Web Data Scraping

Preprint · March 2020

DOI: 10.13140/RG.2.2.15125.76009

1 author:

Rizul Sharma
KIIT University
11 PUBLICATIONS **6** CITATIONS

SEE PROFILE

DATA CRAPER

Rizul Sharma

School of Computer Engineering, KIIT

1605216@kiit.ac.in

Abstract- From the advancement of the World Wide Web, the situation of the web client and information trade has fastly changed. As average citizens join the web and begin to utilize it, heaps of new systems are elevated to help up the system. Simultaneously to improve PCs and system office, new innovations were brought which results in a consequent reduction in the cost of equipment and site's connected expenses. Business, academician, scientists all share their data on the web with the goal that they can be associated with individuals fastly with no problem at all. Because of exchange, share, and storage options for the information on the web, another issue emerged that how to deal with such information overload and how the client will get or get to the best data in the least endeavours. To settle these issues, specialist spot out a new procedure called Web Scraping. Web scraping, or web scratching, is a procedure which is utilized to create organized information based on accessible unstructured information on the web. Created organized information at that point is then put away in focal database to be dissected in spreadsheets. Presently, there are heaps of tools accessible in the market for web scratching. This paper is centred around the overview of the data extraction method and how to implement it using python.

Index Terms- Web scratching, structured information, unstructured information, data extraction, organized data, scraping.

I. INTRODUCTION

In today's time of data science & engineering, it is entirely expected to gather information from sites for examination purposes. Realizing how to scrap site pages will set aside your time and money. A few organizations like Twitter do provide APIs to get their data in a progressively composed manner while we need to scratch different sites to get information in an organized configuration.

The general thought behind web scratching is to recover information that exists on a site and convert it into a configuration that is usable for analysis. Python is one of the most normally utilized programming dialects for data science ventures. Utilizing python with Beautifulsoup makes web scrapping simpler. Through this paper, we will be experiencing a detail however a basic clarification of how to scratch information in Python using BeautifulSoup. This will help information researchers gather and store information from site pages effortlessly without investing an excess of energy in getting ready datasets.

II. STUDY OF SIMILAR PROJECTS OR TECHNOLOGY\ LITERATURE REVIEW

For an experimental analysis of how to perform a scraping using web scraping tools, I studied two tools. Firstly 'Scrapy', a web scraping extension available in the chrome web store. Scrapy is a very simple but limited data mining extension for facilitating online extraction of data for researchers in the format of spreadsheets. Data is in limited format i.e. we can not get proper data in a spreadsheet [4].

After this, I studied the free available scraping tool 'ParseHub'. Users have full control over the extraction of data from targeted websites. It works like a hierarchical base selection of data. At the starting of scraping, the user simply selects the field which he/she want to extract, then ParseHub automatically guesses similar data element from a website [1]. As the user selects a piece of related information which he wants to extract, all similar elements are extracted. For selecting other data elements from a targeted website, a 'relative' search option is available which is subset information about the previously selected element. Likewise, the user extracts all information from the website. At the time of extraction of element from a website, ParseHub provides a URL also. This URL is an optional field. After successful web scraping data sets are saved in a CVS format [1].

III. BASIC CONCEPTS/ TECHNOLOGY USED

This project utilizes many concepts and is been made with the help of various tools and technologies. The required resource analysis is as follows;

Python is an interpreted, high-level, general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

JUPYTER Lab - Project Jupyter is a nonprofit organization created to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

Web Scraping (also termed Screen Scraping, Web Data Extraction, Web Harvesting etc.) is a concept utilized to gather information from sites whereby the information is extricated and spared to a record in your PC or to a database in table (spreadsheet) design [2].

Below, some systems similar to the current project are mentioned in order to highlight the contrast between the current project and these mentioned software tools.

