Indian Institute of Information Technology Surat

Lab Report on
Machine Learning (CS 601) Practical

Submitted by

[RAHUL KUMAR SINGH] (UI21CS44)

Course Faculty

Dr. Pradeep Kumar Roy

Dr. Rajesh K. Ahir

Department of Computer Science and Engineering

Indian Institute of Information Technology Surat

Gujarat-394190, India

Jan-2024

Lab No: 2

Aim:

To perform exploratory data analysis on the attached dataset

Description:

Perform the Exploratory Data Analysis (EDA) by considering the following tasks. Use the attached dataset for the same.

1. Check for Duplication
2. Missing Values Calculation
3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)
4. Feature Engineering
5. Creating Features
6. Data Cleaning/Wrangling
7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)
8. Analyzing/visualizing the dataset by taking one variable at a time
9. Data Transformation

Source Code:

Import Libraries and Read Dataset

```
[ ]: import pandas as pd
     from sklearn.decomposition import PCA
     from sklearn.preprocessing import StandardScaler, LabelEncoder
     import matplotlib.pyplot as plt
     import seaborn as sns
     from datetime import datetime

     df = pd.read_csv('cars_data.csv')
     print(df.head())
```

1. Check for Duplication

```
[ ]: duplicates = df.duplicated()
     print(df[duplicates])
```

```
[ ]: num_duplicates = df.duplicated().sum()
     percentage_duplicates = (num_duplicates / len(df)) * 100

     print(f"Number of duplicate rows: {num_duplicates}")
     print(f"Percentage of duplicate rows: {percentage_duplicates:.2f}%")
```

```
[ ]: df = df.drop_duplicates()
     print(df.head())
```

2. Missing Values Calculation

```
[ ]: total_missing = df.isnull().sum().sum()
     print(total_missing)
```

```
[ ]: missing_by_column = df.isnull().sum()
     print(missing_by_column)
```

```
[ ]: percentage_missing = (df.isnull().sum() / len(df)) * 100
     print(percentage_missing)
```

3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)

```
# Replace missing values
df['Price'].fillna(0, inplace=True)
df['New_Price'].fillna(0, inplace=True)
df.dropna(inplace=True) # Dropping few inconsequntial records
```

```
# Drop irrelevant columns for analysis
cols_to_drop = ['Name','Location','Fuel_Type','Transmission','Owner_Type','Mileage','Engine','Power','New_Price']
dropdf = df.drop(columns=cols_to_drop)

scaler = StandardScaler()
cars_data_scaled = scaler.fit_transform(dropdf)

# Apply Principal Component Analysis (PCA) for dimensionality reduction
pca = PCA(n_components=2)
cars_pca = pca.fit_transform(cars_data_scaled)

plt.figure(figsize=(10, 6))
plt.scatter(cars_pca[:, 0], cars_pca[:, 1])
plt.title('PCA: First Two Principal Components')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

4. Feature Engineering

```
selected_features = df[['S.No.', 'Kilometers_Driven', 'Seats', 'Price']]

scaler = StandardScaler()
scaled_features = scaler.fit_transform(selected_features)

num_components = 2
pca = PCA(n_components=num_components)
reduced_features = pca.fit_transform(scaled_features)
reduced_features_df = pd.DataFrame(data=reduced_features, columns=['PC1', 'PC2'])
final_data = pd.concat([df, reduced_features_df], axis=1)
print(final_data.head())
```

5. Creating Features

```
cars_df = df.copy()
cars_df['Brand'] = cars_df['Name'].str.split().str[0]
cars_df['Mileage'] = cars_df['Mileage'].str.split().str[0]
cars_df['Mileage'] = pd.to_numeric(df['Mileage'], errors='coerce')
current_year = datetime.now().year
cars_df['Age'] = current_year - cars_df['Year']
cars_df['Price_per_Mile'] = cars_df['Price'] / cars_df['Mileage']

print("\nCars Dataset with New Features:")
print(cars_df)
```

```
# Visualization
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.scatterplot(x='Age', y='Price', data=cars_df, hue='Brand', palette='Set1')
plt.title('Age vs Price')

plt.subplot(1, 2, 2)
sns.scatterplot(x='Mileage', y='Price_per_Mile', data=cars_df, hue='Brand', palette='Set2')
plt.title('Mileage vs Price_per_Mile')

