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：202300130041	姓名： 徐守政	班级： 数据
实验题目：数据质量实践		
实验学时：2	实验日期： 2025/9/26	

实验目的：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

硬件环境：

计算机一台

软件环境：

Linux 或 Windows

实验步骤与内容：

首先加载数据集

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# 加载数据
df = pd.read_csv(r"C:\Users\Lenovo\Downloads\Pokemon (1).csv")
```

然后进行筛选数据的第一步，删除最后无意义的值，我们打印出最后 5 行数据，然后发现在最后有两行 undefined 和两行 NaN，因此把他们全部删除

```
# 查看最后几行数据
print("最后 5 行数据:")
print(df.tail())

# 删除最后四行无意义数据
df_clean = df.iloc[:-4].copy()
```

```
最后5行数据:
            #       Name     Type 1     Type 2      Total         HP  \
805       721  Volcanion       Fire      Water        600         80
806 undefined  undefined  undefined  undefined  undefined  undefined
807 undefined  undefined  undefined  undefined  undefined  undefined
808       NaN        NaN        NaN        NaN        NaN        NaN
809       NaN        NaN        NaN        NaN        NaN        NaN

        Attack    Defense    Sp. Atk    Sp. Def      Speed Generation  \
805        110        120        130         90         70          6
806 undefined  undefined  undefined  undefined  undefined  undefined
807 undefined  undefined  undefined  undefined  undefined  undefined
808       NaN        NaN        NaN        NaN        NaN        NaN
809       NaN        NaN        NaN        NaN        NaN        NaN

    Legendary
805      TRUE
806 undefined
807 undefined
808
809
```

然后对 Type 2 列进行筛选，我们先找出 Type 2 列的唯一值，再查看 Type2 列值的分布

```
# 查看 Type2 列的唯一值
print("Type2 列唯一值:")
print(df_clean['Type 2'].unique())

# 首先查看 Type2 列的值分布
type2_counts = df_clean['Type 2'].value_counts()
print("\nType2 值分布:")
print(type2_counts)
```

```
Type2列唯一值:
['Poison' nan 'Flying' 'Dragon' '0' 'Ground' '273' 'Fairy' 'Grass'
 'Fighting' 'Psychic' 'Steel' 'Ice' 'A' 'Rock' 'Dark' 'Water' 'Electric'
 'Fire' 'Ghost' 'Bug' 'BBB' 'Normal']

Type2值分布:
Type 2
Flying      98
Poison      37
Ground      35
Psychic     33
Fighting    26
Grass       25
Fairy       23
Steel       22
Dark        20
Dragon      18
Rock        14
Ghost       14
Water       14
Ice         14
Fire        12
Electric     6
Normal       4
Bug          3
A            1
273          1
0            1
BBB          1
Name: count, dtype: int64
```

从图中我们可以得到 A，273,0，BBB 是错误项，我们便把它们的内容替换为 NaN

```
df_clean['Type 2'] = df_clean['Type 2'].replace(['A', '273', '0', 'BBB'], np.nan)
```

下一步是检查重复行，我们利用 pandas 库中 duplicated()函数，找出重复项并只保留第一个出现的数据

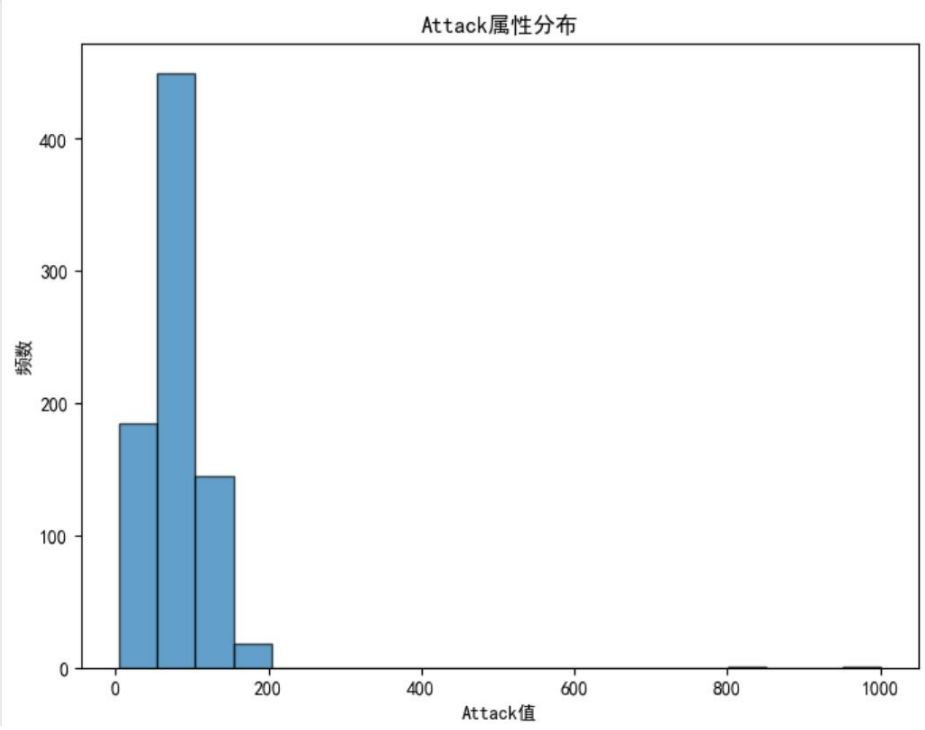
```
# 检查重复行
duplicates = df_clean.duplicated()
print(f"重复行数量: {duplicates.sum()}")

if duplicates.sum() > 0:
    print("重复的行:")
    print(df_clean[duplicates])

