IMDb Movie data using python web-scrapping

Author: D.Sathyakala,

Data analyst Intern @ Senchola Technology Solutions

Web-scrapping

Import library files

In [412…
```
import pandas as pd
from bs4 import BeautifulSoup as bs
import requests
import re
```

Steps

In [430…
```
try:
    # loading url of top 250 movies of IMBb list
    user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0

    # url of the IMDb website
    url = 'https://www.imdb.com/chart/top/?ref_=nv_mv_250'

    # Get the url from requests library
    response = requests.get(url,headers = {'User-Agent':user_agent,'Accept-Language' : 'en-US,en;q=0.5'})
    print(response)

    # Souping and arranging the html code
    soup = bs(response.text,'html.parser')

    # Select table from the soup
    movies = soup.findAll('div', attrs = {'class':'sc-c7e5f54-0 gytZrF cli-children'})
    #print(movies)

    # Select the required coulmn for the dataframe
    movie_name = []
    year = []
    time = []
    rating = []
    rating_type = []
    votes = []

    # Taking each records and building our dataframe
    for source in movies:
        # Get movie name
        name = source.div.a.text
        movie_name.append(name)

        # Get year of release
        year1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').text
        year.append(year1)

        # Get duration of movie
        time1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').find_next('span
        time.append(time1)

        # Get rating type of movie
        rating_type1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').find_nex
        rating_type.append(rating_type1)

        # Get voting rates of movie
        vote1 = source.find('span',class_="ipc-rating-star--voteCount").text.replace('\xa0(','').replace(')','
        votes.append(vote1)

        # Get the rating of movie
        rating1 = source.find('div',class_="sc-e3e7b191-0 iKUUVe sc-c7e5f54-2 hCiLPi cli-ratings-container").te
        rating.append(rating1)
except Exception as e:
    print(e)
```

```
<Response [200]>
```

In [431…
```
# Put all details into the Dataframe
df = pd.DataFrame({'Movie Name':movie_name, 'Duration':time, 'Released year':year, 'Rating': rating, 'Rated typ
df
```

Out[431…

	Movie Name	Duration	Released year	Rating	Rated type	Votes
0	1. The Shawshank Redemption	2h 22m	1994	9.3	R	2.8M
1	2. The Godfather	2h 55m	1972	9.2	R	2M
2	3. The Dark Knight	2h 32m	2008	9.0	PG-13	2.8M
3	4. The Godfather Part II	3h 22m	1974	9.0	R	1.3M
4	5. 12 Angry Men	1h 36m	1957	9.0	Approved	839K
...	...	...	...	...	...	...
245	246. The 400 Blows	1h 39m	1959	8.1	Not Rated	125K
246	247. Persona	1h 23m	1966	8.1	Not Rated	128K
247	248. Aladdin	1h 30m	1992	8.0	G	452K
248	249. Life of Brian	1h 34m	1979	8.0	R	416K
249	250. Dances with Wolves	3h 1m	1990	8.0	PG-13	283K

250 rows × 6 columns

Here I will modify the column Movie Name, Duration, Votes in proper way

In [432…
```
# Convert duration (hours) into minutes
def convert_to_minutes(runtime):
    total_minutes = 0

    parts = runtime.split()
    for part in parts:
        if 'h' in part:
            total_minutes += int(part.replace('h','')) * 60
        elif 'm' in part:
            total_minutes += int(part.replace('m',''))
    return total_minutes
```

In [433…
```
# Convert viewers count (Mega, Kilo) into whole value
def convert_viewer_count(viewer_count_str):
     multiplier = 1
     if 'M' in viewer_count_str:
         multiplier = 1e6
     elif 'K' in viewer_count_str:
         multiplier = 1e3
     return int(float(viewer_count_str.replace('M', '').replace('K', '')) * multiplier)
```

In [434…
```
try:
    # loading url of top 250 movies of IMBb list
    user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0

    # url of the IMDb website
    url = 'https://www.imdb.com/chart/top/?ref_=nv_mv_250'

    # Get the url from requests library
    response = requests.get(url,headers = {'User-Agent':user_agent,'Accept-Language' : 'en-US,en;q=0.5'})
    print(response)

    # Souping and arranging the html code
    soup = bs(response.text,'html.parser')

    # Select table from the soup
    movies = soup.findAll('div', attrs = {'class':'sc-c7e5f54-0 gytZrF cli-children'})
    #print(movies)

    # Select the required coulmn for the dataframe
    rank = []
    movie_name = []
    year = []
    time = []
    rating = []
    rating_type = []
    votes = []

    # Taking each records and building our dataframe
    for source in movies:
        name = source.div.a.text

        # Get the rank of the movie
        rank1 = int(name.split('. ')[0])

        # Get name of the movie
```

```
        name1 = name.split('. ')[1]
        rank.append(rank1)
        movie_name.append(name1)

        # Get year of release
        year1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').text
        year.append(year1)

        # Get duration of movie in minutes
        time1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').find_next('span
        time1 = convert_to_minutes(time1)
        time.append(time1)

        # Get rating type of movie
        rating_type1 = source.find('div',class_="sc-c7e5f54-7 brlapf cli-title-metadata").find('span').find_nex
        rating_type.append(rating_type1)

         # Get voting rates of movie in whole number
        vote1 = source.find('span',class_="ipc-rating-star--voteCount").text.replace('\xa0(','').replace(')','
        vote1 = convert_viewer_count(vote1)
        votes.append(vote1)

        # Get the rating of movie
        rating1 = source.find('div',class_="sc-e3e7b191-0 iKUUVe sc-c7e5f54-2 hCiLPi cli-ratings-container").te
        rating.append(rating1)

except Exception as e:
    print(e)
```

```
<Response [200]>
```

