SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Data Scrapping

Vaibhav P. Vasani

Assistant Professor
Department of Computer Engineering
K. J. Somaiya College of Engineering
Somaiya Vidyavihar University

vaibhav.vasani@gmail.com

Somaiya TRUST

Data Scrapping

- Data Scrapping: Introduction, Need, Sources, Web Scrapping, Scrapping of Images, Data Wrangling, ETL Process

- Data scraping, also known as web scraping, is the process of importing information from a website into a spreadsheet or local file saved on your computer.

Web scraping, also called **web data mining** or **web harvesting**,

What is Web Scraping?

- Web scraping is an automated method used to extract large amounts of data from websites.
- The data on the websites are unstructured.
- Web scraping helps collect these unstructured data and store it in a structured form.
- There are different ways to scrape websites such as online Services, APIs or writing your own code.


edureka!
Apps
Development
XML
CSV
Webpages
Web Scraping
Structured Data

Why is Web Scraping Used?

- **Price Comparison:** Services such as ParseHub use web scraping to collect data from online shopping websites and use it to compare the prices of products.
- **Email address gathering:** Many companies that use email as a medium for marketing, use web scraping to collect email ID and then send bulk emails.
- **Social Media Scraping:** Web scraping is used to collect data from Social Media websites such as Twitter to find out what's trending.
- **Research and Development:** Web scraping is used to collect a large set of data (Statistics, General Information, Temperature, etc.) from websites, which are analyzed and used to carry out Surveys or for R&D.
- **Job listings:** Details regarding job openings, interviews are collected from different websites and then listed in one place so that it is easily accessible to the user.

Uses of data scraping include:

- Research for web content/business intelligence
- Pricing for travel booker sites/price comparison sites
- Finding sales leads/conducting market research by crawling public data sources (e.g. Yell and Twitter)
- Sending product data from an e-commerce site to another online vendor (e.g. Google Shopping)

list's just scratching the surface.

Is Web Scraping Legal?

Warning !!!!!

- The most prevalent misuse of data scraping is email harvesting – the scraping of data from websites, social media and directories to uncover people's email addresses, which are then sold on to spammers or scammers. In some jurisdictions, using automated means like data scraping to harvest email addresses with commercial intent is illegal, and it is almost universally considered bad marketing practice.

Web Crawling v/s Web Scraping

- Often used interchangeably
- Web crawling is basically used to index the information on the page using bots aka crawlers. It is also called **indexing**.
- web scraping is an automated way of extracting the information using bots aka scrapers. It is also called **data extraction**.

Web Crawling	Web Scraping
Refers to downloading and storing the contents of a large number of websites.	Refers to extracting individual data elements from the website by using a site-specific structure.
Mostly done on large scale.	Can be implemented at any scale.
Yields generic information.	Yields specific information.
Used by major search engines like Google, Bing, Yahoo. **Googlebot** is an example of a web crawler.	The information extracted using web scraping can be used to replicate in some other website or can be used to perform data analysis. For example the data elements can be names, address, price etc

Components of a Web Scraper

- A web scraper consists of the following components −
- Web Crawler Module
- A very necessary component of web scraper, web crawler module, is used to navigate the target website by making HTTP or HTTPS request to the URLs. The crawler downloads the unstructured data (HTML contents) and passes it to extractor, the next module.
- Extractor
- The extractor processes the fetched HTML content and extracts the data into semistructured format. This is also called as a parser module and uses different parsing techniques like Regular expression, HTML Parsing, DOM parsing or Artificial Intelligence for its functioning.
- Data Transformation and Cleaning Module
- The data extracted above is not suitable for ready use. It must pass through some cleaning module so that we can use it. The methods like String manipulation or regular expression can be used for this purpose. Note that extraction and transformation can be performed in a single step also.

- Data Transformation and Cleaning Module
  - The data extracted above is not suitable for ready use. It must pass through some cleaning module so that we can use it. The methods like String manipulation or regular expression can be used for this purpose. Note that extraction and transformation can be performed in a single step also.
- Storage Module
  - After extracting the data, we need to store it as per our requirement. The storage module will output the data in a standard format that can be stored in a database or JSON or CSV format.

