragon	634	78	130	1
8	6	CharizardMega Charizard Y	Fire	Flying	634	78	104	
9	7	Squirtle	Water	NaN	314	44	48	

In [10]:
```
data.tail(n=15)
```

Out[10]:

	#	Name	Type 1	Type 2	Total	HP	Attack
785	711	GourgeistSmall Size	Ghost	Grass	494	55	85
786	711	GourgeistLarge Size	Ghost	Grass	494	75	95
787	711	GourgeistSuper Size	Ghost	Grass	494	85	100
788	712	Bergmite	Ice	NaN	304	55	69
789	713	Avalugg	Ice	NaN	514	95	117
790	714	Noibat	Flying	Dragon	245	40	30
791	715	Noivern	Flying	Dragon	535	85	70
792	716	Xerneas	Fairy	NaN	680	126	131
793	717	Yveltal	Dark	Flying	680	126	131
794	718	Zygarde50% Forme	Dragon	Ground	600	108	100

		Forme					
795	719	Diancie	Rock	Fairy	600	50	100
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110
798	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160
799	721	Volcanion	Fire	Water	600	80	110

In [11]:
```
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 [49]:
```
data.isnull().sum()
```

Out[49]:
```
#                   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
Attack_high_low     0
dtype: int64
```

In [ ]:
```

```

In [12]:
```
data.describe()
```

Out[12]:

	#	Total	HP	Attack	Defer
count	800.000000	800.00000	800.000000	800.000000	800.0000
mean	362.813750	435.10250	69.258750	79.001250	73.8425
std	208.343798	119.96304	25.534669	32.457366	31.1835
min	1.000000	180.00000	1.000000	5.000000	5.0000
25%	184.750000	330.00000	50.000000	55.000000	50.0000
50%	364.500000	450.00000	65.000000	75.000000	70.0000
75%	539.250000	515.00000	80.000000	100.000000	90.0000
max	721.000000	780.00000	255.000000	190.000000	230.0000

In [13]:
```
data.head()
```

Out[13]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defens
0	1	Bulbasaur	Grass	Poison	318	45	49	4
1	2	Ivysaur	Grass	Poison	405	60	62	6
2	3	Venusaur	Grass	Poison	525	80	82	8
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	12
4	4	Charmander	Fire	NaN	309	39	52	4

In [16]:
```
data["Type 1"].value_counts()
```

Out[16]:
```
Water       112
Normal       98
Grass        70
Bug          69
Psychic      57
Fire         52
Electric     44
Rock         44
Dragon       32
Ground       32
Ghost        32
Dark         31
Poison       28
Steel        27
Fighting     27
Ice          24
Fairy        17
Flying        4
Name: Type 1, dtype: int64
```

```
Name: Type 1, dtype: int64
```

In [19]:
```
data.sort_values("Defense",ascending=False)
```

Out[19]:

	#	Name	Type 1	Type 2	Total	HP	Attack
230	213	Shuckle	Bug	Rock	505	20	10
224	208	SteelixMega Steelix	Steel	Ground	610	75	125
333	306	AggronMega Aggron	Steel	NaN	630	70	140
223	208	Steelix	Steel	Ground	510	75	85
414	377	Regirock	Rock	NaN	580	80	100
...	...	...	...	...	...	...	...
68	63	Abra	Psychic	NaN	310	25	20
186	172	Pichu	Electric	NaN	205	20	40
261	242	Blissey	Normal	NaN	540	255	10
488	440	Happiny	Normal	NaN	220	100	5
121	113	Chansey	Normal	NaN	450	250	5

800 rows × 13 columns

In [24]:
```
data.sort_values(["HP","Attack","Defense"])
```

Out[24]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Def
316	292	Shedinja	Bug	Ghost	236	1	90	
55	50	Diglett	Ground	NaN	265	10	55	
139	129	Magikarp	Water	NaN	200	20	10	
230	213	Shuckle	Bug	Rock	505	20	10	
381	349	Feebas	Water	NaN	200	20	15	
...	...	...	...	...	...	...	...	
655	594	Alomomola	Water	NaN	470	165	75	
351	321	Wailord	Water	NaN	500	170	90	
217	202	Wobbuffet	Psychic	NaN	405	190	33	
121	113	Chansey	Normal	NaN	450	250	5	
261	242	Blissey	Normal	NaN	540	255	10	

800 rows × 13 columns

Feature Engineering

Add column

In [30]:
```
attack_mean = data["Attack"].mean()

def set_attack(val):
    if val < attack_mean:
        return "Attack Low"
    elif val == attack_mean:
        return "Attack neutral"
    else:
        return "Attack High"
```

In [31]:
```
# "Attack_high_low"  -> "Attack High" , "Attack Low"
data["Attack_high_low"] = data["Attack"].apply(set_a
data
```

Out[31]:

	#	Name	Type 1	Type 2	Total	HP	Attack	I
0	1	Bulbasaur	Grass	Poison	318	45	49	
1	2	Ivysaur	Grass	Poison	405	60	62	
2	3	Venusaur	Grass	Poison	525	80	82	
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	
4	4	Charmander	Fire	NaN	309	39	52	
...	...	...	...	...	...	...	...	
795	719	Diancie	Rock	Fairy	600	50	100	
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160	
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	
798	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160	
799	721	Volcanion	Fire	Water	600	80	110	

800 rows × 14 columns

Column of Speed_high_low and HP_high_low

GroupBy

In [45]:
```
data.groupby("Type 1").groups
```

Out[45]:
```
{'Bug': [13, 14, 15, 16, 17, 18, 19, 51, 52, 53, 54,
132, 136, 137, 179, 180, 181, 182, 208, 219, 220, 22
8, 229, 230, 231, 232, 288, 289, 290, 291, 292, 307,
308, 314, 315, 316, 342, 343, 446, 447, 457, 458, 45
9, 460, 461, 462, 463, 520, 600, 601, 602, 603, 604,
605, 618, 619, 649, 650, 656, 657, 677, 678, 693, 69
7, 698, 717, 732, 733, 734], 'Dark': [212, 213, 233,
246, 247, 248, 284, 285, 326, 327, 392, 393, 478, 51
2, 549, 568, 569, 620, 621, 631, 632, 685, 686, 690,
691, 694, 695, 696, 756, 757, 793], 'Dragon': [159,
160, 161, 365, 366, 406, 407, 408, 409, 417, 418, 41
9, 420, 425, 426, 491, 492, 493, 494, 671, 672, 673,
682, 706, 707, 710, 711, 712, 774, 775, 776, 794],
'Electric': [30, 31, 88, 89, 108, 109, 134, 146, 15
7, 186, 193, 194, 195, 196, 258, 262, 337, 338, 339,
340, 341, 448, 449, 450, 464, 513, 517, 531, 532, 53
3, 534, 535, 536, 581, 582, 648, 663, 664, 665, 704,
705, 764, 765, 772], 'Fairy': [40, 41, 187, 189, 19
0, 225, 226, 519, 737, 738, 739, 752, 753, 754, 755,
770, 792], 'Fighting': [61, 62, 72, 73, 74, 114, 11
5, 255, 256, 320, 321, 334, 335, 336, 496, 497, 498,
592, 593, 594, 598, 599, 680, 681, 742, 743, 771],
'Fire': [4, 5, 6, 7, 8, 42, 43, 63, 64, 83, 84, 135,
147, 158, 169, 170, 171, 236, 237, 259, 263, 270, 27
6, 277, 278, 279, 352, 353, 354, 355, 435, 436, 437,
518, 542, 557, 558, 559, 572, 573, 614, 615, 616, 69
2, 721, 722, 723, 730, 731, 735, 736, 799], 'Flyin
g': [702, 703, 790, 791], 'Ghost': [99, 100, 101, 10
```