In [435…
```
# Put all details into the Dataframe
df1 = pd.DataFrame({'Rank':rank, 'Movie Name':movie_name, 'Year':year, 'Duration':time, 'Rating':rating, 'Ratin
df1
```

Out[435…

	Rank	Movie Name	Year	Duration	Rating	Rating type	Vote
0	1	The Shawshank Redemption	1994	142	9.3	R	2800000
1	2	The Godfather	1972	175	9.2	R	2000000
2	3	The Dark Knight	2008	152	9.0	PG-13	2800000
3	4	The Godfather Part II	1974	202	9.0	R	1300000
4	5	12 Angry Men	1957	96	9.0	Approved	839000
...	...	...	...	...	...	...	...
245	246	The 400 Blows	1959	99	8.1	Not Rated	125000
246	247	Persona	1966	83	8.1	Not Rated	128000
247	248	Aladdin	1992	90	8.0	G	452000
248	249	Life of Brian	1979	94	8.0	R	416000
249	250	Dances with Wolves	1990	181	8.0	PG-13	283000

250 rows × 7 columns

In [436…
```
# Dounload the dataFrame into csv file
df1.to_csv('IMDb dataset.csv',index = False, encoding = 'latin')
```

Data Exploration

In [437…
```
# Load the dataset
data = pd.read_csv("IMDb dataset.csv",encoding="latin")
data
```

Out[437…

	Rank	Movie Name	Year	Duration	Rating	Rating type	Vote
0	1	The Shawshank Redemption	1994	142	9.3	R	2800000
1	2	The Godfather	1972	175	9.2	R	2000000
2	3	The Dark Knight	2008	152	9.0	PG-13	2800000
3	4	The Godfather Part II	1974	202	9.0	R	1300000
4	5	12 Angry Men	1957	96	9.0	Approved	839000
...	...	...	...	...	...	...	...
245	246	The 400 Blows	1959	99	8.1	Not Rated	125000
246	247	Persona	1966	83	8.1	Not Rated	128000
247	248	Aladdin	1992	90	8.0	G	452000
248	249	Life of Brian	1979	94	8.0	R	416000
249	250	Dances with Wolves	1990	181	8.0	PG-13	283000

250 rows × 7 columns

In [438…
```
# Display all the details of dataframe
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Rank         250 non-null    int64
 1   Movie Name   250 non-null    object
 2   Year         250 non-null    int64
 3   Duration     250 non-null    int64
 4   Rating       250 non-null    float64
 5   Rating type  250 non-null    object
 6   Vote         250 non-null    int64
dtypes: float64(1), int64(4), object(2)
memory usage: 13.8+ KB
```

In [439…
```
# Display top 5 rows of the dataset
data.head(5)
```

Out[439…

	Rank	Movie Name	Year	Duration	Rating	Rating type	Vote
0	1	The Shawshank Redemption	1994	142	9.3	R	2800000
1	2	The Godfather	1972	175	9.2	R	2000000
2	3	The Dark Knight	2008	152	9.0	PG-13	2800000
3	4	The Godfather Part II	1974	202	9.0	R	1300000
4	5	12 Angry Men	1957	96	9.0	Approved	839000

In [440…
```
# Display bottom 5 rows of the dataset
data.tail(5)
```

Out[440…

	Rank	Movie Name	Year	Duration	Rating	Rating type	Vote
245	246	The 400 Blows	1959	99	8.1	Not Rated	125000
246	247	Persona	1966	83	8.1	Not Rated	128000
247	248	Aladdin	1992	90	8.0	G	452000
248	249	Life of Brian	1979	94	8.0	R	416000
249	250	Dances with Wolves	1990	181	8.0	PG-13	283000

In [441…
```
# Display size of the dataset
data.shape
```

Out[441… (250, 7)

In [442…
```
# Display the statistics of the dataset
data.describe()
```

Out[442…

	Rank	Year	Duration	Rating	Vote
count	250.000000	250.000000	250.000000	250.000000	2.500000e+02
mean	125.500000	1986.716000	129.108000	8.306400	6.759520e+05
std	72.312977	25.324785	30.002549	0.232757	5.435674e+05
min	1.000000	1921.000000	45.000000	8.000000	3.700000e+04
25%	63.250000	1966.250000	107.250000	8.100000	2.322500e+05
50%	125.500000	1994.000000	126.500000	8.200000	5.405000e+05
75%	187.750000	2007.000000	145.750000	8.400000	9.920000e+05
max	250.000000	2023.000000	238.000000	9.300000	2.800000e+06

In [443…
```
# Display column name of the dataset
data.columns
```

Out[443…
```
Index(['Rank', 'Movie Name', 'Year', 'Duration', 'Rating', 'Rating type',
       'Vote'],
      dtype='object')
```

In [444…
```
# Check the duplicate data
data.duplicated().sum()
```

Out[444…
```
0
```

In [445…
```
# Identify the null values
data.isna()
```

Out[445…

	Rank	Movie Name	Year	Duration	Rating	Rating type	Vote
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
245	False	False	False	False	False	False	False
246	False	False	False	False	False	False	False
247	False	False	False	False	False	False	False
248	False	False	False	False	False	False	False
249	False	False	False	False	False	False	False

250 rows × 7 columns

In [446…
```
# Identify the null values
data.isna().sum()
```

Out[446…
```
Rank           0
Movie Name     0
Year           0
Duration       0
Rating         0
Rating type    0
Vote           0
dtype: int64
```

In [ ]:

In [ ]:

In [ ]: