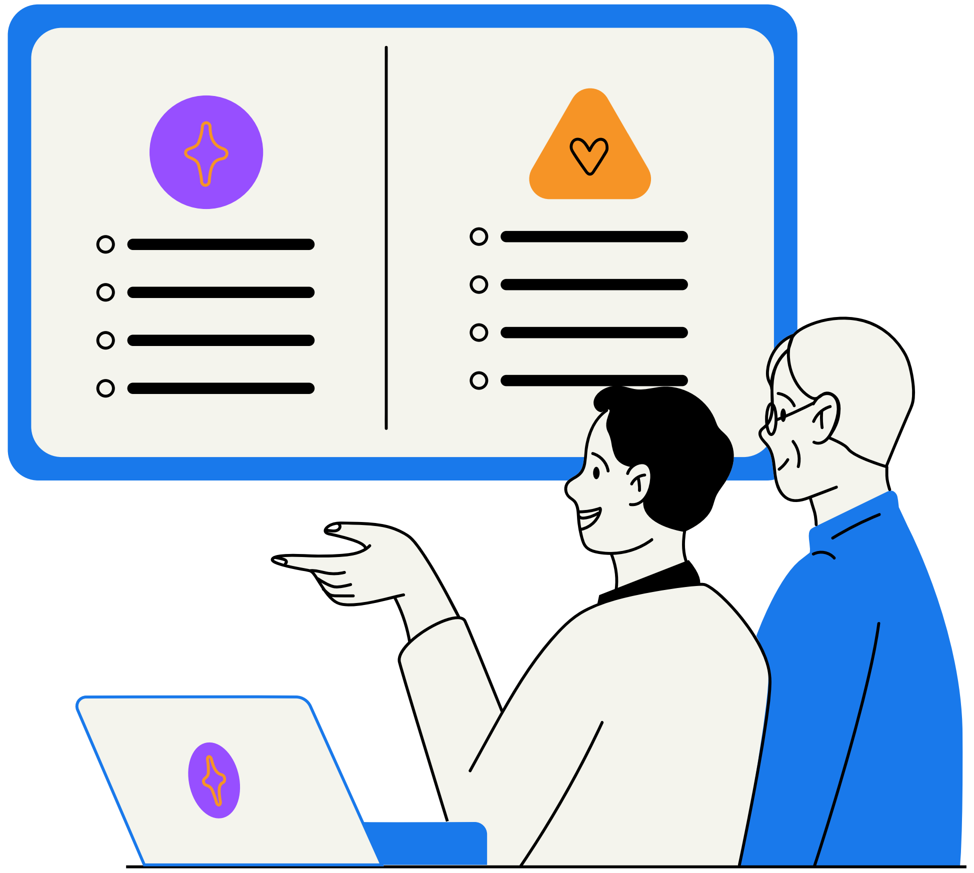
DATA COLLECTION & ANALYSIS

Ramesh

PROBLEM STATEMENT:



Steps to Follow:

1) Research and Planning:

- Understand the structure of Cars24.com.
- Identify the HTML elements containing the required information.

2) Data Extraction:

- Write a script to scrape the necessary details (kilometers driven, year of manufacture, fuel type, transmission, and price) for the assigned brand.

3) Data Cleaning:

- Ensure the scraped data is clean and organized for analysis.

4) Data Presentation:

- Present the collected data in a structured format in a CSV.

DATA COLLECTION:

1) Scraping Target: Cars listed on Cars24.com for the Mumbai location.

2) Data to Collect:

- Kilometers Driven
- Year of Manufacture
- Fuel Type
- Transmission
- Price

WEB SCRAPING WITH BEAUTIFULSOUP DEFINITION:

Web scraping is the process of extracting data from websites using automated scripts.

Tools:

- **BeautifulSoup:** A Python library for parsing HTML and XML documents.
- **Requests:** A Python library for making HTTP requests.

Steps:

Store the website into a variable:

```
In [2]: url = "https://www.cars24.com/buy-used-car?f=make%3A%3D%3Atoyota&sort=bestmatch&serveWarrantyCount=true&listingSou
```

Request the URL:

```
In [3]: response = requests.get(url)
```

To check we use Status Code:

```
In [4]: response.status_code

Out[4]: 200
```

We need to create a Soup object:

```
In [5]: soup = BeautifulSoup(response.text, "lxml")

In [6]: print(soup)
```

```
<!DOCTYPE html>
<html lang="en-IN">
<head>
<link href="https://assets.cars24.com" rel="preconnect"/>
<link href="https://fastly-production.24c.in" rel="preconnect"/>
<link href="https://connect.facebook.net" rel="preconnect"/>
<link href="https://www.googletagmanager.com" rel="preconnect"/>
<link href="https://www.google-analytics.com" rel="preconnect"/>
<link href="https://analytics.twitter.com" rel="preconnect"/>
```

DATA EXTRACT:

CREATE DATAFRAME:

car_name:

```
names = soup.find_all("h3", class_ = "_11dVb")
```

```
car_names = []
for i in names:
    name = i.text
    categorical_name = name[5:] if name[:4].isdigit() and name[4] == ' ' else name
    car_names.append(categorical_name)

print(car_names)
```

```
['Toyota Etios Liva G', 'Toyota YARIS VX MT', 'Toyota URBAN CRUISER PREMIUM GRADE MT', 'Toyota Etios Liva G', 'Toyota URBAN CRU
ISER PREMIUM GRADE AT', 'Toyota Etios Liva G', 'Toyota Corolla Altis VL CVT PETROL', 'Toyota Corolla Altis VL CVT PETROL', 'Toy
ota Glanza G CVT', 'Toyota Corolla Altis VL CVT PETROL', 'Toyota Glanza V CVT', 'Toyota Corolla Altis VL CVT PETROL', 'Toyota G
lanza V CVT', 'Toyota Glanza G CVT', 'Toyota Etios Liva G', 'Toyota URBAN CRUISER HIGH GRADE AT', 'Toyota Innova 2.5 GX 8 STR',
'Toyota Etios Liva G', 'Toyota Etios Liva G']
```

Kilometers_Driven:

```
km_elements = soup.find_all("ul", class_ = "_3J2G-")
```

```
Kilometers_driven = []

for i in km_elements:
    full_text = i.text
    km = full_text.split(' km')[0]  if ' km' in full_text else None
    if km:
        Kilometers_driven.append(km)

print(Kilometers_driven)
```

```
['52,656', '30,509', '18,001', '79,643', '33,986', '77,595', '74,221', '56,595', '16,870', '77,581', '56,916', '61,520', '15,81
```

Create a Data Frame

```
data = {
    'Car_Name': car_names,
    'Kilometers_Driven': Kilometers_driven,
    'Fuel_Type': fuel_types,
    'Price': prices,
    'Year': years,
    'Transmission': transmissions
}

df = pd.DataFrame(data)

print(df)
```

DATA TABLES

During an experiment, data is typically collected and organized using data tables with the independent variable on the left side and the dependent variable on the right side -- with units included!

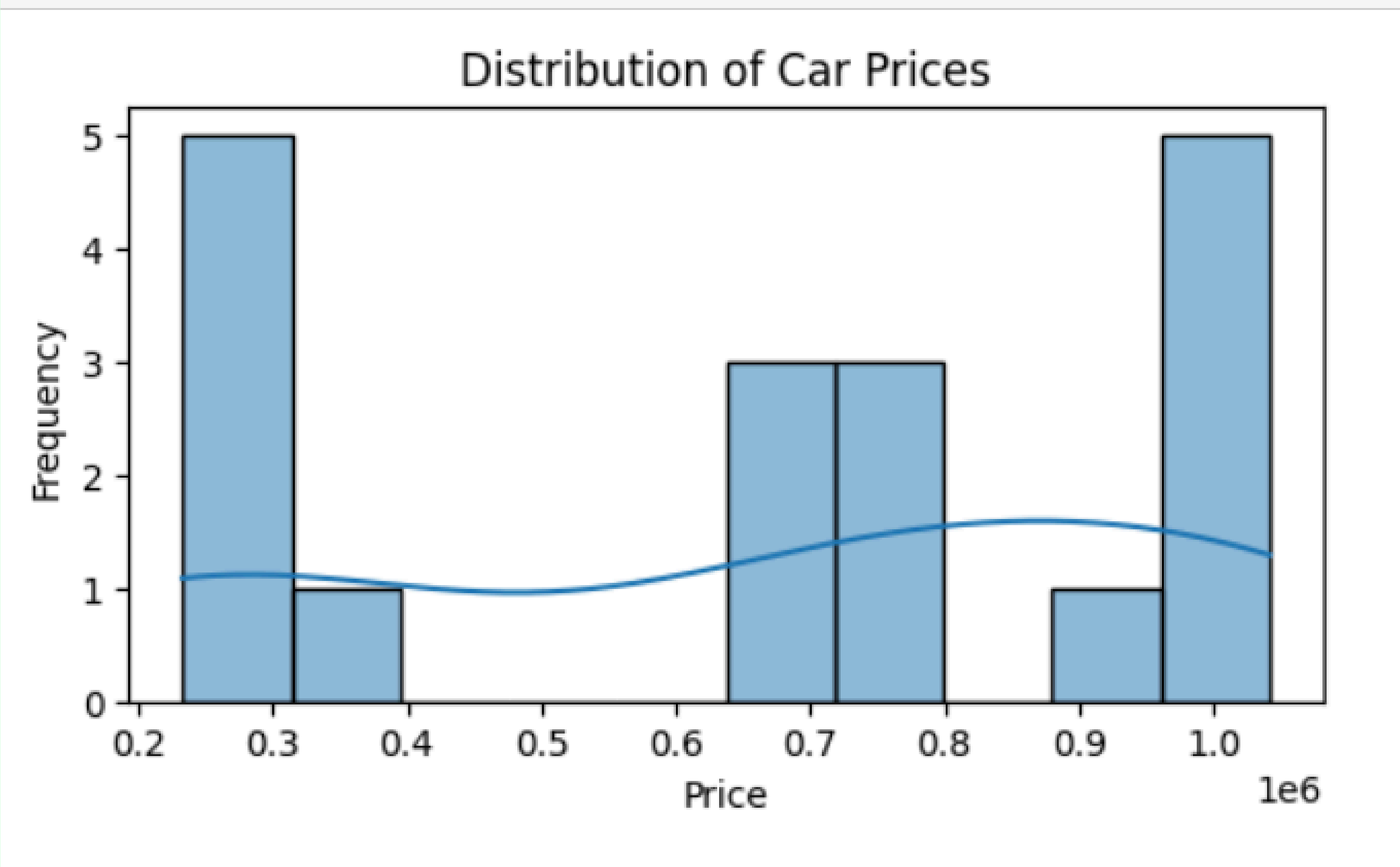
	Car_Name	Year	Kilometers_Driven	Fuel_Type	Transmission	Price
0	Toyota Etios Liva G	2012	52656	Petrol	Manual	237000
1	Toyota YARIS VX MT	2018	30509	Petrol	Manual	783000
2	Toyota URBAN CRUISER PREMIUM GRADE MT	2021	18001	Petrol	Manual	957000
3	Toyota Etios Liva G	2011	79643	Petrol	Manual	233000
4	Toyota URBAN CRUISER PREMIUM GRADE AT	2022	33986	Petrol	Automatic	1011000
5	Toyota Etios Liva G	2011	77595	Petrol	Manual	247000
6	Toyota Corolla Altis VL CVT PETROL	2018	74221	Petrol	Automatic	978000
7	Toyota Corolla Altis VL CVT PETROL	2017	56595	Petrol	Automatic	1033000
8	Toyota Glanza G CVT	2019	16870	Petrol	Automatic	679000
9	Toyota Corolla Altis VL CVT PETROL	2018	77581	Petrol	Automatic	1000000
10	Toyota Glanza V CVT	2020	56916	Petrol	Automatic	713000
11	Toyota Corolla Altis VL CVT PETROL	2017	61520	Petrol	Automatic	1042000
12	Toyota Glanza V CVT	2019	15816	Petrol	Automatic	793000
13	Toyota Glanza G CVT	2019	21695	Petrol	Automatic	728000
14	Toyota Etios Liva G	2013	30154	Petrol	Manual	251000
15	Toyota Etios Liva G	2011	75420	Petrol	Manual	258000
16	Toyota Innova 2.5 GX 8 STR	2012	89683	Diesel	Manual	660000
17	Toyota Etios Liva G	2014	23685	Petrol	Manual	333000

