山东大学计算机科学与技术学院

可视化技术课程实验报告

学号：202302130293	姓名：李嘉欣	班级：数据科学与大数据技术
实验题目：二、数据质量实践	实验日期：2025/9/24	
实验目标：本实验以宝可梦数据集为对象，建立数据预处理清洗流程，处理无意义行、Type 2 列异常值、重复数据、Attack 属性异常高值及属性置换问题，输出规范数据集，提升数据质量问题处理能力。		
实验环境：windows 系统 & python3.11		

实验步骤：

1. 数据的读入

```
def main():
    print('1. 读取数据') #-----
    try:
        encodings = ['utf-8', 'gbk', 'latin-1', 'utf-16']
        data = None
        for encoding in encodings:
            try:
                data = pd.read_csv("Pokemon.csv", encoding=encoding)
                print(f"成功读取文件")
                break
            except UnicodeDecodeError:
                continue
        if data is None:
            raise Exception("无法解析文件")

        print("原始数据形状（行×列）：", data.shape)
        print("原始数据前5行：")
        print(data.head())
        print("数据列名：", data.columns.tolist())

    except Exception as e:
        print(f"读取数据时出错：{str(e)}")
        return
```

输出：

```
1. 读取数据
成功读取文件
原始数据形状（行×列）： (810, 13)
原始数据前5行：
   #                  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
数据列名： ['#', 'Name', 'Type 1', 'Type 2', 'Total', 'HP', 'Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed', 'Generation', 'Legendary']
```

2. 数据清洗

2.1 删除无意义数据行

```
# 2.1 删除无意义数据行
print("\n2.1 删除无意义数据行")
print("数据最后10行：")
print(cleaned_data.tail(10))

cleaned_data = cleaned_data[~cleaned_data['#'].isin(['undefined', np.nan])]
cleaned_data = cleaned_data.dropna(how='all')
print("删除无意义行后的数据形状：", cleaned_data.shape)
```

2.2 处理 Type 2 列异常值

```
print("\n2.2 处理Type 2列异常值")
if 'Type 2' in cleaned_data.columns:
    valid_types = set(cleaned_data['Type 1'].dropna().unique())
    print(f"Type列合理属性值: {sorted(valid_types)}")

    type2_values = cleaned_data['Type 2'].dropna()
    invalid_types = type2_values[~type2_values.isin(valid_types)]
    if not invalid_types.empty:
        print(f"检测到Type 2列异常值: {invalid_types.unique().tolist()}")
        cleaned_data.loc[~cleaned_data['Type 2'].isin(valid_types), 'Type 2'] = np.nan
        print(f"已处理所有异常值，共{len(invalid_types.unique())}种异常类型")
    else:
        print("未检测到Type 2列异常值")

    print("处理后Type 2列缺失值数量: ", cleaned_data['Type 2'].isnull().sum())
```

2.3 删除重复数据

```
print("\n2.3 删除重复数据")
duplicate_count = cleaned_data.duplicated().sum()
print(f"重复数据的数量: {duplicate_count}")
if duplicate_count > 0:
    cleaned_data = cleaned_data.drop_duplicates(keep='first')
    print(f"删除重复数据后的数据形状: {cleaned_data.shape}")
```

2.4 处理 Attack 属性异常高值

```
print("\n2.4 处理Attack属性异常高值")
if 'Attack' in cleaned_data.columns:
    cleaned_data['Attack'] = pd.to_numeric(cleaned_data['Attack'], errors='coerce')
    cleaned_data = cleaned_data.dropna(subset=['Attack'])
    print("已将Attack列转换为数值类型并移除无效值")

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

    outliers = cleaned_data[(cleaned_data['Attack'] < lower_bound) | (cleaned_data['Attack'] > upper_bound)]
    print(f"Attack属性异常值数量: {len(outliers)}")
    cleaned_data = cleaned_data[(cleaned_data['Attack'] >= lower_bound) & (cleaned_data['Attack'] <= upper_bound)]
    print(f"删除Attack异常值后的数据形状: {cleaned_data.shape}")
```

