Webscraping and Social Media Scraping | project description

Katarzyna Piotrowska, Ismayil Ismayilov

May 14, 2022

Contents

Contents 2

1 Topic and web page description 3

2 Scraper mechanics 3

3 Output 4
3.1 Technical description of the output 4
3.2 Analysis of collected data and Consistency of the obtained results 4
3.3 Comparison of scraper performance 6

4 Work devision among group participants 7

1 Topic and web page description

Discount on a product is something everyone looks for. Our team tried to make a user have an access to all discount details and enjoy exploring the best out of those deals. One of the chain drug stores, Rossmann has a very varied product offering. It has also an online store. The website has a separate page called Promotions for discounted products. At the time of this project, there are more than 6000 products on promotion. Going over each of them and finding the best deal can be overwhelming for a user. Scraping discounted products provides detailed information about each product in a short period. In the end, a user has an access to:

- Brand and products names
- The original and discounted price
- Discounted products by percentage, or product category | which can be filtered to see the most discounted products.
- Tp filter for only reviewed/rating products since all of the products are not reviewed.
- To easily detect if a product is not available in an online store.

This data can be used furthermore for data analysis. For example, one may interested in if there is a significant relationship between product category and discount rate. As the promotional products change, two scraping results can be compared to each other for further analysis.

2 Scraper mechanics

Three scrapers have been written, and each of them scrape the same piece of information from the rossmann.pl domain. Below is a description of each scraper's mechanics, focusing on the technical side.

- **scraper using Beautiful Soup** First, the program sends an HTTP request to the Rossmann.pl URL.The HTML content of the home page is returned as the response, and using Beautiful Soup scraper moves through the HTML data tree structure until it find the "Promotions" tab. In the next step, the program loops through the subsequent pages available in the "Promotions" tab. Moving to the next page is done by modifying the url by incrementing the page number.The program goes through the pages, downloads and saves the visible products with their prices and direct links to them. In the last step, the scraper goes through all the saved links to products and retrieves additional information, such as number of reviews, category, image etc.The program allows in the CONSTANS section to declare the number of products to scrape, the name of the csv and txt file created.The program saves the collected results to an external csv file, but every X products the data is saved and memory is freed (X- variable declared by the user). Additionally, the program after finishing the operation creates a log.txt file, which contains, among others, information about the time of operation execution and list of products, which were not scraped. The mechanics of the scraper is described in detail in the comments in the file rossmann_BS.py
- **scraper using Scrapy** The logic for the scraper using Selenium is very similar to that using Beautiful Soup. First, a list of all the url's to be scraped is created by changing the last part of the url for the Promotions tab.Next, the program loops through the list of url's that appear in the Promotions tab and sends a request to each of them.After sending the query, scraper saves the basic information about the products from the specific page in the Promotions tab. As a next step, the program goes to each product page. In each of these requests the argument meta={} is used to pass the product price information to the next parse() method. Scrapy is not able to retrieve the prices from the Product page, but it retrieves the rest of the product information from it. Running the appropriate commend saves the results to a csv file and saves a log.txt with status and execution time. The mechanics of the scraper are described in more detail in the comments in the rossmann_SCRAPY.py file.
- **scraper using Selenium** Rossmann's promotion page is dynamic as the products are continuously being updated. Selenium does not face any problem scraping the products. The scraping procedure starts with getting the URL of the promotion page which is looped for a range, so a user can choose/input how

many pages he wants to be scraped. Then selenium goes over the product by product, opening every product's link in a new tab, scraping the product details, and closing the tab. This procedure continues until every product on-page is scraped. One of the benefits of Selenium is being able to scrape JavaScript-loaded elements. For example, each product's page has two prices: Original and Discounted. The prices are loaded with JavaScript. Selenium had no problem with scraping those details.

3 Output

3.1 Technical description of the output

In case of each program, a csv file is obtained as a result, which contains scraped information about a certain number of products that are currently on sale in Rossmann on the Polish market. When the program opens a link to the next products, it retrieves 11 pieces of information and stores them as one row in a table. Below are listed the scraped information, for each variable its name fully explains what information about the promotional product it stores.

