Methodology Notes

Written by: Chaimaa Lotfi and Swetha Srinivasan

Supervised by: Professors Myriam Ertz and Imen Latrous

September 2021

Table of contents

1 Introduction

Big data analytics gives organizations a way to analyze huge data sets and gather new information. It helps answer basic questions about business operations and business performance. It also helps uncover patterns in the enormous amount of data. In the current data driven world, it is very essential that big data techniques are applied and analyzed for the growth of organizations. There is a lack of investigation on the impact of BDA on firm performance. In this project, we aim to build a tool that helps us analyse if a particular company from the list of S&P 500 and TSX60 companies has used a particular big data technique from the list of techniques available based on the information reported in academic literature. The first part of the project deals with building the updated corpus from the academic literature by developing the web scraping tool. Later, with the help of a text mining tool, we have tried to analyse whether the companies listed have used the techniques listed.

2 Data Collection

With the pre-constituted corpus collected in 2019 as the basis, we collected data from academic literature with the help of a web scraping tool and built an updated corpus.
To update the corpus to include papers from 2020 and 2021, we collected papers which cite the papers already present in the corpus using a web scraping tool. The process of updating the corpus is divided into two modules :

- Extracting cited papers
- Downloading the cited papers

2.1 Extracting cited papers

The pre-constituted corpus had publications related to impact of big data analytics on firm performance. We used the tool to identify publications which cite these research papers to update the corpus with relevant and recent research studies on big data analytics capabilities. [8] [9] We leveraged Google Scholar to identify the citing publications. We built a tool that can search for the title of a research paper in the publication and extract titles that cite that paper.
We tried to build the tool with Selenium, a web automation and testing framework[1]. Selenium is an open-source web-based automation tool which is quite efficient for web scraping. The web driver in Selenium provides numerous features that enable users to navigate through the desired web pages and fetch various contents of the page depending on necessities[3]. Thus, lots of data from various web pages concerning the user's query can be extracted from multiple web pages and grouped. But we faced issues with CAPTCHA and integrating Selenium with proxies and VPNs[5].

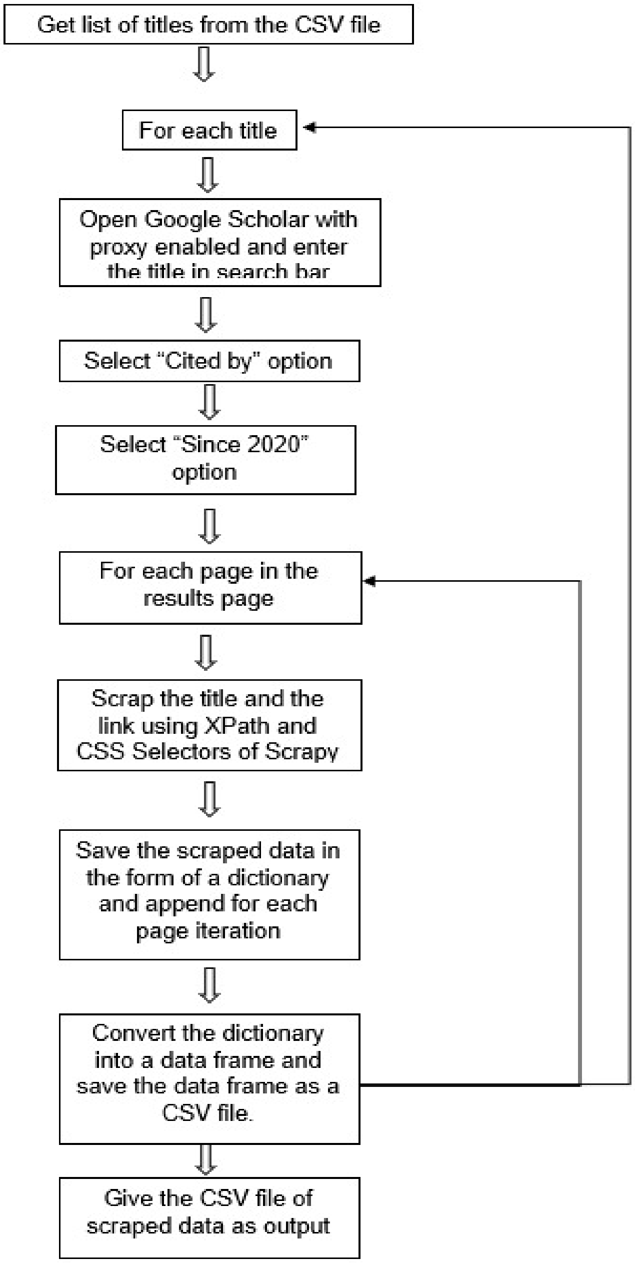
Get list of titles from the CSV file
For each title
Open Google Scholar with proxy enabled and enter the title in search bar
Select "Cited by" option
Select "Since 2020" option
For each page in the results page
Scrap the title and the link using XPath and CSS Selectors of Scrapy
Save the scraped data in the form of a dictionary and append for each page iteration
Convert the dictionary into a data frame and save the data frame as a CSV file.
Give the CSV file of scraped data as output

Figure 1 – Process flow of the web scraping tool

[7] We also tried using Octoparse, a cloud-based web data extraction solution that helps users extract relevant information from various types of websites, but did not get better results. [4] We used ScraperAPI, an API that handles proxies, browsers, and CAPTCHAs to avoid triggering CAPTCHA. We used this API along with Scrapy which supports integration with proxies and VPN.