# plt.tight_layout()
plt.show()
```

6. Data Cleaning/Wrangling

```
clean_df = df.copy()

clean_df['Brand'] = clean_df['Name'].str.split().str[0]
clean_df['Engine'] = clean_df['Engine'].str.extract('(\d+)').astype(float)
clean_df['Mileage'] = clean_df['Mileage'].str.extract('(\d+)').astype(float)
clean_df['Power'] = clean_df['Power'].str.extract('(\d+)').astype(float)
clean_df['New_Price'] = clean_df['New_Price'].str.extract('(\d+)').astype(float)
clean_df['New_Price'].fillna(0, inplace=True)
current_year = datetime.now().year
clean_df['Mileage'][clean_df['Mileage']==0] = 1
clean_df['Age'] = current_year - clean_df['Year']
clean_df['Price_per_Mile'] = clean_df['Price'] / clean_df['Mileage']
clean_df = clean_df.drop(['Name', 'Year'], axis=1)

print("\nCleaned and Wrangled Dataset:")
print(clean_df)
```

7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)

```
print("Dataset Information:")
print(df.info())

print("\nSummary Statistics:")
print(df.describe())
```

```
cars_data = dropdf.copy()
summary_stats = {
    'Count': cars_data.shape[0],
    'Mean': cars_data.mean(),
    'Standard Deviation': cars_data.std(),
    'Median': cars_data.median(),
    'Mode': cars_data.mode().iloc[0],
    'Minimum Value': cars_data.min(),
    'Maximum Value': cars_data.max(),
    'Range': cars_data.max() - cars_data.min(),
}
summary_df = pd.DataFrame(summary_stats)

print("\nStatistics Summary:")
print(summary_df)
```

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.show()
```

8. Analyzing/visualizing the dataset by taking one variable at a time

```
[ ]: cars_data = clean_df.copy()
def visualize_variable(variable_name):
    plt.figure(figsize=(8, 6))
    plt.hist(cars_data[variable_name], bins=20, color='skyblue', edgecolor='black')
    plt.title(f'Distribution of {variable_name}')
    plt.xlabel(variable_name)
    plt.ylabel('Frequency')
    plt.show()

numerical_variables = cars_data.select_dtypes(include='number').columns
for variable in numerical_variables:
    visualize_variable(variable)
```

9. Data Transformation

```
[ ]: cars_data = clean_df.copy()
# Encode Categorical Variables
label_encoder = LabelEncoder()
cars_data['Brand'] = label_encoder.fit_transform(cars_data['Brand'])
cars_data['Fuel_Type'] = label_encoder.fit_transform(cars_data['Fuel_Type'])
cars_data['Transmission'] = label_encoder.fit_transform(cars_data['Transmission'])

# Scale Numerical Features
numerical_features = ['Price', 'Mileage', 'Engine']
scaler = StandardScaler()
cars_data[numerical_features] = scaler.fit_transform(cars_data[numerical_features])

print("\nTransformed Dataset:")
print(cars_data.head())
```

Output:

1. Check for Duplication

1. Check for Duplication

```
[2]: duplicates = df.duplicated()
print(df[duplicates])
```

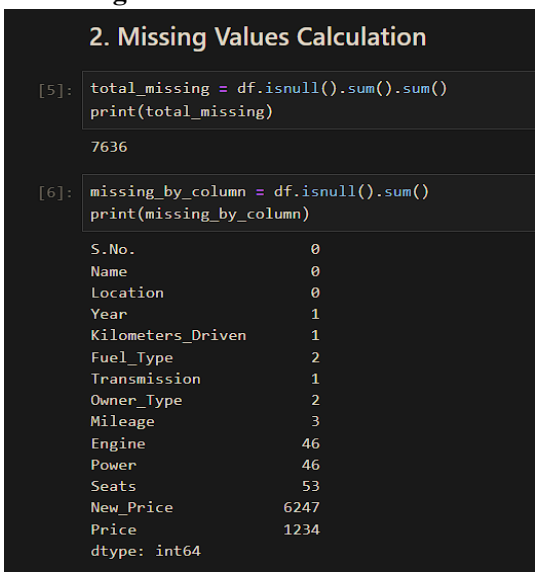
```
Empty DataFrame
Columns: [S.No., Name, Location, Year, Kilometers_Driven, Fuel_Type, Transmission, Owner_Type, Mileage, Engine, Power, Seats, New_Price, Price]
Index: []
```

```
[3]: num_duplicates = df.duplicated().sum()
percentage_duplicates = (num_duplicates / len(df)) * 100

print(f"Number of duplicate rows: {num_duplicates}")
print(f"Percentage of duplicate rows: {percentage_duplicates:.2f}%")
```

```
Number of duplicate rows: 0
Percentage of duplicate rows: 0.00%
```

Figure 2.1 Output for task 1

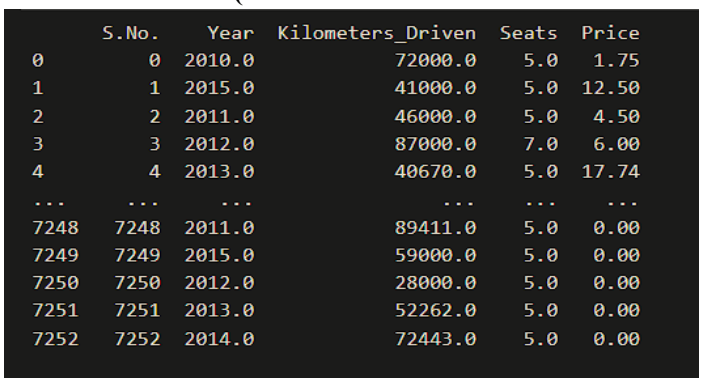
2. Missing Values Calculation



```
2. Missing Values Calculation

[5]: total_missing = df.isnull().sum().sum()
     print(total_missing)

     7636

[6]: missing_by_column = df.isnull().sum()
     print(missing_by_column)

     S.No.                  0
     Name                   0
     Location               0
     Year                   1
     Kilometers_Driven      1
     Fuel_Type              2
     Transmission           1
     Owner_Type             2
     Mileage                3
     Engine                46
     Power                 46
     Seats                 53
     New_Price           6247
     Price               1234
     dtype: int64
```

Figure 2.2 Output for task 2

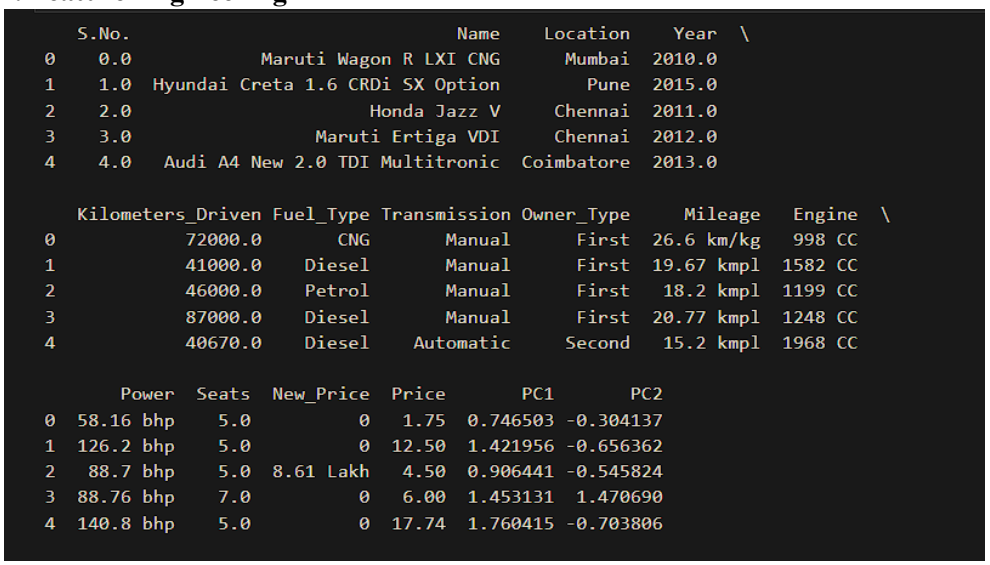
3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)



