Team E (Mini Project)

Team Leads	Co-Leads
Vaibhavi Kapse	Nandita Kaushik
	Akshar

Team Members

Aachal Gupta	Ketan Kumar Sendre
Aarjav Jain	Alfed Khan
Pabitra Goswami	Juweriya Shayhmeen
Saili Ashok Dhuri	Dayal Kumar Sarkar
Pratham Jindal	Ashmit Zanzote
Shubham Arvind Rangari	Kunal Kumbhar
Abeed Mohammed	Sudarshan Sopane

Project Name: *Web Scraping of Tata Cars in Mumbai from Cars24.com*

Steps Performed by Each Team Member

Task Allotment:

- **Research and Planning**
  - Nandita and Akshar
- **Data Extraction**
  - Team Alpha: Aarjav, Alfed, Shubham, Nandita
  - Team Beta: Pratham, Pabitra, Saili, Akshar
- **Data Cleaning**
  - Ketan, Aachal, Kunal
- **Data Presentation**
  - Juweriya, Abeed, Sudarshan
- **Data Analysis**
  - Vaibhavi
- **Quality Assurance**
  - Ashmit, Dayal Kumar
- **Reporting**
  - Saili, Aachal, Pabitra, Nandita
- **PowerPoint Presentation**
  - Shubham, Ketan, Akshit, Akshar

Web Scraping using BeautifulSoup and Selenium

```
# Import necessary libraries
from selenium import webdriver
from bs4 import BeautifulSoup
import pandas as pd
import time

# Initialize Chrome WebDriver
driver = webdriver.Chrome()

# Navigate to the Cars24 website with the specified filters
url = 'https://www.cars24.com/buy-used-car?f=make%3A%3D
%3Atata&sort=bestmatch&serveWarrantyCount=true&storeCityId=2378'
driver.get(url)

# Wait for the page to load completely
time.sleep(5) # adjust this delay based on your internet speed and
page loading time

# Scroll down to the bottom of the page to load more results (if any)
driver.execute_script("window.scrollTo(0,
document.body.scrollHeight);")
time.sleep(5)  # wait for new elements to load

# Get the page source after scrolling
page_source = driver.page_source

# Parse the page source using BeautifulSoup
soup = BeautifulSoup(page_source, 'html.parser')

# Find all car results on the page
results = soup.find_all('div', {'class': '_2YB7p'})
print(f'Total results: {len(results)}')

# Initialize an empty list to store car data
car_data = []

# Iterate over each car result and extract relevant information
for result in results:
    # Extract car name
    name = result.find('h3', {'class': '_11dVb'}).get_text() if
result.find('h3', {'class': '_11dVb'}) else 'N/A'

    # Extract kilometers driven
    kilometers = result.find('ul', {'class': '_3J2G-'}).find_all('li')
[0].get_text() if result.find('ul', {'class':
'_3J2G-'}).find_all('li') else 'N/A'

    # Extract fuel type
    fuel = result.find('ul', {'class': '_3J2G-'}).find_all('li')
[2].get_text() if len(result.find('ul', {'class':
```

```
'_3J2G-'}).find_all('li')) > 2 else 'N/A'

    # Extract transmission type
    transmission = result.find('ul', {'class':
'_3J2G-'}).find_all('li')[4].get_text() if len(result.find('ul',
{'class': '_3J2G-'}).find_all('li')) > 4 else 'N/A'

    # Extract price
    price = result.find('strong', {'class': '_3RL-I'}).get_text() if
result.find('strong', {'class': '_3RL-I'}) else 'N/A'

    # Extract EMI (if available)
    emi = result.find('span', {'class': '_2O0yU'}).get_text() if
result.find('span', {'class': '_2O0yU'}) else 'N/A'

    # Append extracted data as a dictionary to car_data list
    car_data.append({
        'Name': name,
        'Kilometers Driven': kilometers,
        'Fuel': fuel,
        'Transmission': transmission,
        'Price': price,
        'EMI': emi
    })

# Close the WebDriver
driver.quit()

# Create a DataFrame from car_data list
df = pd.DataFrame(car_data)

# Specify the path where you want to save the CSV file
csv_path = 'C:\\Users\\Vaibhavi\\Desktop\\tata-vk.csv' # Replace with
your desired path

# Save the DataFrame to a CSV file without including the index
df.to_csv(csv_path, index=False)

# Print confirmation message
print(f'Data saved to {csv_path}')
```

