Movies Dataset from Pirated Sites的数据分析及预处理

学号：1120202579 姓名：彭高鹏

In [1]:
```
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as plt
import seaborn as sns
```

1. 数据集简介

该数据集是从盗版网站收集的，该网站每月约有 200 万访问者的用户群。该数据包含来自好莱坞、宝莱坞、动漫等所有行业的 20,000+ 多部电影。

数据集的各个属性的简要描述如下：
id: 电影的唯一 ID
title: 电影的标题
storyline: 电影的简短描述
views: 每部电影的点击次数
downloads: 每部电影的下载量
IMDb-rating: IMDb上的评分
appropriate_for: 适合人群分级，有R-rated，PG-13等
language: 电影的语言
industry: 电影的制作行业，如好莱坞，宝莱坞等
posted_date: 电影在平台上发布的时间
release_date: 这部电影在全球上映的时间
runtime: 电影的时长，以分钟或小时为单位
director: 电影导演的名称
writer: 所有编剧的名单

下面对数据集的csv文件进行读取，并展示前几个数据对象

In [2]:
```
# 自定义函数，用于将时间转换为分钟
def transform_runtime(runtime):
    total_minutes = 0
    if runtime.strip() == '':
        return None
    if 'h' in runtime:
        hours, minutes = runtime.split('h')
        hours = int(hours)
        if minutes:
            minutes = int(minutes[0:2].strip())
            total_minutes = hours * 60 + minutes
        else:
            total_minutes = hours * 60
    else:
        total_minutes = int(runtime.replace('min', '').strip())
    return total_minutes
```

In [3]:
```
df = pd.read_csv('movies_dataset.csv',thousands=',',converters={'run_time': transfo
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Unnamed: 0      20548 non-null  int64  
 1   IMDb-rating     19707 non-null  float64
 2   appropriate_for 11072 non-null  object 
 3   director        18610 non-null  object 
 4   downloads       20547 non-null  float64
 5   id              20548 non-null  int64  
 6   industry        20547 non-null  object 
 7   language        20002 non-null  object 
 8   posted_date     20547 non-null  object 
 9   release_date    20547 non-null  object 
 10  run_time        18780 non-null  float64
 11  storyline       18847 non-null  object 
 12  title           20547 non-null  object 
 13  views           20547 non-null  float64
 14  writer          18356 non-null  object 
dtypes: float64(4), int64(2), object(9)
memory usage: 2.4+ MB
```

In [4]:
```
df.head(5)
```

Out[4]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language
0	0	4.8	R	John Swab	304.0	372092	Hollywood / English	English
1	1	6.4	TV-PG	Paul Ziller	73.0	372091	Hollywood / English	English
2	2	5.2	R	Ben Wheatley	1427.0	343381	Hollywood / English	English,Hindi
3	3	8.1	NaN	Venky Atluri	1549.0	372090	Tollywood	Hindi
4	4	4.6	NaN	Shaji Kailas	657.0	372089	Tollywood	Hindi

定义分析数据的方案，包括展示数据信息、绘制数据的直方图和盒图

In [5]:
```
#定义标称型数据的分析方案
def analyze_freq_nominal(name,data,draw=True):
    freq = data.value_counts()
    print("Frequency information of the {}:\n".format(freq))
    print("Missing Value Count:",data.isnull().sum())

    if draw:
    # 绘制处理数据的频数直方图
        plt.figure(figsize=(12,5))
        plt.bar(freq.index,freq.values)
        plt.title(name + ' Frequency Histogram')
        plt.xlabel(name)
        plt.ylabel(' Frequency')
        plt.grid(axis='y')
        plt.tick_params(axis='x', labelsize=8)
        plt.xticks(rotation=-50)
        plt.tight_layout()
        plt.show()

#定义数值型数据的分析方案
def analyze_freq_and_box_numeric(name,data,draw=True):
    print("The Information of {}:\n".format(name))
    print(data.describe())
    print("Missing Value Count:",data.isnull().sum())

    if draw:
        # 绘制数据的频数直方图
        plt.figure(figsize=(24,5))
        plt.subplot(1, 2, 1)
        sns.histplot(data, kde=True)
        plt.title(name + ' Frequency Histogram')
        plt.xlabel(name)
        plt.ylabel(' Frequency')
        plt.grid(axis='y')

        #绘制数据的盒图
        plt.subplot(1, 2, 2)
        sns.boxplot(data)
        plt.title( name + ' Boxplot')
        plt.ylabel(name)
        plt.grid(axis='y')

        plt.tight_layout()
        plt.show()
```