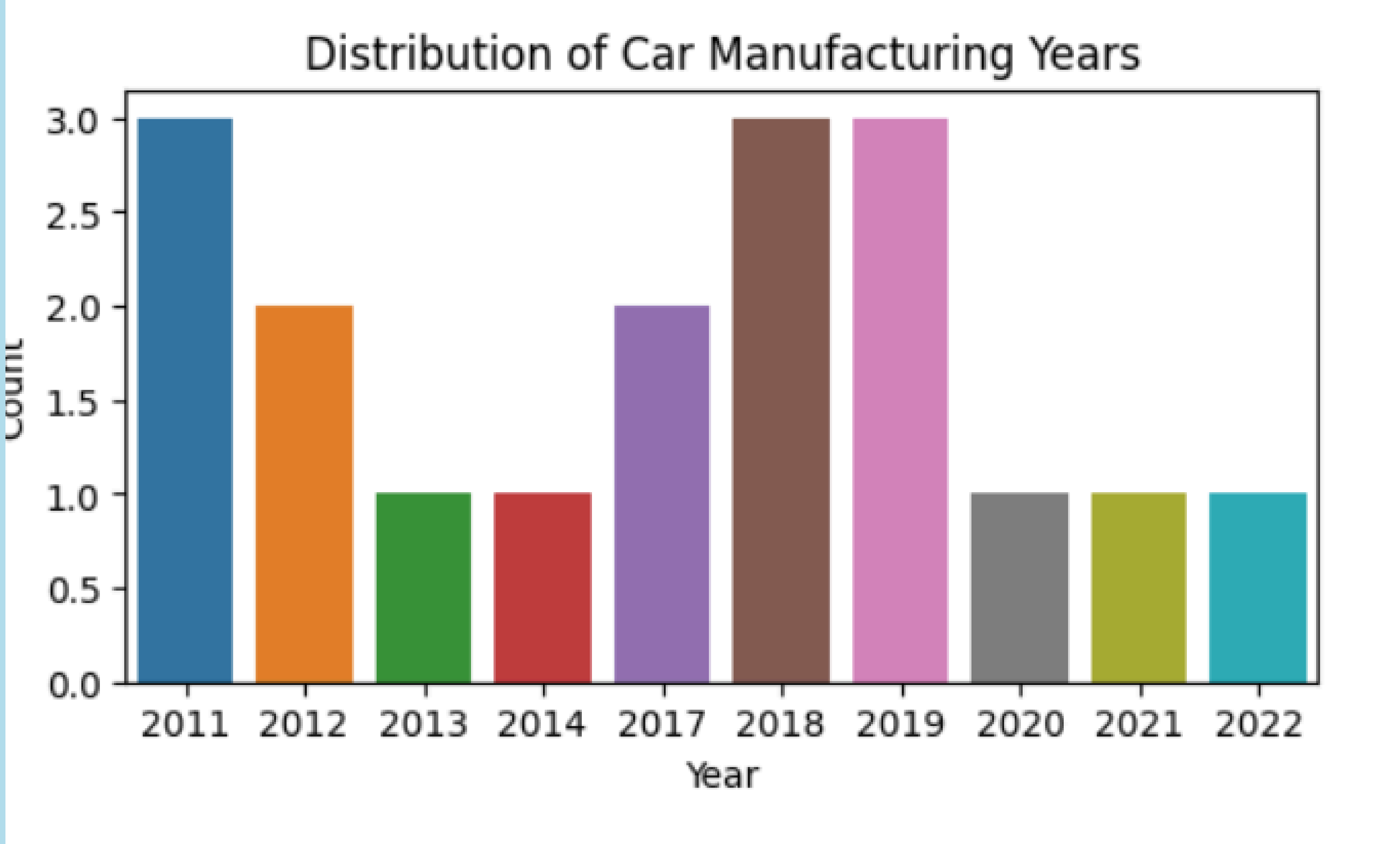
TYPES OF DATA

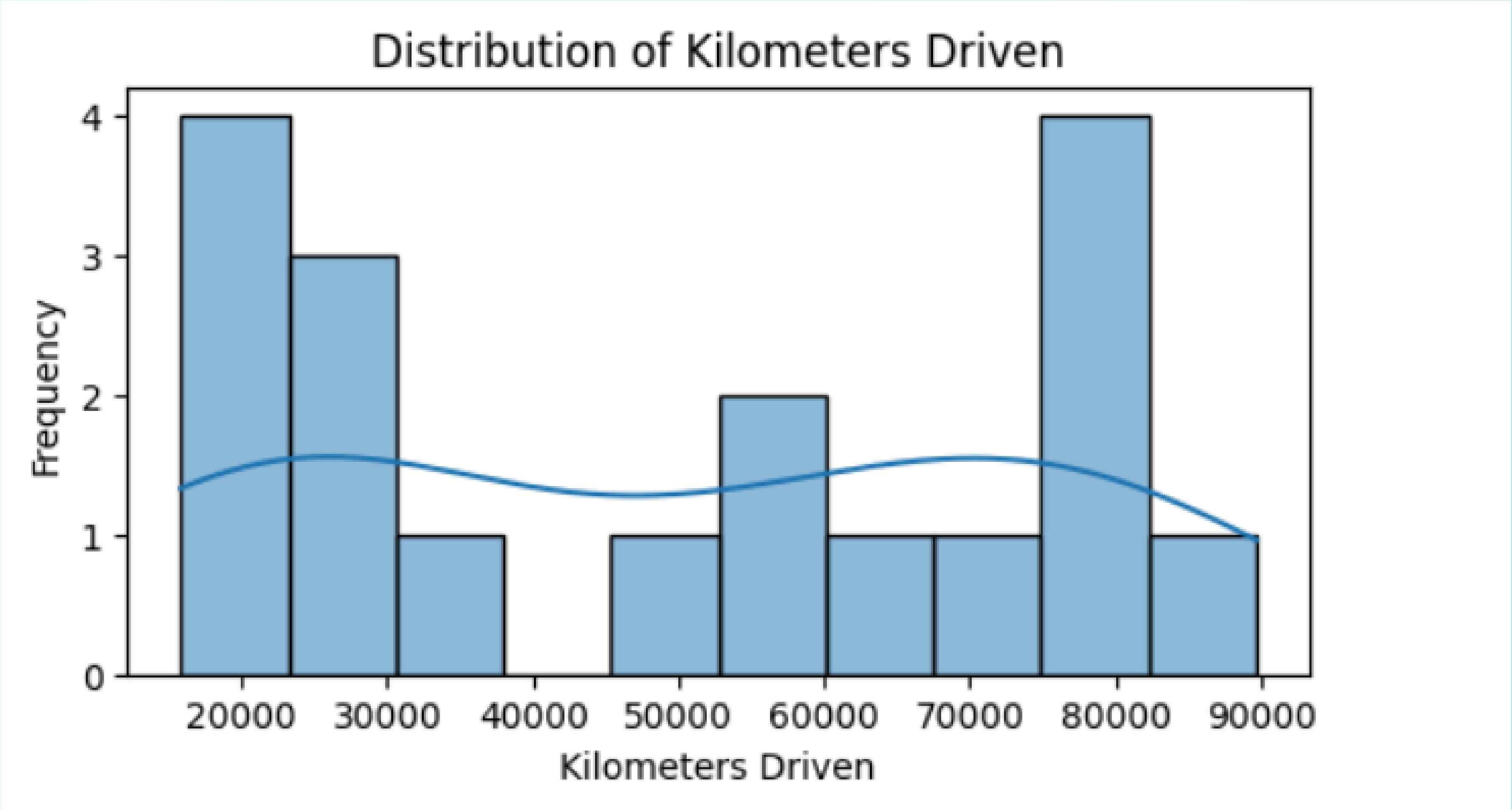
There are two main types of data: qualitative data and quantitative data.

Qualitative data is descriptive. It includes things like color, texture, and taste.
Year, Kilometers_Driven, Price

Quantitative data is numerical. It includes things like height, rate, and speed.
Car_Name, Fuel_Type, Transmission

