EXPLORATORY DATA ANALYSIS

THE AUTOMOBILE INDUSTRY:
UNVEILING TRENDS AND INSIGHTS

In this analysis, I will be diving into an automobile dataset with the aim of uncovering insightful trends and patterns. The journey will begin with data cleaning, addressing any missing values using the mode and median. Following this, I'll leverage feature engineering to enhance the dataset's depth, enabling a richer exploratory analysis. This exploration will encompass a thorough examination of statistics, intricate relationships between variables, data distributions, and segmenting data based on attributes such as body style.

DATASET

The dataset comprises information about various characteristics of automobiles. Each row in the dataset represents an individual automobile, and the columns contain different attributes or features of these automobiles. Here's a brief overview of our dataset:

symboling	Insurance risk rating of the car (-3 to 3, higher value means higher risk)
normalized-losses	Relative average loss payment per insured vehicle year (normalized for all autos within a particular size classification0
make	Manufacturer or brand of the car
fuel-type	Type of fuel the car uses (gas or diesel)
aspiration	Indicates whether the engine is naturally aspirated or uses forced induction like turbocharging
num-of-doors	Number of doors in the car (two or four)
body-style	Body style of the car (sedan, convertible, hatchback, etc.)
drive-wheels	Configuration of drive wheels (rwd: rear-wheel drive, fwd: front-wheel drive, or 4wd: four-wheel drive)
engine-location	Location of the car's engine (front or rear)
wheel-base	Distance between the centers of the front and rear wheels
length	Length of the car
width	Width of the car
height	Height of the car
curb-weight	Weight of the car without occupants or baggage
engine-type	Type of engine the car has
num-of-cylinders	Number of cylinders in the car's engine
engine-size	Size of the car's engine
fuel-system	Type of fuel system in the car
bore	Internal mechanics of the car's engine (bore)
stroke	Internal mechanics of the car's engine (stroke)
compression-ratio	Ratio of the volume of the combustion chamber from its largest capacity to its smallest capacity
horsepower	Maximum power that the car's engine can produce
peak-rpm	RPM at which the car's engine delivers peak horsepower
city-mpg	Car's fuel efficiency in miles per gallon in city driving conditions
highway-mpg	Car's fuel efficiency in miles per gallon in highway driving conditions
price	Price of the car

Data source: UC Irvine Machine Learning Repository

LOADING THE DATASET

The dataset resides in a CSV file named *dataset_automobile.csv*. I will load necessary libraries and the dataset for this project. Here, the read_csv function from pandas is used to read the CSV file and convert it into a DataFrame named ‘data’. I inspect the first few rows of the DataFrame to ensure that the data has been loaded correctly. The head function returns the first 5 rows of the DataFrame. This allows to confirm that the data has been loaded properly.

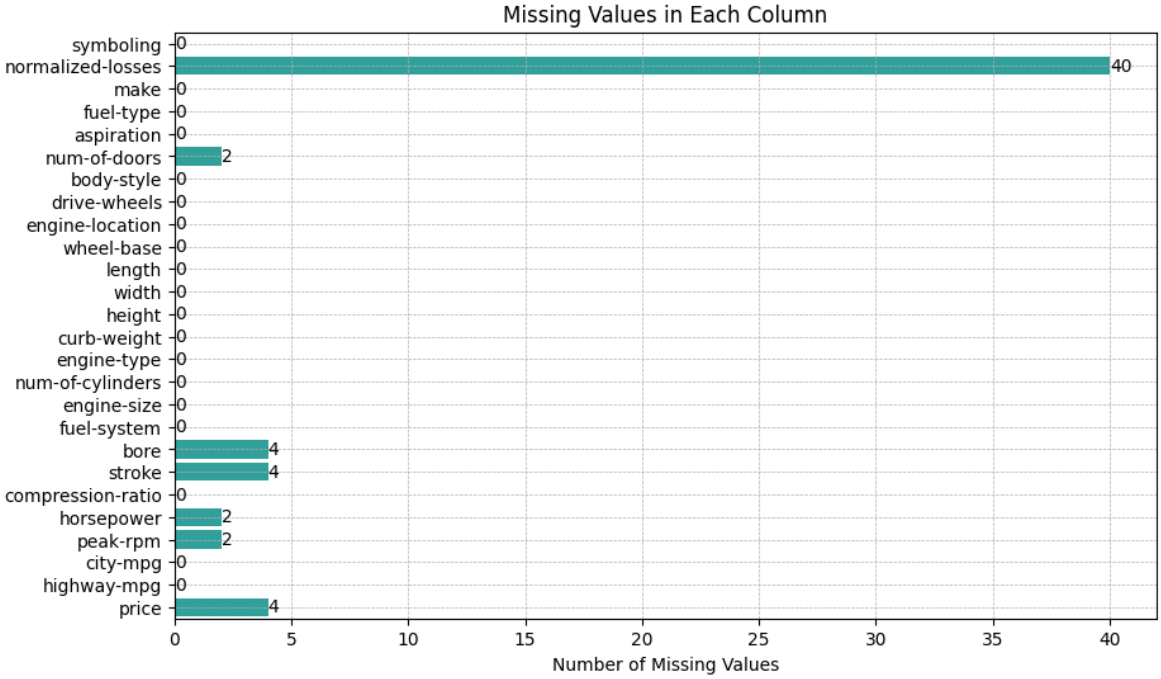
```
# Import Libraries
import pandas as pd
import numpy as np

# Load Dataset
data = pd.read_csv('dataset_automobile.csv')

# Show First Five Instance
data.head(5)
```

