JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Online Product Comparison System For E-commerce Websites

Prof.Shaikh mam, Sudhir Pakhare, Rohit Thorat, Himanshu Kapse,Bhagyashree Bansode
Associate Professor ,Student, Student, Student,Student
Computer Engineering Department,
Sinhgad Academy of Engineering Kondhwa, Pune, India

Abstract : Web scraping is a data-mining technique which is used to extract information from various web content. Nowadays, Everyone uses E-commerce websites to buy products online. These giant companies always continue to come up with new strategies and attract more customers. Even they track customer behavior and psychology to increase their sales. As there are many competitors coming in this industry it becomes difficult for customers to compare and buy at best deal. Using our proposed model users can easily compare and buy products at best deal. To get such best deals we are using web scraping and web crawling techniques. Using this, users can easily grab the best deal with minimum efforts,time and money. Manually visiting each website and comparing each deal on these sites is time consuming and costly. Our model will compare all the product details and features which will make it easier for users to make their decision.

Index Terms **- Web scraping, E-commerce, Data extraction, Web crawler.**

I. INTRODUCTION

The Internet is a major source for many online shoppers to shop at ease and in peace of their homes. Younger generation prefers online shopping. The Internet consists of many online shopping websites,portals,suggestions and various information related to new products,devices,etc. To fetch and gain insight from such information becomes time and money consuming for all of us. The purpose of the program is to extract all such information from multiple E-commerce websites and display all product information in a single web interface for comparison. Web scraping is a time consuming and resource consuming task when performed manually. This complexity increases with more no. of sites and dynamic websites. For dynamic websites, it is very complicated to extract data due to its dynamic nature. There are many techniques[1] to retrieve information from such sites like, Cut/Paste, http, Query languages for DOM [2],semi-structured Data [3], or even Web-Scraping [4]. A web scraper is a software which simulates human browsing on the web to collect detailed information or data from different websites. The scrapper has advantage on its speed and its capacity to be automated and programmed. Although whatever method we use, the approach remains the same- capturing the web data and presenting it to users in simple and easy to read format. A user can gain insight from it and make his/her decision easily. Data extraction is a method of collecting data from various sources. User gives input then our model does analysis and web scraping in the background and then represents results to the user in a better way. We did a survey of our website and 83.7% of people found our website helpful for their online shopping.

II. LITERATURE SURVEY

Following are some of the approaches for web scraping:

a. Manual cut-paste :

This is the simplest approach to web scraping. We manually cut-paste data from a web page into a text file or spreadsheet. As we are personally giving the data and testing it manually this approach gives more accurate and better results. Some sites explicitly set up protection barriers from automation. In such a case this approach still works as we are manually taking the data. But this is more time and resource consuming. As the number of sites and their complexity increases it becomes impossible to gather data from all of them. As we scale more this approach becomes difficult to follow.

b. Direct HTTP requests (HTML code update):

By using this method, you will be able to find dynamic pages by sending HTTP requests to remote servers. Given the behavior of client side rendered websites, the approach for direct HTTP requests becomes interesting. This method mainly uses sockets that organize all responses using pre-made data in targeted packages. Scraping becomes a bit faster since we make a direct HTTP request and never need to load the full web page. Data that we receive is richer since we receive data from servers than what the website shows. It is as simple as making an HTTP request. In fact, after identifying and obtaining the link, we can open that link in a new tab and see the results in its bare form mostly in JSON format. Suitable for client side rendered websites or sites that incorporate infinite loading especially those that heavily rely on REST API. There are certain things which we need to take care of in this approach. We must make sure that the websites we are about to scrape use client side rendering methods of populating data. Each website can have various endpoints. Using trial and error we can't find HTTP requests. since each website is unique and it can have different implementations. We need to take care of Cross-Origin Resource Sharing, Same-Origin policies, Cookies, and Authentication Tokens. When the website updates its HTTP-requests, we will also need to update our scraper algorithm and accessible endpoints.

c. DOM analyzer:

The web scraping can be done by targeting the selected DOM components and then processing or storing the text between that DOM element of a web page.It becomes a challenge in web crawling with powerful content websites. These browser controls also browse web pages in the DOM tree, depending on which programs can find parts of the pages. Developer Tools mainly operate on a browser DOM, what you will actually see when inspecting the page source is not the original HTML, but a modified one after applying some browser clean up and executing Javascript code. Languages such as Xpath can be used to scan a growing DOM tree.However results are noisy.

