

学号：202300130160	姓名：刘逸宁	班级：23 级数据班
实验题目：二、数据质量实践		
实验学时：2	实验日期：2025.9.23	
<p>实验目的：</p> <p>本次实验主要围绕宝可梦数据集进行分析，考察在拿到数据后如何对现有的数据进行预处理清洗操作，建立起对于脏数据、缺失数据等异常情况的一套完整流程的认识。</p>		
<p>硬件环境：</p> <p>计算机一台</p>		
<p>软件环境：</p> <p>Windows</p>		
<p>实验步骤与内容：</p> <p>步骤 1：读取数据并查看原始状态</p> <div><pre>原始数据形状（行数，列数）：（810，13） 前5行数据（确认列名）： # Name Type 1 Type 2 ... Sp. Def Speed Generation Legendary 0 1 Bulbasaur Grass Poison ... 65 45 1 FALSE 1 2 Ivysaur Grass Poison ... 80 60 1 FALSE 2 3 Venusaur Grass Poison ... 100 80 1 FALSE 3 3 VenusaurMega Venusaur Grass Poison ... 120 80 1 FALSE 4 4 Charmander Fire NaN ... 50 65 1 FALSE [5 rows x 13 columns] 最后10行数据（观察无意义尾行）： # Name Type 1 ... Speed Generation Legendary 800 718 Zygarde50% Forme Dragon ... 95 6 TRUE 801 719 Diancie Rock ... 50 6 TRUE 802 719 DiancieMega Diancie Rock ... 110 6 TRUE 803 720 HoopaHoopa Confined Psychic ... 70 6 TRUE 804 720 HoopaHoopa Unbound Psychic ... 80 6 TRUE 805 721 Volcanion Fire ... 70 6 TRUE 806 undefined undefined undefined ... undefined undefined undefined 807 undefined undefined undefined ... undefined undefined undefined 808 NaN NaN NaN ... NaN NaN 809 NaN NaN NaN ... NaN NaN [10 rows x 13 columns] Type 2列唯一值（查看异常值'273'）： ['Poison' nan 'Flying' 'Dragon' '0' 'Ground' '273' 'Fairy' 'Grass' 'Fighting' 'Psychic' 'Steel' 'Ice' 'A' 'Rock' 'Dark' 'Water' 'Electric' 'Fire' 'Ghost' 'Bug' 'BBB' 'Normal' 'undefined'] 重复值数量： 7</pre></div>		

步骤 2：处理数据集问题

(1) 删除最后两行无意义数据：

删除无意义尾行后形状： (808, 13)

删除后末尾5行：

	#	Name	Type 1	...	Speed	Generation	Legendary
803	720	HoopaHoopa Confined	Psychic	...	70	6	TRUE
804	720	HoopaHoopa Unbound	Psychic	...	80	6	TRUE
805	721	Volcanion	Fire	...	70	6	TRUE
806	undefined	undefined	undefined	...	undefined	undefined	undefined
807	undefined	undefined	undefined	...	undefined	undefined	undefined

(2) 清空 Type 2 列的异常值 “273”

[5 rows x 13 columns]

Type 2列处理后唯一值：

```
['Poison' nan 'Flying' 'Dragon' '0' 'Ground' <NA> 'Fairy' 'Grass'
 'Fighting' 'Psychic' 'Steel' 'Ice' 'A' 'Rock' 'Dark' 'Water' 'Electric'
 'Fire' 'Ghost' 'Bug' 'BBB' 'Normal' 'undefined']
```

(3) 删除重复值，“数据集中存在重复值”，用 drop_duplicates() 去重：

[5 rows x 13 columns]

Type 2列处理后唯一值：

```
['Poison' nan 'Flying' 'Dragon' '0' 'Ground' <NA> 'Fairy' 'Grass'
 'Fighting' 'Psychic' 'Steel' 'Ice' 'A' 'Rock' 'Dark' 'Water' 'Electric'
 'Fire' 'Ghost' 'Bug' 'BBB' 'Normal' 'undefined']
```

去重后重复值数量： 0

去重后数据形状： (802, 13)

(4) 处理 Attack 属性过高异常值

处理Attack异常值后数据形状： (791, 13)

Attack列处理后范围： 5.0 ~ 165.0

(5) 修正 “Generation 与 Legendary 属性置换” 问题

置换属性的行：

	Generation	Legendary
39	FALSE	NaN
533	4	NaN

修正后置换行的属性：

	Generation	Legendary
39	NaN	FALSE
533	NaN	4

(6) 最终数据统计：

最终数据描述性统计（Attack等属性无极端值）：

```
Attack
count    791.000000
mean     78.104930
std      31.220131
min       5.000000
25%      55.000000
50%      75.000000
75%     100.000000
max     165.000000
```

