Web Scraping Project Report: Hatla2ee Egyptian Car Site

Amr Khaled Abdeltawab, ID:224363, MSA University
Ahmed Mohamed Mounir, ID:224643, MSA University

May 20, 2024

Contents

Chapter 1

Introduction

1.1 Chosen Topic

The chosen topic for this project is the web scraping of the Hatla2ee Egyptian car site. Hatla2ee is a popular online marketplace for buying and selling new and used cars in Egypt. The site hosts a vast amount of data on various car models, prices, locations, and other attributes, making it a valuable resource for analysis.

Web scraping involves extracting large amounts of data from websites, and in this project, it allows us to collect real-time data on the car market in Egypt. The data gathered includes car titles, model names, colors, mileage, locations, listing dates, prices, and images. By systematically extracting and analyzing this data, we aim to uncover patterns and insights that can inform various stakeholders, including car buyers, sellers, and market analysts.

Additionally, we utilize machine learning techniques, specifically linear regression, to predict car prices based on the collected data. This adds a predictive analytics component to our project, enhancing the practical applications of our findings.

1.2 Objectives

The main objectives of this project are:

- To extract data from the Hatla2ee website using web scraping techniques. This involves navigating the site, identifying relevant data elements, and systematically extracting this information into a structured format.

- To analyze the extracted data to gain insights into the car market in Egypt. This includes identifying trends in car prices, popular models and colors, and the impact of mileage and location on car prices.

- To present the findings using data visualization tools, specifically Power BI. Visualization helps in summarizing complex data into digestible insights, making it easier to communicate our findings.

- To implement a linear regression model to predict car prices. This involves preprocessing the data, training the model, and evaluating its performance. The model helps in understanding the relationship between car prices and various features, providing a predictive tool for estimating car values.

Chapter 2

Methodology

2.1 Planned Approach for Data Scraping

The data scraping process involves several steps:

1. Identifying the target URL: The base URL for the car listings on Hatla2ee, specifically for Mercedes cars, was chosen as `https://eg.hatla2ee.com/ar/car/mercedes`.
2. Setting up the scraping environment: Using Python with the BeautifulSoup and Requests libraries to handle HTTP requests and parse HTML content.
3. Extracting data: Writing a script to extract specific details about each car, such as title, model name, color, mileage, location, date, price, and image.
4. Saving the data: Storing the extracted data in a CSV file for further analysis.

2.2 Adhering to Robots.txt Restrictions

While scraping the Hatla2ee website, it was crucial to adhere to the restrictions presented in the site's robots.txt file. The robots.txt file specifies which parts of the site can be accessed by web crawlers and which parts are restricted. Our approach to working with these restrictions included:

- Carefully reviewing the robots.txt file to identify any disallowed paths and ensuring our scraping script avoided these areas.
- Implementing a polite scraping strategy by adding delays between requests to avoid overloading the server and to comply with the site's rate limits.
- Respecting the site's terms of service and ensuring that the scraping activities were conducted ethically and legally.

The robots.txt file can be accessed at `https://eg.hatla2ee.com/robots.txt`, and it provides guidance on which parts of the site are accessible to web crawlers.

