山东大学计算机科学与技术学院

大数据分析与实践课程实验报告

学号：202300130205	姓名：李尚远	班级：23 级数据班
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。

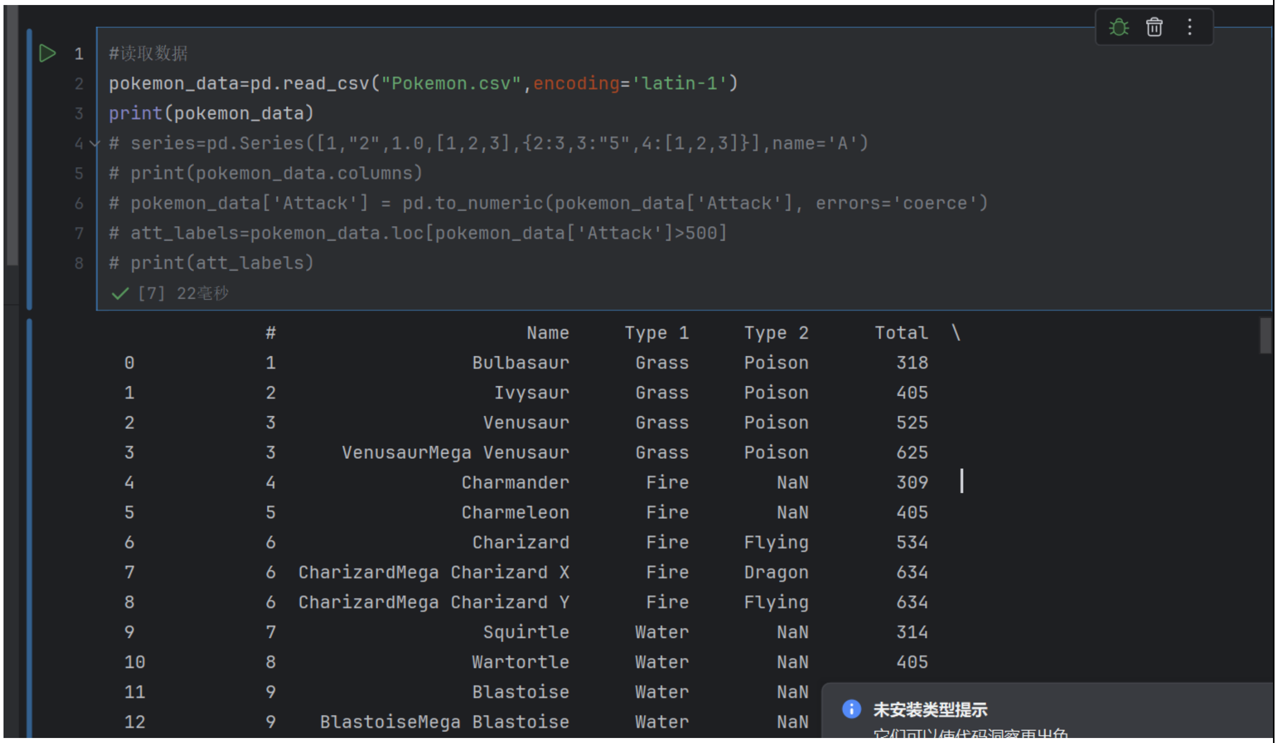
实验环境

python3.9， jupyter notebook

实验步骤：

一、

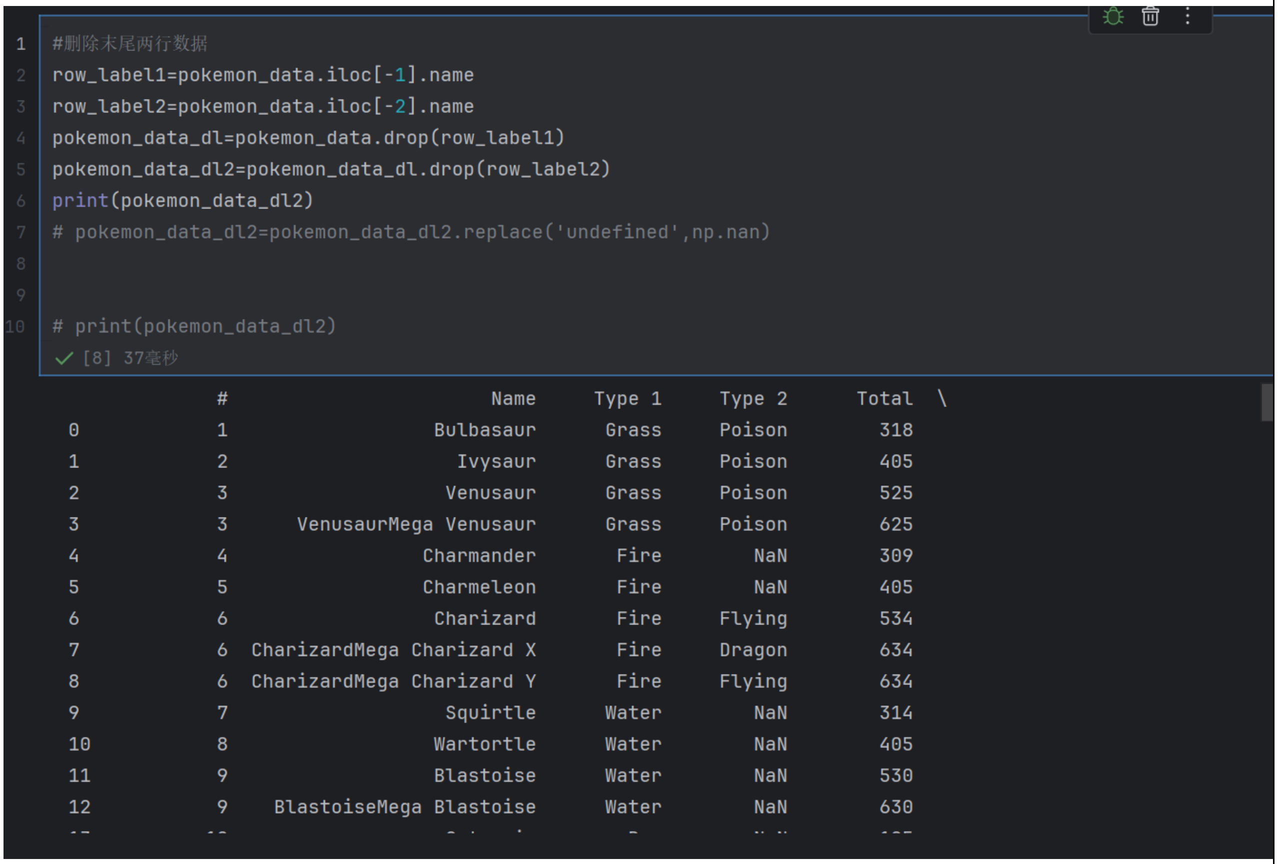
导入数据集



```
#读取数据
pokemon_data=pd.read_csv("Pokemon.csv",encoding='latin-1')
print(pokemon_data)
# series=pd.Series([1,"2",1.0,[1,2,3],{2:3,3:"5",4:[1,2,3]}],name='A')
# print(pokemon_data.columns)
# pokemon_data['Attack'] = pd.to_numeric(pokemon_data['Attack'], errors='coerce')
# att_labels=pokemon_data.loc[pokemon_data['Attack']>500]
# print(att_labels)
✓ [7] 22毫秒
```

```
             #                   Name   Type 1  Type 2   Total  \
0            1              Bulbasaur    Grass  Poison     318
1            2                Ivysaur    Grass  Poison     405
2            3               Venusaur    Grass  Poison     525
3            3  VenusaurMega Venusaur    Grass  Poison     625
4            4             Charmander     Fire     NaN     309
5            5             Charmeleon     Fire     NaN     405
6            6              Charizard     Fire  Flying     534
7            6 CharizardMega Charizard X  Fire  Dragon     634
8            6 CharizardMega Charizard Y  Fire  Flying     634
9            7               Squirtle    Water     NaN     314
10           8              Wartortle    Water     NaN     405
11           9              Blastoise    Water     NaN
12           9  BlastoiseMega Blastoise  Water     NaN
```