Import.io is an online instrument for extracting information from a site without composing code. In the event when the client needs a quick outcome, at that point he will go after this approach with the goal that he can change data of the site in a brief timeframe. For extricating information, the client enters URL and the application naturally gets the information which client needs if the client doesn't keen on the programmed extraction, the point and click interface helps to select data fields on the site [1]. As the information extraction is finished, the separated dataset is stored on Import.io cloud server and further downloaded in CSV, Exceed expectations, JSON design.

Scrapy is intended to scratch web content from locales that are made out of numerous pages of comparative semantic structure. An open-source and community system for extricating the information you need from sites. The framework is actualized as a Firefox browser extension and works in three principal stages to scratch web information [1]. Initial, a client explores to a page that he would like to scratch and creates a format for the substance that he might want from that page. Next, the client chooses a lot of links that point to pages matching the substance layout characterized by the client. At last, the client chooses a final output to information group and Scrapy slithers the connections determined by the client and scratches content comparing to the client's template [1]. Scrapy is composed in Python and runs on Linux, Windows, and Macintosh.

IV. Proposed Model / Architecture / Methodology/ Model Tool

The proposed work centers around dissecting the website pages (HTML code). Right now, have built up a working model. By utilizing this procedure weblink change into visual blocks. A visual block is actually a section of web page. The framework is programmed top-down and deals to recognize web content structure. Basically, the block-based page content structure is obtained by using a python script in BeautifulSoup in order to further save it as a CSV file. Simulation of experimental work shown below [3];

A. Installation of BeautifulSoup and Requests

B. Python scripting

C. Execution of the python code

D. Content structure construction

E. Saving it as a CSV file

V. Implementation and results

To automatically extract HTML tables from web pages and save them in a proper format in your computer, I used Requests and BeautifulSoup libraries to convert any table in any web page and save it in the drive. I used Pandas to easily convert to CSV format.

I first initialized a requests session, I used the User-Agent header to indicate that a regular browser, and not a bot (some websites block them), is in the use to get the HTML content, which was accessed using session.get() method.

After that, I constructed a BeautifulSoup object using html.parser. Since I wanted to extract every table in any page, I needed to find the table HTML tag and return it. Then to get the table headers & the column names, "get_all_tables" function finds the first row of the table and extracts all <th> tags (table headers).

"get_table_rows" function is finding <tr> tags (table rows) and extracting <td> elements which then appends them to a list. The reason I used table.find_all("tr")[1:] and not all tr tags, is because the first tr tag corresponds to the table headers. "save_as_csv" function takes the table name, table headers and all the rows and saves them as CSV format.

VI. CONCLUSION

It completely meets the objectives and requirements of the system. The framework has achieved an unfaltering state where all the bugs have been disposed of [1]. The framework cognizant clients who are familiar with it and comprehend it's html parser and the fact that it takes care of the issue of collecting and storing data for individuals in the field of data analysis to help them manage time and reduce efforts.

The framework can be improved steadily to extract information from the hidden web. Hidden Web information mix is a significant test these days. In further research work, different difficulties in the region of Hidden Web information extraction and their potential arrangements can be talked about. Right now, internet search engine shell has been made which was tried on different areas. This work could be reached out for onion spaces by coordinating this work with the unified search interface [2].

REFERENCES

[1] Saurkar, Anand V., Kedar G. Pathare and Shweta A. Gode, *An Overview On Web Scraping Techniques And Tools,* International Journal on Future Revolution in Computer Science & Communication Engineering, pages 363-367, 2018.

[2] Liu B., *Sentiment Analysis and Subjectivity,* Handbook of Natural Language Processing, pages 627-666, 2010.

[3] Pratiksha Ashiwal, S.R. Tandan, Priyanka Tripathi and Rohit Miri, *Web Information Retrieval Using Python and BeautifulSoup,* International Journal for Research in Applied Science & Engineering Technology (IJRASET), pages 335-339, 2016.

[4] Rahul Dhawani, Marudav Shukla, Priyanka Puvar, Bhagirath Prajapati, *A Novel Approach to Web Scraping Technology,* International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 5, 2015.