2. 数据摘要和可视化

- **数据摘要**

  标称属性，给出每个可能取值的频数</br>
  数值属性，给出5数概括及缺失值的个数

- **数据可视化**

  使用直方图、盒图等检查数据分布及离群点

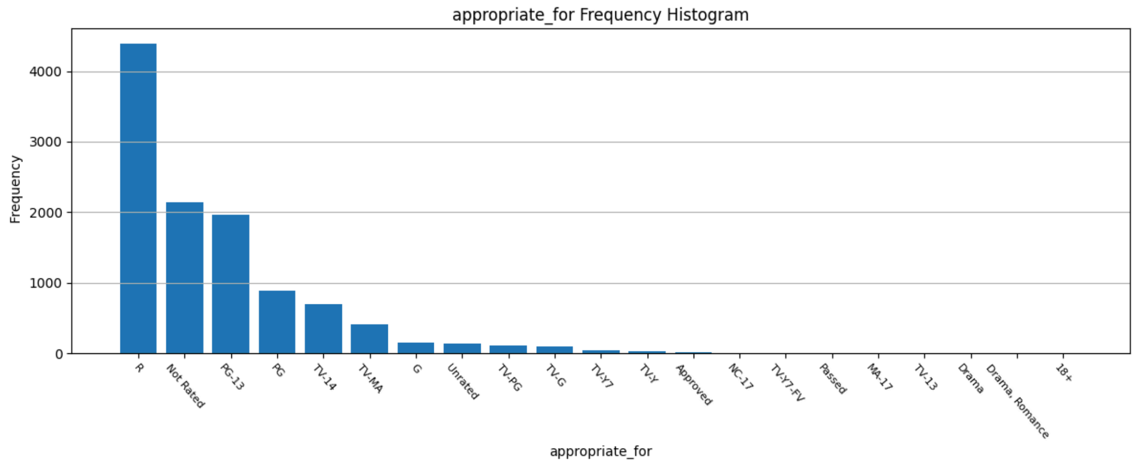
2.1 分析标称型数据

数据集中的标称型数据有：电影的分级appropriate_for，电影的导演director，电影的标识id，电影的制作行业industry，电影的语言language，故事简述storyline，电影题目title，电影编剧writer 下面我们对以上几个标称型数据逐一进行分析 其中对于数据类型较多的数据，我们不进行画图，只进行频数分析

In [6]:
```
analyze_freq_nominal("appropriate_for",df['appropriate_for'])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("director",df['director'],draw=False)
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("id",df['id'],draw=False)
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("industry",df['industry'])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("language",df['language'],draw=False)
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("storyline",df['storyline'],draw=False)
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>")
analyze_freq_nominal("title",df['title'],draw=False)
```

```
Frequency information of the appropriate_for
R                 4384
Not Rated         2142
PG-13             1968
PG                 886
TV-14              694
TV-MA              406
G                  152
Unrated            132
TV-PG              115
TV-G                99
TV-Y7               45
TV-Y                25
Approved             9
NC-17                4
TV-Y7-FV             3
Passed               3
MA-17                1
TV-13                1
Drama                1
Drama, Romance       1
18+                  1
Name: count, dtype: int64:

Missing Value Count: 9476
```

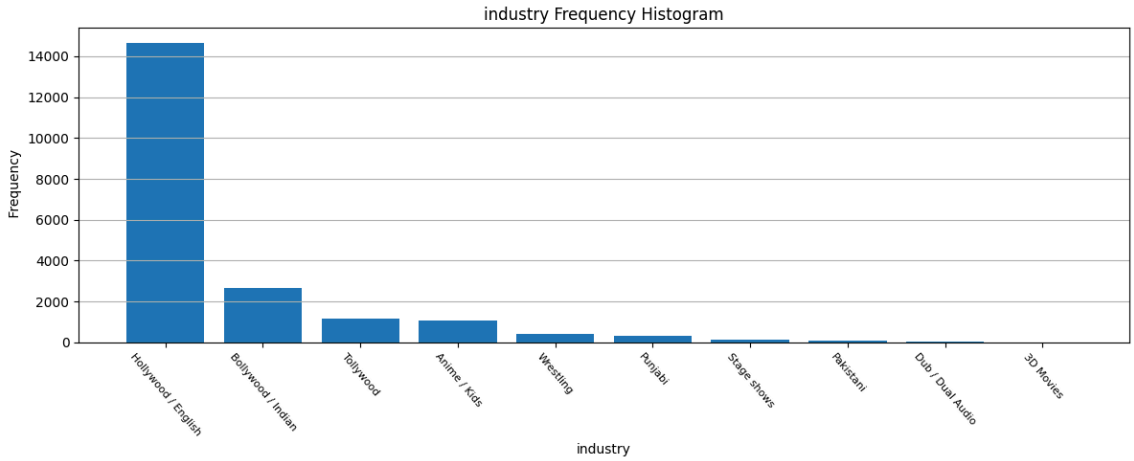


```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the director
Venky Atluri                                   405
Simone Stock                                   403
Xavier Manrique                                403
John Swab                                      205
Neil Jordan                                    205
                                              ...
Agnieszka Smoczynska                             1
Dylan Thomas Ellis                               1
Sunil Thakur, Sunil Dhawan, Shivani Thakur       1
Suman Mukhopadhyay                               1
Shea Sizemore                                    1
Name: count, Length: 9672, dtype: int64:

Missing Value Count: 1938
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the id
372090    402
371744    402
371877    402
372092    202
371991    202
         ...
303381      1
303380      1
303379      1
303377      1
30459       1
Name: count, Length: 17086, dtype: int64:

Missing Value Count: 0
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the industry
Hollywood / English    14649
Bollywood / Indian      2645
Tollywood               1172
Anime / Kids            1049
Wrestling                433
Punjabi                  332
Stage shows              129
Pakistani                 92
Dub / Dual Audio          45
3D Movies                  1
Name: count, dtype: int64:

Missing Value Count: 1
```


industry Frequency Histogram
Frequency
14000
12000
10000
8000
6000
4000
2000
0
Hollywood / English
Bollywood / Indian
Tollywood
Anime / Kids
Wrestling
Punjabi
Stage shows
Pakistani
Dub / Dual Audio
3D Movies
industry

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the language
English                                 12657
Hindi                                    2558
English,Spanish                           391
Punjabi                                   310
English,Hindi                             304
                                        ...
English,Korean,Spanish                      1
Norwegian,Swedish                           1
Spanish,Chinese,English,Maori,French        1
Urdu,Punjabi,English                        1
Spanish,German,English                      1
Name: count, Length: 1167, dtype: int64:

Missing Value Count: 546
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the storyline
The life of a young man and his struggles against the privatization of education.
402
Follows\r\n a New York City family hiding out in the Hamptons whose bubble is \r
\npopped when a Bloody Mary-swilling, pot-smoking 'Charlie' comes to bring\r\n a
lifetime of hurt that might heal them all.
402
It follows Kara Robinson as she survives an abduction and ultimately brings down
a serial killer.
402
Doc\r\n facilitates a fragile truce between the Governor and Cartel, trading \r\n
prosecutorial leniency for finance. With no more truce, Doc is left to \r\nfend f
or himself and protect the one untainted thing in his life: his \r\ndaughter, Lit
tle Dixie.
202
A\r\n young, gay Black man, rejected by his mother and with few options for \r\nh
is future, decides to join the Marines, doing whatever it takes to \r\nsucceed in
a system that would cast him aside.
202