	S.No.	Year	Kilometers_Driven	Seats	Price
0	0	2010.0	72000.0	5.0	1.75
1	1	2015.0	41000.0	5.0	12.50
2	2	2011.0	46000.0	5.0	4.50
3	3	2012.0	87000.0	7.0	6.00
4	4	2013.0	40670.0	5.0	17.74
...	...	...	...	...	...
7248	7248	2011.0	89411.0	5.0	0.00
7249	7249	2015.0	59000.0	5.0	0.00
7250	7250	2012.0	28000.0	5.0	0.00
7251	7251	2013.0	52262.0	5.0	0.00
7252	7252	2014.0	72443.0	5.0	0.00

Figure 2.3 Output for task 3

4. Feature Engineering



	S.No.	Name	Location	Year	\
0	0.0	Maruti Wagon R LXI CNG	Mumbai	2010.0	
1	1.0	Hyundai Creta 1.6 CRDi SX Option	Pune	2015.0	
2	2.0	Honda Jazz V	Chennai	2011.0	
3	3.0	Maruti Ertiga VDI	Chennai	2012.0	
4	4.0	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013.0	

	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	\
0	72000.0	CNG	Manual	First	26.6 km/kg	998 CC	
1	41000.0	Diesel	Manual	First	19.67 kmpl	1582 CC	
2	46000.0	Petrol	Manual	First	18.2 kmpl	1199 CC	
3	87000.0	Diesel	Manual	First	20.77 kmpl	1248 CC	
4	40670.0	Diesel	Automatic	Second	15.2 kmpl	1968 CC	

	Power	Seats	New_Price	Price	PC1	PC2
0	58.16 bhp	5.0	0	1.75	0.746503	-0.304137
1	126.2 bhp	5.0	0	12.50	1.421956	-0.656362
2	88.7 bhp	5.0	8.61 Lakh	4.50	0.906441	-0.545824
3	88.76 bhp	7.0	0	6.00	1.453131	1.470690
4	140.8 bhp	5.0	0	17.74	1.760415	-0.703806

Figure 2.4 Output for task 4

5. Creating Features

2	2							Honda Jazz V	Chennai			
3	3							Maruti Ertiga VDI	Chennai			
4	4							Audi A4 New 2.0 TDI Multitronic	Coimbatore			
...	...							...	...			
7248	7248							Volkswagen Vento Diesel Trendline	Hyderabad			
7249	7249							Volkswagen Polo GT TSI	Mumbai			
7250	7250							Nissan Micra Diesel XV	Kolkata			
7251	7251							Volkswagen Polo GT TSI	Pune			
7252	7252							Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Kochi			

	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	\
0	2010.0	72000.0	CNG	Manual	First	NaN	
1	2015.0	41000.0	Diesel	Manual	First	NaN	
2	2011.0	46000.0	Petrol	Manual	First	NaN	
3	2012.0	87000.0	Diesel	Manual	First	NaN	
4	2013.0	40670.0	Diesel	Automatic	Second	NaN	
...	...	...	...	...	...	...	
7248	2011.0	89411.0	Diesel	Manual	First	NaN	
7249	2015.0	59000.0	Petrol	Automatic	First	NaN	
7250	2012.0	28000.0	Diesel	Manual	First	NaN	
7251	2013.0	52262.0	Petrol	Automatic	Third	NaN	
7252	2014.0	72443.0	Diesel	Automatic	First	NaN	