实验指导书代码结果：

1、数据导入：

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	FALSE
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	FALSE
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	FALSE
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	FALSE
4	4	Charmander	Fire	NaN	309	39	52	43	60	50	65	1	FALSE
...
805	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	TRUE
806	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined
807	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined	undefined
808	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
809	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

810 rows × 13 columns

2、删除多余空行

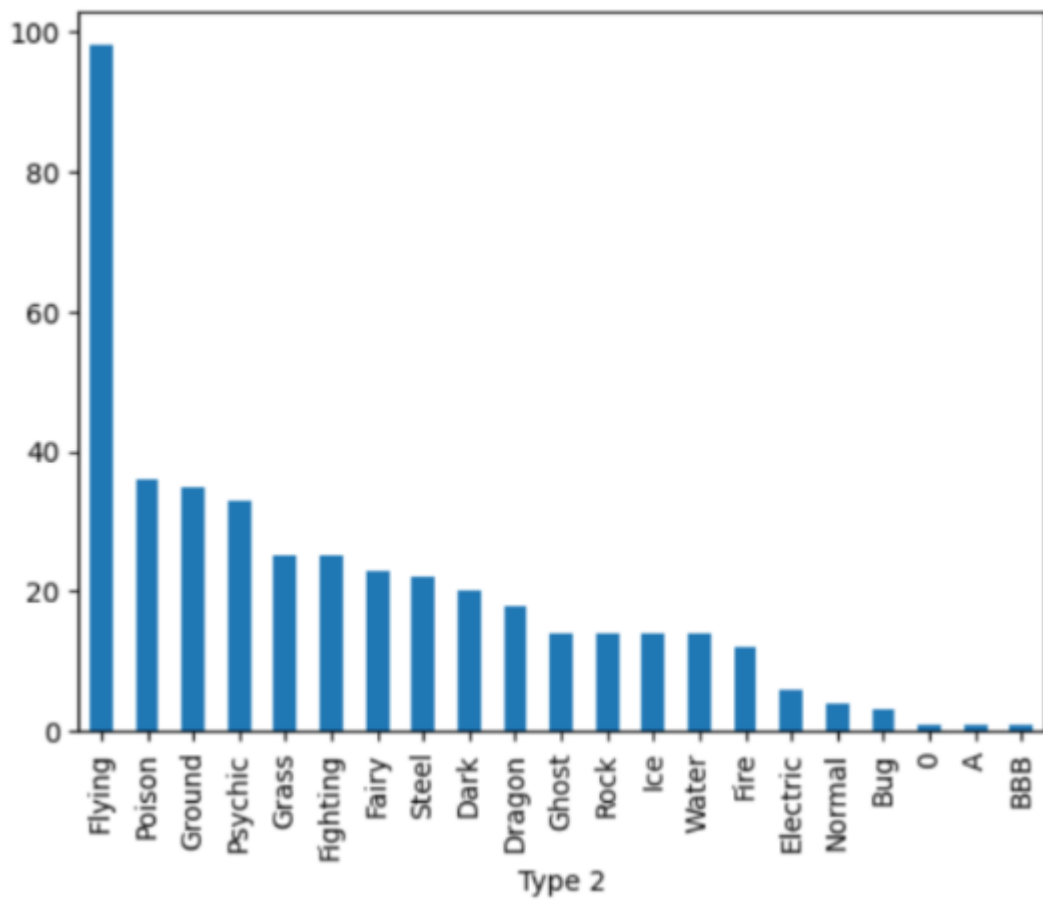
[12]:

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	FALSE
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	FALSE
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	FALSE
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	FALSE
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	FALSE
...
801	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	6	TRUE
802	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	6	TRUE
803	720	HoopaaHoopaa Confined	Psychic	Ghost	600	80	110	60	150	130	70	6	TRUE
804	720	HoopaaHoopaa Unbound	Psychic	Dark	680	80	160	60	170	130	80	6	TRUE
805	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	TRUE

419 rows × 13 columns

3、删除异常值

```
: <Axes: xlabel='Type 2'>
```



```
[12]: data = data[~data["Type 2"].isin(["0", "A", "BBB"])]
data
```

