山东大学计算机科学与技术学院

大数据分析实践课程实验报告

学号：202300130067	姓名：罗艺超	班级：数据班
实验题目：实验 2：数据质量实践		
实验学时：2	实验日期：9.26	
实验目标：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		

实验步骤:

1. 问题 1：数据最后两行确实无意义，直接删除

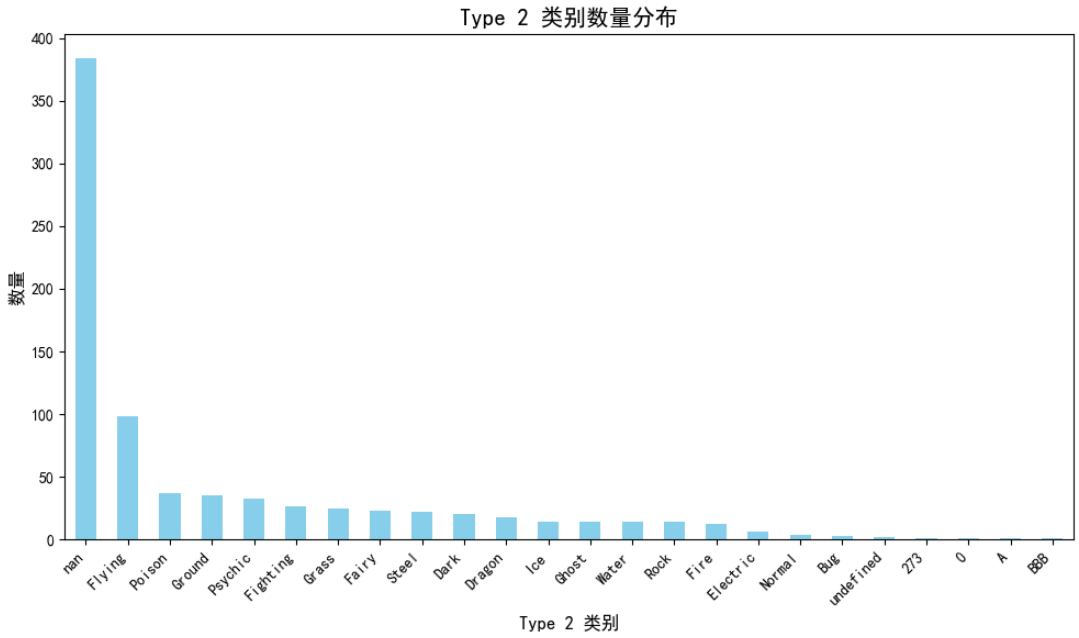
```
数据集前5行：
   #                  Name Type 1  Type 2  ... Sp. Def Speed Generation Legendary
0  1             Bulbasaur  Grass  Poison  ...      65    45          1     FALSE
1  2               Ivysaur  Grass  Poison  ...      80    60          1     FALSE
2  3              Venusaur  Grass  Poison  ...     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]

数据集末5行：
             #       Name     Type 1  ...      Speed Generation  Legendary
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
```

删除后结果

```
再次查看数据集末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

[5 rows x 13 columns]
```