Read Yourself Slide

Many web users have adopted techniques to help reduce the risk of email harvesters getting hold of their email address, including:

- Address munging: changing the format of your email address when posting it publicly, e.g. typing ‘patrick[at]gmail.com’ instead of ‘patrick@gmail.com’. This is an easy but slightly unreliable approach to protecting your email address on social media – some harvesters will search for various munged combinations as well as emails in a normal format, so it’s not entirely airtight.
- Contact forms: using a contact form instead of posting your email address(es) on your website.
- Images: if your email address is presented in image form on your website, it will be beyond the technological reach of most people involved in email harvesting.

The Data Scraping Future

- There are now data scraping **AI on the market that can use machine learning** to keep on getting better at recognising inputs which only humans have traditionally been able to interpret – like images.

- there’s the biggest data scraper of all – Google.

Why is Python Good for Web Scraping?

- Here is the list of features of Python which makes it more suitable for web scraping.
- **Ease of Use:** Python is simple to code. You do not have to add semi-colons ";" or curly-braces "{}" anywhere. This makes it less messy and easy to use.
- **Large Collection of Libraries:** Python has a huge collection of libraries such as Numpy, Matlplotlib, Pandas etc., which provides methods and services for various purposes. Hence, it is suitable for web scraping and for further manipulation of extracted data.
- **Dynamically typed:** In Python, you don't have to define datatypes for variables, you can directly use the variables wherever required. This saves time and makes your job faster.

Why is Python Good for Web Scraping?

- **Easily Understandable Syntax:** Python syntax is easily understandable mainly because reading a Python code is very similar to reading a statement in English. It is expressive and easily readable, and the indentation used in Python also helps the user to differentiate between different scope/blocks in the code.
- **Small code, large task:** Web scraping is used to save time. But what's the use if you spend more time writing the code? Well, you don't have to. In Python, you can write small codes to do large tasks. Hence, you save time even while writing the code.
- **Community:** What if you get stuck while writing the code? You don't have to worry. Python community has one of the biggest and most active communities, where you can seek help from.

How Do You Scrape Data From A Website?

- When you run the code for web scraping, a request is sent to the URL that you have mentioned. As a response to the request, the server sends the data and allows you to read the HTML or XML page. The code then, parses the HTML or XML page, finds the data and extracts it.
- To extract data using web scraping with python, you need to follow these basic steps:
  - Find the URL that you want to scrape
  - Inspecting the Page
  - Find the data you want to extract
  - Write the code
  - Run the code and extract the data
  - Store the data in the required format

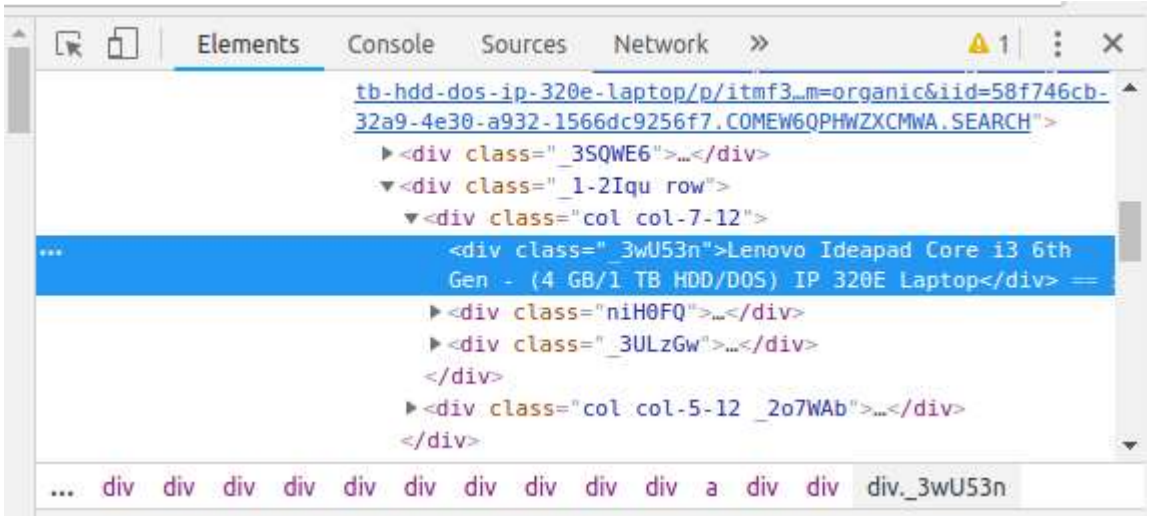
Libraries used for Web Scraping

- **Selenium**: Selenium is a web testing library. It is used to automate browser activities.
- **BeautifulSoup**: Beautiful Soup is a Python package for parsing HTML and XML documents. It creates parse trees that is helpful to extract the data easily.
- **Pandas**: Pandas is a library used for data manipulation and analysis. It is used to extract the data and store it in the desired format.
-

Web Scraping Example : Scraping Flipkart Website

- **Step 1: Find the URL that you want to scrape**
- For this example, we are going scrape **Flipkart** website to extract the Price, Name, and Rating of Laptops. The URL for this page is https://www.flipkart.com/laptops/~buyback-guarantee-on-laptops-/pr?sid=6bo%2Cb5g&uniqBStoreParam1=val1&wid=11.productCard.PMU_V2.

- **Step 2: Inspecting the Page**
- The data is usually nested in tags. So, we inspect the page to see, under which tag the data we want to scrape is nested. To inspect the page, just right click on the element and click on “Inspect”.
- When you click on the “Inspect” tab, you will see a “Browser Inspector Box” open.

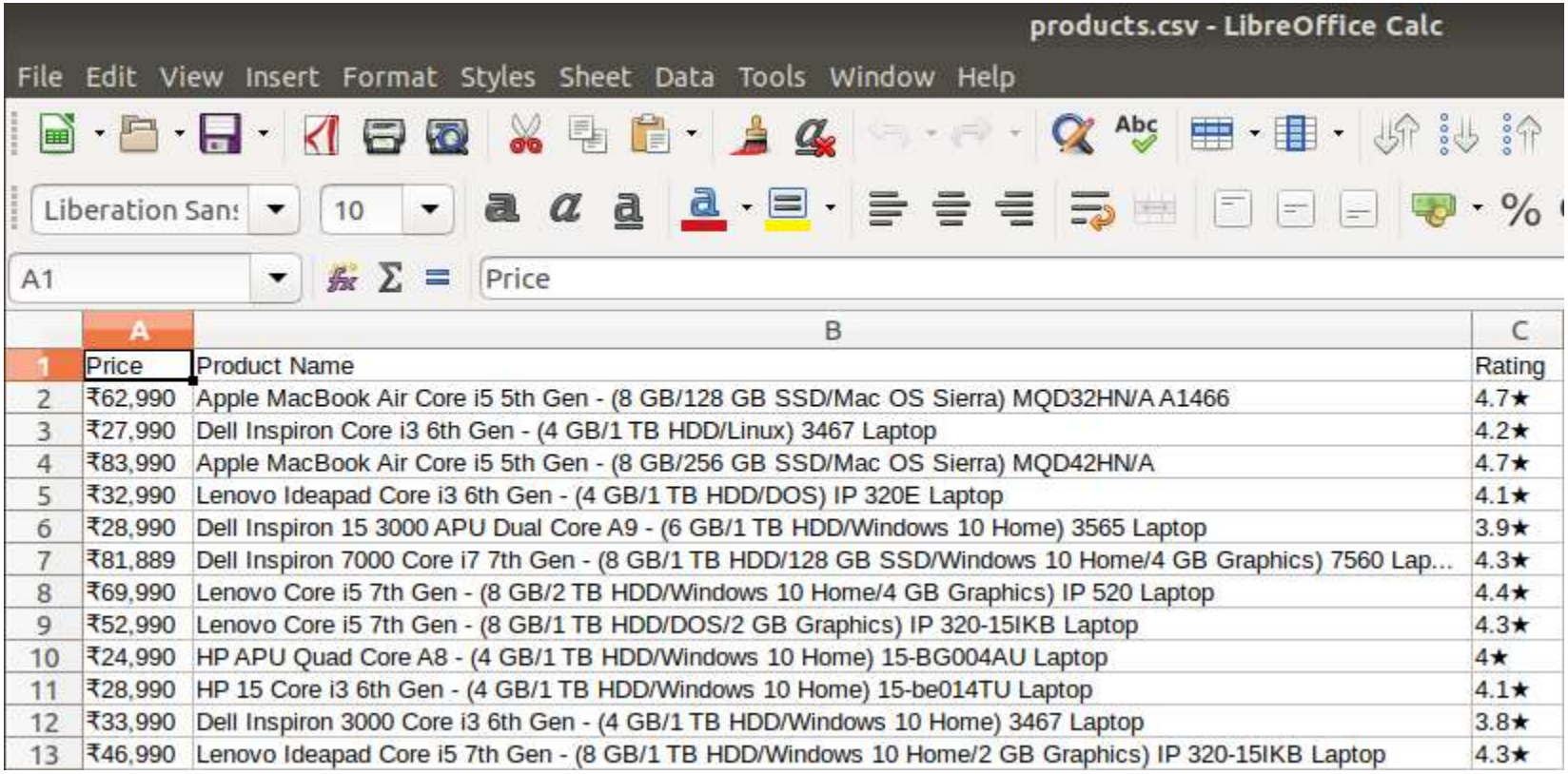


- **Step 3: Find the data you want to extract**
- Let's extract the Price, Name, and Rating which is in the "div" tag respectively.

- **Step 4: Write the code**
- First, let’s create a Python file.

- **Step 5: Run the code and extract the data**

- **Step 6: Store the data in a required format**
- After extracting the data, you might want to store it in a format. This format varies depending on your requirement. For this example, we will store the extracted data in a CSV (Comma Separated Value) format.



	A	B	C
1	Price	Product Name	Rating
2	₹62,990	Apple MacBook Air Core i5 5th Gen - (8 GB/128 GB SSD/Mac OS Sierra) MQD32HN/A A1466	4.7★
3	₹27,990	Dell Inspiron Core i3 6th Gen - (4 GB/1 TB HDD/Linux) 3467 Laptop	4.2★
4	₹83,990	Apple MacBook Air Core i5 5th Gen - (8 GB/256 GB SSD/Mac OS Sierra) MQD42HN/A	4.7★
5	₹32,990	Lenovo Ideapad Core i3 6th Gen - (4 GB/1 TB HDD/DOS) IP 320E Laptop	4.1★
6	₹28,990	Dell Inspiron 15 3000 APU Dual Core A9 - (6 GB/1 TB HDD/Windows 10 Home) 3565 Laptop	3.9★
7	₹81,889	Dell Inspiron 7000 Core i7 7th Gen - (8 GB/1 TB HDD/128 GB SSD/Windows 10 Home/4 GB Graphics) 7560 Lap...	4.3★
8	₹69,990	Lenovo Core i5 7th Gen - (8 GB/2 TB HDD/Windows 10 Home/4 GB Graphics) IP 520 Laptop	4.4★
9	₹52,990	Lenovo Core i5 7th Gen - (8 GB/1 TB HDD/DOS/2 GB Graphics) IP 320-15IKB Laptop	4.3★
10	₹24,990	HP APU Quad Core A8 - (4 GB/1 TB HDD/Windows 10 Home) 15-BG004AU Laptop	4★
11	₹28,990	HP 15 Core i3 6th Gen - (4 GB/1 TB HDD/Windows 10 Home) 15-be014TU Laptop	4.1★
12	₹33,990	Dell Inspiron 3000 Core i3 6th Gen - (4 GB/1 TB HDD/Windows 10 Home) 3467 Laptop	3.8★
13	₹46,990	Lenovo Ideapad Core i5 7th Gen - (8 GB/1 TB HDD/Windows 10 Home/2 GB Graphics) IP 320-15IKB Laptop	4.3★

First, let us import all the necessary libraries:

- from selenium import webdriver
- from BeautifulSoup import BeautifulSoup
- import pandas as pd

To configure webdriver to use Chrome browser, we have to set the path to chromedriver

- driver = webdriver.Chrome("/usr/lib/chromium-browser/chromedriver")

Refer the below code to open the URL:

  - products=[] #List to store name of the product
  - prices=[] #List to store price of the product
  - ratings=[] #List to store rating of the product
  - driver.get("https://www.flipkart.com/laptops/~buyback-guarantee-on-laptops-/pr?sid=6bo%2Cb5g&uniq")

Now that we have written the code to open the URL, it's time to extract the data from the website. As mentioned earlier, the data we want to extract is nested in tags. So, I will find the div tags with those respective class-names, extract the data and store the data in a variable. Refer the code below:

- content = driver.page_source
- soup = BeautifulSoup(content)
- for a in soup.findAll('a',href=True, attrs={'class':'_31qSD5'}):
- name=a.find('div', attrs={'class':'_3wU53n'})
- price=a.find('div', attrs={'class':'_1vC4OE _2rQ-NK'})
- rating=a.find('div', attrs={'class':'hGSR34 _2beYZw'})
- products.append(name.text)
- prices.append(price.text)
- ratings.append(rating.text)

#Store the data

- df = pd.DataFrame({'Product Name':products,'Price':prices,'Rating':ratings})
- df.to_csv('products.csv', index=False, encoding='utf-8')

- A web scraper (also known as web crawler) is a tool or a piece of code that performs the process to extract data from web pages on the Internet. ...
- Scrapy.
- Heritrix.
- Web-Harvest.
- MechanicalSoup.
- Apify SDK.
- Apache Nutch.
- Jaunt.

Image Scraping

- Scraping static pages
- Scraping interactive pages
- Scraping images from Google

Scraping static pages

- Scraping static pages (i.e., pages that don't utilize JavaScript to create a high degree of interaction on the page) is extremely simple. A static webpage is pretty much just a large file written in a markup language that defines how the content should be presented to the user. You can very quickly get the raw content without the markup being applied.

Scraping interactive pages

- Most modern web pages are quite interactive.
- The concept of “single-page application” means that the web page itself will change without the user having to reload or getting redirected from page to page all the time.
- Because this happens only after specific user interactions, there are few options when it comes to scraping the data (as those actions do have to take place).

- Sometimes the user action might trigger a call to an exposed backend API. In which case, it might be possible to directly access the API and fetch the resulting data without having to go through the unnecessary steps in-between. Most of the time, however, you will have to go through the steps of clicking buttons, scrolling pages, waiting for loads and all of that … or at least you have to make the webpage think **you** are doing all of that. Selenium to the rescue!

- **Selenium**
- Selenium can be used to automate web browser interaction with Python (also other languages).
- In layman's term, selenium pretends to be a real user, it opens the browser, "moves" the cursor around and clicks buttons if you tell it to do so.
- The initial idea behind Selenium, is automated testing.
- However, Selenium is equally powerful when it comes to automating repetitive web-based tasks.

Scraping images from Google

- Searching for a specific term & get image links
- Downloading the images

Data Wrangling

- Data wrangling is the process of cleaning and unifying messy and complex data sets for easy access and analysis.

- Process typically includes manually converting and mapping data from one raw form into another format to allow for more convenient consumption and organization of the data.

The Goals of Data Wrangling

- Reveal a "deeper intelligence" by gathering data from multiple sources
- Provide accurate, actionable data in the hands of business analysts in a timely matter
- Reduce the time spent collecting and organizing unruly data before it can be utilized
- Enable data scientists and analysts to focus on the analysis of data, rather than the wrangling
- Drive better decision-making skills by senior leaders in an organization

DATA WRANGLING STEPS

- 1. Discovery
- Discovery refers to the process of familiarizing yourself with data so you can conceptualize how you might use it. You can liken it to looking in your refrigerator before cooking a meal to see what ingredients you have at your disposal.
- During discovery, you may identify trends or patterns in the data, along with obvious issues, such as missing or incomplete values that need to be addressed. This is an important step, as it will inform every activity that comes afterward.
- 2. Structuring
- Raw data is typically unusable in its raw state because it's either incomplete or misformatted for its intended application. Data structuring is the process of taking raw data and transforming it to be more readily leveraged. The form your data takes will depend on the analytical model you use to interpret it.
- 3. Cleaning
- Data cleaning is the process of removing inherent errors in data that might distort your analysis or render it less valuable. Cleaning can come in different forms, including deleting empty cells or rows, removing outliers, and standardizing inputs. The goal of data cleaning is to ensure there are no errors (or as few as possible) that could influence your final analysis.

DATA WRANGLING STEPS

- 4. Enriching
- Once you understand your existing data and have transformed it into a more usable state, you must determine whether you have all of the data necessary for the project at hand. If not, you may choose to enrich or augment your data by incorporating values from other datasets. For this reason, it's important to understand what other data is available for use.
- If you decide that enrichment is necessary, you need to repeat the steps above for any new data.
- 5. Validating
- Data validation refers to the process of verifying that your data is both consistent and of a high enough quality. During validation, you may discover issues you need to resolve or conclude that your data is ready to be analyzed. Validation is typically achieved through various automated processes and requires programming.
- 6. Publishing
- Once your data has been validated, you can publish it. This involves making it available to others within your organization for analysis. The format you use to share the ainformation—such as a written report or electronic file—will depend on your data and the organization's goals.

Question ?