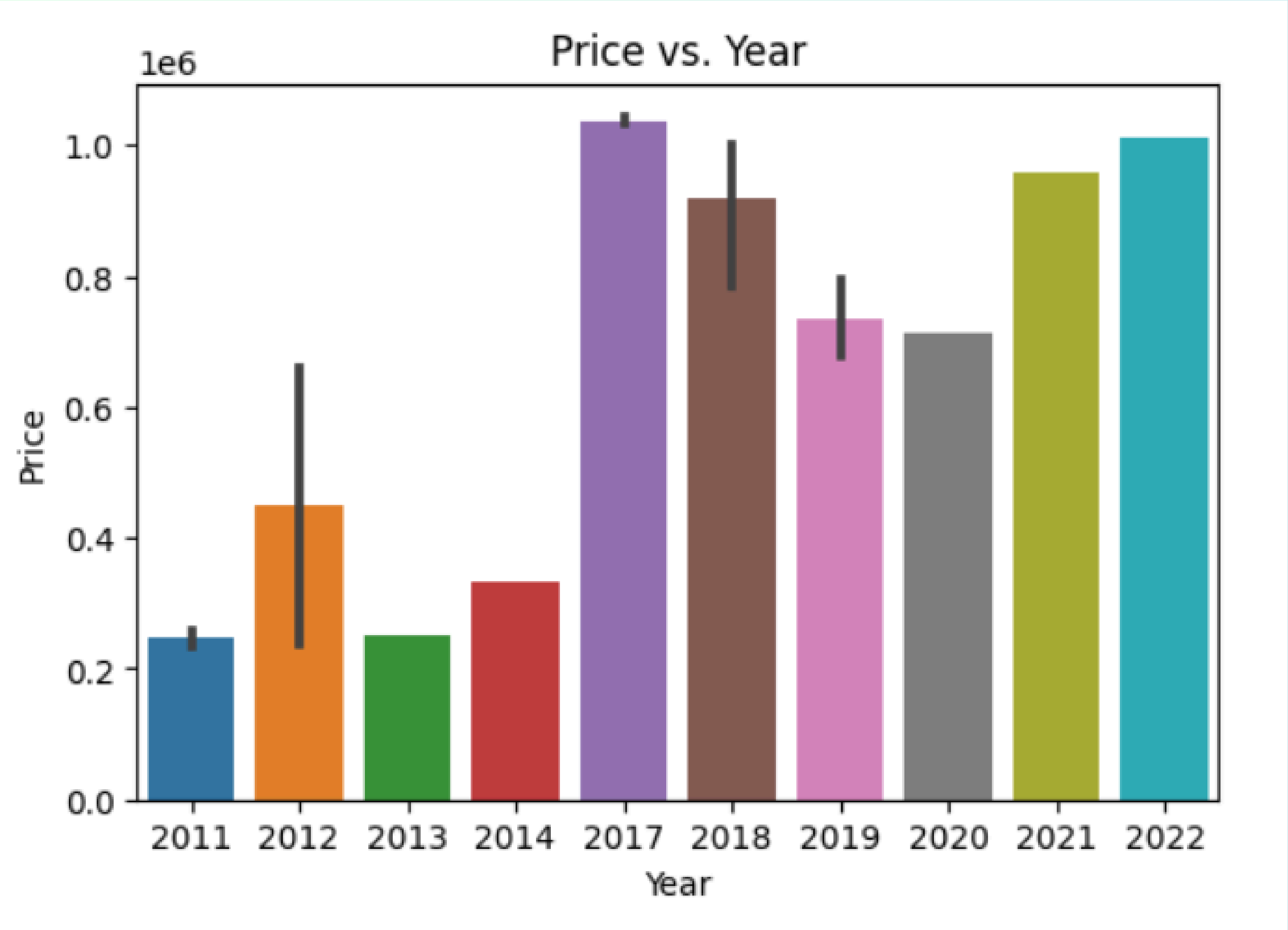






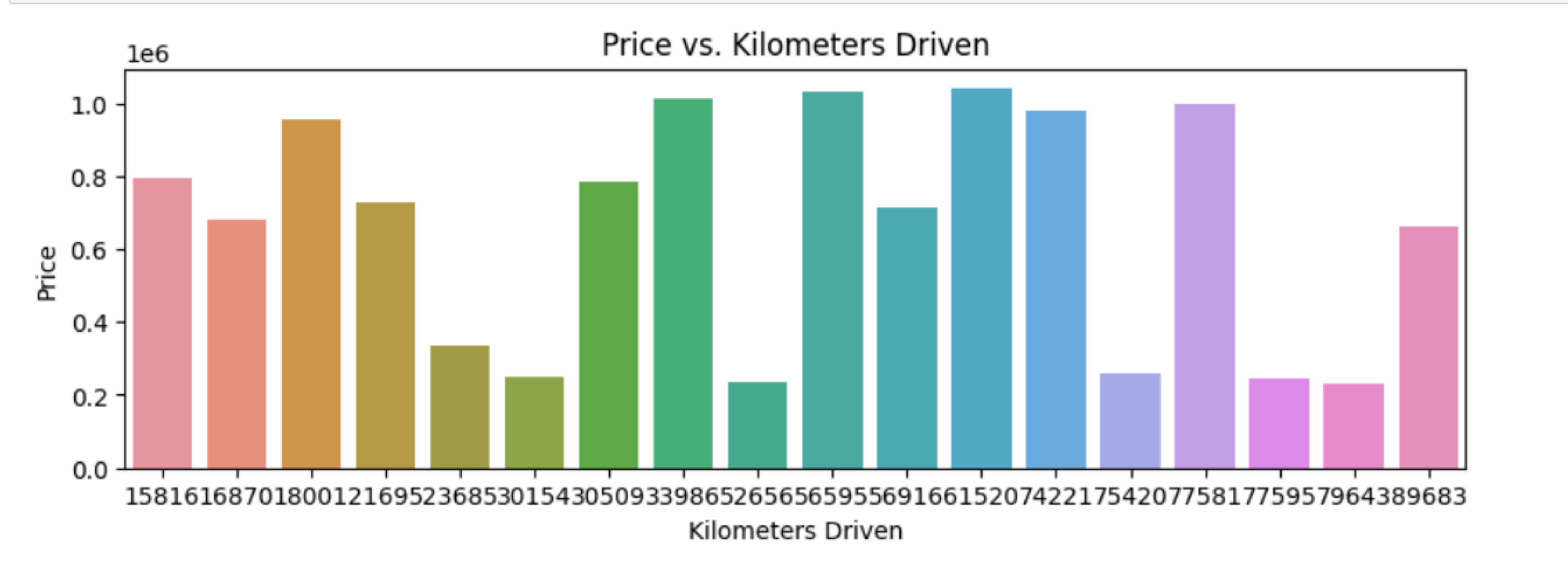
Conclusion:

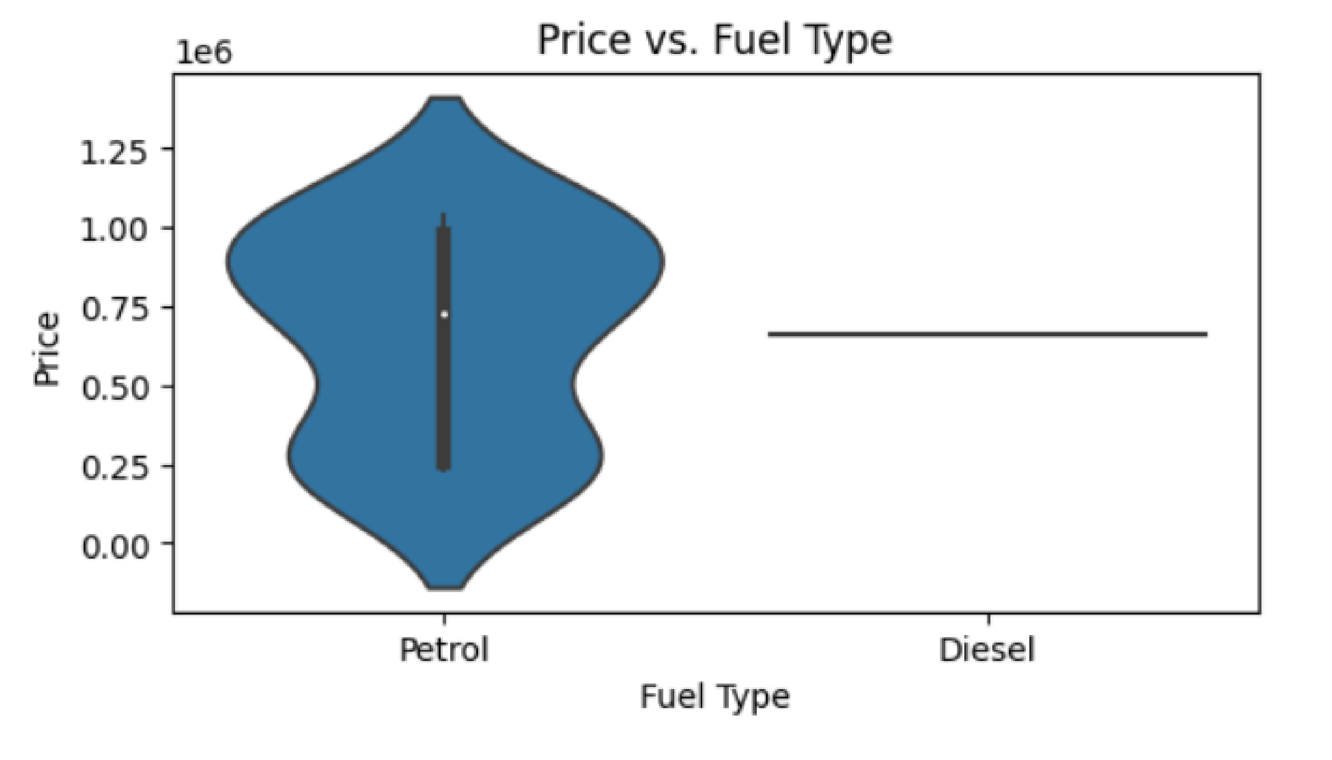
- The price distribution is right-skewed, with most cars priced in the lower range and a few high-priced outliers.
- The dataset includes cars from a wide range of years, with a higher concentration of newer models.
- The distribution of kilometers driven is right-skewed, indicating that most cars in the dataset have lower mileage, with some high-mileage outliers.