二、删除末尾两行数据



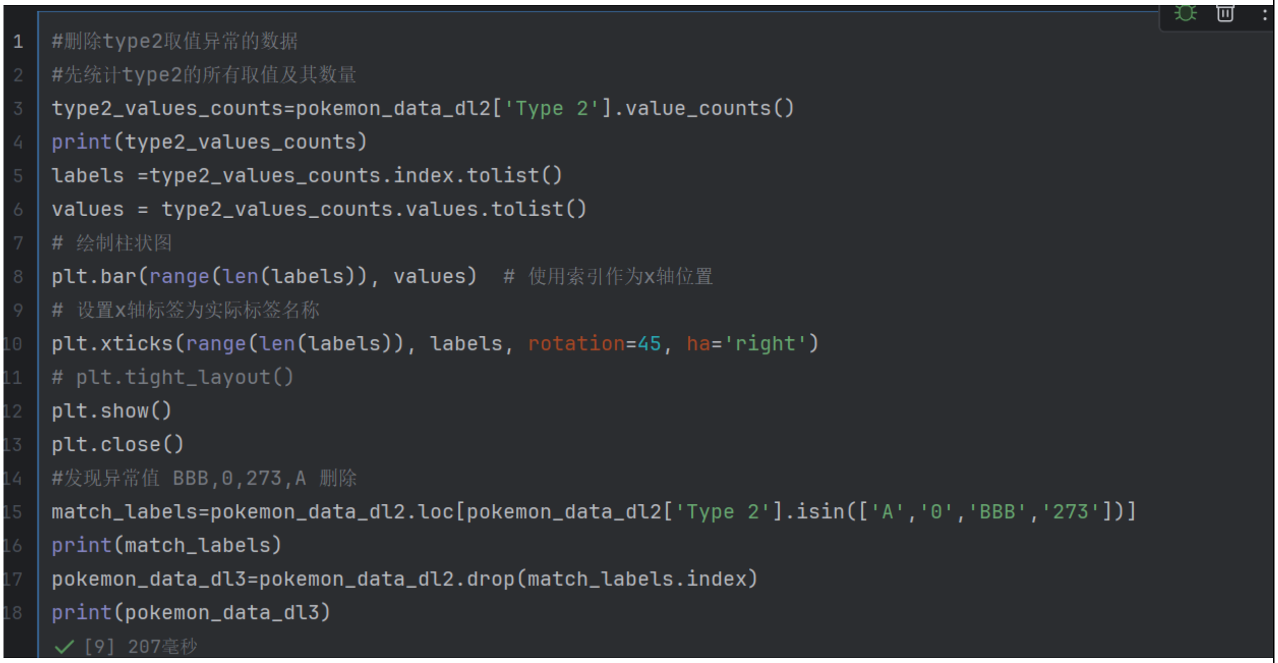
```
#删除末尾两行数据
row_label1=pokemon_data.iloc[-1].name
row_label2=pokemon_data.iloc[-2].name
pokemon_data_dl=pokemon_data.drop(row_label1)
pokemon_data_dl2=pokemon_data_dl.drop(row_label2)
print(pokemon_data_dl2)
# pokemon_data_dl2=pokemon_data_dl2.replace('undefined',np.nan)


# print(pokemon_data_dl2)
```

✓ [8] 37毫秒

```
           #                   Name   Type 1    Type 2     Total  \
0          1              Bulbasaur    Grass    Poison       318
1          2                Ivysaur    Grass    Poison       405
2          3               Venusaur    Grass    Poison       525
3          3     VenusaurMega Venusaur  Grass    Poison       625
4          4             Charmander     Fire       NaN       309
5          5             Charmeleon     Fire       NaN       405
6          6              Charizard     Fire    Flying       534
7          6  CharizardMega Charizard X  Fire    Dragon       634
8          6  CharizardMega Charizard Y  Fire    Flying       634
9          7               Squirtle    Water       NaN       314
10         8              Wartortle    Water       NaN       405
11         9              Blastoise    Water       NaN       530
12         9    BlastoiseMega Blastoise  Water       NaN       630
```