In addition to the csv file, a log.txt file is created for each program, which contains information about its status and/or execution time.

```
## Rows: 1,008
## Columns: 11
## $ Name            <chr> "NUMEE", "NUMEE", "NUMEE", "NEUTROGENA Hydro Boost", "~
## $ RegularPrice    <dbl> 29.99, 29.99, 29.99, 56.99, 56.99, 44.99, 26.99, 33.99~
## $ PromoPrice      <dbl> 17.99, 17.99, 17.99, 42.99, 42.99, 33.99, 19.99, 24.99~
## $ Rate            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ NumberOfReviews <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ Gender          <fct> NA, NA, NA, kobieta, kobieta, NA, NA, kobieta, kobieta~
## $ Categories      <chr> "Twarz/Pielęgnacja twarzy/Serum", "Twarz/Pielęgnacja t~
## $ Availability    <fct> Available online, Available online, Available online, ~
## $ Description     <chr> "serum złuszczające do twarzy30 ml, nr kat. 393360", "~
## $ Link            <chr> "https://www.rossmann.pl/Produkt/Serum/Numee-serum-zlu~
## $ Image           <chr> "https://www.ros.net.pl/GalleryImages/product_photos/1~
```

3.2 Analysis of collected data and Consistency of the obtained results

Written scrapers with different approaches may vary in performance or complexity, but the important thing is that the data extracted from the website should be correct and consistent between the different approaches. After careful analysis of the csv files, we found that the results obtained are consistent between the different approaches. For illustration, below are three tables with 5 continuous variables representing 5 features of a particular product from the promotion tab on rossmann's website. The tables represent the results obtained for 240 products using BS scraper, Scrapy and SELENIUM respectively.Looking at the three basic statistics, as well as the number of observations and missing data, it can be concluded that the tools are fully compatible. The results for 120, 480 and 1000 products were also compared and were also consistent.

BS RESULTS:

Variable	max	mean	min	N	NA.
Discount	43.35	29.96350	22.74	240	0
NumberOfReviews	57.00	15.35294	5.00	240	223
PromoPrice	66.99	23.30292	4.49	240	0
Rate	5.00	4.44118	3.00	240	223
RegularPrice	89.99	33.23583	5.99	240	0

SCRAPY RESULTS:

Variable	max	mean	min	N	NA.
Discount	43.35	29.96350	22.74	240	0
NumberOfReviews	57.00	15.35294	5.00	240	223
PromoPrice	66.99	23.30292	4.49	240	0
Rate	5.00	4.44118	3.00	240	223
RegularPrice	89.99	33.23583	5.99	240	0

SELENIUM RESULTS:

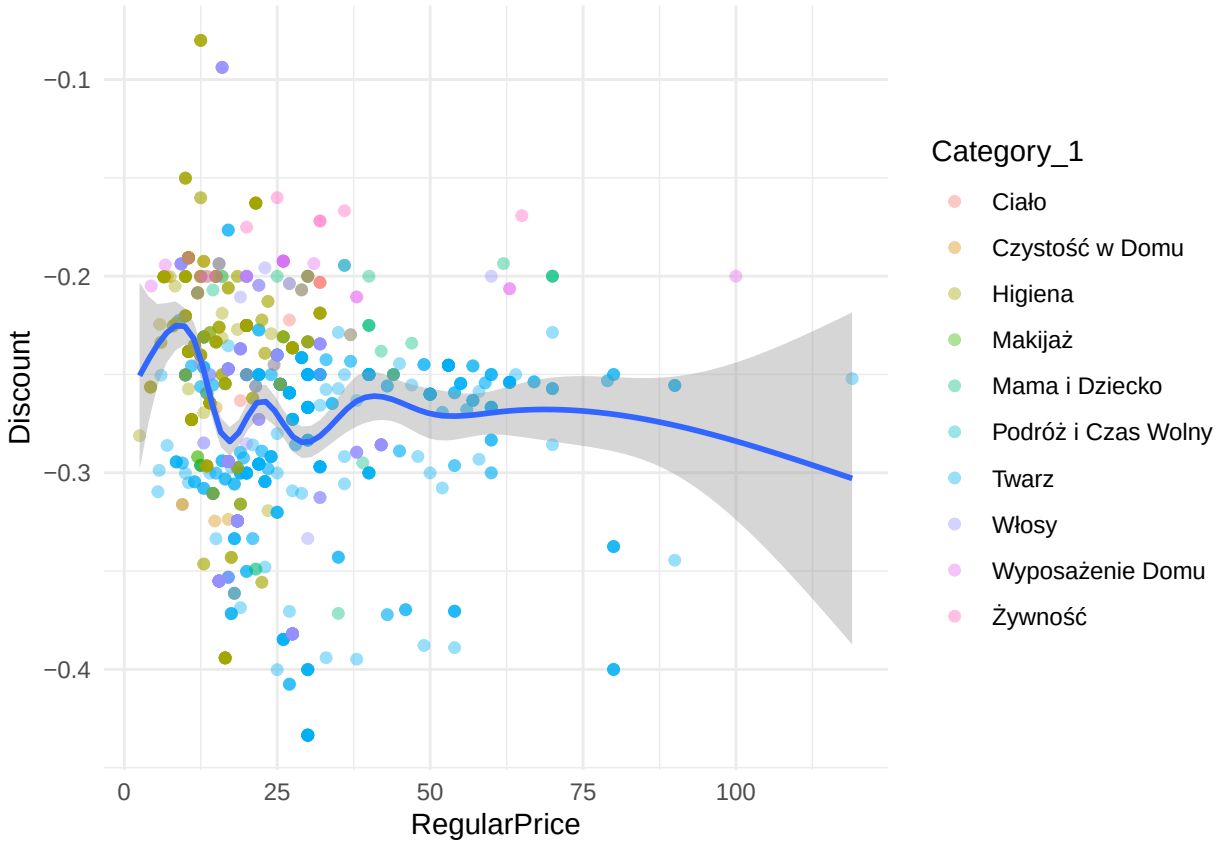
Variable	max	mean	min	N	NA.
Discount	43.35	29.96350	22.74	240	0
NumberOfReviews	57.00	15.35294	5.00	240	223
PromoPrice	66.99	23.30292	4.49	240	0
Rate	5.00	4.44118	3.00	240	223
RegularPrice	89.99	33.23583	5.99	240	0