...
Four waves of increasingly deadly attacks have left most of Earth in ruin. Agains
t a backdrop of fear and distrust, Cassie is on the run, desperately trying to sa
ve her younger brother. As she prepares for the inevitable and lethal fifth wave,
Cassie teams up with a young man who may become her final hope - if she can only
trust him.
1
Yamuna along with her son Laxman locates to Mumbai leaving behind her abusive hus
band.She takes shelter in the house of her aunt Chandra whom she calls Akka.Yamun
a's only aim is to give a better education to her son.Chandra finds her a job as
sweeper in a art school.Yamuna finds that Chandra poses as a nude model to the st
udents of the school.Chandra confines Yamuna to take up the job being nude out th
ere the students don't look at you in lust but as a project.      1
A young violinist struggles to assert her individuality amidst the intense pressu
re of her pianist father, and the weight of her own musical ability.
1
A right wing talk show host's life takes a sudden turn when his 16 year old niece
comes crashing into his life.
1
While driving his car on a rainy night, Anand's car breaks down, and he goes to s
eek shelter in a nearby house. He is let into the house by the servant, and he is
permitted to stay until the rains stop be able to get his car fixed. It is here t
hat he will find out about his previous birth, his true love, Madhumati, their il
l-fated, star-crossed and tragic romance, and how events in his previous birth ar
```

```
e going to effect him in this life-time.                          1
Name: count, Length: 15748, dtype: int64:

Missing Value Count: 1701
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Frequency information of the title
The Girl Who Escaped: The Kara Robinson Story    402
Vaathi                                           402
Who Invited Charlie?                             402
Little Dixie                                     202
The Inspection                                   202
                                                ...
Kesari                                             1
Old Boys                                           1
American Exit                                      1
Adventures of Aladdin                              1
Madhumati                                          1
Name: count, Length: 16572, dtype: int64:

Missing Value Count: 1
```

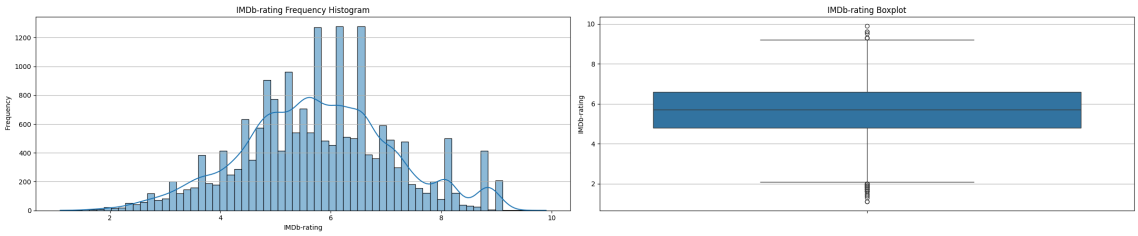
2.2 分析数值型数据

数据集中的数值型数据有：电影的IMDb评分IMDb-rating，电影的下载次数downloads，电影的时长run_time，电影的点击量views 下面我们对以上几个数值型数据逐一进行分析

In [7]:
```
analyze_freq_and_box_numeric("IMDb-rating",df["IMDb-rating"])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("downloads",df["downloads"])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("downloads",df["downloads"])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("run_time",df["run_time"])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("views",df["views"])
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

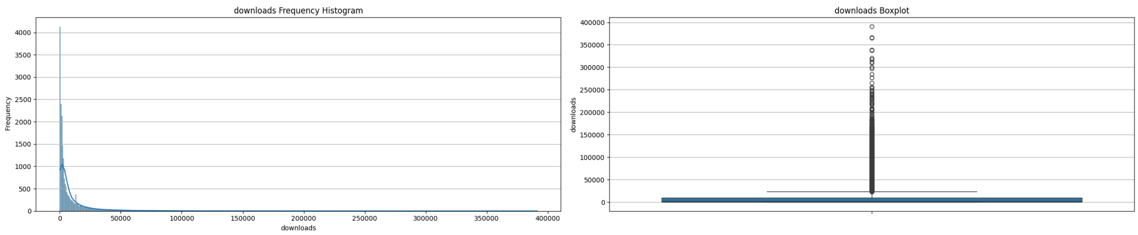
```
The Information of IMDb-rating:

count    19707.000000
mean         5.762151
std          1.374041
min          1.100000
25%          4.800000
50%          5.700000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 841
```



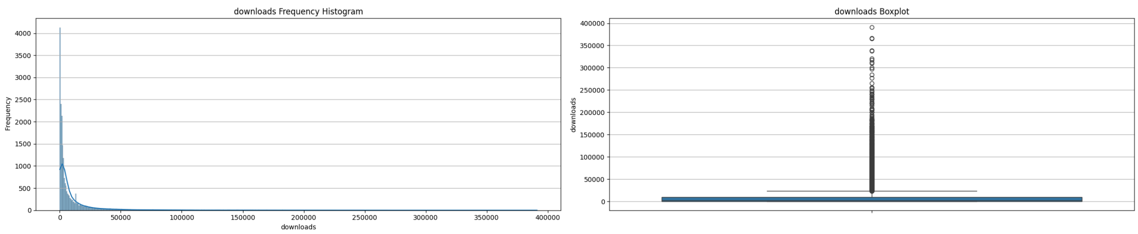

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>
The Information of downloads:

count     20547.000000
mean      10795.238916
std       23716.181987
min           0.000000
25%         855.500000
50%        2716.000000
75%       10070.000000
max      391272.000000
Name: downloads, dtype: float64
Missing Value Count: 1
```



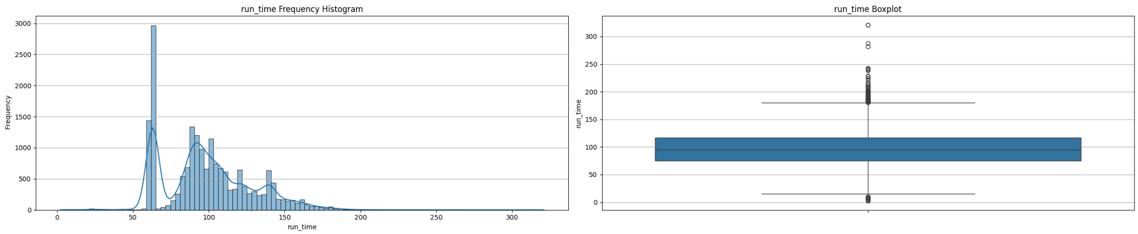
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>
The Information of downloads:

count     20547.000000
mean      10795.238916
std       23716.181987
min           0.000000
25%         855.500000
50%        2716.000000
75%       10070.000000
max      391272.000000
Name: downloads, dtype: float64
Missing Value Count: 1
```



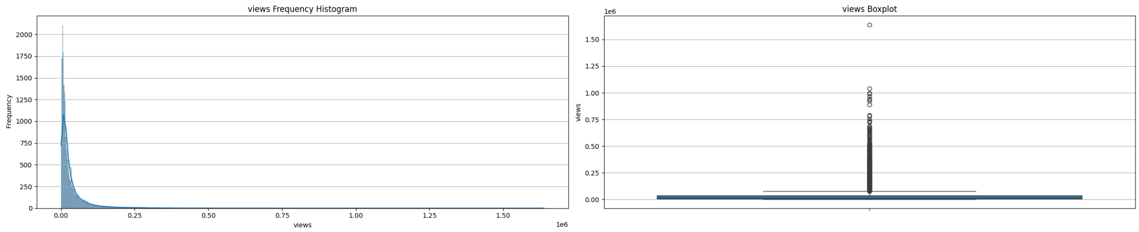

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>
The Information of run_time:

count    18780.000000
mean        98.724654
std         29.417936
min          2.000000
25%         75.000000
50%         95.000000
75%        117.000000
max        321.000000
Name: run_time, dtype: float64
Missing Value Count: 1768
```




```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>
The Information of views:

count    2.054700e+04
mean     3.559551e+04
std      6.247242e+04
min      6.670000e+02
25%      7.571500e+03
50%      1.522200e+04
75%      3.657100e+04
max      1.638533e+06
Name: views, dtype: float64
Missing Value Count: 1
```



>>>
>>>>>>>>>>>>>>>>>>>

3. 数据缺失的处理

观察数据集中缺失数据，分析其缺失的原因。分别使用下列四种策略对缺失值进行处理:

- 将缺失部分剔除
- 用最高频率值来填补缺失值
- 通过属性的相关关系来填补缺失值
- 通过数据对象之间的相似性来填补缺失值

3.1 分析数据缺失的原因

该数据集几乎所有的属性都有或多或少数量的缺失值，从前面对标称型数据和数值型数据的分析来看，二者也各自都有部分缺失值。 数据缺失的原因可能有以下几点：

- 在盗版电影网站上该电影的部分信息不完整
- 爬取电影信息的程序算法不完备

3.2 剔除有缺失值的数据对象

In [8]:
```
data = df.dropna()
```

展示填补前的数据信息

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Unnamed: 0       20548 non-null  int64  
 1   IMDb-rating      19707 non-null  float64
 2   appropriate_for  11072 non-null  object 
 3   director         18610 non-null  object 
 4   downloads        20547 non-null  float64
 5   id               20548 non-null  int64  
 6   industry         20547 non-null  object 
 7   language         20002 non-null  object 
 8   posted_date      20547 non-null  object 
 9   release_date     20547 non-null  object 
 10  run_time         18780 non-null  float64
 11  storyline        18847 non-null  object 
 12  title            20547 non-null  object 
 13  views            20547 non-null  float64
 14  writer           18356 non-null  object 
dtypes: float64(4), int64(2), object(9)
memory usage: 2.4+ MB
```

In [10]:
```
df.head(5)
```

Out[10]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language
0	0	4.8	R	John Swab	304.0	372092	Hollywood / English	English
1	1	6.4	TV-PG	Paul Ziller	73.0	372091	Hollywood / English	English
2	2	5.2	R	Ben Wheatley	1427.0	343381	Hollywood / English	English,Hindi
3	3	8.1	NaN	Venky Atluri	1549.0	372090	Tollywood	Hindi
4	4	4.6	NaN	Shaji Kailas	657.0	372089	Tollywood	Hindi