2.3 Code Implementation

The following Python code was used to scrape the car data from the Hatla2ee website:

```
import requests
from bs4 import BeautifulSoup
import csv
import time

def get_soup(url):
    retries = 5
    backoff_factor = 1

    for attempt in range(retries):
        try:
            response = requests.get(url)
            response.raise_for_status()
            return BeautifulSoup(response.content, ’html.parser’)
        except requests.exceptions.RequestException as e:
            if attempt < retries - 1:
                time.sleep(backoff_factor * (2 ** attempt))
                continue
            else:
                print(f"Failed to retrieve {url}: {e}")
                return None

def scrape_car_data(base_url, num_pages):
    car_data = []

    for page in range(1, num_pages + 1):
        url = f"{base_url}/page/{page}"
        soup = get_soup(url)

        if soup is None:
            continue

        for car in soup.find_all(’div’, class_=’newCarListUnit_wrap’):
            title = car.find(’div’, class_=’newCarListUnit_header’).find(’a’).text.st
            car_model_name = car.find(’div’, class_=’newCarListUnit_metaTags’).find_a
            car_color = car.find(’span’, class_=’newCarListUnit_metaTag mob_hidden’).
            car_mileage = ""
            meta_tags = car.find_all(’span’, class_=’newCarListUnit_metaTag’)
            for tag in meta_tags:
                if "" in tag.text:
                    car_mileage = tag.text.strip()
                    break
            location = car.find(’div’, class_=’newCarListUnit_metaTags’).find_all(’sp
            date = car.find(’div’, class_=’otherData_Date’).find(’span’).text.strip()
            price = car.find(’div’, class_=’main_price’).find(’a’).text.strip()
            image = car.find(’img’, class_=’lazy’)[’data-original’]

            car_data.append({
                ’title’: title,
                ’car_model_name’: car_model_name,
                ’car_color’: car_color,
                ’car_mileage’: car_mileage,
                ’location’: location,
                ’date’: date,
                ’price’: price,
                ’image’: image
            })

        if len(car_data) >= 1000:
            break

    return car_data[:1000]

def save_to_csv(data, filename):
    with open(filename, mode=’w’, newline=’’, encoding=’utf-8-sig’) as file:
        writer = csv.DictWriter(file, fieldnames=[’title’, ’car_model_name’, ’car_col
        writer.writeheader()
        writer.writerows(data)

    print(f’Data has been written to {filename}’)

def download_csv(base_url, num_pages, filename):
    data = scrape_car_data(base_url, num_pages)
    save_to_csv(data, filename)

base_url = ’https://eg.hatla2ee.com/ar/car/mercedes’
num_pages = 50
csv_file = ’car_data.csv’
download_csv(base_url, num_pages, csv_file)
```

Chapter 3

Data Analysis

3.1 Power BI Analysis

The data extracted from the Hatla2ee website was analyzed using Power BI. The key findings and visualizations include:

- Distribution of car prices across different models and locations.
- Trends in car listings over time.
- Analysis of car mileage and its impact on price.
- Visualization of car colors and their popularity.

3.2 Analysis Snippets

Below are some snippets from the Power BI analysis:

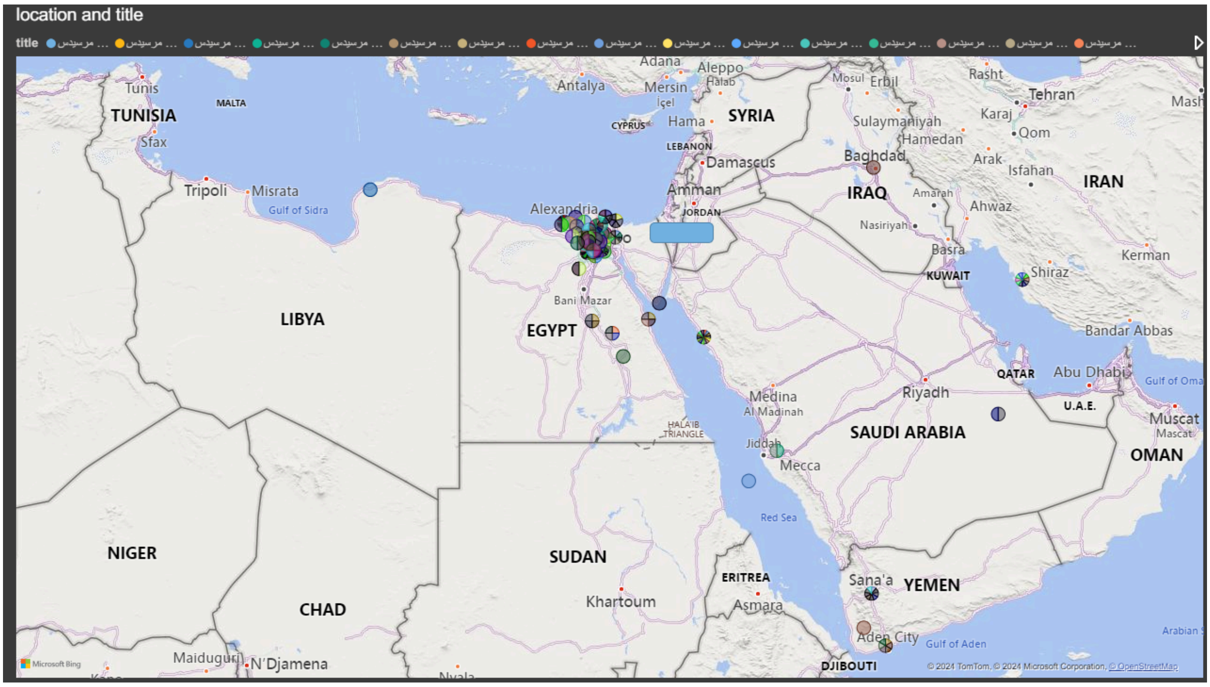

Figure 3.1: Distribution of Car Prices Across Different Models and Locations

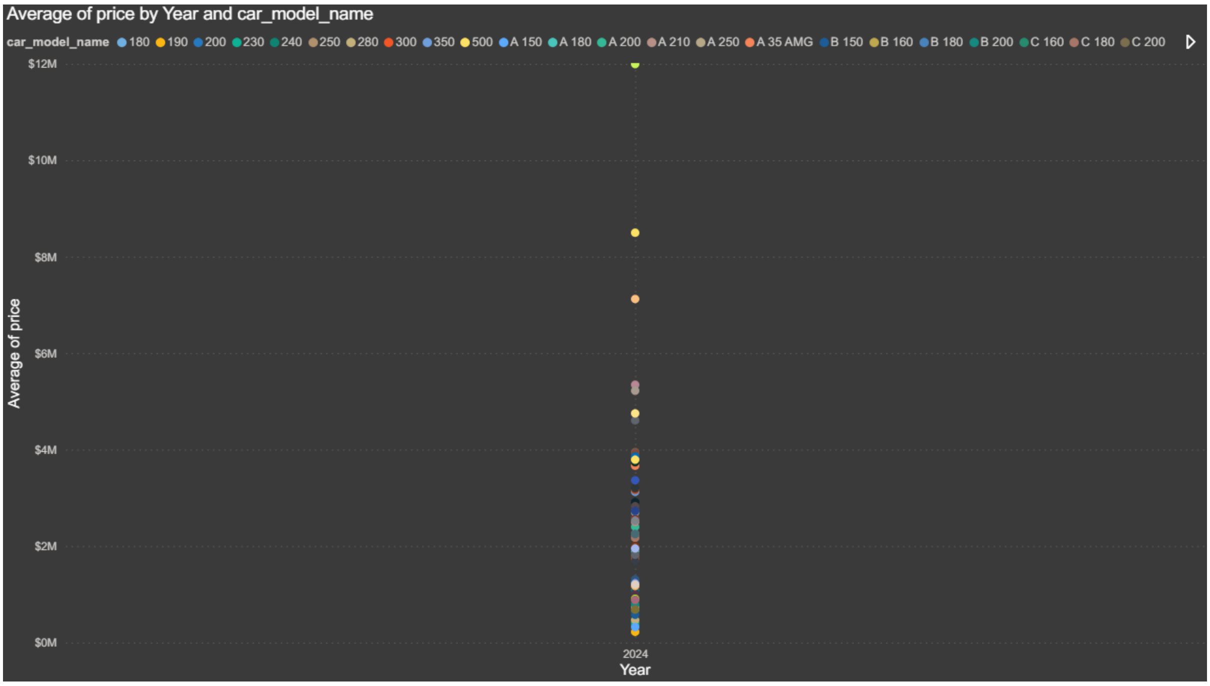

Figure 3.2: Trends in Car Listings Over Time

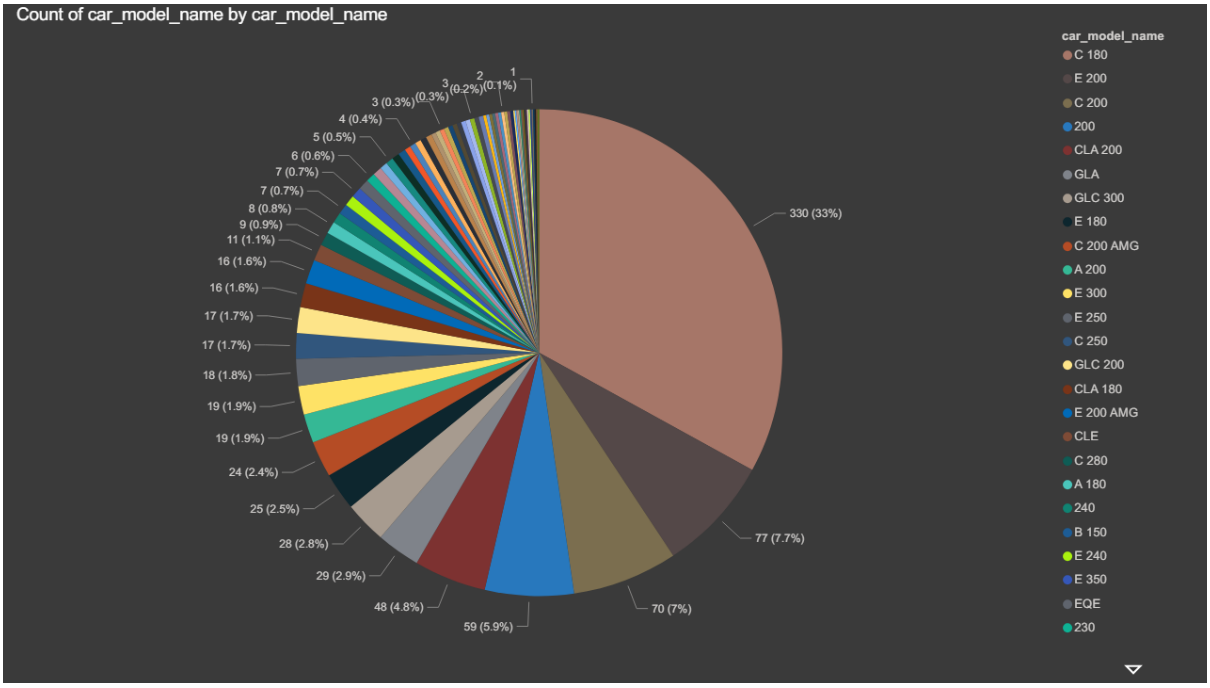

Figure 3.3: Analysis of Models Count

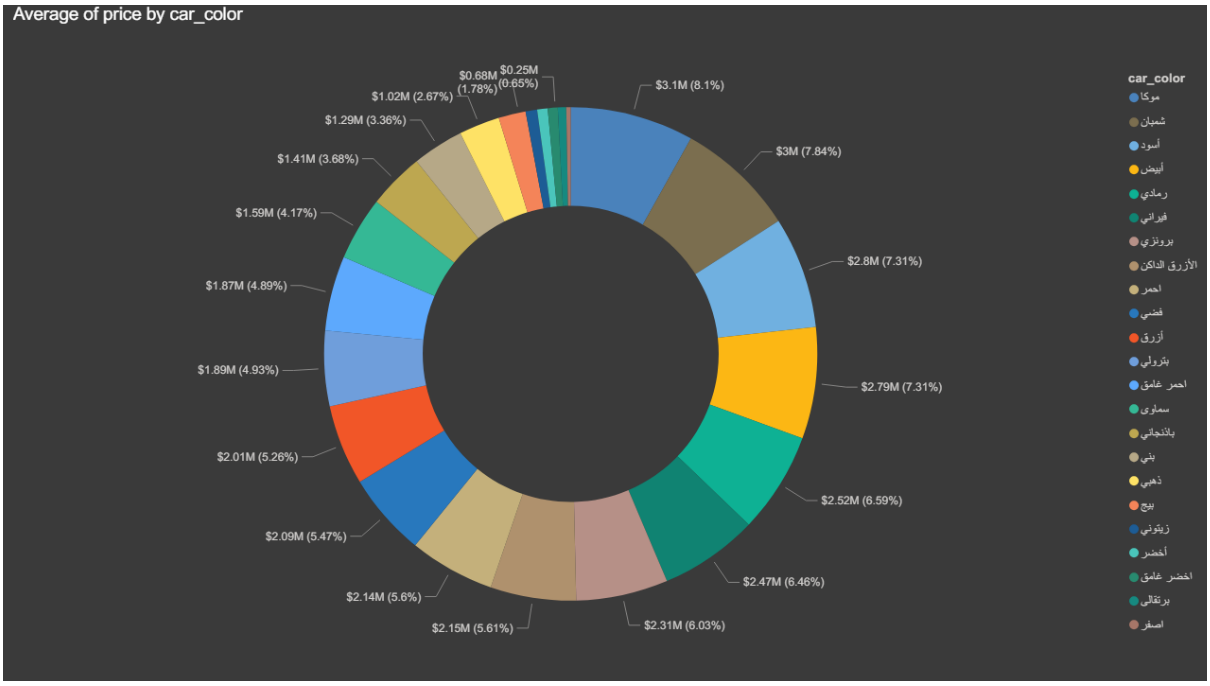

Figure 3.4: Relationship between Car Color and Price

Chapter 4

Linear Regression Model

4.1 Introduction

To predict car prices based on various features, we implemented a linear regression model. This model helps understand the relationship between car prices and attributes such as mileage, model, color, and location.

4.2 Data Preprocessing

The data preprocessing steps included:

- Converting categorical variables to numerical values using one-hot encoding.
- Splitting the data into training and testing sets.

4.3 Model Implementation

The following Python code was used to implement the linear regression model:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the dataset
file_path = ’car_data.csv’
car_data = pd.read_csv(file_path)

# Function to convert mileage and price to numerical values
def convert_to_numeric(value):