The data collected is useful, it gives the opportunity to analyze price trends in the cosmetics industry, or provide information about the range of competitors in this case rossmann. And the reproduction of such analysis over time can allow for the analysis of customer behavior, but also for the analysis of long-term pricing policy of competitors. The collected information is not only the prices of products, but also their evaluation by users, which can also be use in the analysis.
To show that the collected information has a positive value and can be used for analysis, below we present a table showing the amount of discount by product category. It was created using data from scraping 1000 products. We can see in the table that bb and cc creams are currently the most discounted, perhaps it is a strategy replicated by rossmann, which noticed that it is now spring and at that time people who wears make up switch to light creams. The same table was recreated for a generated set of 3000 products and this category kept the high place. in the table.

Percentage of discount by product category:

Categories	mean	min	max
Makijaż/Twarz/Kremy bb i cc	36.13000	36.13	36.13
Twarz/Oczyszczanie i demakijaż/Toniki do twarzy	32.20778	29.45	37.16
Twarz/Oczyszczanie i demakijaż/Żele i pianki do twarzy	31.95636	29.17	37.16
Twarz/Oczyszczanie i demakijaż/Peelingi do twarzy	30.82200	29.75	35.02
Twarz/Pielęgnacja twarzy/Kremy do twarzy	29.99991	22.86	43.35
Twarz/Oczyszczanie i demakijaż/Płyny micelarne	29.94417	22.74	33.35
Twarz/Pielęgnacja twarzy/Kremy pod oczy	29.68538	24.25	43.35
Makijaż/Twarz/Pudry	29.40500	29.19	29.62
Twarz/Pielęgnacja twarzy/Serum	29.15957	22.86	40.01
Twarz/Pielęgnacja twarzy/Maseczki	25.02750	25.01	25.04

We also presented a graph showing the slight correlation between the regular price of the product and the discount in percentage. The more expensive the product, the higher the discount. Moreover, you can see that the blue dots are slightly shifted to the right, which is connected with the table analysis done below.



This is a very simple and preliminary analysis of the data, but it seems that the dataset can be used for some analysis and scraping more products, perhaps extending the dataset with reviews, can increase the value of this dataset.

3.3 Comparison of scraper performance

Our team tested and measured the timing of every scraping method by scraping 120, 240, and 480 products which are 5, 10, and 20 promotion pages, respectively. The data shows that Scrapy is an undisputed winner among the three options. Scrapy managed to scrape 480 products in just 10 seconds, while the closest was BeautifulSoup, which took 4 minutes and 6 seconds. Scrapy is also the only framework that scraping speed was consistent and scraping more products didn't cause diminishing results in time. Out of three frameworks, Selenium showed pretty slow results. It took whopping 19 minutes and 43 seconds to scrape 480 products(~110 times slower than Scrapy). The slow scraping is caused by the mechanics of Selenium. Every time to scrape each product, Selenium opened a product page, scraped, and closed. It also lost time going to the next page, since the next page opened in a new window.

#PROD	BS [SEC]	SCRAPY [SEC]	SELENIUM [SEC]	BS [MINS]	SCRAPY [MINS]	SELENIUM [MINS]
120	87	3	298	1.4583	0.0500	4.9804
240	110	5	562	1.8358	0.0833	9.3694
480	246	10	1183	4.0941	0.1667	19.7164

4 Work devision among group participants

Team Members:

- Katarzyna Piotrowska || studentID 397061
- Ismayil Ismayilov || studentID 444459.

BeautifulSoup and Scrapy frameworks are done by Katarzyna Piotrowska. Selenium framework is done by Ismayil Ismayilov.

Side note: Ismayil Ismayilov also tried to do the Scrapy framework. His method scraped the products but faced an issue while exporting CSV in the correct way. Katarzyna Piotrowska had tried another method and her scraping worked perfectly.