    # 删除重复行，保留第一个出现的数据
    df_clean = df_clean.drop_duplicates()
```

```
print(f"删除重复值后数据形状: {df_clean.shape}")
```

```
重复行数量: 5
重复的行:
        #       Name  Type 1  Type 2 Total  HP Attack Defense Sp. Atk Sp. Def  \
15     11    Metapod     Bug     NaN   205  50     20      55      25      25
23     17  Pidgeotto  Normal  Flying   349  63     60      55      50      50
185   168    Ariados     Bug  Poison   390  70     90      70      60      60
186   168    Ariados     Bug  Poison   390  70     90      70      60      60
187   168    Ariados     Bug  Poison   390  70     90      70      60      60

     Speed Generation Legendary
15      30          1     FALSE
23      71          1     FALSE
185     40          2     FALSE
186     40          2     FALSE
187     40          2     FALSE
删除重复值后数据形状: (801, 13)
```

我们先绘制 ATTACK 的直方图，来观察出是否有明显的极大值和极小值

```
# 设置中文字体
plt.rcParams['font.sans-serif'] = ['SimHei', 'Microsoft YaHei', 'DejaVu Sans']
plt.rcParams['axes.unicode_minus'] = False

# 如果数据类型不是数值型，先进行转换
df_clean['Attack'] = pd.to_numeric(df_clean['Attack'], errors='coerce')

# 绘制直方图
plt.figure(figsize=(8, 6))
plt.hist(df_clean['Attack'].dropna(), bins=20, alpha=0.7, edgecolor='black')
plt.title('Attack 属性分布')
plt.xlabel('Attack 值')
plt.ylabel('频数')
plt.show()
```



然后我们根据 IQR 的方法来识别并替换这些异常值

```
# 使用 IQR 方法识别异常值
```

```
Q1 = df_clean['Attack'].quantile(0.25)
Q3 = df_clean['Attack'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

print(f"Q1: {Q1}, Q3: {Q3}, IQR: {IQR}")
print(f"异常值边界: [{lower_bound}, {upper_bound}]")

outliers = df_clean[(df_clean['Attack'] < lower_bound) | (df_clean['Attack'] > upper_bound)]
print(f"Attack 异常值数量: {len(outliers)}")

if len(outliers) > 0:
    print("异常值数据:")
    print(outliers[['Name', 'Attack']])

    # 使用中位数替换异常值
    median_attack = df_clean['Attack'].median()
    df_clean.loc[df_clean['Attack'] > upper_bound, 'Attack'] = median_attack
    df_clean.loc[df_clean['Attack'] < lower_bound, 'Attack'] = median_attack
else:
    print("未发现异常值")
```

```
Q1: 55.0, Q3: 100.0, IQR: 45.0
异常值边界: [-12.5, 167.5]
Attack异常值数量: 9
异常值数据:
                       Name  Attack
9                  Squirtle   840.0
140                  Tauros  1000.0
165      MewtwoMega Mewtwo X   190.0
237  HeracrossMega Heracross   185.0
430    GroudonPrimal Groudon   180.0
432    RayquazaMega Rayquaza   180.0
435       DeoxysAttack Forme   180.0
500    GarchompMega Garchomp   170.0
717       KyuremBlack Kyurem   170.0
```

识别并修正 generation 和 Legendary 属性被置换的两条数据，因为 Generation 是文本，Legendary 是数字，如果 Generation 是文本，Legendary 是数字，说明需要交换

```
# 查找明显异常的行（Generation 是文本，Legendary 是数字）
for idx, row in df_clean.iterrows():
    gen_val = str(row['Generation'])
    leg_val = str(row['Legendary'])

    # 如果 Generation 是文本，Legendary 是数字，说明需要交换
    if gen_val.isalpha() and leg_val.isdigit():
        print(f"发现置换行: {row['Name']}")
        print(f"    原始: Generation={gen_val}, Legendary={leg_val}")
        # 交换值
```

```
            df_clean.at[idx, 'Generation'] = leg_val
            df_clean.at[idx, 'Legendary'] = gen_val
            print(f"    修正后: Generation={leg_val}, Legendary={gen_val}")

# 转换数据类型
df_clean['Generation'] = pd.to_numeric(df_clean['Generation'], errors='coerce')
```

```
发现置换行: Blastoise
  原始: Generation=FALSE, Legendary=1
  修正后: Generation=1, Legendary=FALSE
发现置换行: Pikachu
  原始: Generation=FALSE, Legendary=0
  修正后: Generation=0, Legendary=FALSE
```

结论分析与体会：

通过本次宝可梦数据集的数据质量实践，我深刻认识到数据清洗是数据分析过程中至关重要且耗时的基础环节。原始数据集中存在的多种质量问题——包括无意义的末尾行、Type2 列的异常取值、重复记录、Attack 属性的异常值以及 generation 与 Legendary 属性的置换错误，展现了真实世界数据的复杂性和不完美性。这次实验让我体会到，有效的数据清洗不仅需要技术手段，更需要结合业务理解进行判断。每一个数据问题的发现与解决过程，都是对数据敏感性和逻辑思维能力的锻炼，也让我明白了高质量的数据是保证后续分析结果可靠性的根本前提。