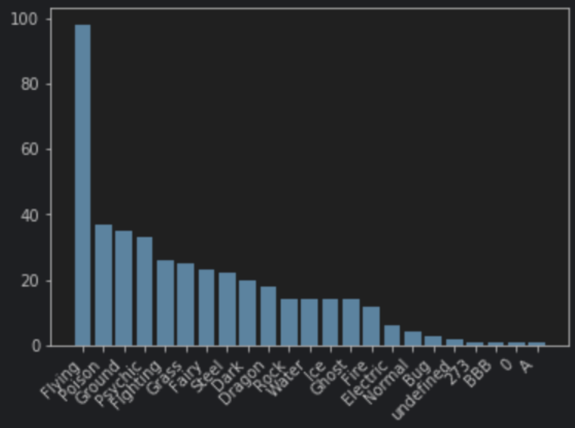
三、统计 type2 中取值的分布找到异常类型并删除



```
#删除type2取值异常的数据
#先统计type2的所有取值及其数量
type2_values_counts=pokemon_data_dl2['Type 2'].value_counts()
print(type2_values_counts)
labels =type2_values_counts.index.tolist()
values = type2_values_counts.values.tolist()
# 绘制柱状图
plt.bar(range(len(labels)), values)  # 使用索引作为x轴位置
# 设置x轴标签为实际标签名称
plt.xticks(range(len(labels)), labels, rotation=45, ha='right')
# plt.tight_layout()
plt.show()
plt.close()
#发现异常值 BBB,0,273,A 删除
match_labels=pokemon_data_dl2.loc[pokemon_data_dl2['Type 2'].isin(['A','0','BBB','273'])]
print(match_labels)
pokemon_data_dl3=pokemon_data_dl2.drop(match_labels.index)
print(pokemon_data_dl3)
```

✓ [9] 207毫秒

```
A              1
Name: Type 2, dtype: int64
```



```
796           6      FALSE
797           6      FALSE
798           6       TRUE
799           6       TRUE
800           6       TRUE
801           6       TRUE
802           6       TRUE
803           6       TRUE
804           6       TRUE
805           6       TRUE
806   undefined  undefined
```

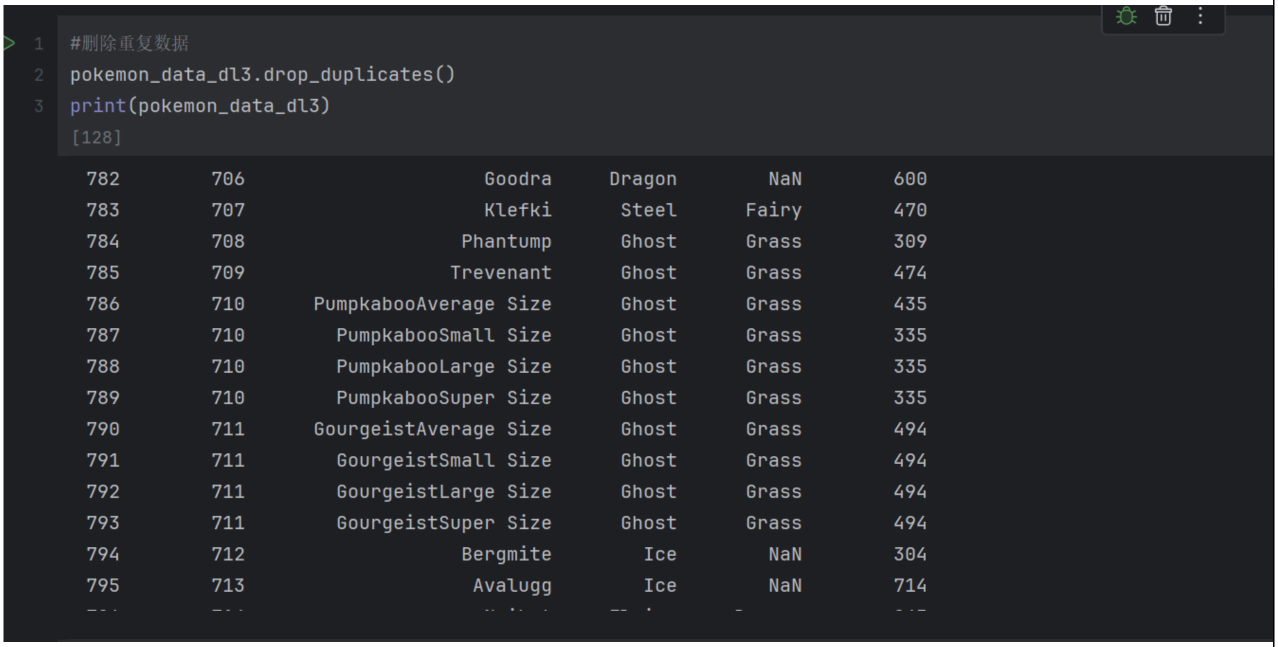
四、随机抽样

```python
#随机采样前的预处理
data_before_sample=data_after_filter_2
random_sample=data_before_sample
columns=data_before_sample.columns  #用于后续回复数据列结构
```
[5]

```python
#随机抽样
random_sample_finish=random_sample.sample(n=50)
print(random_sample_finish)
random_sample_finish=random_sample_finish[columns]
# print(random_sample_finish)
```
[6]

```
      from_dev  from_port from_city from_level  to_dev  to_port to_city  \
799        180         52      呼和浩特       一般节点     474      460     哈尔滨
51          96        156      呼和浩特       一般节点    3227      103      济南
44          96        127      呼和浩特       一般节点    1756     1027      北京
423        591        558        绥化       一般节点     180       20    呼和浩特
124        474       1311       哈尔滨       一般节点    2549     1570      沈阳
9           47        252        通辽       一般节点      96      134    呼和浩特
41          96        120      呼和浩特       一般节点    1997      250      天津
```