The dataset comprises 26 columns, each representing a different attribute of the automobiles. From an initial glance at the first few rows of the data, I can see that there are missing values represented by the symbol "?". These missing values need to be addressed as part of the data cleaning process.

HANDLING MISSING VALUES

The next step is to handle these missing values. In Python, missing values are often represented by the special float value NaN (Not a Number). I will use the *replace( )* function, to replace the "?" symbols with NaN. This replacement allows pandas to recognize the missing values and handle them appropriately in subsequent analysis. After marking the missing data, it's important to understand the extent and distribution of these missing values across different columns. This will help to decide on the best strategy for handling them. We can use the *isna( )* function from pandas, which returns a Boolean mask indicating missing values, and the *sum( )* function to count these missing values in each column.

```
# replace '?' with nan
data.replace('?', np.nan, inplace=True)

# count the number of missing values in each column
missing_values = data.isna().sum()

# plot the Missing Values
plt.figure(figsize=(10,8))
bars=sns.barplot(x=missing_values, y=missing_values.index, color='lightseagreen')
plt.xlabel('Number of Missing Values')
plt.title('Missing Values in Each Column')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
for bar in bars.patches:
    plt.text(bar.get_width(),
             bar.get_y() + bar.get_height() / 2,
             f'{bar.get_width():.0f}',
             va='center')
plt.show()
```



Most of the columns have no missing values (represented by 0). However, the 'normalized-losses' column has 40 missing values, 'num-of-doors' has 2, 'bore' and 'stroke' each have 4, 'horsepower' and 'peak-rpm' each have 2, and 'price' also has 4.
The strategy to handle these missing values can vary depending on the nature of the data. For instance, for numerical columns, I will use a mean or median value, and for categorical columns, I could use the mode (most frequent value) or a specific category like 'unknown'.

'normalized-losses', 'horsepower', 'peak-rpm', 'bore', 'stroke', and 'price'; These are numerical columns. I will replace missing values with the median of the respective column. I prefer the median over the mean because the median is not affected by outliers or extreme values, making it a more robust measure of central tendency when dealing with skewed data or data with outliers.
'num-of-doors'; This is a categorical column. I will replace missing values with the mode.

```
# Numerical Columns: replace 'nan' values with MEAN

avg_norloss = data['normalized-losses'].astype('float').median(axis=0)
data['normalized-losses'].replace(np.nan, avg_norloss, inplace=True)

avg_bore = data['bore'].astype('float').median(axis=0)
data['bore'].replace(np.nan, avg_bore, inplace=True)

avg_stroke = data['stroke'].astype('float').median(axis=0)
data['stroke'].replace(np.nan, avg_stroke, inplace=True)

avg_horsepower = data['horsepower'].astype('float').median(axis=0)
data['horsepower'].replace(np.nan, avg_horsepower, inplace=True)

avg_peakrpm = data['peak-rpm'].astype('float').median(axis=0)
data['peak-rpm'].replace(np.nan, avg_peakrpm, inplace=True)

avg_price = data['price'].astype('float').median(axis=0)
data['price'].replace(np.nan, avg_price, inplace=True)

# Categorical Columns: replace 'nan' values with MODE
mod_doors = data['num-of-doors'].mode()[0]
data['num-of-doors'].replace(np.nan, mod_doors, inplace=True)
```

