Huseyin POLAT
437969
Gizem Gulsiye GULELI
449872

Exploring Nonfiction Self-Help Books through Web Scraping

Description of Project

In this project, we look into the world of nonfiction self-help books by leveraging web scraping techniques. We scrape book information from the "Nonfiction with a Side of Self-Help" list on Goodreads using techniques including BeautifulSoup, Scrapy, and Selenium. Our main source of useful information is Goodreads, a well-known website for book suggestions and readers.

The purpose of this project is to gather important data, including things like book titles, authors, Kindle prices, typical ratings, review numbers, and more. This project provides an example of web scraping's usefulness and efficiency in obtaining data from online sources.

The final dataset, which is stored in a CSV file, serves as a useful tool for future research, machine learning models, data visualization, and the extraction of useful information in the areas of literature and self-help.

Scraper Mechanics

1- BeautifulSoup:

First, the script makes a request to the list's webpage, parsing the HTML text with BeautifulSoup. The pagination links are extracted, and a DataFrame is used to store them.

Secondly, the book links are retrieved from each page of the list by using the pagination links that navigate through its many pages. The URLs to the books are maintained in another dataframe.

Finally, the script asks for information about each book from its webpage using the book URLs. ThreadPoolExecutor is used to run the scraping procedure simultaneously, which increases productivity.

The script uses a while loop to handle any potential request failures and adds a 1-second delay between requests to avoid overly demanding the server.

After being placed in a data frame, the scraped data is processed to remove any non-numeric characters from the rating and review count columns.

The script logs the start and finish times and computes the total amount of time that has passed to determine the scraping time.

2- Scrapy:

The scraper consists of 3 spiders named ‘pagination_links’, ‘book_links’ and ‘book_details’.

- **‘pagination_links’** spider visits a Goodreads list and scrapes the pagination links from the response using XPath. Each link is stored in an instance of the class and yielded as a scraped element.
- **‘book_links’** spider visits each pagination link retrieved in the previous step and scrapes the book links from the response using XPath, stores them in instances of the class, and yields them as scraped elements.
- **‘book_details’** spider visits each book link retrieved in the previous step and generates requests for each link. The spider checks if the desired element (book title) is present in the response and makes another request if it's missing. If the element is found, the spider extracts the book details using XPath, stores them in instances of the class, and yields them as scraped elements.

3- Selenium:

This scraper automates browsing on Goodreads to collect pagination links, book links and book details. It uses Selenium with Firefox in headless mode to navigate through the collected links. Firstly, it scrapes pagination links from a Goodreads list and later it visits each pagination link to scrape individual book links. Later, it visits each book link captured in the previous step and scrapes book details.

Output

The information retrieved from each book page includes the title of the book, the author's name and a link, the Kindle price (if available), the average rating, the number of reviews, the number of ratings, and the number of pages.

Field	**Example**
link	https://www.goodreads.com/book/show/59788656-soul-cure
title	Soul Cure: How to Heal Your Pain and Discover Your Purpose
author_name	Gregory Dickow
author_link	https://www.goodreads.com/author/show/163477.Gregory_Dickow
kindle_price	Kindle $12.01
average_rating	3.52
rating_count	1,417
review_count	15
n_pages	272 pages, Hardcover

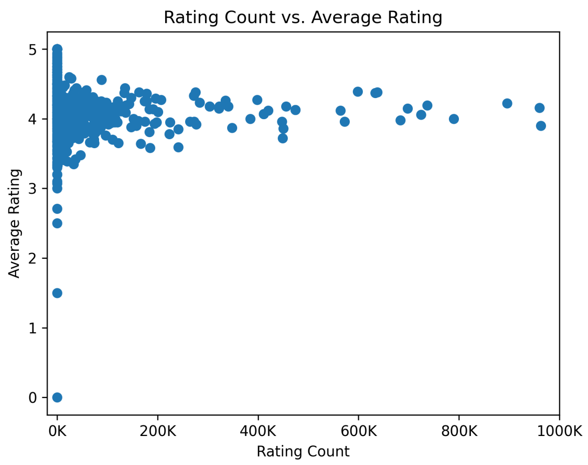
EDA

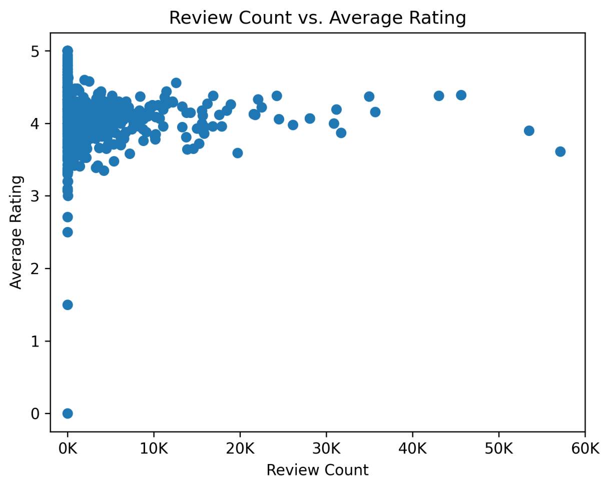
In total, 880 book links are visited, and relevant data is extracted. The table below only shows the summary statistics for the numeric data that was scraped.