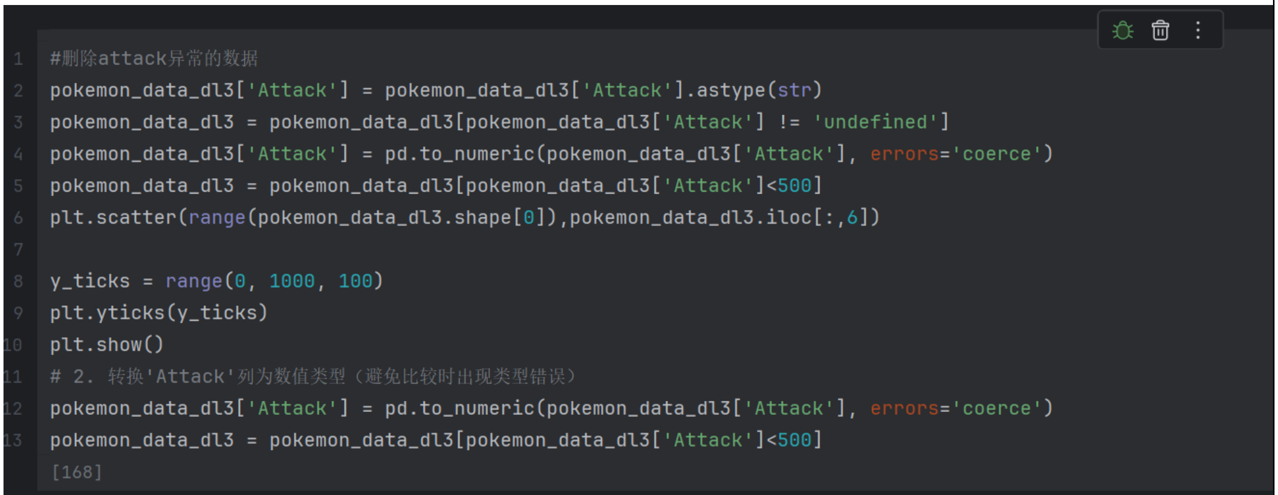
五、删除重复数据



```
#删除重复数据
pokemon_data_dl3.drop_duplicates()
print(pokemon_data_dl3)
[128]
```

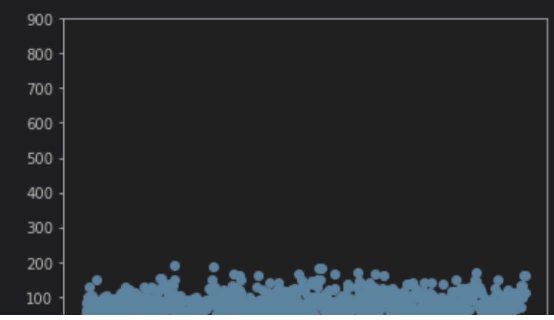
```
782        706                   Goodra    Dragon      NaN       600
783        707                   Klefki     Steel    Fairy       470
784        708                 Phantump     Ghost    Grass       309
785        709                Trevenant     Ghost    Grass       474
786        710     PumpkabooAverage Size    Ghost    Grass       435
787        710       PumpkabooSmall Size    Ghost    Grass       335
788        710       PumpkabooLarge Size    Ghost    Grass       335
789        710       PumpkabooSuper Size    Ghost    Grass       335
790        711     GourgeistAverage Size    Ghost    Grass       494
791        711       GourgeistSmall Size    Ghost    Grass       494
792        711       GourgeistLarge Size    Ghost    Grass       494
793        711       GourgeistSuper Size    Ghost    Grass       494
794        712                 Bergmite       Ice      NaN       304
795        713                  Avalugg       Ice      NaN       714
```