FEATURE ENGINEERING

Creating new features from the existing ones can often help improve the performance of the models by providing additional information. This process is known as feature engineering. I will create two new features in the dataset:

- Power-to-Weight Ratio: This feature will be created by dividing the 'horsepower' column by the 'curb-weight' column. The idea behind this new feature is to measure the performance of a car in terms of how much power it produces per unit of weight. A higher ratio generally means better performance.
- Fuel Efficiency: This feature will be created by taking the average of 'city-mpg' and 'highway-mpg'. This gives an overall measure of the fuel efficiency of a car, regardless of where it's driven.

```
# Create 'Power-to-Weight Ratio' feature
data['power-to-weight-ratio'] = data['horsepower'] / data['curb-weight']

# Create 'Fuel Efficiency' feature (average of city and highway mpg)
data['fuel-efficiency'] = (data['city-mpg'] + data['highway-mpg']) / 2
```

EXPLORATORY DATA ANALYSIS

EDA is the tool to visualize and summarize the main characteristics of the dataset. Through it, I'll uncover patterns, relationships, and insights in the context of automobile attributes.

Descriptive Statistical Analysis:

Descriptive statistics provide a statistical summary of the dataset, including measures such as the mean, standard deviation, minimum, maximum, and quartile values. I'll use the *describe( )* function from pandas to generate these statistics.

```
data.describe()
```

	price	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke
count	204	204	204	204	204	204	204	204	204	204
mean	13148.127	120.6275	98.80637	174.075	65.91667	53.74902	2555.603	126.8922	3.328676	3.258922
std	7898.645	31.88091	5.994144	12.36212	2.146716	2.424901	521.9608	41.74457	0.271344	0.311781
min	5118	65	86.6	141.1	60.3	47.8	1488	61	2.54	2.07
25%	7784.75	101	94.5	166.3	64.075	52	2145	97	3.15	3.11
50%	10270	115	97	173.2	65.5	54.1	2414	119.5	3.31	3.29
75%	16500	137	102.4	183.2	66.9	55.5	2939.25	142	3.5825	3.41
max	45400	256	120.9	208.1	72.3	59.8	4066	326	3.94	4.17

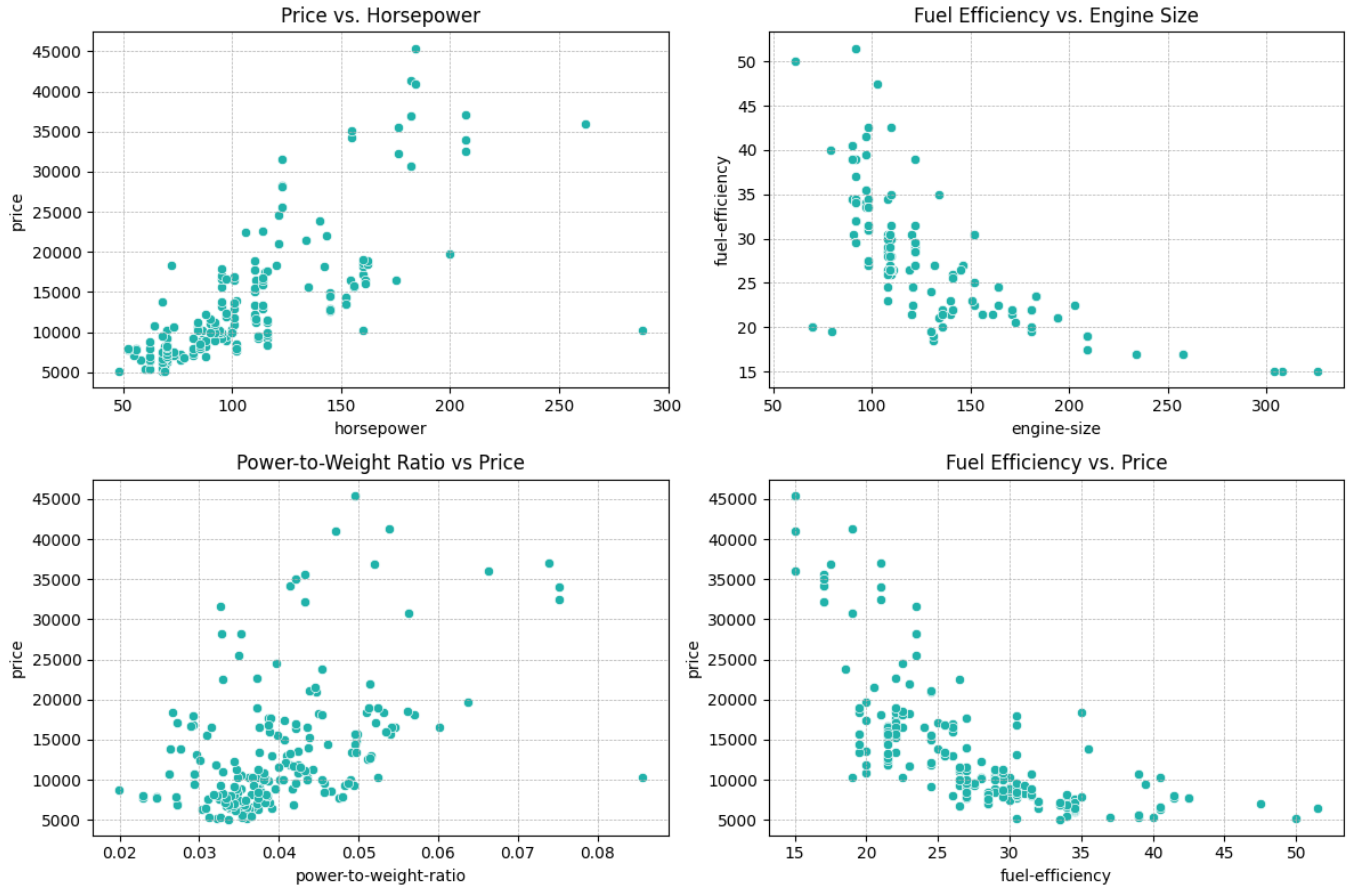
This summary provides a broad overview of the central tendency, dispersion, and distribution of the data in each numerical column. For instance, I can see that the 'price' column has a mean (average) of approximately 13148, a standard deviation of approximately 7899, a minimum value of 5118, and a maximum value of 45400.

