

Webscrapping final project “fimweb scrapers”



UNIwersytet Warszawski
Wydział Nauk Ekonomicznych

Prepared by:

- Aleksander Partyga (435032)
- Adam Dudek (436353)

Table of contents











Short description of the topic and the web page.....	2
Beautiful Soup- why did we omit	2
Short description of our scraper mechanics	3
• Selenium.....	3
• Scrapy	3
Short technical description of the output we get.	3
Elementary data analysis.....	4
Graph.....	4
Summary of data	4
Mean rating summary	5
Running time comparison	5
Detailed description which participant wrote which part of the project.	5

Short description of the topic and the web page.

For the purpose of our project we scraped data from the filmweb.pl webpage about the most popular TV series in Poland. We scraped data of around 1000 most popular TV Series among polish people based on the filmweb's popularity ranking. Filmweb is an online database that stores information about different movies, series, actors and film crew personnel. It is the most known and the largest Polish film database where many users share their feedback and rate movies according to their opinions.

Beautiful Soup- why did we omit

Filmweb.pl is a dynamic website since users interact with it on a daily basis. Each user can have a unique account with a unique set of characteristics (different group of favourite movies that user saved as favourite). On top of that users' rate movies and TV series every day so the information presented on this website will change very often. Consequently, the webpage has to be dynamic to adapt the changes on the fly. To prove that from the technical point of view we present a screenshot from the Wappalyzer (browser extension), which provides a set of technologies used to build the filmweb webpage. One of the technologies used to build Filmweb is Varnish – a program which is “HTTP accelerator designed for content-heavy dynamic web sites ”

Widżet	Narzędzia cache
 Facebook	 Varnish
Statystyki	Język programowania
 Google Analytics	 Java
 Gemius	Menedżer tagów
Skrypt czcionek	 Google Tag Manager
 Google Font API	Cookie compliance ⓘ
Framework webowy	 Didomi
 Google Web Toolkit	Social logins
	 Facebook Login

Short description of our scraper mechanics

- **Selenium:**

Firstly, after we imported the Selenium library, we define an instance of a webdriver class that will perform operations, written in the `filmweb_selenium.py` script. We pass the url of the web page with the most popular TV series as the argument to the `get` function, to tell where the driver should perform its actions. The last character in the url is a variable which changes in every iteration and corresponds to the number of the webpage.

Later, we find a list of series according to their class names defined in the HTML code. From every single series, we extract the information about the title, year, rate and votes once again finding the elements either by class name or xpath.

- **Scrapy:**

At the beginning we created a new scrapy project called "scrapy startproject filmweb_spider02" in bash.

In python code we imported 3 libraries: Scrapy, csv and pandas. Afterwards created a class of our scrapy spider named 'filmweb_spider02'. Similarly as in selenium scraper we passed the url of the web page with the most popular TV series to tell where the driver should perform its actions. We scrap title of movie, year of its production, rate and number of votes from all movies from each website by providing our scraper html path to all of these elements. After each website our scraper click 'next page' button and goes to next page to scrap similar information. We scrape like this 100 pages. At the end we save scraped data to csv file.

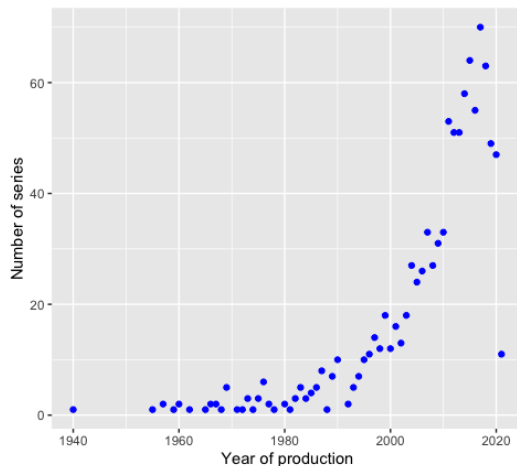
Short technical description of the output we get.

The output of the scraping process is a data frame that contains 1000 rows (each of the row describes characteristics of particular series) and 4 columns (characteristics: title of series, year - when the first episode was launched, average rate of users, number of votes).

Elementary data analysis

Graph

To conduct the EDA, we used the R programming language which provides various inbuilt functions for manipulating and analyzing data. Both tables and graphs were used to get the insights from gathered dataset.



The graph shows the relationship between the number of series produced each year and the year of production. Clearly there is a positive (almost exponential) trend which indicates that more and more series are being produced nowadays in comparison to the previous century or first decade of XXI century.

Summary of data

We also summarized the most popular series, which received at least 10 000 votes. “Nasza planeta” a documentary series about nature took the first place.

	title	year	rate	votes
1	Nasza planeta	2019	9.1	27138
2	Czarnobyl	2019	8.9	182568
3	Gra o tron	2011	8.8	360762
4	Breaking Bad	2008	8.8	262253
5	Kompania braci	2001	8.7	125829
6	Rick i Morty	2013	8.7	97628
7	Biuro	2005	8.7	44571
8	Ostatni taniec	2020	8.7	29916
9	Sherlock	2010	8.6	207799
10	Peaky Blinders	2013	8.6	99767

Mean rating summary

Lastly, we summarized the mean rating according to each year (with a condition that number of series produced in that year > 10). It turned out that 2005 was the year in which the average rate of series is the best.

year	mean	n
<int>	<dbl>	<int>
2005	7.59	24
2013	7.52	51
2014	7.51	58
2015	7.48	64
2016	7.46	55
2019	7.46	49
2001	7.43	16
2002	7.4	13
2011	7.4	53
2007	7.36	33

In these three examples we presented that the gathered data can be useful for various analyses. There are many more insights that one can get out of data about series. It is only a matter of what is the main interest of the analyst.

Running time comparison

As we expected, scraper that used SCRAPY library occurred to be much faster than Selenium scraper.

Running time of 'Scrapy' scraper was only 33 seconds, while Selenium scraper needed 72 seconds.

Detailed description which participant wrote which part of the project.

Aleksander: Prepared filmweb_selenium.py script, descriptions of topic and web page, selenium mechanics, technical description of output, EDA.

Adam: Prepared filmweb_scrapy02.py file and it's description, running time comparison of each scraper, merging and uploading results to github.