Output:

```
Total results: 36
Data saved to C:\Users\Vaibhavi\Desktop\tata-vk.csv
```

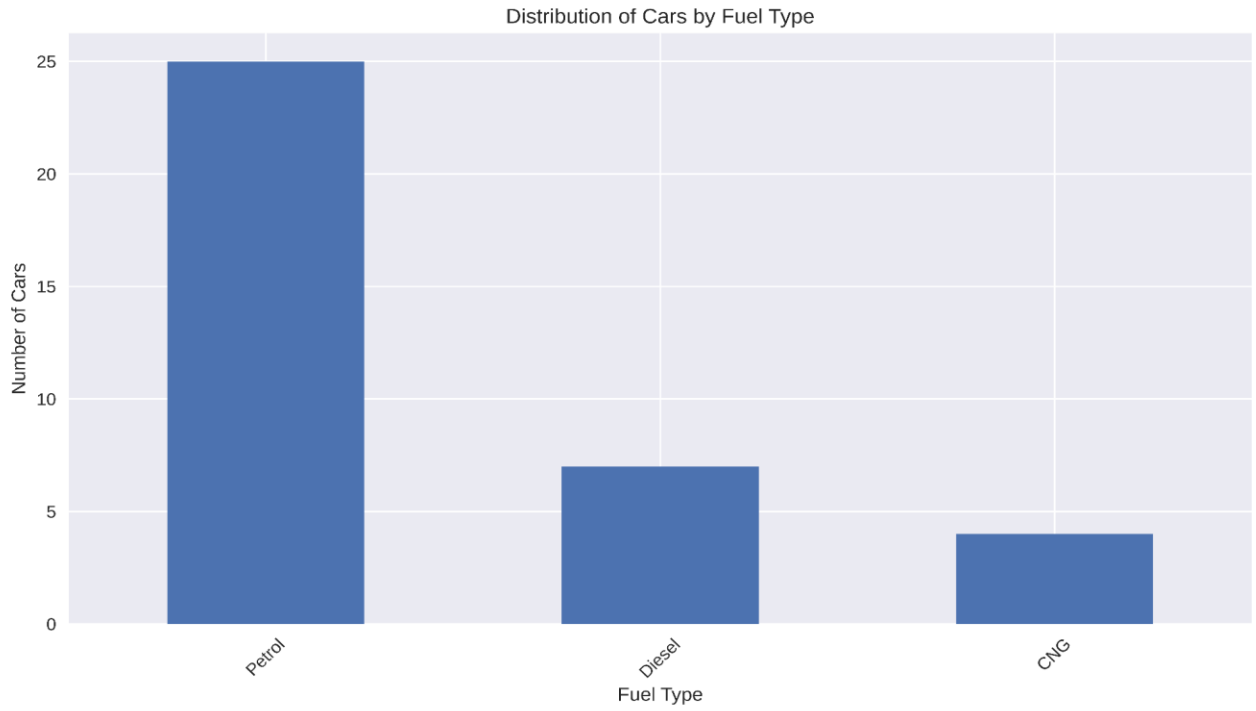
CSV file as Output of TATA Cars in Mumbai from Cars24.com

