

WebCrawler

Websites to be consumed.

For this NLP prototype as stated previously i have chosen a content-based movie recommendation and have scrapped the top 1000 movies on IMDb's website (https://www.imdb.com/search/title/?groups=top_1000&sort=user_rating,desc). I decided to leave this at only the top 1000 movies as this turned out to be a good starting base and not overload IMDb which is a popular website. This website was chosen for its large database of movies giving both titles, genres, descriptions, directions, and ratings all on a singular page rather than scraping multiple websites this provided the information needed to complete the task provided. I had found other sites that I was considering like Rotten Tomatoes or Metacritic however these websites did not provide the data that I found was a sufficient choice to be evaluated.

Rationale for chosen Data.


As stated, before the website was chosen for its large dataset containing all the information that needed to be scraped was provided all on a few webpages. This had made it easier to scrape a singular website rather than multiple as well as IMDb allowing their website to be scraped for data being in accordance with copyright was also a deciding factor. However, if this were to be used publicly, we would need to give the rights to IMDb's team.


Content Coverages of the Data

IMDb "Top 1000" (Sorted by IMDb Rating Descending)

1-100 of 1,000 titles. | [Next »](#) View Mode: [Compact](#) | [Detailed](#)

Sort by: [Popularity](#) | [A-Z](#) | [User Rating ▼](#) | [Number of Votes](#) | [US Box Office](#) | [Runtime](#) | [Year](#) | [Release Date](#) | [Date of Your Rating](#) | [Your Rating](#)



1. [The Shawshank Redemption](#) (1994) 


MA | 142 min | Drama


★ **9.3** ☆ [Rate this](#) **80** Metascore

Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.

Director: [Frank Darabont](#) | Stars: [Tim Robbins](#), [Morgan Freeman](#), [Bob Gunton](#), [William Sadler](#)

Votes: **2,397,913** | Gross: **\$28.34M**



2. [The Godfather](#) (1972) 


R | 175 min | Crime, Drama


★ **9.2** ☆ [Rate this](#) **100** Metascore

An organized crime dynasty's aging patriarch transfers control of his clandestine empire to his reluctant son.

Director: [Francis Ford Coppola](#) | Stars: [Marlon Brando](#), [Al Pacino](#), [James Caan](#), [Diane Keaton](#)

Votes: **1,660,744** | Gross: **\$134.97M**



3. [Soorai Pottru](#) (2020) 

153 min | Drama

★ **9.1** ☆ [Rate this](#)

Nedumaaran Rajangam "Maara" sets out to make the common man fly and in the process takes on the world's most capital intensive industry and several enemies who stand in his way.

Director: [Sudha Kongara](#) | Stars: [Suniya](#), [Madhavan](#), [Paresh Rawal](#), [Aparna Balamurali](#)

Votes: **80,317**

A Picture detailed the website and how it functions.



1. [The Shawshank Redemption](#) (1994) 

MA | 142 min | Drama

★ **9.3** ☆ [Rate this](#) **80** Metascore

Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.

Director: [Frank Darabont](#) | Stars: [Tim Robbins](#), [Morgan Freeman](#), [Bob Gunton](#), [William Sadler](#)

Votes: **2,397,913** | Gross: **\$28.34M**

As Seen in the Photo this was what was chosen for the WebCrawler to scrape but this could definitely be further expanded on scraping the Actors, votes and gross of the movie to further expand the dataset and improve accuracy but for the prototype I felt this to be unnecessary. The python packages that were used for the web scraping was the “Beautiful Soup” package. The BeautifulSoup is what was used for extracting and saving the data from the page chosen and had given the web scraper the ability to go through all the pages until the end of the top 1000 as IMDb would only display 250 at a time. I also expanded on the use of tqdm a progress bar which would display how long the web scraper would take to scrape the 1000 movies.

```
url = "https://www.imdb.com/search/title/?groups=top_1000&sort=user_rating,desc"
```

```
def 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_='listner-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
```

The start of the code showcasing its ability to go through all the pages.

```
main_array = []
for url in tqdm(all_page_link("https://www.imdb.com/search/title/?groups=top_1000&sort=user_rating,desc")):
    soup = BeautifulSoup(requests.get(url).text,"html.parser")
    for link in soup.find_all(class_='listner-item-content'):
        name = link.find('a').text
        year = years = link.h3.find('span', class_='listner-item-year').text
        director=link.find('p',class_='').find_all('a')[0].text
        about = link.find_all('p')[1].text[5:]
        rating = imdb = float(link.strong.text)
        run_time = link.find('span',{'class':"runtime"}).text
        genre = link.find('span',{'class':"genre"}).text[1:]
        list_of_all = [name,year,run_time,genre,director,about,rating]
        main_array.append(list_of_all)
```

100%|██████████| 20/20 [00:42<00:00, 2.12s/it]

An example of the scraper displaying the HTML structure of the website and how it needs to search through the containers. Also displaying the progress bar.