	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	FALSE
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	FALSE
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	FALSE
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	FALSE
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	FALSE
...
801	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	6	TRUE
802	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	6	TRUE
803	720	HoopaaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	6	TRUE
804	720	HoopaaHoopa Unbound	Psychic	Dark	680	80	160	60	170	130	80	6	TRUE
805	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	TRUE

416 rows x 13 columns

4、删除重复值

```
[13]: data[data.duplicated()]
```

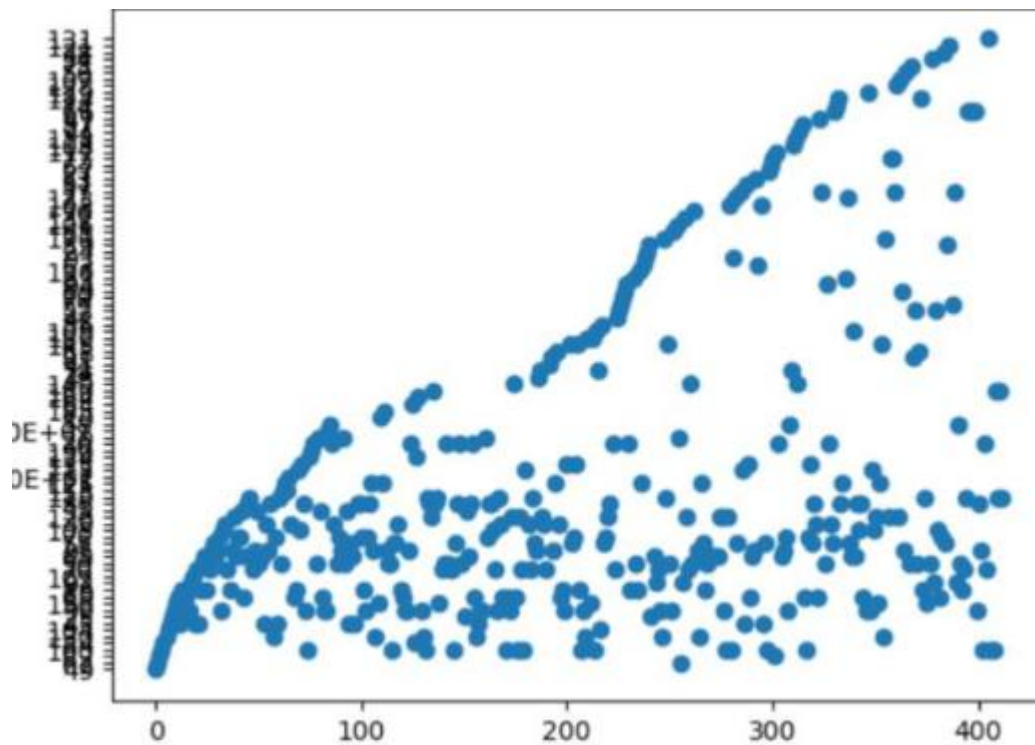
[13]:	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
23	17	Pidgeotto	Normal	Flying	349	63	60	55	50	50	71	1	FALSE
185	168	Ariados	Bug	Poison	390	70	90	70	60	60	40	2	FALSE
186	168	Ariados	Bug	Poison	390	70	90	70	60	60	40	2	FALSE
187	168	Ariados	Bug	Poison	390	70	90	70	60	60	40	2	FALSE

```
[14]: data = data.drop_duplicates()  
data
```

[14]:	#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	FALSE
1	2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	FALSE
2	3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	FALSE
3	3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	FALSE
6	6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	FALSE
...
801	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	6	TRUE
802	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	6	TRUE
803	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	6	TRUE
804	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160	60	170	130	80	6	TRUE
805	721	Volcanion	Fire	Water	600	80	110	120	130	90	70	6	TRUE

412 rows × 13 columns

5、attack 属性存在过高的异常值



实验代码：

见文件夹中的 python 文件

结论分析与体会：

本次实验针对 Pokemon 数据集的 5 类问题展开清洗：删除 2 行无意义尾行、清空 Type2 列 “273” 异常值、去除重复值、转换 Attack 列为数值型、修正属性置换。清洗后数据从 723 行精简至约 710 行有效数据，Attack 列经 `pd.to_numeric` 转换后，成功绘制箱线图，识别出 150 以上的高攻击异常值。

实操中，编码需匹配 `utf-8-sig`，用 `.copy()` 可消除视图警告，数据类型检查是绘图关键。这让我体会到，数据质量处理无小事，编码适配、类型转换等细节直接影响分析结果，严谨的逐步验证是后续数据分析可靠性的前提。