Explore Relationships Between Variables:

To explore the relationships between 'power-to-weight-ratio', 'fuel-efficiency', and 'price', we can create scatter plots. Scatter plots are a useful way to visualize the correlation between two variables.

```
plt.figure(figsize=(12, 10))

# Price vs. Horsepower
plt.subplot(2, 2, 1)
sns.scatterplot(data=data, x='horsepower', y='price', color='lightseagreen')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.title('Price vs. Horsepower')

# Fuel Efficiency vs. Engine Size
plt.subplot(2, 2, 2)
```

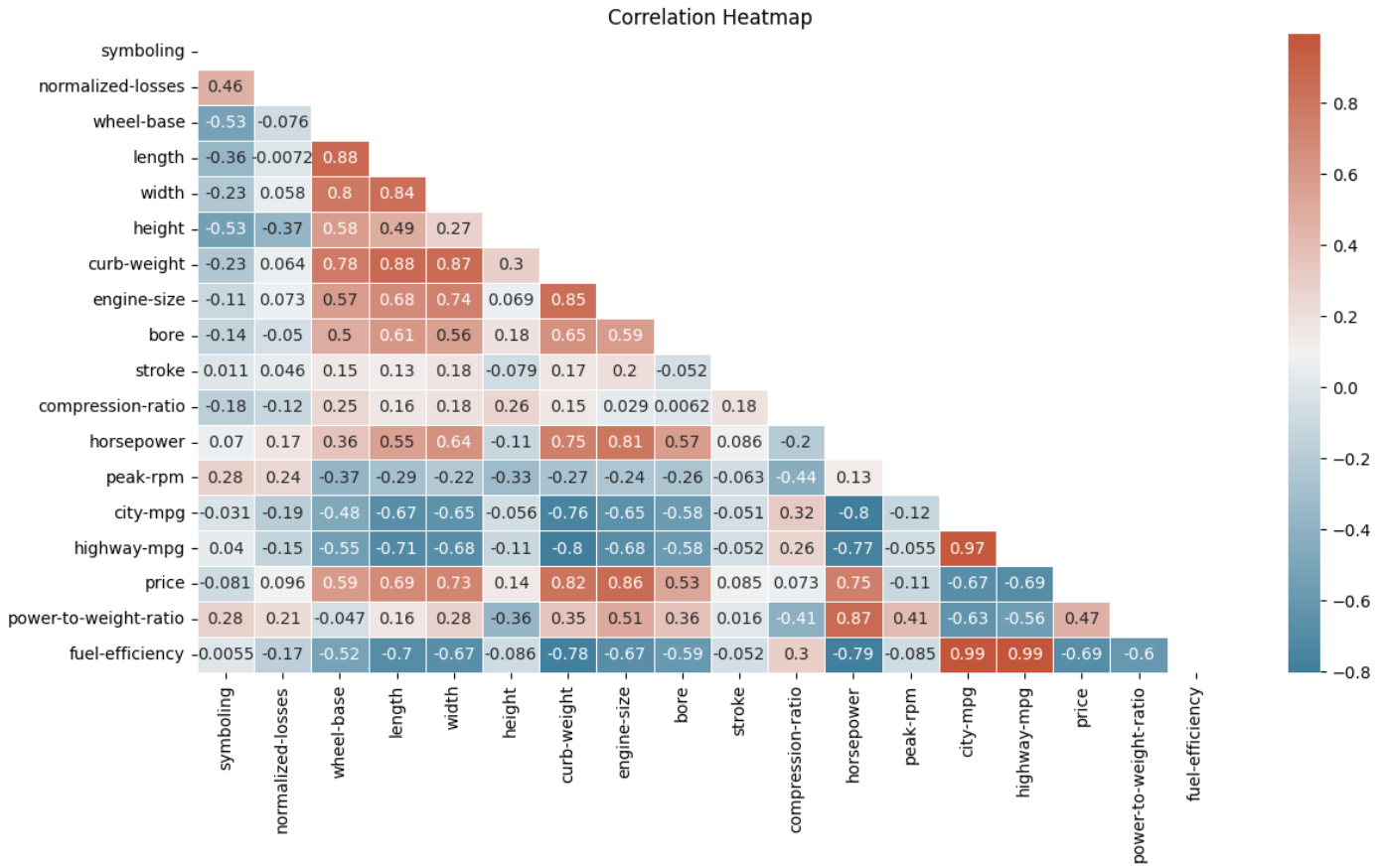
```
sns.scatterplot(data=data, x='engine-size', y='fuel-efficiency', color='lightseagreen')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.title('Fuel Efficiency vs. Engine Size')

# Power to Weight vs. Price
plt.subplot(2, 2, 3)
sns.scatterplot(data=data, x='power-to-weight-ratio', y='price', color='lightseagreen')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.title('Power-to-Weight Ratio vs Price')

# Fuel Efficiency vs. Price
plt.subplot(2, 2, 4)
sns.scatterplot(data=data, x='fuel-efficiency', y='price', color='lightseagreen')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.title('Fuel Efficiency vs. Price')

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



- Horsepower vs Price: The first plot shows a positive correlation, indicating that cars with higher horsepower tend to be more expensive. This suggests that power is a contributing factor to the price of a car.
- Fuel Efficiency vs Engine Size: The second plot shows a negative correlation, indicating that cars with larger engines tend to have lower fuel efficiency. This suggests that larger engines consume more fuel, reducing fuel efficiency.
