Speed Skating web scraping - WS2023

Names and ID’s of all participants

Hubert Wojewoda 406420
Michał Grzyb 419327

Short description of the topic and the web page

Our project involves scraping a list of speed skaters from speedskatingresults.com. This site is a database of information regarding all speed skaters and competitions that take place around the world. You can find there information about the skater's personal records as well as information about which competitions they competed in and what times they achieved. Also you can find information like track, country and world records by age category and much more. In our project we focused on scraping 100 speed skaters in alphabetical order. Along with this we will scrap information such as nationality, date of birth and personal records at each distance

Short description of the scraper mechanics

All three scripts operate in a similar manner. They follow a 3-step process:

1. **Navigating to the Website:** All scripts generate a URL and use an HTTP GET request to navigate to the webpage containing a list of athletes. BeautifulSoup/lxml and Scrapy use their built-in HTTP request functionalities, while Selenium controls a browser instance to do this.
2. **Extracting URLs of Athlete Profiles:** Once on the page, they find the URLs of each athlete's profile. BeautifulSoup and lxml parse the HTML and extract the href attribute of the relevant links. Selenium interacts with the browser to get the same links. Scrapy's Spider does the same but uses the response object passed to it by the Scrapy engine.
3. **Scraping Data from Athlete Profiles:** The scripts then visit each athlete's profile using the URLs obtained in the previous step and extract the required information (name, country, birthdate, personal records). They do this by locating the HTML elements containing the data and extracting the text. BeautifulSoup and lxml parse the HTML, Selenium interacts with the rendered webpage, and Scrapy's Spider uses the response object.

Finally, all scripts compile the scraped data into a pandas DataFrame and export it to a CSV file. They also have a mechanism to control the number of profiles to be scraped, as well as pagination logic if there is a need to browse more than one page of athletes. Each page has 66 athletes so in the 100 links test case, two pages are checked for links.

Short technical description of the output you get

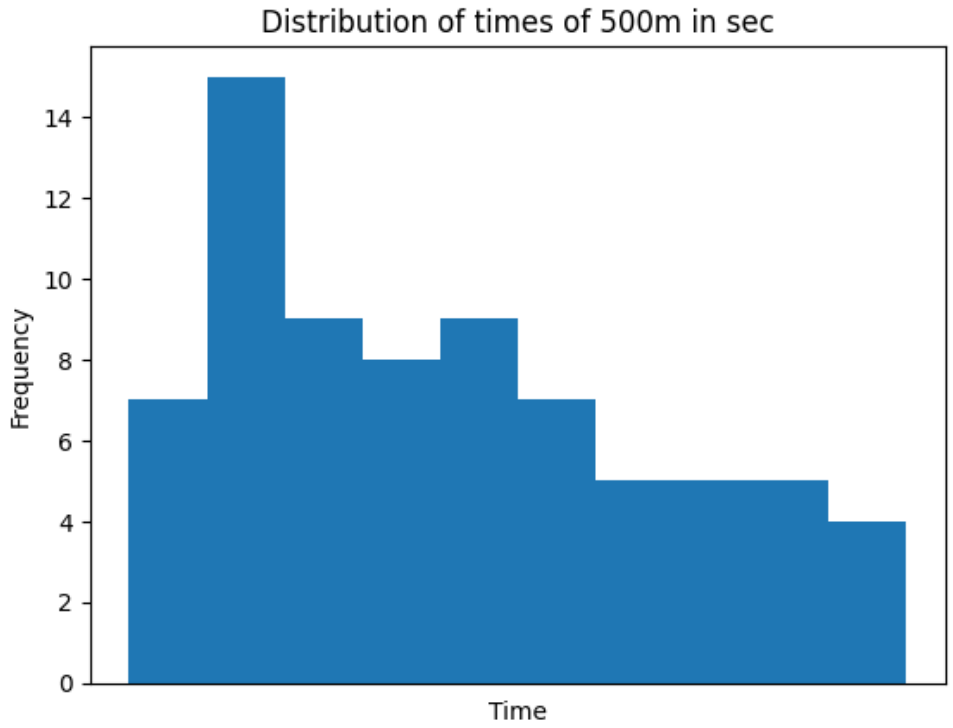
The output of each of the scrapers is a dataframe that contains the first 100 speed skating athletes. Each row has information such as name of the athlete, country which they represent, their birthdate as provided on the site and their respective personal best times for each distance that they started in. If they don't have a PB in a certain distance, the value for the respective column is NaN. In addition to printing a dataframe, each scraper also saves the data as a csv file for further analysis. The final output line is the time in which the scraping task was performed, denoted in seconds.

Elementary data analysis

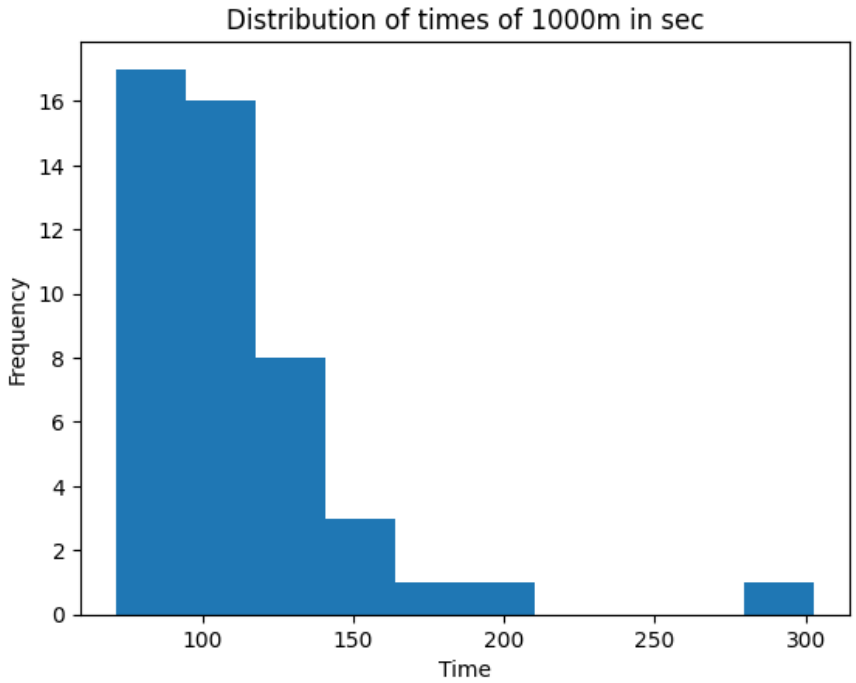
Scraped data is 100 unique values with 9 different features. In this case, after deleting the rows with no values, consecutively there are so many values left

500m - 74
1000m - 47
1500m - 38
3000m - 10
5000m - 9
10000m - 5

It can be noted that as the distance increases, the number of people who competed at a given distance decreases.



The 500m time distribution chart shows that the best times under 40 seconds are achieved by only 7 athletes. Times between 40 and 45 seconds are the most common . From this, as the time increases, the number of athletes decreases



Of the 47 athletes who rode the 1000m az 33 fall under 2 minutes and the other 14 over 2 minutes.

Which group participant wrote which part of the project

Michał Grzyb:

- BeautifulSoup scraper
- Selenium scrape_pages function
- Pagination logic design

Hubert Wojewoda:

- Selenium athlete link and profile scrapers
- Scrapy spider and pipeline logic
- Timing and saving to csv for each method