Name	Year of Manufacture	Distance Driven(in Km)	Fuel Type	Transmission	Price(in Lakhs)	EMI (₹/ month)
Tata Tiago XT PETROL	2017	24,888	Petrol	Manual	4.02	7,859
Tata Tiago XZ PETROL	2017	37,602	Petrol	Manual	4.12	8,055
Tata TIGOR XZ PETROL	2018	94,586	Petrol	Manual	4.2	8,211
Tata NEXON XZA PLUS DIESEL DUAL TONE	2018	1,04,792	Diesel	Automatic	6.96	15,482
Tata NEXON XZA PLUS PETROL	2018	26,184	Petrol	Automatic	7.75	15,151
Tata Tiago XZ PETROL	2018	76,127	Petrol	Manual	4.24	8,289
Tata Tiago XZ PLUS PETROL	2018	36,638	CNG	Manual	4.63	9,052
Tata Tiago XZ PETROL	2018	90,317	Petrol	Manual	4.1	8,016
Tata Tiago XZA PETROL	2019	32,045	Petrol	Automatic	5.36	10,479
Tata Harrier XZ 2.0L	2019	52,045	Diesel	Manual	12.68	24,135
Tata Harrier XM 2.0L KRYOTEC	2019	44,461	Diesel	Manual	10.63	20,233
Tata TIGOR XZ PLUS PETROL	2019	87,229	Petrol	Manual	4.91	9,599
Tata Tiago XZA PLUS PETROL	2019	18,218	Petrol	Automatic	5.59	10,929
Tata NEXON XZA PLUS PETROL	2019	61,806	Petrol	Automatic	7.73	15,112
Tata NEXON XZA PLUS DIESEL	2019	51,506	Diesel	Automatic	8.53	16,236
Tata ALTROZ XZ PETROL	2020	47,992	Petrol	Manual	6.54	12,786
Tata ALTROZ XZ PETROL	2020	71,356	Petrol	Manual	6.9	13,490
Tata ALTROZ XZ PETROL	2020	11,670	Petrol	Manual	7.07	13,828
Tata ALTROZ XZ PETROL	2020	27,102	Petrol	Manual	7.15	13,978
Tata ALTROZ XZ PETROL	2020	23,825	Petrol	Manual	7.37	14,408
Tata Tiago XT PETROL	2020	42,969	Petrol	Manual	4.67	9,130
Tata ALTROZ XZ PLUS PETROL	2021	13,348	Petrol	Manual	7.7	15,054
Tata NEXON XZ PETROL	2021	13,342	Petrol	Manual	8.8	16,750
Tata Tiago XZA PLUS DUAL TONE PETROL	2021	34,270	Petrol	Automatic	6.02	11,763
Tata Tiago XZA PLUS PETROL	2021	42,528	Petrol	Automatic	5.34	10,440
Tata Safari XZ PLUS	2021	16,603	Diesel	Manual	19.21	36,564
Tata TIGOR XZA PLUS PETROL	2021	39,046	Petrol	Automatic	7.04	13,763
Tata TIGOR XZ PLUS CNG	2022	24,490	CNG	Manual	7.89	15,425
Tata NEXON XZA PLUS DIESEL KAZIRANGA	2022	37,027	Diesel	Automatic	12.65	24,078
Tata TIGOR XZ PLUS CNG	2022	5,837	CNG	Manual	8.19	15,589
Tata PUNCH CREATIVE 1.2 AMT KAZIRANGA EDITION	2022	21,714	Petrol	Automatic	8.14	15,494
Tata TIGOR XZ PLUS CNG	2022	18,438	CNG	Manual	7.33	14,330
Tata ALTROZ XZA PLUS	2022	2,973	Petrol	Automatic	9.15	17,416
Tata NEXON XZ PLUS PETROL	2022	18,598	Petrol	Manual	9.17	17,454
Tata Safari XZA PLUS ADVENTURE	2022	42,408	Diesel	Automatic	20.27	38,582
Tata NEXON XZA PLUS PETROL	2023	6,437	Petrol	Automatic	10.69	20,347

TATA Cars in Mumbai on cars24.com

Analysis and Insights

1. Distribution of Cars by Fuel Type



Petrol cars dominate the dataset with 25 cars compared to 7 diesel cars and 4 CNG cars. The bar chart provides a clear visual comparison of the distribution of cars by fuel type, showing the predominance of petrol cars.

2. Price vs Kilometers Driven by Fuel Type

This scatter plot illustrates the relationship between the price of cars and the kilometers driven with different colors representing fuel types. We can observe that there's a general trend of lower prices for cars with higher mileage, which is expected. The plot also shows that diesel cars tend to be priced higher than petrol cars, especially for lower mileage vehicles.