Copyright Considerations

After a bit of googling i found there that IMDB was okay with people scraping their website in small amounts however if this is further expanded on you would apply for an API key and have their permission to scrape their large dataset. Moreover, IMDb does provide a few datasets which could also help expand this own dataset providing reviews and user ids to do a collaborative based recommendation system.

Metadata supplementation

For this prototype, I decided that 1000 movies would be more than sufficient for this task however i did spend time debating on whether the choice of using IMDb's on dataset to help supplement my own but decided against due to the sheer size of the ones provided but, in the future, this would definitely be a route that could be chosen.

Demonstration of WebCrawler

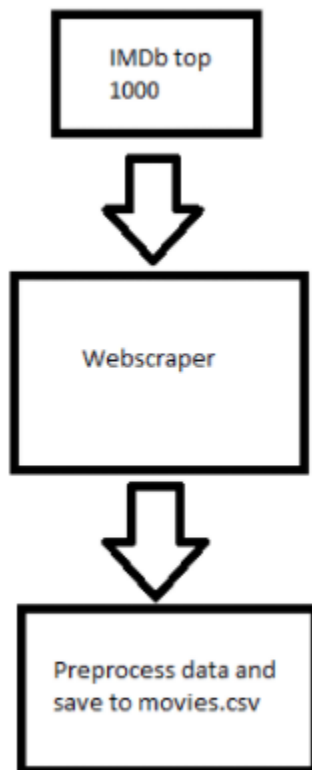
```
main_array = []
for url in tqdm(all_page_link("https://www.imdb.com/search/title/?groups=top_1000&sort=user_rating,desc")):
    soup = BeautifulSoup(requests.get(url).text,"html.parser")
    for link in soup.find_all(class_='list-item-content'):
        name = link.find('a').text
        year = years = link.h3.find('span', class_='list-item-year').text
        director=link.find('p',class_='').find_all('a')[0].text
        about = link.find_all('p')[1].text[5:]
        rating = imdb = float(link.strong.text)
        run_time = link.find('span',{'class':"runtime"}).text
        genre = link.find('span',{'class':"genre"}).text[1:]
        list_of_all = [name,year,run_time,genre,director,about,rating]
        main_array.append(list_of_all)
```

5% | 1/20 [00:01<00:37, 1.97s/it]

```
main_array = []
for url in tqdm(all_page_link("https://www.imdb.com/search/title/?groups=top_1000&sort=user_rating,desc")):
    soup = BeautifulSoup(requests.get(url).text,"html.parser")
    for link in soup.find_all(class_='list-item-content'):
        name = link.find('a').text
        year = years = link.h3.find('span', class_='list-item-year').text
        director=link.find('p',class_='').find_all('a')[0].text
        about = link.find_all('p')[1].text[5:]
        rating = imdb = float(link.strong.text)
        run_time = link.find('span',{'class':"runtime"}).text
        genre = link.find('span',{'class':"genre"}).text[1:]
        list_of_all = [name,year,run_time,genre,director,about,rating]
        main_array.append(list_of_all)
```

30% | 6/20 [00:11<00:26, 1.92s/it]

Processing, Cleaning, and storing



Summary and visualisation of the harvested data

	movie	year	run_time	genre	director	about	rating
0	The Shawshank Redemption	1994	142 min	Drama	Frank Darabont	Two imprisoned men bond over a number of years...	9.3
1	The Godfather	1972	175 min	Crime Drama	Francis Ford Coppola	An organized crime dynasty's aging patriarch L...	9.2
2	Soorai Pottu	2020	153 min	Drama	Sudha Kongara	Nedumaaran Rajangam "Maara" sets out to make L...	9.1
3	The Dark Knight	2008	152 min	Action Crime Drama	Christopher Nolan	When the menace known as the Joker wreaks havo...	9.0
4	The Godfather: Part II	1974	202 min	Crime Drama	Francis Ford Coppola	The early life and career of Vito Corleone in ...	9.0

The final table shown is the product of the web scraper after having been slightly cleaned.