六、删除 attack 异常的数据

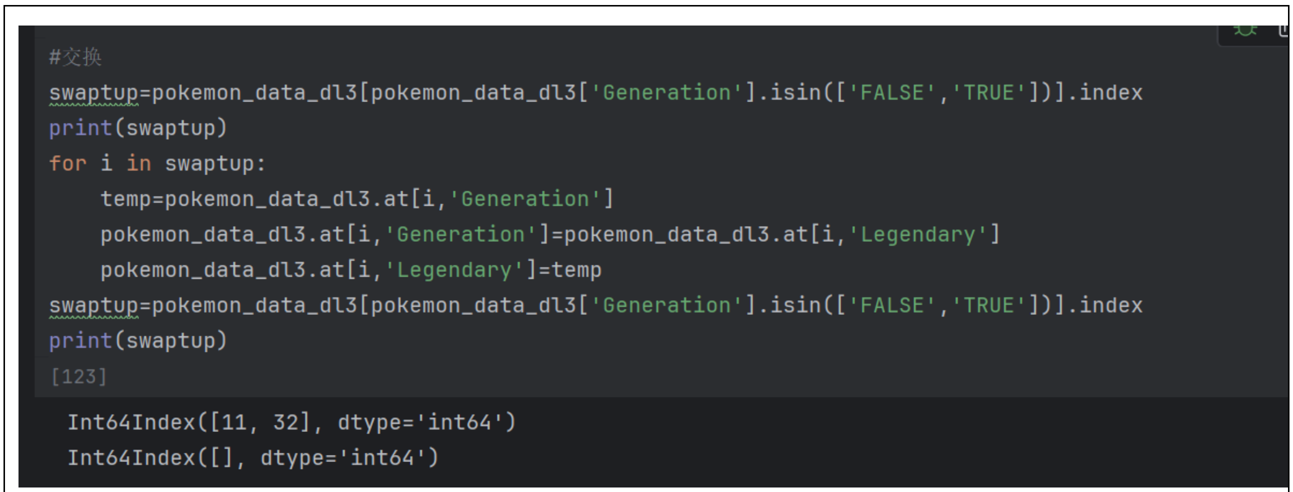


```
#删除attack异常的数据
pokemon_data_dl3['Attack'] = pokemon_data_dl3['Attack'].astype(str)
pokemon_data_dl3 = pokemon_data_dl3[pokemon_data_dl3['Attack'] != 'undefined']
pokemon_data_dl3['Attack'] = pd.to_numeric(pokemon_data_dl3['Attack'], errors='coerce')
pokemon_data_dl3 = pokemon_data_dl3[pokemon_data_dl3['Attack']<500]
plt.scatter(range(pokemon_data_dl3.shape[0]),pokemon_data_dl3.iloc[:,6])

y_ticks = range(0, 1000, 100)
plt.yticks(y_ticks)
plt.show()
# 2. 转换'Attack'列为数值类型（避免比较时出现类型错误）
pokemon_data_dl3['Attack'] = pd.to_numeric(pokemon_data_dl3['Attack'], errors='coerce')
pokemon_data_dl3 = pokemon_data_dl3[pokemon_data_dl3['Attack']<500]
[168]
```



七、交换 generation 和 legendary 中错误置换的数据



```
#交换
swaptup=pokemon_data_dl3[pokemon_data_dl3['Generation'].isin(['FALSE','TRUE'])].index
print(swaptup)
for i in swaptup:
    temp=pokemon_data_dl3.at[i,'Generation']
    pokemon_data_dl3.at[i,'Generation']=pokemon_data_dl3.at[i,'Legendary']
    pokemon_data_dl3.at[i,'Legendary']=temp
swaptup=pokemon_data_dl3[pokemon_data_dl3['Generation'].isin(['FALSE','TRUE'])].index
print(swaptup)
[123]
```

```
Int64Index([11, 32], dtype='int64')
Int64Index([], dtype='int64')
```