填补后的数据集信息如下：

In [11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9902 entries, 0 to 20540
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Unnamed: 0       9902 non-null   int64  
 1   IMDb-rating      9902 non-null   float64
 2   appropriate_for  9902 non-null   object 
 3   director         9902 non-null   object 
 4   downloads        9902 non-null   float64
 5   id               9902 non-null   int64  
 6   industry         9902 non-null   object 
 7   language         9902 non-null   object 
 8   posted_date      9902 non-null   object 
 9   release_date     9902 non-null   object 
 10  run_time         9902 non-null   float64
 11  storyline        9902 non-null   object 
 12  title            9902 non-null   object 
 13  views            9902 non-null   float64
 14  writer           9902 non-null   object 
dtypes: float64(4), int64(2), object(9)
memory usage: 1.2+ MB
```

In [12]:
```
data.head(5)
```

Out[12]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language
0	0	4.8	R	John Swab	304.0	372092	Hollywood / English	English
1	1	6.4	TV-PG	Paul Ziller	73.0	372091	Hollywood / English	English
2	2	5.2	R	Ben Wheatley	1427.0	343381	Hollywood / English	English,Hindi
7	7	6.5	R	Benjamin Caron	1781.0	371751	Hollywood / English	English
8	8	6.9	PG-13	Ravi Kapoor	458.0	372042	Hollywood / English	English

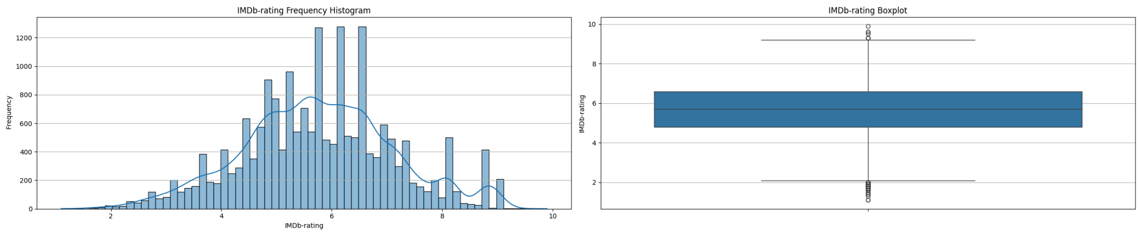
比较处理前后数据集差异

下面以IMDb-rating这一属性为例，对比数据集在剔除含缺失值数据对象后的差异

In [13]:
```
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",df["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",data["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

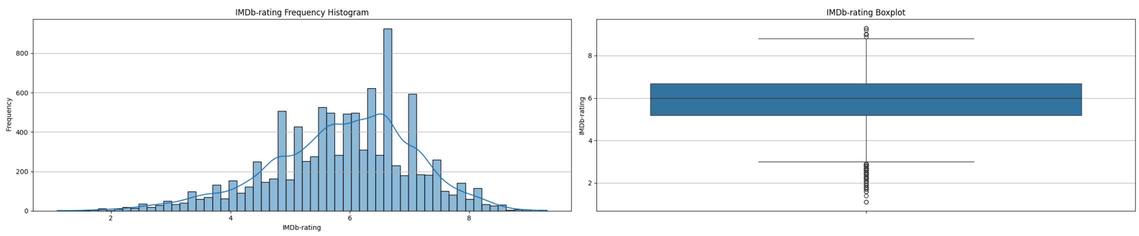
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    19707.000000
mean         5.762151
std          1.374041
min          1.100000
25%          4.800000
50%          5.700000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 841
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    9902.000000
mean        5.878489
std         1.195440
min         1.100000
25%         5.200000
50%         6.000000
75%         6.675000
max         9.300000
Name: IMDb-rating, dtype: float64
Missing Value Count: 0
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
```

3.3 用最高频率值来填补缺失值

In [14]:
```
data = df.copy(deep=True)
```

展示填补前的数据信息

In [15]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Unnamed: 0       20548 non-null  int64  
 1   IMDb-rating      19707 non-null  float64
 2   appropriate_for  11072 non-null  object 
 3   director         18610 non-null  object 
 4   downloads        20547 non-null  float64
 5   id               20548 non-null  int64  
 6   industry         20547 non-null  object 
 7   language         20002 non-null  object 
 8   posted_date      20547 non-null  object 
 9   release_date     20547 non-null  object 
 10  run_time         18780 non-null  float64
 11  storyline        18847 non-null  object 
 12  title            20547 non-null  object 
 13  views            20547 non-null  float64
 14  writer           18356 non-null  object 
dtypes: float64(4), int64(2), object(9)
memory usage: 2.4+ MB
```

In [16]:
```
data.head(5)
```

Out[16]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language
0	0	4.8	R	John Swab	304.0	372092	Hollywood / English	English
1	1	6.4	TV-PG	Paul Ziller	73.0	372091	Hollywood / English	English
2	2	5.2	R	Ben Wheatley	1427.0	343381	Hollywood / English	English,Hindi
3	3	8.1	NaN	Venky Atluri	1549.0	372090	Tollywood	Hindi
4	4	4.6	NaN	Shaji Kailas	657.0	372089	Tollywood	Hindi

In [17]:
```
for i in data.columns:
    if not data[i].isnull().any():
        continue
    data[i] = data[i].fillna(data[i].dropna().mode()[0])
```

填补后的数据集信息如下：

In [18]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Unnamed: 0       20548 non-null  int64  
 1   IMDb-rating      20548 non-null  float64
 2   appropriate_for  20548 non-null  object 
 3   director         20548 non-null  object 
 4   downloads        20548 non-null  float64
 5   id               20548 non-null  int64  
 6   industry         20548 non-null  object 
 7   language         20548 non-null  object 
 8   posted_date      20548 non-null  object 
 9   release_date     20548 non-null  object 
 10  run_time         20548 non-null  float64
 11  storyline        20548 non-null  object 
 12  title            20548 non-null  object 
 13  views            20548 non-null  float64
 14  writer           20548 non-null  object 
dtypes: float64(4), int64(2), object(9)
memory usage: 2.4+ MB
```

In [19]:
```
data.head(5)
```

Out[19]:

	Unnamed: 0	IMDb-rating	appropriate_for	director	downloads	id	industry	language
0	0	4.8	R	John Swab	304.0	372092	Hollywood / English	English
1	1	6.4	TV-PG	Paul Ziller	73.0	372091	Hollywood / English	English
2	2	5.2	R	Ben Wheatley	1427.0	343381	Hollywood / English	English,Hindi
3	3	8.1	R	Venky Atluri	1549.0	372090	Tollywood	Hindi
4	4	4.6	R	Shaji Kailas	657.0	372089	Tollywood	Hindi

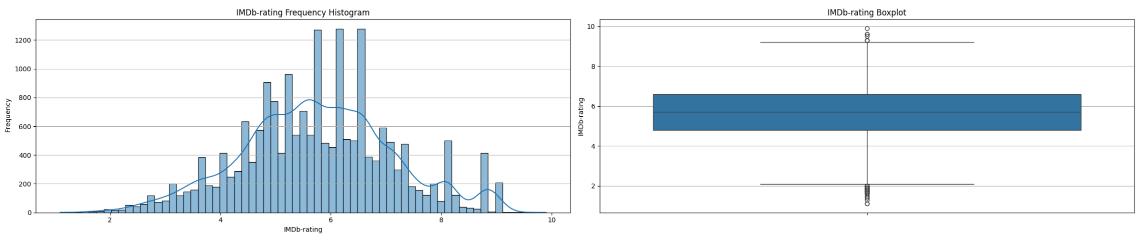
比较处理前后数据集差异

下面以IMDb-rating这一属性为例，对比数据集在剔除含缺失值数据对象后的差异

In [20]:
```
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",df["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",data["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