    if isinstance(value, str):
        return int(value.replace(’,’, ’’).replace(’ ’, ’’).replace(’ ’, ’’).replace(’
    return value

# Apply the conversion function to car_mileage and price columns
```

```
car_data[’car_mileage’] = car_data[’car_mileage’].apply(convert_to_numeric)
car_data[’price’] = car_data[’price’].apply(convert_to_numeric)

# Encode categorical variables using one-hot encoding
car_data_encoded = pd.get_dummies(car_data, columns=[’car_model_name’, ’car_color’, ’

# Drop unnecessary columns
car_data_encoded = car_data_encoded.drop(columns=[’title’, ’date’, ’image’])

# Split the data into features and target variable
X = car_data_encoded.drop(columns=[’price’])
y = car_data_encoded[’price’]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state

# Initialize and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Calculate the performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("R-squared (R²) Score:", r2)

# Test the model with a specific input
sample_input = {
    ’car_mileage’: 50000,  # example mileage
    ’car_model_name_230’: 1,  # example model
    ’car_color_’: 1,  # example color
    ’location_’: 1  # example location
}

# Convert sample input to DataFrame
sample_df = pd.DataFrame([sample_input])

# Predict the price for the sample input
predicted_price = model.predict(sample_df)
print("Predicted Price:", predicted_price[0])
```

4.4 Results

The performance for the linear regression model are as follows in this example:

- Predicted Price of this input: car mileage = 50,000 - model year = 2020 - car model name 230 - car color = black - location = Cairo ,was Predicted Price: 2833527.074283967

Chapter 5

Conclusion

5.1 Summary

The project successfully achieved the objectives of extracting and analyzing car data from the Hatla2ee website. The use of web scraping techniques allowed us to gather valuable information, which was then analyzed and visualized using Power BI. The linear regression model provided insights into the relationship between car prices and various attributes, although there is room for improvement in the model's accuracy.

5.2 Future Work

Future work could include:

- Expanding the scraping to include other car brands and models.
- Implementing more advanced machine learning models to improve price predictions.
- Enhancing the data visualization with interactive dashboards.