

# **WEB SCRAPING AND DATA ANALYSIS USING PYTHON**

**Assignment Report**

**Submitted by**

**Name: Vasavi Gangisetty**

**Roll Number: 23471A05DO**

**Branch: CSE**

**Section: C2**

**College Name: Narasaraopeta Engineering College**

```
def get_water_bottle_data(url):
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'}
    response = requests.get(url, headers=headers)
    soup = BeautifulSoup(response.text, 'lxml')
    water_bottles = []
    for item in soup.select('.product-item'):
        title = item.select_one('.product-item__title').text
        price = item.select_one('.product-item__price').text
        rating = item.select_one('.product-item__rating').text
        reviews = item.select_one('.product-item__reviews').text
        water_bottles.append((title, float(price), float(rating), int(reviews)))
    return water_bottles
```

The figure below shows a line chart comparing the price of different water bottles over time. The x-axis represents time in months, and the y-axis represents price in rupees. The chart displays several trends, with some bottles showing a steady increase in price while others remain relatively stable.

# Web Scraping of water bottle data

A Flipkart Product Analysis Case Study

This presentation delves into a comprehensive web scraping and data analysis project, focusing on Flipkart product data. We'll explore the process from initial data extraction to insightful visualisations, all powered by Python.

## OVERVIEW

# File Overview: WEBSCRAP.ipynb

The WEBSCRAP.ipynb Jupyter Notebook is a powerful tool designed to scrape water bottle product data from Flipkart across multiple pages (1 to 10). It's a full-stack solution for data acquisition and preliminary analysis.

- Extracts product information using robust web scraping techniques.
- Cleanses and structures raw data for optimal use.
- Stores data efficiently in Pandas DataFrames.
- Combines data from various pages into a single, cohesive dataset.
- Performs extensive exploratory data analysis (EDA).
- Answers 20 analytical questions with statistics and visualisations.
- Exports the refined dataset as a CSV file for further applications.

The screenshot shows a Jupyter Notebook interface with the title 'WEBSCRAP.ipynb'. The code cell contains Python code for web scraping, specifically for extracting data from Flipkart. The code uses BeautifulSoup to parse HTML, finds specific product details like 'name', 'rating', and 'price', and then stores this data in a Pandas DataFrame. It also includes logic to handle multiple pages (1 to 10) and perform some initial data cleaning and analysis. The code is well-structured with comments explaining the steps.

```
1 # Importing libraries
2 import requests
3 from bs4 import BeautifulSoup
4 import pandas as pd
5
6 # Function to extract product details
7 def extract_product_info(html):
8     soup = BeautifulSoup(html, 'lxml')
9     products = []
10    for item in soup.find_all('div', class_='col col-12'):
11        name = item.find('div', class_='product__title').text
12        rating = item.find('div', class_='product__rating').text
13        price = item.find('div', class_='product__price').text
14        products.append({'name': name, 'rating': rating, 'price': price})
15    return pd.DataFrame(products)
16
17 # Main function to scrape data
18 def scrape_water_bottles():
19     url = "https://www.flipkart.com/search?q=water+bottle"
20     headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36"}
21     df = pd.DataFrame()
22    for page in range(1, 11):
23        response = requests.get(url + f"&page={page}", headers=headers)
24        html_content = response.text
25        df_page = extract_product_info(html_content)
26        df = df.append(df_page)
27    return df
28
29 # Export to CSV
30 df = scrape_water_bottles()
31 df.to_csv("water_bottles.csv", index=False)
32
33 # Basic EDA
34 df.describe()
35
36 # Data Cleaning
37 df['rating'] = df['rating'].str.replace('Rating', '')
38 df['rating'] = df['rating'].str.replace('Avg.', '')
39 df['rating'] = df['rating'].str.replace('No reviews', '0')
40 df['rating'] = df['rating'].str.replace('No ratings', '0')
41 df['rating'] = df['rating'].str.replace('No reviews available', '0')
42 df['rating'] = df['rating'].str.replace('No ratings available', '0')
43 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
44 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
45 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
46 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
47 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
48 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
49 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
50 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
51 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
52 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
53 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
54 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
55 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
56 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
57 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
58 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
59 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
60 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
61 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
62 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
63 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
64 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
65 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
66 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
67 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
68 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
69 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
70 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
71 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
72 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
73 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
74 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
75 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
76 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
77 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
78 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
79 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
80 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
81 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
82 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
83 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
84 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
85 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
86 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
87 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
88 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
89 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
90 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
91 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
92 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
93 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
94 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
95 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
96 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
97 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
98 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
99 df['rating'] = df['rating'].str.replace('No reviews available yet', '0')
100 df['rating'] = df['rating'].str.replace('No ratings available yet', '0')
```

## OBJECTIVES

# *Key Objectives of This Analysis*



### *Master Web Scraping*

**Demonstrate advanced web scraping techniques without relying on Selenium.**



### *Collect Real-World Data*

**Gather authentic e-commerce data for practical analysis.**



### *Analyze Product Metrics*

**Examine pricing, ratings, discounts, and availability patterns.**



### *Practice Data Skills*

**Refine data cleaning, transformation, and visualisation capabilities.**



### *Create Usable Datasets*

**Generate a clean dataset suitable for dashboards or Machine Learning tasks.**

# Foundation: Python's Role in Data Science

Python serves as the backbone of this project, offering a versatile ecosystem essential for various data-related tasks. Its extensive libraries make it the go-to language for:

**Web Scraping:** Efficiently extracting data from websites.

**Data Analysis:** Performing complex computations and statistical analysis.

**Visualisation:** Creating compelling charts and graphs.





## LIBRARIES

# Requests Library: Fetching Web Content

### Purpose

To send HTTP requests to web servers and retrieve raw webpage content. It acts as the bridge between our Python script and the target website.

### Usage Example

```
import requests
page = requests.get(url)
```

This simple code snippet demonstrates how to initiate a GET request and store the response.

### Definition

Requests is a Python library that simplifies making HTTP requests, allowing developers to programmatically interact with web services and download HTML for further processing.



## LIBRARIES

# *BeautifulSoup (bs4): Parsing HTML*

### *Purpose*

To parse HTML and XML documents, enabling easy extraction of data by navigating the document as a tree structure.

### *Usage Example*

```
from bs4 import  
BeautifulSoup  
soup = BeautifulSoup(page.text,  
'html.parser')
```

This transforms raw HTML into a navigable object.

### *Definition*

**BeautifulSoup** is a Python library for pulling data out of HTML and XML files. It provides methods for searching and modifying the parse tree, making data extraction intuitive.



## LIBRARIES

# Regular Expressions (re): Text Cleaning

## Purpose

- 1 To define and apply patterns for advanced text searching, manipulation, and cleaning, crucial for standardising scraped data.

## Usage Example

```
import re
re.findall(r"\w+", text)
```

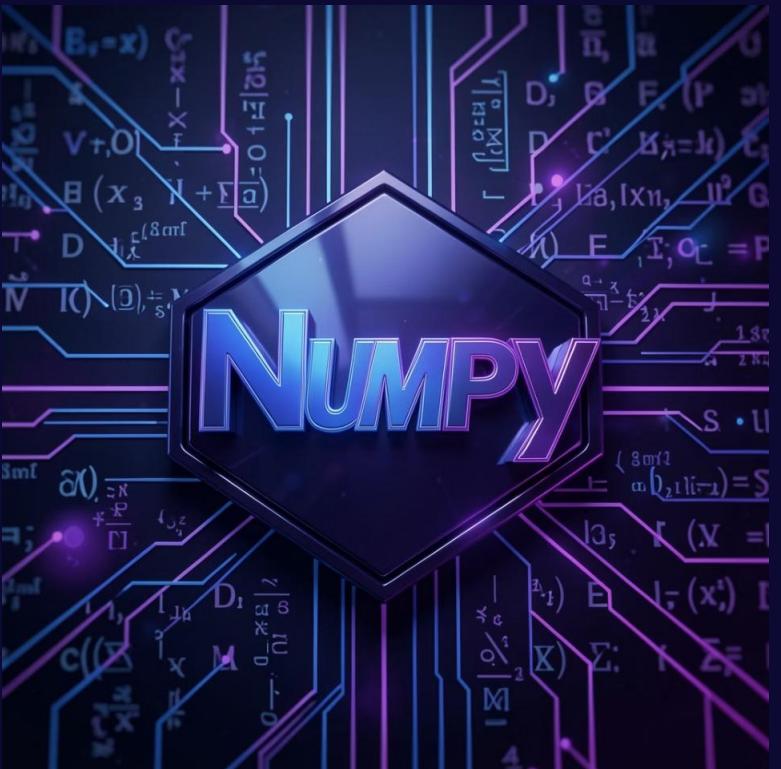
- 2

This example finds all word characters at the beginning of a string.

## Definition

- 3 **re** is Python's built-in module for regular expressions, enabling powerful pattern matching operations like searching, splitting, and replacing strings based on complex rules.

# NumPy & Pandas: The Data Handling Duo



## NumPy

**Purpose:** Essential for high-performance numerical operations, especially with arrays and matrices. It forms the mathematical foundation for many data science libraries.

**Usage Example:** `import numpy as np` Used for efficient calculations and handling large datasets.



## Pandas

**Purpose:** The cornerstone for data manipulation and analysis, providing powerful DataFrame and Series objects for structured data.

**Usage Examples:** `import pandas as pd`, `pd.DataFrame()`, `pd.concat()` Used for creating, combining, and querying tabular data.



# Matplotlib: Visualising Insights

## 1 Purpose

To create static, animated, and interactive visualisations in Python, making complex data interpretable through various plots.

## 2 Usage Example

```
import matplotlib.pyplot as plt
```

The primary interface for creating plots.

## 3 Charts Created

- Scatter plots to show relationships.
- Histograms for data distribution.
- Rating distributions to understand user feedback.

## 4 Definition

Matplotlib is a comprehensive library for creating static, animated, and interactive visualisations in Python, providing a vast array of plotting functions.



## DATA SCHEMA

# *Key Data Extracted: Product Features*

For each water bottle product scraped from Flipkart, the following crucial details are extracted and stored:

Brand	Product brand name
Rating	User rating (1–5 stars)
Price	Selling price
Original_Price	MRP before discount
Stock	Availability status
Ratings_Reviews_Count	Number of ratings and reviews
Discounts	Discount percentage

Thank You