3. Average Price by Transmission Type

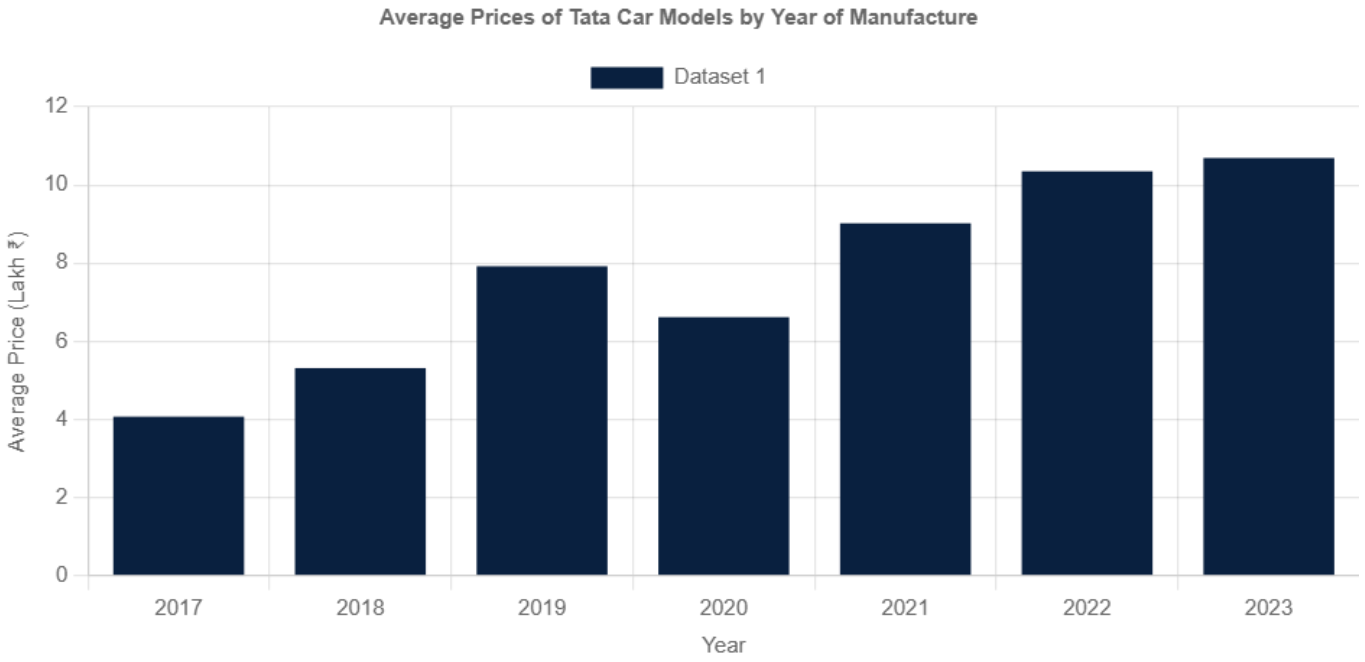
Transmission	Price
Automatic	8.65857142857143
Manual	7.341818181818183

This data shows that automatic transmission cars have a higher average price (₹ 8.66 Lakh) compared to manual transmission cars (₹ 7.34 Lakh). This suggests that automatic transmission is considered a premium feature in the Tata cars market in Mumbai.

4. Correlation between Year and Price

The correlation coefficient between the year of manufacture and the price of the car is 0.51, indicating a moderate positive correlation. This suggests that newer models tend to be more expensive, which is a common trend in the automotive market.

5. Average Prices of Tata Car Models by Year of Manufacture



This graph visually represents the average prices of Tata cars for each year from 2017 to 2023. We can see a general upward trend in prices over the years.

year	Price
2017	4.07
2018	5.3133333333333335
2019	7.918571428571428
2020	6.616666666666667
2021	9.018333333333333
2022	10.348749999999999
2023	10.69

As we can see:

- 2017 models have the lowest average price at ₹ 4.07 Lakh.
- There's a significant jump in average price for 2019 models (₹ 7.92 Lakh).
- The most recent years (2021-2023) show the highest average prices, with 2023 models averaging ₹ 10.69 Lakh.

It's important to note that these averages might be influenced by the specific models available for each year and the number of cars in the dataset for each year.

	Price
count	36.0
mean	7.853888888888889
std	3.684949526909835
min	4.02
25%	5.355
50%	7.24
75%	8.5975
max	20.27

These statistics give us a broader view of the price range:

- The average price across all years is about ₹ 7.85 Lakh.
- The cheapest car in the dataset is priced at ₹ 4.02 Lakh.
- The most expensive car is priced at ₹ 20.27 Lakh.
- 50% of the cars (median) are priced at ₹ 7.24 Lakh or below.