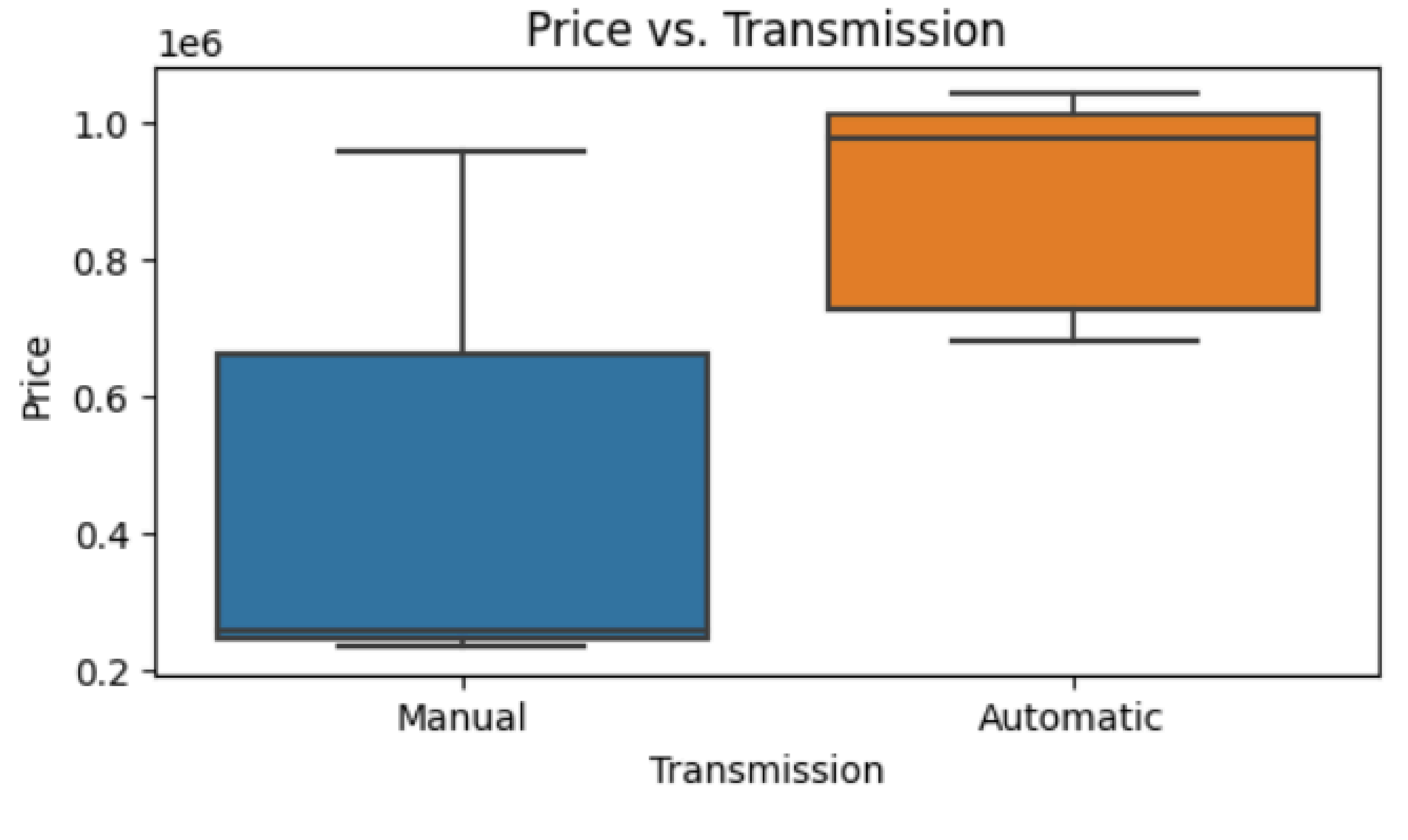


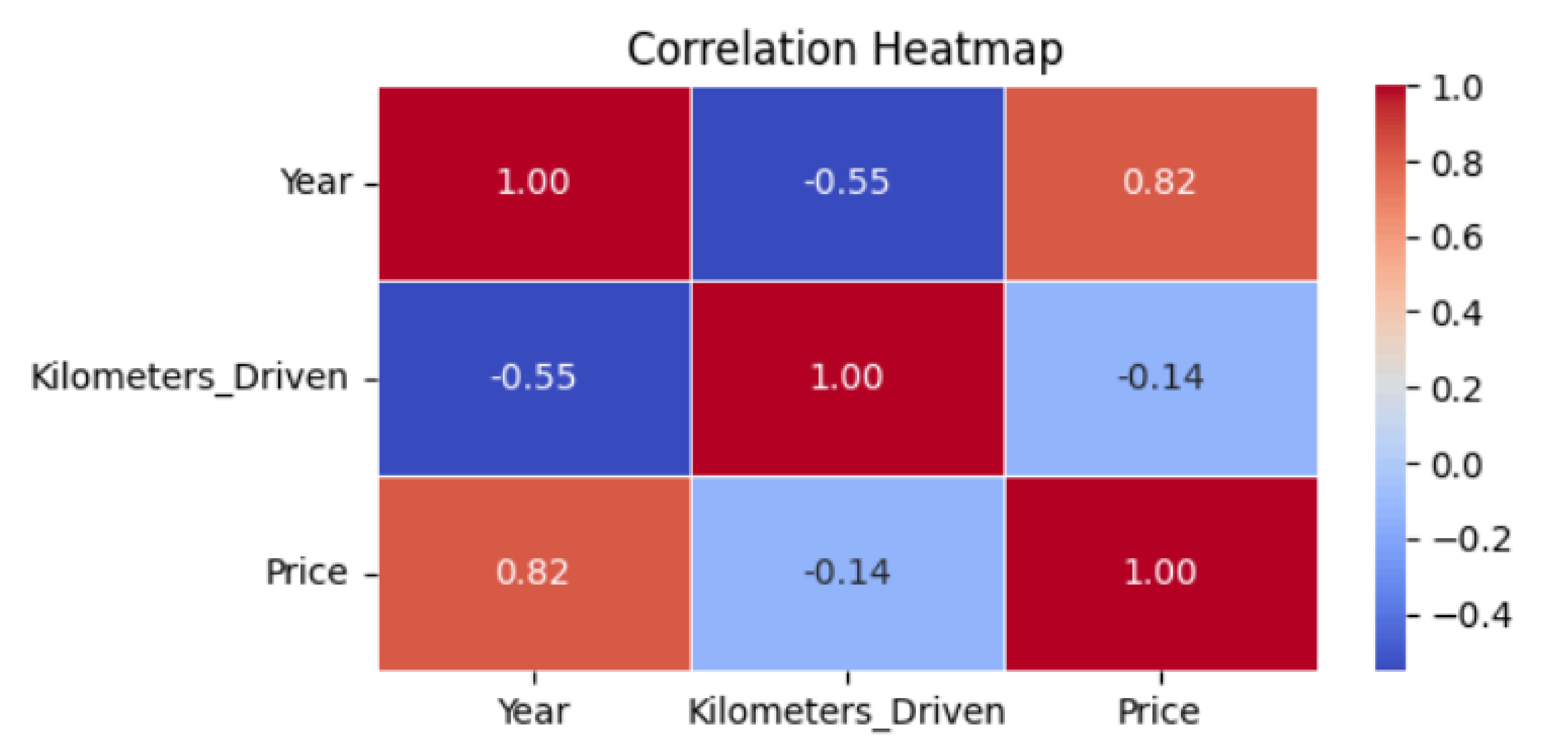
Conclusion:

- There is a positive correlation between the car's manufacturing year and its price, suggesting newer cars tend to be more expensive.
- A negative correlation is visible between kilometers driven and price, implying that cars with higher mileage generally have lower prices.
- This visualization shows how prices vary across different fuel types. Further analysis might be needed to draw specific conclusions.









Conclusion:

- This plot illustrates the price differences between transmission types. Further analysis might be needed to determine if there's a significant price difference between manual and automatic transmissions.
- A strong positive correlation of 82 between year and price indicates that newer cars tend to have higher prices

Final Conclusion:

The analysis reveals that newer cars tend to have higher prices, indicated by a strong positive correlation of 82 between year and price. There is also a negative correlation between kilometers driven and price, implying higher mileage cars generally have lower prices. The price distribution is right-skewed with most cars in the lower price range and a few high-priced outliers. Further analysis is needed to determine the impact of transmission types and fuel types on prices.

LABEL ENCODING:

The categorical data is convert into numerical data. Identify the columns with categorical data that need to be converted to numerical data. In this case, the columns are **Fuel_Type and Transmission**.

Data Cleaning:

If there are any unnecessary columns in the dataset that are not required for analysis or modeling, remove them.
like remove **Car_Name** columns

EXPERIMENTAL DATA

When designing an experiment, you must decide what type of data you will collect. This is related to your **dependent variable**. It should be the goal of all researchers to report data that is both **accurate** (matching known results) and **reliable** (matching other experimental results), no matter what type of data it is.

Independent variable is X:

X = Car_Name, Year, Kilometers_Driven, Fuel_Type, Transmission

Dependent Variable is y:

y = Price

MODEL BUILDING:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

from sklearn.linear_model import LinearRegression

# Initialize the model
lr_model = LinearRegression()

# Train the model on the training data
lr_model.fit(X_train_scaled, y_train)
# Predict the target variable for the testing data
y_pred_lr = lr_model.predict(X_test_scaled)

# Calculate evaluation metrics
mae = mean_absolute_error(y_test, y_pred_lr)
mse = mean_squared_error(y_test, y_pred_lr)
r2 = r2_score(y_test, y_pred_lr)

print("Mean Absolute Error (MAE):", mae)
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)
```

```
Mean Absolute Error (MAE): 98338.16986622484
Mean Squared Error (MSE): 11321816922.884563
R-squared (R2): 0.814835829357638
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Initialize the model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model on the training data
rf_model.fit(X_train_scaled, y_train)

# Predict the target variable for the testing data
y_pred_rf = rf_model.predict(X_test_scaled)

# Calculate evaluation metrics
mae_rf = mean_absolute_error(y_test, y_pred_rf)
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

print("Random Forest Regressor:")
print("Mean Absolute Error (MAE):", mae_rf)
print("Mean Squared Error (MSE):", mse_rf)
print("R-squared (R2):", r2_rf)
```

```
Random Forest Regressor:
Mean Absolute Error (MAE): 70435.0
Mean Squared Error (MSE): 6199641550.0
R-squared (R2): 0.8986071322558356
```

Conclusion:

Based on the analysis, the Random Forest Regressor outperforms Linear Regression in predicting car prices, showing lower Mean Absolute Error (MAE) and Mean Squared Error (MSE), and a higher R-squared (R^2) value. This indicates its superior ability to capture and explain the variance in pricing data, making it the preferred model for accurate and reliable predictions in the automotive market. Implementing this model can significantly enhance pricing strategies and operational efficiencies, driving competitive advantage and informed decision-making in the industry.

LEARNING OUTCOMES FROM THE WEB SCRAPING PROJECT:

1. Research and Planning:
    - Understanding Website Structure: Learn how to inspect and understand the structure of a website, including the HTML elements and their attributes.
    - Identifying Relevant Data: Develop the skill to identify and target specific HTML elements that contain the necessary information for your project.
2. Data Extraction:
    - Writing Scraping Scripts: Gain experience in writing Python scripts using libraries like BeautifulSoup and Requests to scrape data from websites.
    - Handling Different Data Points: Learn to extract various details such as kilometers driven, year of manufacture, fuel type, transmission, and price for different car models.
3. Data Cleaning:
    - Ensuring Data Quality: Understand the importance of data cleaning and learn techniques to clean and organize the scraped data for accurate analysis.
    - Handling Inconsistencies: Learn to deal with inconsistent data formats and missing values, ensuring the dataset is reliable and usable.
4. Data Presentation:
    - Structuring Data: Develop the ability to structure and present the collected data in a clear and organized format, such as CSV.
    - Preparation for Analysis: Learn how to prepare data for further analysis, making it easier to derive insights and make data-driven decisions.

THANK YOU