web scraping

IMDb "Top 250"
+
rotten tomatoes

Matthis.assalier

+

Marianne D.

+

Eden_Lecarpentie

+

Imen Fehri

+

Nidhal

Plan

- Scraping imDB250
- Exporter le data frame
- Utiliser le fichier scrapy _ final.py
- Scraping Rotten Tomatoes
- Extraire la base de données finale
- Visualisation

scraping imDB250

```
fine header language for getting data in english
ers = {"Accept-Language": "en-US,en;q=0.5"}
```

```
op all the page baby
all_page_link(start_url):
all_urls = []
url = start_url
while(url != None):
    all_urls.append(url)
    soup = BeautifulSoup(requests.get(url).text,"html.parser")
    next_links = soup.find_all(class_='flat-button lister-page-next next-page')
    if (len(next_links) == 0):
        url = None
    else:

        next_page = "https://www.imdb.com" + next_links[0].get('href')
        url = next_page
return all_urls
```

```
_array = []
url in tqdm(all_page_link("https://www.imdb.com/list/ls068082370")):
soup = BeautifulSoup(requests.get(url,headers=headers).text,"html.parser")
for link in soup.find_all(class_='lister-item-content'):
```

Exporter le data frame

```
df
```

]:

	name	year	runtime	genre
0	The Shawshank Redemption	1994	142	Drama
1	The Godfather	1972	175	Crime, Drama
2	The Dark Knight	2008	152	Action, Crime, Drama
3	The Godfather: Part II	1974	202	Crime, Drama
4	Pulp Fiction	1994	154	Crime, Drama
...	...	...	...	...
245	Nausicaä of the Valley of the Wind	1984	117	Animation, Adventure, Fantasy
246	The Maltese Falcon	1941	100	Crime, Film-Noir, Mystery
247	Persona	1966	83	Drama, Thriller
248	The Grapes of Wrath	1940	129	Drama
249	Jaws	1975	124	Adventure, Thriller

250 rows × 4 columns

```
df.to_csv("movie.csv", index=False, encoding='utf-8')
```

Scraping Rotten Tomatoes

```python
#Permet de créer la BeatifulSoup à partir d'une URL et avec les paramètres que l'on souhaite en headers
def soup_url(url,headers={'User-Agent': 'python-requests/2.25.1',
                          'Accept-Encoding': 'gzip, deflate',
                          'Accept': '*/*', 'Connection': 'keep-alive'}):
        req = requests.get(url,headers=headers)
        soup = BeautifulSoup(req.text, "lxml")
        return soup
```

```python
df['tomatometer'] = None
df['audience_score'] = None
df['url_tomatoes'] = None
```

```python
def fetch_film_rt(df):
    #Pour chaque ligne du DataFrame on va effectuer une recherche à partir du titre du film
    for ligne in range(len(df)):
        recherche = df['titre'].iloc[ligne]
        annee = df['annee'].iloc[ligne]
        recherche = recherche.replace(" ","%20")
        rt2_url = f"https://www.rottentomatoes.com/search?search={recherche}"
        soup2 = soup_url(rt2_url)
        #Sélection du bon film à enregistrer selon le match avec sa date de sortie
        if soup2.find_all("search-page-media-row",{"releaseyear":annee}):
            new_url=soup2.find_all("search-page-media-row",{"releaseyear":annee})[0]("a")[0]["href"]
            df['url_tomatoes'].iloc[ligne] = new_url
            #On crée une nouvelle requete qui va sur la page du film visé
            soup3 = soup_url(new_url)
            df["tomatometer"].iloc[ligne] = soup3.find('score-board')['tomatometerscore']
            df["audience_score"].iloc[ligne] = soup3.find('score-board')['audiencescore']
    return df
```

scrapping sur rottentomatoes

Nous avons fait du scrapping sur :
le tomatometer et le audiencescore

Se sont 2 scores qui permettent d'évaluer la qualité d'un film

Ensuite on essayer de récupérer les données sur le Top des films actuellement au cinéma

Scrapping en utilisant json

Pour avoir accès a certaine donnée sur le site de rottentomatoes on a du utilisé du json .
Les données des noms des films sur se site web était rendue difficilement accessible. En utilisant l'inspecteur on tomber sur les sois dissentes données mai rien était retourner .On a du chercher dans tout le html notamment les script pour trouver les vrais donner

```
import json
temp = json.loads(soup.find('script', type='application/ld+json').contents[0])

urls = []
for i,v in enumerate(temp.get('itemListElement')):
    url = v.get('url')
    urls.append(url)
urls
df = pd.DataFrame({'Current_Top_Box_Office':urls})
df
```

Exporter le data frame

```
df2
```

Out[103]:

	titre	annee	duree	genre	tomatometer	audience_score	url_tomatoes
0	The Shawshank Redemption	1994	142	Drama	91	98	https://www.rottentomatoes.com/m/shawshank_red...
1	The Godfather	1972	175	Crime, Drama	97	98	https://www.rottentomatoes.com/m/godfather
2	The Dark Knight	2008	152	Action, Crime, Drama	94	94	https://www.rottentomatoes.com/m/the_dark_knight
3	The Godfather: Part II	1974	202	Crime, Drama	96	97	https://www.rottentomatoes.com/m/godfather_par...
4	12 Angry Men	1957	96	Crime, Drama	100	97	https://www.rottentomatoes.com/m/1000013_12_an...
5	The Lord of the Rings: The Return of the King	2003	201	Action, Adventure, Drama	93	86	https://www.rottentomatoes.com/m/the_lord_of_t...
6	Pulp Fiction	1994	154	Crime, Drama	92	96	https://www.rottentomatoes.com/m/pulp_fiction
7	Schindler's List	1993	195	Biography, Drama, History	98	97	https://www.rottentomatoes.com/m/schindlers_list
8	Inception	2010	148	Action, Adventure, Sci-Fi	87	91	https://www.rottentomatoes.com/m/inception
9	Fight Club	1999	139	Drama	79	96	https://www.rottentomatoes.com/m/fight_club

Entrée [ ]:

```
df2.to_csv('top_film_imdb_rt.csv', index = False, header=True,sep='|')
```

Utiliser le fichier scrapy_final.py

- A executer
- 2 fonctions : fetch_film_IMDb() & fetch_film_rt(df)

Visualisation

TOP 250 film

Liste des films

Titre de film

- 12 Angry Men
- 12 Years a Slave
- 1917
- 2001: A Space Odyssey
- 3 Idiots
- A Beautiful Mind
- A Clockwork Orange
- A Separation
- A Silent Voice: The Movie
- Alien
- Aliens
- All About Eve
- Amadeus
- Amélie
- American Beauty
- American History X
- Andhadhun
- Andrei Rublev
- Apocalypse Now
- Autumn Sonata
- Avengers: Endgame
- Avengers: Infinity War
- Back to the Future
- Barry Lyndon

annee

1921 2021

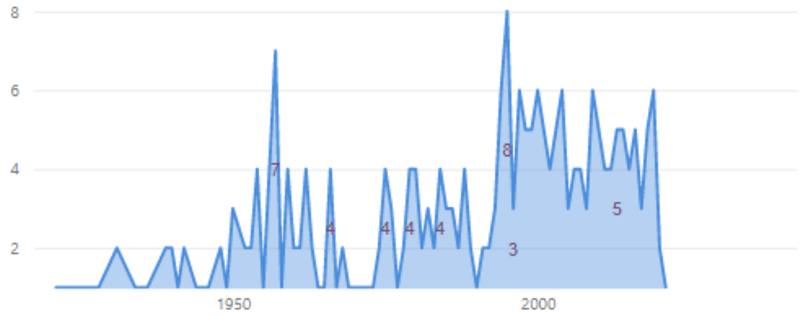
AUDIENCE SCORE

TOMATOMETER

92,5
Moyenne de audience_score

duree des films par genre

Nombre de film par année



genre des film