Web Scraping Car Details from Cars24.com

Assigned Brand : Tata

Evoastra Mini Project

By

Team E

B Deepakjeswi (Team Lead)
Jaimon Jose (Co-Lead)
Abinarthana Jayakumar (Member)
Digvijay Ingale (Member)
Veerta Rajput (Member)
Urmi Thakkar (Member)
Chandan R (Member)
G Vishnu Chakradhar (Member)
Meda Srinivasa Rao (Member)

Introduction

The used car market presents a challenge for timely and accurate analysis due to its vast, dynamic, and often opaque nature, where manually gathering data on vehicle prices and inventory is highly inefficient. To address this, there is a clear need for structured data to accurately understand regional market dynamics and the key value drivers that influence vehicle pricing, such as kilometers driven, fuel type, transmission, and age. This mini-project was designed to directly tackle this problem by focusing on developing core Web Scraping and Data Engineering skills.

Our primary objective was to transform unstructured web data into a robust, analysis-ready dataset. This involved systematically extracting high-value information on pre-owned cars listed on the Cars24.com platform across three specific target locations. The data collection focused on five critical metrics: Price, Kilometers Driven, Year of Manufacture, Fuel Type, and Transmission. By converting this raw web content into structured data, this project successfully demonstrates proficiency in data acquisition and preparation. Ultimately, the successful completion of this project converts unstructured web content into actionable market intelligence suitable for deeper analysis.

Methodology

1. Research and Planning

- Tool Selection: Used Python, the Selenium library for HTML parsing, and the requests library for page retrieval.
- Target Analysis (DOM Inspection): Employed Browser Developer Tools to identify and map specific CSS selectors for the five required attributes: Price, Kilometers Driven, Year of Manufacture, Fuel Type, and Transmission.
- Scope Definition: Established logic for pagination handling and ensured complete coverage across the three specified regional entry points for the assigned vehicle brand.

2. Data Extraction

- Extraction Script: Developed a Python script to automate navigation and data collection, iterating through all available listing pages.
- Ethical Request Management: Incorporated a controlled delay (sleep function) between HTTP requests to prevent overloading the target server.
- Resilience and Error Handling: Integrated robust try-except blocks to manage missing data fields or connection errors, maximizing data completeness.

3. Data Cleaning

- Numeric Conversion: Systematically cleaned raw strings for Price and Kilometers Driven by removing non-numeric artifacts (e.g., currency symbols, "KMs" units) and converting them to appropriate numeric data types.
- Feature Engineering: Calculated the high-value feature Age of the Vehicle (in years) using the raw Year of Manufacture field.
- Data Structure: Loaded all cleaned and standardized fields into a Pandas DataFrame for highly organized, consistent, and type-correct output.

4. Data Presentation

- Storage Format: Exported the final cleaned and structured DataFrame into a Comma Separated Values (.csv) file.

- Compatibility: Selected the CSV format to ensure universal compatibility and immediate ingestion by statistical tools, BI platforms, and machine learning pipelines.

Team Roles and Responsibilities

1. Core Data Acquisition & Structuring

Responsible for all phases of data collection, including planning, scripting, extraction, and cleaning.

- Task: Web Scraping & Cleaning
    - Deliverable: Final Structured CSV File
    - Team Members: Vishnu, Jaimon, Deepakjeswi, Urmi

2. Reporting & Communication

Responsible for synthesizing the technical results into professional, client-facing formats.

- Project Report
    - Deliverable: Formal Analytical Report
    - Team Members: Abirnarthana Jayakumar , Digvijay Ingale
- Presentation (PPT)
    - Deliverable: Executive Summary Slides
    - Team Members: Veerta Rajput, Chandan R , Meda Srinivasa Rao

Tools and Technologies :

The project was created in Python as it is simple to work with and has many libraries around handling and scraping data from the web. The Jupyter Notebook interface was also used to build and run the code in an organized manner.

The following libraries and tools were used in the process:

- **Selenium WebDriver:** Used to automate web browsing and extract data from dynamically loaded pages on Cars24.com.
- **Pandas:** Used to organize, clean, and store structured data of the scraped data into a structured DataFrame for further analysis.
- **Time Library:** Used to add delays (time.sleep) so that pages fully load before data is extracted.
- **CSV Module:** Used to save cleaned and structured data set into a CSV format to present and analyze the data.
- **ChromeDriver Manager:** Simplified the setup of Chrome WebDriver, ensuring compatibility without manual driver installation.

All the tools and libraries worked together to make scraping, cleaning, and storing car data as smooth as possible while keeping accuracy and consistency in the data.

Challenges Faced and Solutions:

- **Dynamic Loading of Content:** The Cars24 website dynamically loads automobile listings via JavaScript. The conventional static scraping technique didn't function, so Selenium simulated the behavior of a genuine browser to load all elements completely before scraping.
- **Identifying Appropriate HTML Tags:** The HTML structures and the class names were complicated and occasionally inconsistent. Chrome Developer Tools were used by us to thoroughly examine every page and pick proper CSS selectors for attributes like price, kilometers, and transmission.

- **Data Cleaning:** Unwanted characters like "₹" and "km" had been included in the scraped data, and numerical values had been stored as strings. Pandas string functions and numeric conversion functions were utilized to clean and normalize the dataset.
- **Page Loading and Time Out Problems:** Certain pages were slower to load, resulting in partial scraping. Inserting a five-second delay (time.sleep(5)) prior to data extraction helped all car listings come through correctly.
- **Few Tata Listings in Mumbai:** Certain pages had only a few Tata cars listed. To address this, the script cycled through several pages (1 to 5) to collect enough entries to enable meaningful analysis.

Result:

This analysis, based on 50 Tata vehicles in the Mumbai used car market, clearly defines the current inventory profile and pricing structure. The data shows that the market supply is heavily weighted toward three main models: the Tata Harrier (10 cars), Tata Altroz (9 cars), and Tata Nexon (9 cars). Together, these models make up more than half (56%) of the total cars analyzed, demonstrating their strong presence in the used car inventory.

Regarding vehicle value, the market exhibits a wide price range, from a minimum of ₹2.45 lakh to a maximum of ₹15.70 lakh. The average price is ₹8.54 lakh, but the slightly lower median price of ₹7.55 lakh suggests that most listings fall toward the mid-range, with a few high-value, premium models pulling the average up.

The available cars are relatively new, with the most common manufacturing years being 2020 and 2021 (20 cars combined), meaning a significant portion of the inventory is less than five years old. Usage is typical for the segment, with an average mileage of 48.5k km. Finally, the technical distribution shows a clear preference for Petrol fuel types, which account for 48.0%, and a slight edge for Manual transmission, which makes up 54.0% of the total cars.

Conclusion:

This web scraping mini-project of Tata car details from Cars24.com provided useful hands-on experience with the processing of real-world, dynamically generated web data. Employing Selenium automation, the Team was able to extract vital car details like year of production, kilometers run, fuel type, transmission type and price successfully.

The project enhanced our knowledge of how to deal with JavaScript-based sites, determining and parsing HTML tags and cleaning incomplete data with Python libraries. It also stressed the significance of inserting proper delays and creating efficient scraping codes to prevent errors.

Overall, the project strengthened our technical background in data cleaning and collection, giving us practical experience in one of the most frequent tasks in data science pipelines.