feature	count	mean	std	min	25%	50%	75%	max
average_rating	880	4.14	0.42	0	3.93	4.11	4.32	5
kindle_price*	508	10.81	4.41	0.45	7.99	11.24	13.99	39.99
rating_count	880	51320.78	180041.68	0	20	9420	33573.25	2812368
review_count	880	2743.77	7619.63	0	4	633	2165	109905

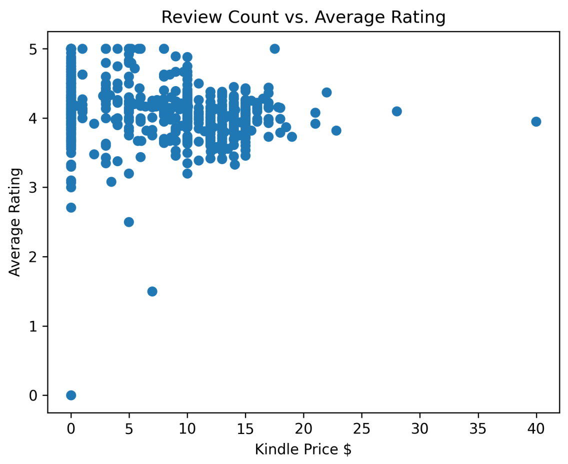
*empty values are not included in kindle_price feature

1. average_rating: The average rating is 4.14 out of 5, with a standard deviation of 0.42, indicating a generally positive and consistent reception among readers.
2. kindle_price: The average price of a Kindle book is $10.81, with a maximum price of $39.99.
3. rating_count: The standard deviation of 180,041 indicates a wide variability in the number of ratings, with some books receiving significantly higher engagement from readers.
4. review_count: The standard deviation of 7,619 indicates a wide variability in the number of reviews, with some books generating significantly more discussion and feedback from readers.





Looking at the graphs above for the relationship between rating count, review count and average rating, it is concluded that there is no correlation between rating count, review count and average rating.



Looking at the right graph for the relationship between Kindle price and average rating, it is concluded that there is a slightly negative correlation between them.

Contributions

This project was developed by Huseyin and Gizem, who collaborated to implement various components of the web scraping project. Our individual contributions are outlined below:

Huseyin's

Scrapy and Selenium: Huseyin was responsible for implementing the Scrapy and Selenium components of the project. He utilized Scrapy, a powerful web scraping framework, to extract data from the Goodreads website. Huseyin also utilized Selenium, a browser automation tool, to interact with web pages that required JavaScript rendering. This combination allowed for dynamic scraping and improved data collection capabilities.

Documentation: Huseyin took charge of documenting the Scrapy and Selenium parts of the project. He provided clear instructions on how to set up and run the scraping scripts using these tools. Huseyin also documented any specific configuration or requirements for successful scraping with Scrapy and Selenium.

Eda Analysis: Huseyin contributed to the exploratory data analysis (EDA) phase of the project. He used the collected data to perform insightful analyses, such as identifying trends, correlations, or patterns within the book details. Huseyin visualized the results and provided meaningful insights into the dataset.

Gizem's

Beautiful Soup: Gizem was responsible for implementing the BeautifulSoup component of the project. She utilized BeautifulSoup for web scraping, to parse the HTML content of the Goodreads website and extract the required information. Gizem employed BeautifulSoup's intuitive methods and selectors to navigate and extract data from web pages.

Documentation: Gizem took charge of writing the general description and also the description for BeutifulSoap. She provided a clear overview of the project's objectives, the importance of the collected dataset, and its potential applications. Gizem also ensured the project documentation was well-structured, with detailed explanations of the implemented components and their functionalities.

We worked closely together throughout the project, exchanging our knowledge and insights to produce a complete web scraping solution. Additionally, we talked about design choices, shared ideas, and handled any problems that arose during the development process. We successfully created a reliable and effective web scraping project by combining our skills, which allowed us to collect book details from Goodreads.