	Engine	Power	Seats	New_Price	Price	Brand	Age	\
0	998 CC	58.16 bhp	5.0	0	1.75	Maruti	14.0	
1	1582 CC	126.2 bhp	5.0	0	12.50	Hyundai	9.0	
2	1199 CC	88.7 bhp	5.0	8.61 Lakh	4.50	Honda	13.0	
3	1248 CC	88.76 bhp	7.0	0	6.00	Maruti	12.0	
4	1968 CC	140.8 bhp	5.0	0	17.74	Audi	11.0	
...	...	...	...	...	...	...	...	
7248	1598 CC	103.6 bhp	5.0	0	0.00	Volkswagen	13.0	
7249	1197 CC	103.6 bhp	5.0	0	0.00	Volkswagen	9.0	
7250	1461 CC	63.1 bhp	5.0	0	0.00	Nissan	12.0	
7251	1197 CC	103.6 bhp	5.0	0	0.00	Volkswagen	11.0	
7252	2148 CC	170 bhp	5.0	0	0.00	Mercedes-Benz	10.0	

Figure 2.5.1 Tabular Representation

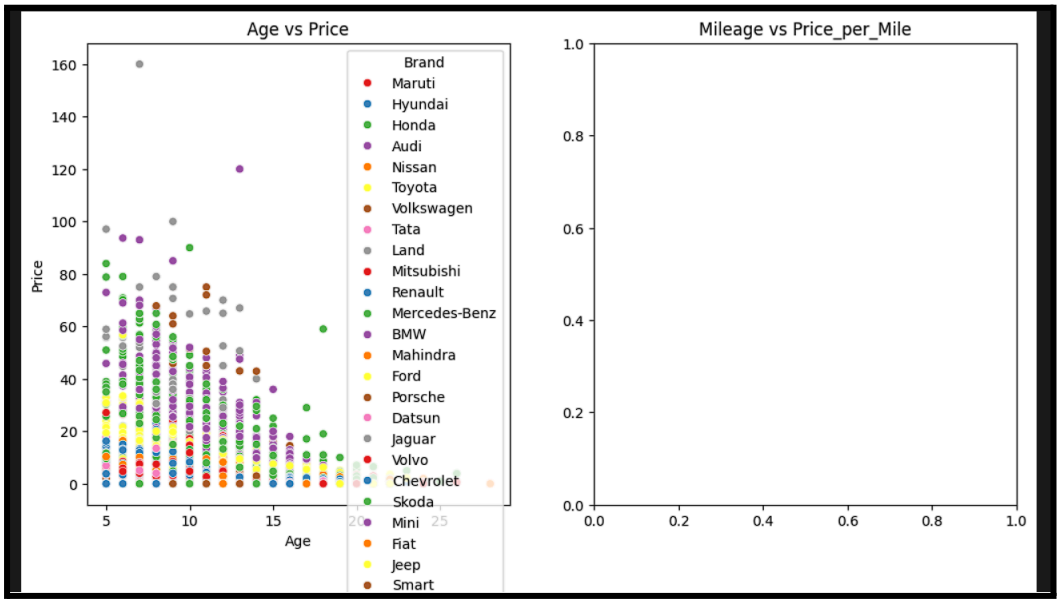


Figure 2.5.2 Graphical Representation

6. Data Cleaning/Wrangling

Cleaned and Wrangled Dataset:

	S.No.	Location	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	\
0	0	Mumbai	72000.0	CNG	Manual	First	
1	1	Pune	41000.0	Diesel	Manual	First	
2	2	Chennai	46000.0	Petrol	Manual	First	
3	3	Chennai	87000.0	Diesel	Manual	First	
4	4	Coimbatore	40670.0	Diesel	Automatic	Second	
...	...	...	...	...	...	...	
7248	7248	Hyderabad	89411.0	Diesel	Manual	First	
7249	7249	Mumbai	59000.0	Petrol	Automatic	First	
7250	7250	Kolkata	28000.0	Diesel	Manual	First	
7251	7251	Pune	52262.0	Petrol	Automatic	Third	
7252	7252	Kochi	72443.0	Diesel	Automatic	First	