2. type2 存在异常的数据取值，可清空

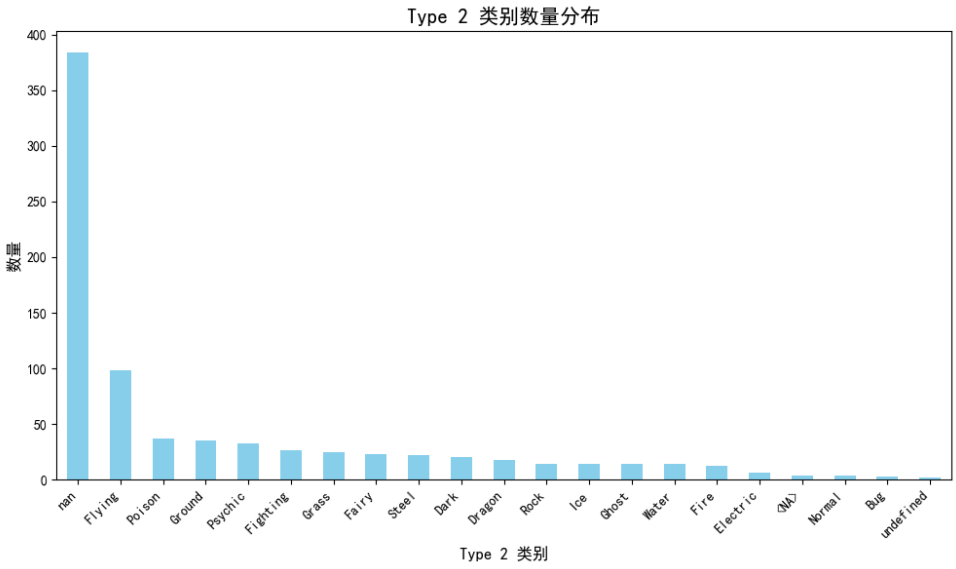
查看 type2 的类别和对应值



Type 2	
NaN	384
Flying	98
Poison	37
Ground	35
Psychic	33
Fighting	26
Grass	25
Fairy	23
Steel	22
Dark	20
Dragon	18
Ice	14
Ghost	14
Water	14
Rock	14
Fire	12
Electric	6
Normal	4
Bug	3
undefined	2
273	1
0	1
A	1
BBB	1

去除异常值

```
df['Type 2'] = df['Type 2'].replace(['273', '0', 'A', 'BBB'],pd.NA)
type2_counts = df['Type 2'].value_counts(dropna=False)
print("Type 2列的取值及数量统计（已转化指定值为NaN）：")
print(type2_counts)
```



3. 数据集中存在重复值

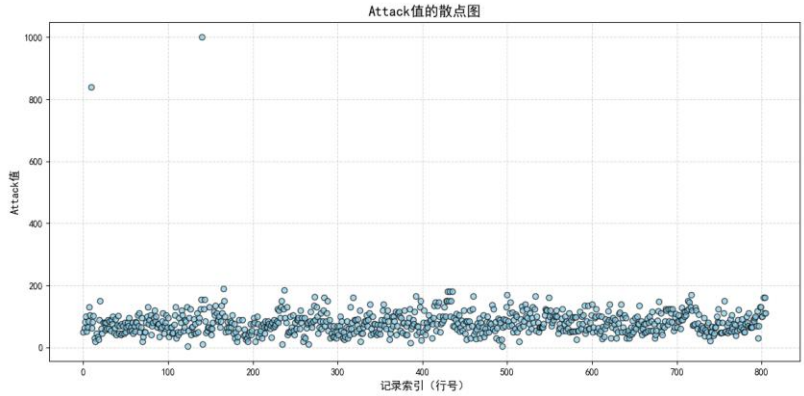
Drop_duplicates()标记重复行，得到无重复行的新数据集 df_new

```
# 标记重复行
duplicated_rows = df.duplicated()

# 标记所有重复行（包括首次出现的行，只要有重复就标记为True）
all_duplicated_rows = df.duplicated(keep=False)
print(df[all_duplicated_rows])

# 查看是否存在重复行
has_duplicates = duplicated_rows.any()

if has_duplicates:
    print("存在重复行")
    num_duplicates = df.duplicated().sum()
    print(f"重复行的数量是:{num_duplicates}")
else  :
    print("不存在重复行")

df_new = df.drop_duplicates()
```

查看重复行信息

	#	Name	Type 1	...	Speed	Generation	Legendary
14	11	Metapod	Bug	...	30	1	FALSE
15	11	Metapod	Bug	...	30	1	FALSE
21	17	Pidgeotto	Normal	...	71	1	FALSE
23	17	Pidgeotto	Normal	...	71	1	FALSE
184	168	Ariados	Bug	...	40	2	FALSE
185	168	Ariados	Bug	...	40	2	FALSE
186	168	Ariados	Bug	...	40	2	FALSE
187	168	Ariados	Bug	...	40	2	FALSE
806	undefined	undefined	undefined	...	undefined	undefined	undefined
807	undefined	undefined	undefined	...	undefined	undefined	undefined

