E-Commerce Web Scraping

Name: Avani Gupta

- **Introduction:**

  The E-Commerce Web Scraping project involves extracting product details from a major e-commerce platform through meticulous web scraping. It focuses on key categories to gather information like product names, prices, descriptions, reviews, and image URLs for analysis. Emphasizing data cleanliness, consistency, and structured formatting, the project ensures compatibility with machine learning models for further refinement. Ethical considerations are upheld, ensuring adherence to website terms of service. Transparent documentation and precise execution aim to extract valuable insights, enabling informed decision-making and insightful analysis from the vast e-commerce dataset.

- **Objectives:**

  The objectives of the E-Commerce Web Scraping project are :

  - To extract detailed product information from Amazon.in
  - Organize the data into a structured format suitable for analysis(CSV file)
  - Ensure data cleanliness and consistency
  - Address ethical considerations in data collection
  - Coming up with suitable data frame to fine a LLM like GPT-4

- **Methodology:**

  1. **Website Selection:**
     The website selected for this project is Amazon.in as it is a popular e-commerce website with multiple product categories.

  2. **Data Extraction:**
     The process followed for scrapping product information is as follows:

     1. Category Selection: Choose two or more product categories from the e-commerce website, ensuring diversity and relevance to the project's objectives. The categories selected are watches, dresses and earrings.

2. HTTP Request: Use the requests library in Python to send HTTP GET requests to the URLs corresponding to the selected categories on the e-commerce website.

3. Response Handling: Check the response status code to ensure successful retrieval of the webpage content (status code 200). Handle cases where the status code indicates an error (e.g., status code 503).

4. HTML Parsing: Utilize BeautifulSoup library to parse the HTML content of the webpage and extract relevant information such as product names, prices, descriptions, ratings, reviews, and image URLs.

5. Iterative Extraction: Iterate through each product listing on the webpage within each category, extracting the desired information for each product. This includes:

- Product Name: Extract the name of the product from the HTML structure, typically found within specific HTML tags such as `<span>` or `<h2>` with appropriate attributes.
- Price: Extract the price of the product, usually located within designated HTML elements such as `<span>` with specific class attributes or within a specific HTML structure.
- Description: Extract the product description, which may be present within `<div>` or `<p>` tags with relevant attributes. This description provides additional details about the product.
- Reviews: Extract user reviews and ratings, typically found within designated HTML elements such as `<span>` or `<div>` with specific class attributes. This includes extracting both the review text and the associated rating.
- Image URL: Extract the URL of the product image, allowing for the retrieval and display of product images in the dataset.

6. Data Storage: Store the extracted product information in a structured format of a Pandas DataFrame, to facilitate further processing and analysis. Each row of the dataset represents a unique product, with columns corresponding to the extracted details (Product Name, Price, Description, Reviews, Image URL).

3. **Data Formatting**
The scraped data is structured into CSV format for compatibility with machine learning models like GPT-4. Each row represents a unique product, with columns for features such as name, price, description,

ratings, reviews, and image URLs. This tabular layout facilitates easy manipulation and analysis, providing a standardized input format for model training. Additionally, CSV ensures portability and accessibility, enabling integration into different machine learning frameworks. Overall, organizing the data in this structured format makes e-commerce website information readily available for analysis and model development.

4. **Challenges Faced**
    - Header issue – Defining and verifying my identity for scraping and to ensure to comply with the security policies of Amazon
    - Network Error – To handle the problem of Multiple requests in a short time span I added a code snippet that configures retry logic for connection errors and sends an HTTP GET request to the specified URL.
    - Connection Error – Slow internet speed, the packet travel time had to be increased in order to the packet to reach the destination IP address. Blocker resolved after increasing the *Packet Transmission Time* while performing the request (GET API call).

      Packet Transmission Time = Packet Size / Bit Rate

    - Error of HTTPSConnectionPool – some of the products around 2 out of 100 weren't able to cater to my requests of fetching the relevant data. Eventually, caught the error and threw it as an exception.
    - Status code after performing get request. The status code resulted in 503 – Error from the server-side, essentially the server is unable to cater to my requests
    - Fetching image URLs : While fetching image URLs care had to be taken to extract the vertical as well as the horizontal images.

- **Data Cleaning and Consistency:**

  Data cleaning and consistency are crucial aspects addressed to ensure the quality and usability of the extracted product information as follows:
    - Replace missing values with NaN (Not a Number) using the `replace()` method to standardize representation and ensure uniformity.
    - Remove "\nRead more" substring from each review text in the 'Top Review' column using a custom function and the `apply()` method to eliminate extraneous characters.
    - Rename the 'User Ratings' column to 'User Ratings (out of 5)' for clarity and remove the " out of 5" suffix from each value to maintain consistent formatting.
    - Rename the 'Price' column to 'Price(₹)' to explicitly denote the currency used, enhancing clarity and consistency.

  These measures collectively standardize the extracted product information, ensuring consistency and readiness for further analysis or model training. The cleaned and consistent data is saved to a CSV file, providing a structured and accessible format for downstream processing.

- **Ethical Considerations:**
  In this project several ethical considerations are addressed to ensure responsible web scraping practices.
  - The code demonstrates respect for the terms of service of the targeted e-commerce website by employing proper web scraping techniques. Specifically, the code utilizes a session object and HTTP adapter to handle multiple GET requests with retry logic, preventing excessive and disruptive scraping behaviour that could potentially overload the website's servers.
  - The code includes custom headers to mimic a web browser's user agent, promoting transparency and adherence to the website's guidelines.
  - By handling pagination and limiting the number of requests per category, the code mitigates the risk of overwhelming the website and respects its bandwidth limitations.
  - The code includes exception handling mechanisms to gracefully handle errors and exceptions, reducing the likelihood of disrupting the website's normal operation.

  Overall, these ethical considerations ensure that the web scraping process is conducted responsibly and in accordance with the website's terms of service, thereby fostering positive relationships with the website and its users.

- **Tools and Libraries used**

  1. Python Requests: Used for making HTTP requests to retrieve webpage content.
  2. BeautifulSoup (bs4): Utilized for parsing HTML content and extracting relevant information from webpages.
  3. requests.adapters.HTTPAdapter: Part of the Requests library, used to configure HTTP adapter settings for handling retries.
  4. urllib3.util.retry.Retry: Provides retry logic settings for handling connection errors during HTTP requests.
  5. numpy (np) and pandas (pd): These libraries are used for data manipulation and analysis, especially in handling and processing data after extraction.