The tool was built on the basis of Scrapy architecture. [2]We use Scrapy to navigate to the page that will provide the accurate results. The tool initially gets the titles from a CSV file and for each title in the file, the tool opens Google Scholar, enters the title in the search box, selects the "Cited by" option to identify papers which cite these publications and selects the year "2020" to identify more recent publications. This will lead Google Scholar to give us recent and updated publications which cite the research papers that are in the pre-constituted corpus. From the results obtained, the tool will further identify the title of each paper that cites the publication and extract the title and link to each of the papers with which the paper can be downloaded. We employ Scrapy's CSS and XPath selectors to identify title and link of each citing publication and extract those results in the form of a CSV file.

2.2 Downloading the cited papers

1. Open the CSV file of cited papers from previous module
2. Convert the contents of CSV file into a data frame
3. For each link in the data frame
    - Identify the links with PDF extension
    - Enter the link
    - Click on save

To download the cited papers, we have to look if the extension of the link is PDF or not, if it's the case, we have used the two libraries requests and mimetypes, the first one is a simple HTTP library for python, with its method GET to retrieve the PDF from the link provided. The second library is a module that converts between a URL and the mime type associated with the filename extension, it helps us to see if a link has a PDF extension or not[9].

Next step, we download the paper if the link's extension is PDF, using a simple code with requests library, otherwise, we keep the link in a CSV file to download it manually.

If the extension of the link is not PDF, we manually download each paper using sci-hub if the link is not accessible.

We have tried to use Selenium and try to save the paper using Sci-hub without checking if the extension is PDF or not but we always faced the CAPTCHA problem[5], and when the link is not accessible by sci-hub it shows a blank page which stops the execution of the program.

3 Data Processing

Once the corpus is updated, a text mining tool is built to help us extract meaningful insights from the data collected. This process is further divided into two modules :

- Converting PDF to text
- Searching for companies' names and technique names

3.1 Converting PDF to text

The papers are stored in Google Drive to deal with storage constraints and access constraints. The papers stored in the drive are accessed individually in PDF format. We parse those files individually and store the contents in the format of a text file in Google Drive. We use Apache Tika library to parse the file content and store them in a text file. [5] Apache Tika is a library that is used for document type detection and content extraction from various file formats. Using this, one can develop a universal type detector and content extractor to extract both structured text and metadata from different types of documents such as spreadsheets, text documents, images, PDF's, and even multimedia input formats to a certain extent.

3.2 Searching for companies' names and technique names

After converting all the PDFs into text files we use a simple function that search for the occurrence of the company's name in text file, if the company's name is found we start searching for the technique name, and then if both are found we print the matched lines of the document. Following this,we manually read these lines and tried to figure out if that company is using that particular technique.

We have also tried implementing spaCy library, an open-source NLP library. Matcher, a rule-matching engine featured by spaCy would have helped us find what we are looking for in the text document. It works better than regular expressions because the rules can refer to annotations as well. The tool needs to be better refined to acquire accurate results.

4 Observation

- We were able to find a lot of instances of company name and technique name, but both were in different contexts.
- We could find a lot of familiar and popular company names in the academic literature
- Generic technique names like "artificial intelligence" or "optimization" or "descriptive", "predictive" and "prescriptive analytics" or just "analytics" or "big data" appeared a lot more

than specific technique names.

5 References

1. K. U. Manjari, S. Rousha, D. Sumanth and J. Sirisha Devi, "Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), 2020, pp. 648-652, doi: 10.1109/ICOEI48184.2020.9142938.
2. Asikri[1], M & Chaib, Hassan & Salah-ddine, Krit. (2020). Using Web Scraping In A Knowledge Environment To Build Ontologies Using Python And Scrapy. European Journal of Translational and Clinical Medicine. 7. 433-442.
3. Gunawan, Rohmat & Rahmatulloh, Alam & Darmawan, Irfan & Firdaus, Firman. (2019). Comparison of Web Scraping Techniques : Regular Expression, HTML DOM and Xpath. 10.2991/icoiese-18.2019.50.
4. ScraperAPI
5. How to handle CAPTCHA in Selenium
6. Apache Tika
7. Octoparse
8. Suganya, E., Vijayarani, S. Firefly Optimization Algorithm Based Web Scraping for Web Citation Extraction. Wireless Pers Commun 118, 1481–1505 (2021). https://doi.org/10.1007/s11277-021-08093-z
9. Rahmatulloh, Alam & Gunawan, Rohmat. (2020). Web Scraping with HTML DOM Method for Data Collection of Scientific Articles from Google Scholar. Indonesian Journal of Information Systems. 2. 16. 10.24002/ijis.v2i2.3029.
10. Scrapy documentation.