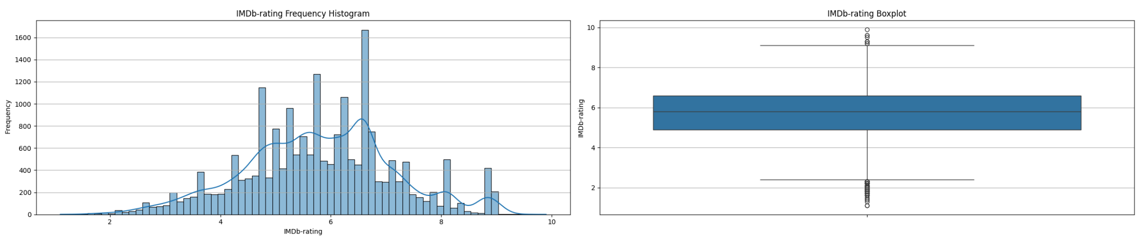
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    19707.000000
mean         5.762151
std          1.374041
min          1.100000
25%          4.800000
50%          5.700000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 841
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    20548.000000
mean         5.796442
std          1.355827
min          1.100000
25%          4.900000
50%          5.800000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 0
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
```

3.4 根据属性的相关关系填补缺失值

我们采用相关系数来判断两个属性间的相似度，并根据属性间的相似度，来寻找可替代的同类型属性，若相似度较低，则用该属性的均值填充，可填充的数值属性有"IMDb-rating"、"downloads"、"run_time"、"views"

In [21]:
```
selected_columns = ["IMDb-rating","downloads","run_time","views"]
data = df[selected_columns].copy(deep=True)
```

展示填补前的数据信息

In [22]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   IMDb-rating  19707 non-null  float64
 1   downloads    20547 non-null  float64
 2   run_time     18780 non-null  float64
 3   views        20547 non-null  float64
dtypes: float64(4)
memory usage: 642.2 KB
```

In [23]:
```
data.head(20)
```

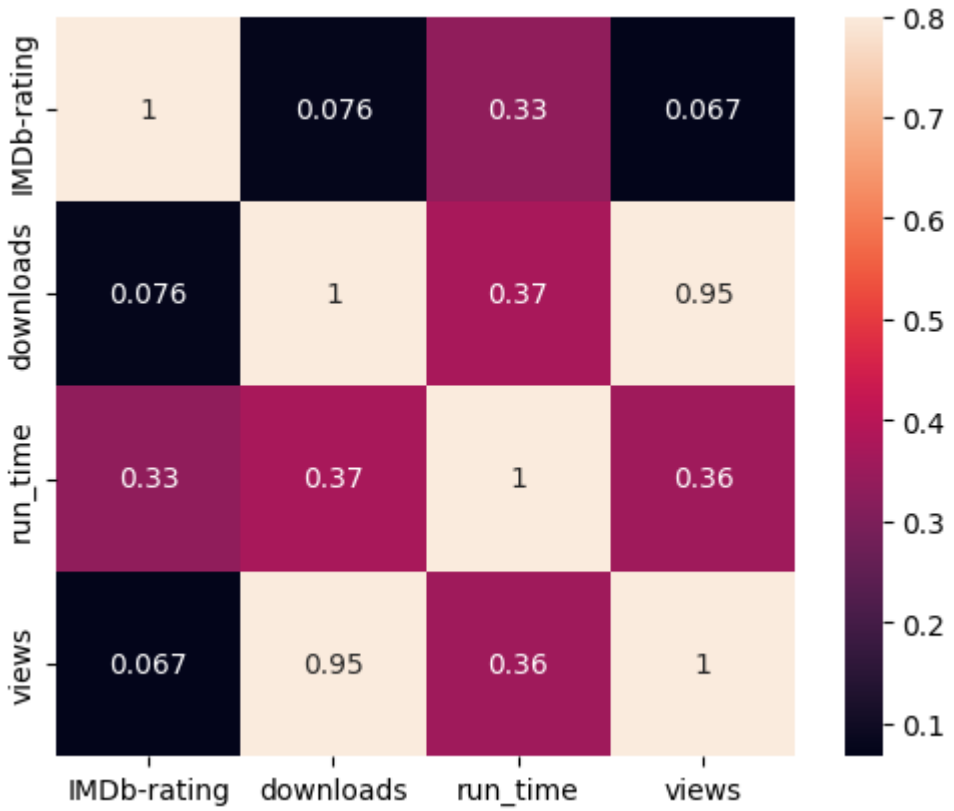
Out[23]:

	IMDb-rating	downloads	run_time	views
0	4.8	304.0	105.0	2794.0
1	6.4	73.0	84.0	1002.0
2	5.2	1427.0	64.0	14419.0
3	8.1	1549.0	139.0	4878.0
4	4.6	657.0	122.0	2438.0
5	5.4	746.0	131.0	2940.0
6	NaN	5332.0	200.0	11978.0
7	6.5	1781.0	116.0	18225.0
8	6.9	458.0	80.0	6912.0
9	4.2	1965.0	80.0	9710.0
10	6.2	742.0	NaN	4618.0
11	9.0	12954.0	142.0	35831.0
12	NaN	2253.0	NaN	5468.0
13	6.6	14867.0	125.0	39399.0
14	7.1	463.0	95.0	5763.0
15	6.5	41512.0	139.0	126897.0
16	NaN	2785.0	NaN	12968.0
17	6.4	12642.0	131.0	37664.0
18	NaN	171.0	NaN	667.0
19	4.7	1453.0	93.0	11626.0

下面以热力图的形式展示数据属性间的相关系数

In [24]:
```
sns.heatmap(data.corr(), vmax=.8, square=True, annot=True)
```

Out[24]: <Axes: >



可以看到只有downloads和views相关性较高，相关系数为0.95，所以只有这两个变量能勉强相互替代，对于其他的属性，我们用属性的均值填补

In [25]:
```
data["IMDb-rating"] = data["IMDb-rating"].fillna(data["IMDb-rating"].mean())
data["run_time"] = data["run_time"].fillna(data["run_time"].mean())
data["downloads"] = data["downloads"].fillna(data["views"])
data["views"] = data["views"].fillna(data["downloads"])
data["downloads"] = data["downloads"].fillna(data["downloads"].mean())
data["views"] = data["views"].fillna(data["views"].mean())
```

填补后的数据集部分信息如下：

In [26]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   IMDb-rating  20548 non-null  float64
 1   downloads    20548 non-null  float64
 2   run_time     20548 non-null  float64
 3   views        20548 non-null  float64
dtypes: float64(4)
memory usage: 642.2 KB
```

In [27]:
```
data.head(20)
```

Out[27]:

	IMDb-rating	downloads	run_time	views
0	4.800000	304.0	105.000000	2794.0
1	6.400000	73.0	84.000000	1002.0
2	5.200000	1427.0	64.000000	14419.0
3	8.100000	1549.0	139.000000	4878.0
4	4.600000	657.0	122.000000	2438.0
5	5.400000	746.0	131.000000	2940.0
6	5.762151	5332.0	200.000000	11978.0
7	6.500000	1781.0	116.000000	18225.0
8	6.900000	458.0	80.000000	6912.0
9	4.200000	1965.0	80.000000	9710.0
10	6.200000	742.0	98.724654	4618.0
11	9.000000	12954.0	142.000000	35831.0
12	5.762151	2253.0	98.724654	5468.0
13	6.600000	14867.0	125.000000	39399.0
14	7.100000	463.0	95.000000	5763.0
15	6.500000	41512.0	139.000000	126897.0
16	5.762151	2785.0	98.724654	12968.0
17	6.400000	12642.0	131.000000	37664.0
18	5.762151	171.0	98.724654	667.0
19	4.700000	1453.0	93.000000	11626.0

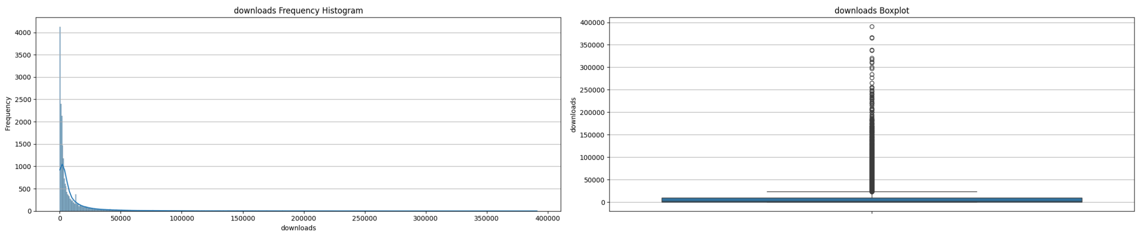
比较处理前后数据集差异

下面以downloads这一属性为例，对比数据集在剔除含缺失值数据对象后的差异

In [28]:
```
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("downloads",df["downloads"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("downloads",data["downloads"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

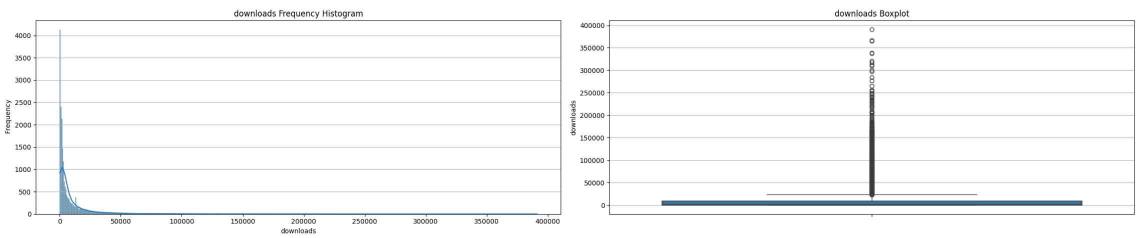
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>
The Information of downloads:

count     20547.000000
mean      10795.238916
std       23716.181987
min           0.000000
25%         855.500000
50%        2716.000000
75%       10070.000000
max      391272.000000
Name: downloads, dtype: float64
Missing Value Count: 1
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>
The Information of downloads:

count     20548.000000
mean      10795.238916
std       23715.604860
min           0.000000
25%         855.750000
50%        2716.000000
75%       10073.250000
max      391272.000000
Name: downloads, dtype: float64
Missing Value Count: 0
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
```

3.5 通过数据对象之间的相似性来填补缺失值

我们将数值属性向量化，然后使用K临近算法——KNN来计算数据对象间的距离，以此来判断数据对象间的相似性，再根据寻找的k个相似的数据对象的相关信息来填补当前对象的缺失值。可填充的数值属性有"IMDb-rating"、"downloads"、"run_time"、"views"

In [29]:
```
data = df[selected_columns].copy()
```

展示填补前的数据信息

In [30]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   IMDb-rating  19707 non-null  float64
 1   downloads    20547 non-null  float64
 2   run_time     18780 non-null  float64
 3   views        20547 non-null  float64
dtypes: float64(4)
memory usage: 642.2 KB
```