This analysis provides a comprehensive view of how Tata car prices vary by year of manufacture in the Mumbai market based on the data provided.

Web Scraping Experience Report

Introduction

Our team embarked on a project to web scrape car details from the Cars24.com website, specifically targeting Tata brand cars in the Mumbai location. This task aimed to extract, clean, and compile data into CSV files for further analysis. Throughout this process, we gained valuable experience and faced several challenges, ultimately enhancing our understanding and skills in web scraping.

Steps Performed

Initial Setup: We began by setting up our development environment, installing necessary libraries such as BeautifulSoup and Selenium. We also ensured our web drivers were correctly configured to work seamlessly with our browsers.

Data Extraction: Our initial approach involved using BeautifulSoup to parse HTML content and extract relevant data. We wrote scripts to navigate through the website, identify the structure of the HTML elements, and capture the required information such as car model, price, mileage, and other specifications.

Overcoming Challenges: During the extraction process, we encountered a significant challenge: BeautifulSoup alone was not sufficient to extract all elements, especially those loaded dynamically through JavaScript. To address this, we integrated Selenium with our scripts. Selenium, with its web driver capabilities, allowed us to automate browser actions, handle JavaScript-loaded content, and ensure we could capture all necessary data.

Data Cleaning and Transformation: Once the data was extracted, we faced the task of cleaning and transforming it into a structured format. We handled missing values, standardized formats, and ensured consistency across our dataset. This step was crucial to prepare the data for analysis and storage in CSV files.

Quality Assurance: Our quality assurance team, consisting of Ashmit and Dayal Kumar, conducted thorough checks to ensure data accuracy and completeness. They compared the extracted data with the original website content and identified any discrepancies, which were promptly addressed.

Reporting and Presentation: Our reporting team, including Saili, Aachal, and Pabitra, compiled the findings into detailed reports, highlighting key insights and trends. Shubham, Ketan, and Akshit prepared PowerPoint presentations to communicate our results effectively to stakeholders.

Experience Gained

This project provided us with hands-on experience in web scraping, a valuable skill in data science and analytics. We learned to navigate complex web structures, handle dynamically loaded content, and automate repetitive tasks. Additionally, working with both BeautifulSoup and Selenium expanded our technical toolkit, allowing us to choose the most appropriate tool for different scenarios.

Lessons Learned

Technical Skills: We enhanced our proficiency in Python programming, particularly in using libraries such as BeautifulSoup and Selenium. We also gained a deeper understanding of HTML and web technologies, enabling us to efficiently parse and manipulate web content.

Problem-Solving: The challenges we faced taught us the importance of adaptability and problem-solving. When BeautifulSoup alone was insufficient, integrating Selenium demonstrated our ability to pivot and find effective solutions to complex problems.

Collaboration: This project underscored the importance of teamwork and communication. Each team member's contribution was vital to the project's success, and our coordinated efforts ensured a smooth workflow from data extraction to final presentation.

Challenges Faced

One of the primary challenges was handling dynamically loaded content. Many websites, including Cars24.com, use JavaScript to load data after the initial page load. BeautifulSoup, being a static HTML parser, struggled with these elements. Integrating Selenium, a tool designed for automating web browsers, allowed us to overcome this obstacle. Additionally, managing large volumes of data and ensuring its accuracy required meticulous attention and rigorous quality checks.

Thoughts on Web Scraping

Web scraping is a powerful technique for data collection, enabling access to vast amounts of information available on the internet. However, it comes with its challenges, including dealing with dynamic content, ensuring data accuracy, and adhering to legal and ethical guidelines. Despite these challenges, web scraping remains an invaluable skill for data analysts, researchers, and businesses seeking to harness the power of data for informed decision-making.

Conclusion

Our journey through this web scraping project was both challenging and rewarding. We developed technical skills, learned to navigate obstacles, and appreciated the value of teamwork. This experience has equipped us with the knowledge and confidence to tackle future web scraping projects, leveraging data to uncover insights and drive innovation.