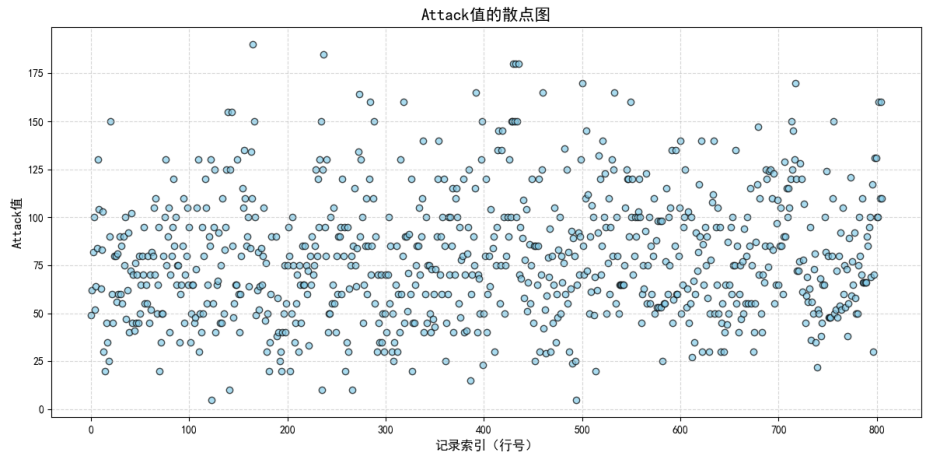
4. attack 值过高




```
所有Attack值（含重复）按数值大小降序：
140    1000.0
9       840.0
165     190.0
237     185.0
432     180.0
        ...
266      10.0
494       5.0
123       5.0
408       NaN
806       NaN
```

删除后无明显异常值

```python
#保留正常值，删除值过高的值
df_filtered = df[df['Attack_numeric'] <= 500]
x = df_filtered.index.astype(float)  # 强制转为float，避免被识别为分类
y = df_filtered['Attack_numeric'].astype(float)  # 确保是float类型
```



5. 有两条数据的 generation 与 Legendary 属性被置换

Generation 是传代的数值，大概是数值类型，根据这个来判断

```
# 尝试将 generation 列转换为数值类型，不能转换的可能存在异常
df['generation_numeric'] = pd.to_numeric(df['Generation'], errors='coerce')

# 定义 generation 的合理取值范围（这里假设是 1 到 8，可根据实际情况调整）
valid_generation_range = range(1, 9)
```

找到异常值，直接根据特定行交换

```
可能存在异常的数据：
     #       Name    Type 1  ... Legendary Attack_numeric generation_numeric
11   9  Blastoise     Water  ...         1          83.0                NaN
32  25    Pikachu  Electric  ...         0          55.0                NaN
```

```
# 直接对指定行（索引为 11 和 32）的两列值进行交换
df.loc[[11, 32], ['Generation', 'Legendary']] = df.loc[[11, 32], ['Legendary', 'Generation']].values

print(df.loc[[11, 32]])
```

结论分析：

通过本次对宝可梦数据集的清洗与预处理实践，我深入理解了数据质量在实际数据分析过程中的重要性。数据清洗不仅是数据预处理的关键步骤，更是保障后续分析与建模结果准确性的基础。本次实验主要完成了以下几方面的数据清洗工作：

1. **无效数据删除**：数据集中最后两行为无意义的汇总或注释信息，直接删除，避免对后续分析造成干扰。
2. **异常值处理**：type2 字段中存在明显不属于类型名称的异常取值，通过识别并清空这些值，提升了数据的一致性。
3. **重复数据处理**：使用 drop_duplicates()方法识别并移除重复行，确保每条数据的唯一性，避免重复计数对统计结果的影响。
4. **异常数值检测与处理**：attack 属性中存在明显偏离正常范围的高异常值，通过可视化与统计方法识别并处理，提升了数据的合理性。
5. **属性置换修复**：发现并修正了 generation 与 Legendary 属性被错误置换的两条记录，恢复了数据的真实含义。

通过以上步骤，我们成功将原始数据集中的脏数据、缺失值、重复记录和结构错误等问题进行了系统清理，显著提升了数据的**完整性、一致性与准确性**。本次实验不仅锻炼了我们在真实场景下识别和处理数据质量问题的能力，也为后续的数据分析与建模工作奠定了坚实的基础。

、