In [31]:
```
data.head(20)
```

Out[31]:

	IMDb-rating	downloads	run_time	views
0	4.8	304.0	105.0	2794.0
1	6.4	73.0	84.0	1002.0
2	5.2	1427.0	64.0	14419.0
3	8.1	1549.0	139.0	4878.0
4	4.6	657.0	122.0	2438.0
5	5.4	746.0	131.0	2940.0
6	NaN	5332.0	200.0	11978.0
7	6.5	1781.0	116.0	18225.0
8	6.9	458.0	80.0	6912.0
9	4.2	1965.0	80.0	9710.0
10	6.2	742.0	NaN	4618.0
11	9.0	12954.0	142.0	35831.0
12	NaN	2253.0	NaN	5468.0
13	6.6	14867.0	125.0	39399.0
14	7.1	463.0	95.0	5763.0
15	6.5	41512.0	139.0	126897.0
16	NaN	2785.0	NaN	12968.0
17	6.4	12642.0	131.0	37664.0
18	NaN	171.0	NaN	667.0
19	4.7	1453.0	93.0	11626.0

调用KNN的包来对数据进行填补，我们选择k=2作为参数

In [32]:
```
from sklearn.impute import KNNImputer
data = df[selected_columns].copy()
knn_imputer = KNNImputer(n_neighbors=2)

df_imputed = knn_imputer.fit_transform(data)

data = pd.DataFrame(df_imputed, columns=data.columns)
```

展示填补后的数据信息

In [33]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20548 entries, 0 to 20547
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   IMDb-rating  20548 non-null  float64
 1   downloads    20548 non-null  float64
 2   run_time     20548 non-null  float64
 3   views        20548 non-null  float64
dtypes: float64(4)
memory usage: 642.2 KB
```

In [34]:
```
data.head(20)
```

Out[34]:

	IMDb-rating	downloads	run_time	views
0	4.80	304.0	105.0	2794.0
1	6.40	73.0	84.0	1002.0
2	5.20	1427.0	64.0	14419.0
3	8.10	1549.0	139.0	4878.0
4	4.60	657.0	122.0	2438.0
5	5.40	746.0	131.0	2940.0
6	4.65	5332.0	200.0	11978.0
7	6.50	1781.0	116.0	18225.0
8	6.90	458.0	80.0	6912.0
9	4.20	1965.0	80.0	9710.0
10	6.20	742.0	62.5	4618.0
11	9.00	12954.0	142.0	35831.0
12	7.00	2253.0	101.0	5468.0
13	6.60	14867.0	125.0	39399.0
14	7.10	463.0	95.0	5763.0
15	6.50	41512.0	139.0	126897.0
16	5.05	2785.0	94.5	12968.0
17	6.40	12642.0	131.0	37664.0
18	5.95	171.0	73.0	667.0
19	4.70	1453.0	93.0	11626.0

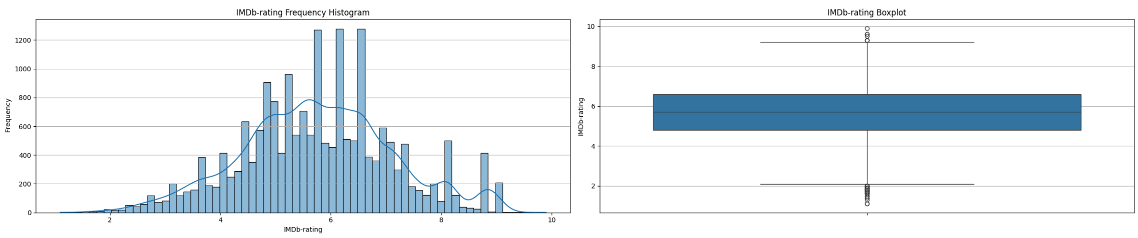
比较处理前后数据集差异

下面以IMDb-rating这一属性为例，对比数据集在剔除含缺失值数据对象后的差异

In [35]:
```
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",df["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
print(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>
analyze_freq_and_box_numeric("IMDb-rating",data["IMDb-rating"])
print("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

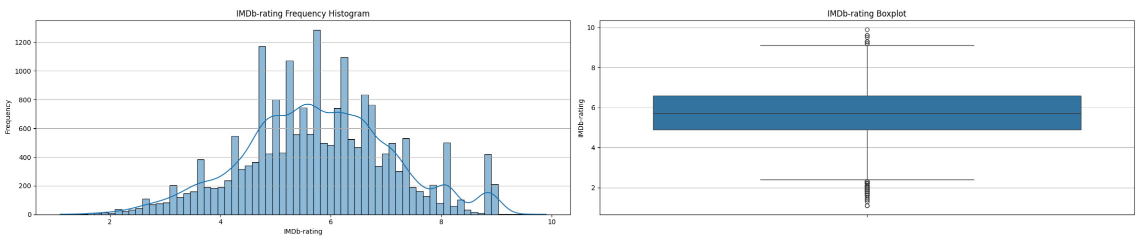
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Before Processing>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    19707.000000
mean         5.762151
std          1.374041
min          1.100000
25%          4.800000
50%          5.700000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 841
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>After Processing>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>
The Information of IMDb-rating:

count    20548.000000
mean         5.767924
std          1.362141
min          1.100000
25%          4.900000
50%          5.700000
75%          6.600000
max          9.900000
Name: IMDb-rating, dtype: float64
Missing Value Count: 0
```




```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<
```

总结

至此，我们对Movies Dataset from Pirated Sites数据集的预处理和探索性分析全部完成