	Mileage	Engine	Power	Seats	New_Price	Price	Brand	Age	\
0	26.0	998.0	58.0	5.0	0.0	1.75	Maruti	14.0	
1	19.0	1582.0	126.0	5.0	0.0	12.50	Hyundai	9.0	
2	18.0	1199.0	88.0	5.0	8.0	4.50	Honda	13.0	
3	20.0	1248.0	88.0	7.0	0.0	6.00	Maruti	12.0	
4	15.0	1968.0	140.0	5.0	0.0	17.74	Audi	11.0	
...	...	...	...	...	...	...	...	...	
7248	20.0	1598.0	103.0	5.0	0.0	0.00	Volkswagen	13.0	
7249	17.0	1197.0	103.0	5.0	0.0	0.00	Volkswagen	9.0	
7250	23.0	1461.0	63.0	5.0	0.0	0.00	Nissan	12.0	
7251	17.0	1197.0	103.0	5.0	0.0	0.00	Volkswagen	11.0	
7252	10.0	2148.0	170.0	5.0	0.0	0.00	Mercedes-Benz	10.0	

Figure 2.6 Output for task 6

7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)

Statistics Summary:

	Count	Mean	Standard Deviation	Median	Mode	\
S.No.	7191	3627.190655	2094.568997	3629.0	0.0	
Year	7191	2013.391322	3.235169	2014.0	2014.0	
Kilometers_Driven	7191	58606.050897	84711.727076	53226.0	60000.0	
Seats	7191	5.279516	0.811614	5.0	5.0	
Price	7191	7.888618	10.819356	4.7	0.0	

	Minimum Value	Maximum Value	Range
S.No.	0.0	7252.0	7252.0
Year	1996.0	2019.0	23.0
Kilometers_Driven	171.0	6500000.0	6499829.0
Seats	0.0	10.0	10.0
Price	0.0	160.0	160.0