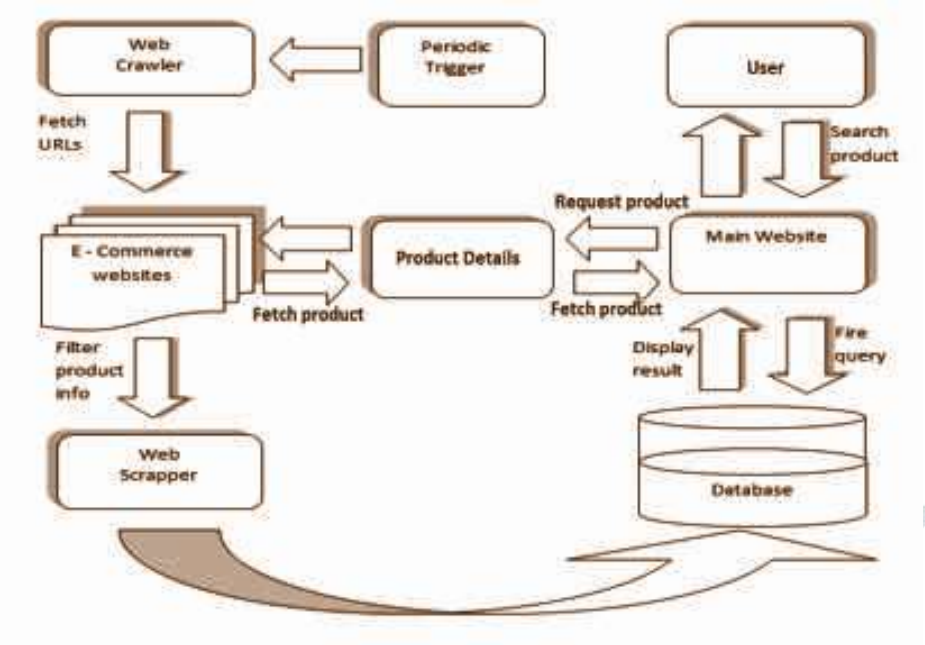
d. Machine Learning Approach

The general principle is to train an algorithm on a large sample of web pages. To train such modules we can use manually analyzed web pages. Machine learning is based on geographical indicators of the text blocks on the page statistical measure is done where the main text block is located as compared to the other text blocks. The machine is then able to deduce by itself where the text is usually located. If we train our machine Learning model on larger sample data we get more accurate results.

III. SYSTEM ARCHITECTURE

Refer to below diagram of system architecture. User searches for a product on the main website. That product details are fetched from all other E-commerce sites. Web crawler fetches all the relevant E-commerce or review websites related to that product. Those results are passed to the

web scraping model. Web scraper filters all such product information and saves it in the Mysql database. Then users can see those results on the main website in simplified format using those saved details in the database. Already Saved results of a particular can be fetched directly from the database.



IV. IMPLEMENTATION

1. Project Modules

a. Beautiful Soup:
Beautiful Soup is a powerful python library designed for quick projects like screen-scraping. It provides simple methods and Pythonic idioms for navigating, searching, and modifying a parse tree. It uses popular Python parsers like lxml and html5lib, allowing you to try out different parsing strategies or trade speed for flexibility.

b. Selenium:
Selenium is used to rip large amounts of data such as text and images in a very short time. Use WebDriver protocol to control a web browser, such as Firefox or Chrome. In browser cross-sectional testing and end-to-end testing using selenium.

c. Mysql-connector:
It allows Python programmers to connect to MySql databases easily. Programmers can easily save,delete or edit data in MySql.

2. Algorithm

a. Algorithm Description:
We take the product name and other details as an input from the user. Using those details we do web scraping and crawling on all relevant E-commerce sites. As we cannot fetch data directly from any website, so for that we use different web scraping libraries. Our system uses Selenium and Beautiful Soup for extraction. Selenium is used for automating web based applications. Selenium has a webdriver which is used to control a web browser. As per the product those related websites are found automatically using webdriver. Each and every website has HTML tags through which we can fetch the required data, for this purpose Beautiful Soup library is used. When the web page is loaded on every website using webdriver, Beautiful Soup fetches the HTML tags of that page using class name and the required data of the respective tag is extracted. The

scraped data is stored in the Mysql database. Python uses a mysql connector module to connect with the database. The product details which are stored in the database are then displayed on the users screen.

b. Algorithm Steps:
Step 1. Take all product details as input from user
Step 2. Find all relevant sites related to that product
Step 3. Extract all the product related data from these websites
Step 4. Store it in database
Step 5. Display the information on the users window

V. CONCLUSION

We tested our model and website among 300 people out of them 251 found this model to be helpful and efficient. We did a survey of our website and 83.7% of people found our website helpful for their online shopping. Our system scrapes the data from different websites and compares the product detail on a website.

VI. REFERENCES

[1]. Glez-Peña, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M., & Fdez-Riverola, F. (2013). Web scraping technologies in an API world. Briefings in bioinformatics, 15(5), 788-797.
[2]. Gupta, S., Kaiser, G., Neistadt, D., & Grimm, P. (2003, May). DOM-based content extraction of HTML documents. In Proceedings of the 12th international conference on World Wide Web (pp. 207-214). ACM.
[3]. Boag, S., Chamberlin, D., Fernández, M. F., Florescu, D., Robie, J., Siméon, J., & Stefanescu, M. (2002). XQuery 1.0: An XML query language.
[4]. Sirisuriya, D. S. (2015). A comparative study on web scraping.