2.5 修正属性置换问题

```
print("\n2.5 修正属性置换问题")
if 'Generation' in cleaned_data.columns and 'Legendary' in cleaned_data.columns:
    cleaned_data = cleaned_data.dropna(subset=['Generation', 'Legendary'])
    print(f"移除Generation/Legendary列NaN值后, 数据形状: {cleaned_data.shape}")

    def check_swap(row):
        try:
            int(row['Generation'])
            if row['Legendary'] in ['TRUE', 'FALSE']:
                return False
            else:
                return True
        except (ValueError, TypeError):
            return True

    swap_rows = cleaned_data.apply(check_swap, axis=1)
    swap_count = sum(swap_rows)
    print(f"发现{swap_count}行存在属性置换问题")

    if swap_count > 0:
        cleaned_data.loc[swap_rows, ['Generation', 'Legendary']] = cleaned_data.loc[swap_rows, ['Legendary', 'Generation']].values
        print(f"已交换{swap_count}行的Generation和Legendary列值")

    cleaned_data['Generation'] = pd.to_numeric(cleaned_data['Generation'], errors='coerce').fillna(0).astype(int)
    cleaned_data['Legendary'] = cleaned_data['Legendary'].str.upper().isin(['TRUE']).astype(bool)

    print(f"修正后Generation列数据类型: {cleaned_data['Generation'].dtype}")
    print(f"修正后Legendary列数据类型: {cleaned_data['Legendary'].dtype}")
```

输出：

```
2.1 删除无意义数据行
数据最后10行：
            #                  Name     Type 1     Type 2      Total         HP     Attack    Defense    Sp. Atk    Sp. Def      Speed Generation  Legendary
800       718     Zygarde50% Forme     Dragon     Ground        600        108        100        121         81         95         95          6       TRUE
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
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
809       NaN                  NaN        NaN        NaN        NaN        NaN        NaN        NaN        NaN        NaN        NaN        NaN
删除无意义行后的数据形状： (805, 13)

2.2 处理Type 2列异常值
Type列合理属性值：['Bug', 'Dark', 'Dragon', 'Electric', 'Fairy', 'Fighting', 'Fire', 'Flying', 'Ghost', 'Grass', 'Ground', 'Ice', 'Normal', 'Poison', 'Psychic', 'Rock', 'Steel', 'Water']
检测到Type 2列异常值：['0', '273', 'A', 'BBB']
已处理所有异常值，共4种异常类型
处理后Type 2列缺失值数量： 387

2.3 删除重复数据
重复数据的数量：5
删除重复数据后的数据形状：(800, 13)

2.4 处理Attack属性异常高值
已将Attack列转换为数值类型并移除无效值
Attack属性异常值数量：9
删除Attack异常值后的数据形状：(791, 13)

2.5 修正属性置换问题
移除Generation/Legendary列NaN值后，数据形状：(789, 13)
发现7行存在属性置换问题
已交换7行的Generation和Legendary列值
修正后Generation列数据类型：int64
修正后Legendary列数据类型：bool
```

3. 清洗后的数据导出

```
cleaned_data.to_csv('cleaned_pokemon.csv', index=False, encoding='utf-8-sig')
print('\n3. 清洗后的数据导出为: cleaned_pokemon.csv')
```

输出：

```
3. 清洗后的数据导出为: cleaned_pokemon.csv
PS D:\1\学习\大三\大三上\大数据分析实践\实验\实验2>
```

实验分析与体会：

一、实验分析

本次实验围绕宝可梦数据集展开，通过多步骤清洗验证了数据预处理的关键逻辑。无意义行删除使数据从 810 行精简至有效行数，Type 2 列通过合理属性集合识别并处理 “0” “A” 等异常值，重复数据清零保障唯一性；Attack 属性经数值转换与 IQR 方法剔除 9 个异常值，属性置换问题修正后两列数据类型分别统一为 int64 和 bool。整个过程中发现，Type 2 列的缺失值包含合理空值与异常值转换结果，需结合业务逻辑区分处理，避免过度清洗破坏数据真实性。

二、实验体会

通过本次实验，深刻认识到数据质量是后续分析的基础。前期仅针对性处理 “273” 异常值导致遗漏其他问题，提醒我需建立全面的异常检测思维；重复数据与属性置换的处理过程，让我掌握了数据一致性校验的实用方法。同时发现，清洗需平衡 “完整性” 与 “合理性”，如宝可梦单属性导致的 Type 2 列空值无需填充，这为后续处理类似数据集提供了宝贵经验，也提升了自身解决实际数据问题的能力。