Figure 2.7 Output for task 7

8. Analyzing/visualizing the dataset by taking one variable at a time

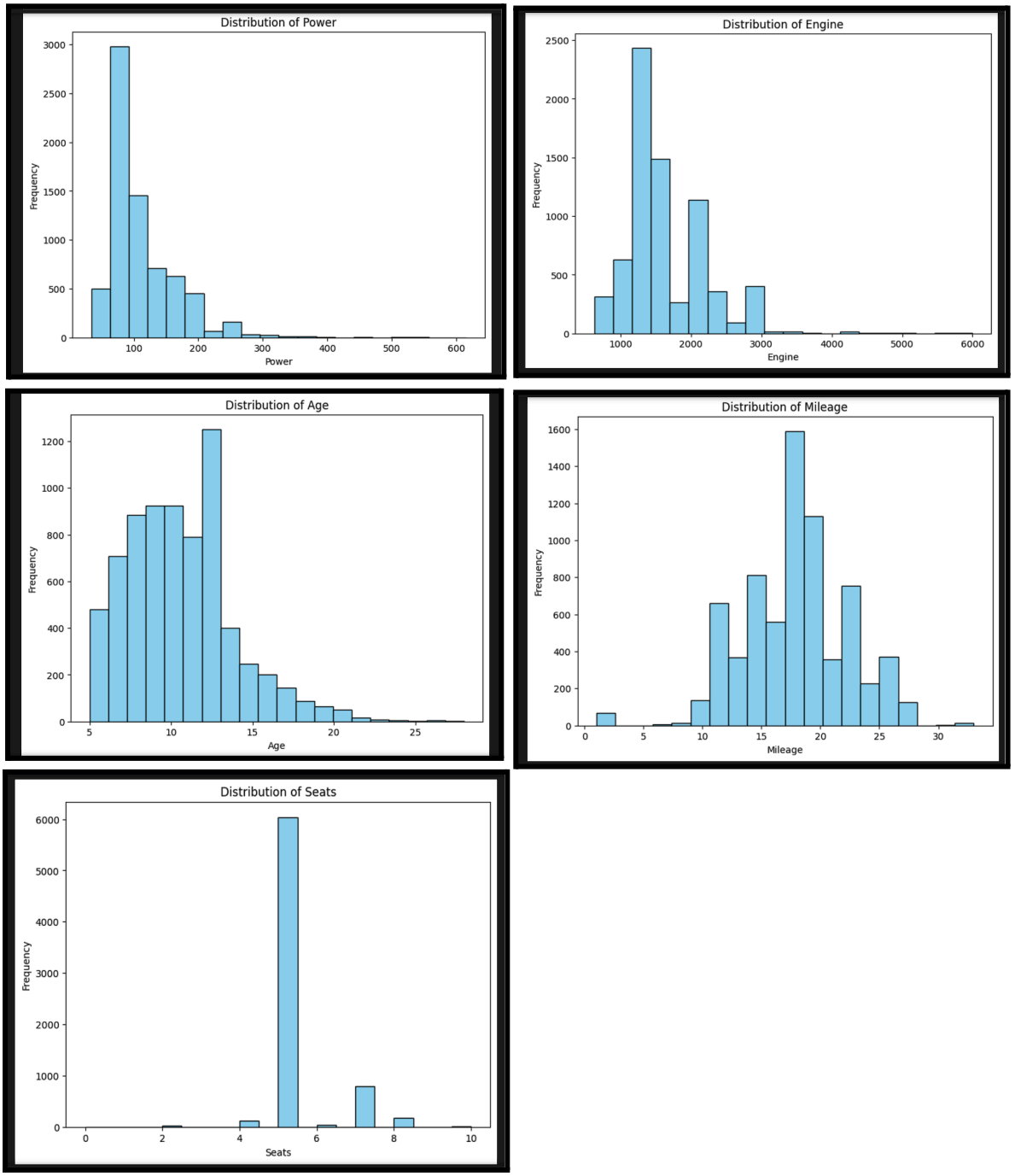


Figure 2.8 Output for task 8

9. Data Transformation

```
Transformed Dataset:
   S.No.    Location  Kilometers_Driven  Fuel_Type  Transmission Owner_Type  \
0      0      Mumbai            72000.0          0             1      First
1      1        Pune            41000.0          1             1      First
2      2     Chennai            46000.0          3             1      First
3      3     Chennai            87000.0          1             1      First
4      4  Coimbatore            40670.0          1             0     Second

    Mileage    Engine  Power  Seats  New_Price     Price  Brand   Age  \
0  1.842662 -1.039810   58.0    5.0        0.0 -0.567413     19  14.0
1  0.275923 -0.058350  126.0    5.0        0.0  0.426246     11   9.0
2  0.052103 -0.702013   88.0    5.0        8.0 -0.313221     10  13.0
3  0.499743 -0.619664   88.0    7.0        0.0 -0.174571     19  12.0
4 -0.619356  0.590354  140.0    5.0        0.0  0.910596      1  11.0

   Price_per_Mile
0        0.067308
1        0.657895
2        0.250000
3        0.300000
4        1.182667
```

Figure 2.9 Output for task 9

Conclusion:

- EDA provides a comprehensive overview of the cars dataset
- Identification and handling of missing values, outliers, and anomalies ensure data integrity and improve analysis accuracy.
- Descriptive statistics, including mean, median, and standard deviation, offer a summary of numerical attributes, aiding in understanding central tendencies and data dispersion.
- Visualization techniques, such as histograms and kernel density plots, reveal the distributions of key features, providing insights into the data's underlying patterns.
- Techniques like correlation, mutual information, or model-based feature importance assessments help prioritize variables based on their impact on the target variable.