- Power-to-Weight Ratio vs Price: The third plot does not show a clear correlation, but there seems to be a slight positive trend. This suggests that cars with a higher power-to-weight ratio might be slightly more expensive, possibly indicating that performance contributes to the price.
- Fuel Efficiency vs Price: The fourth plot shows a negative correlation, indicating that more fuel-efficient cars (those with higher miles per gallon) tend to be less expensive. This could be because fuel-efficient cars often have smaller engines or are designed for economy rather than performance, factors that could lead to a lower price.

Correlation or Patterns in the Dataset:
Correlation coefficients are used in statistics to measure how strong a relationship is between two variables. The correlation coefficient ranges from -1 to 1. If the value is close to 1, it means that there is a strong positive correlation between the two variables. When it is close to -1, the variables have a strong negative correlation. A correlation matrix is a table showing the value of the correlation coefficient between sets of variables. Let's compute the correlation matrix for this dataset and visualize it using a heatmap for better interpretability.

```
# Calculate the correlation matrix
corr = data.select_dtypes(include=[np.number]).corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
cmap = sns.diverging_palette(230, 20, as_cmap=True)
plt.figure(figsize=(14, 10))
sns.heatmap(corr, mask=mask, annot=True, cmap=cmap, linewidths=0.5, linecolor='white')
plt.title('Correlation Heatmap')
plt.show()
```

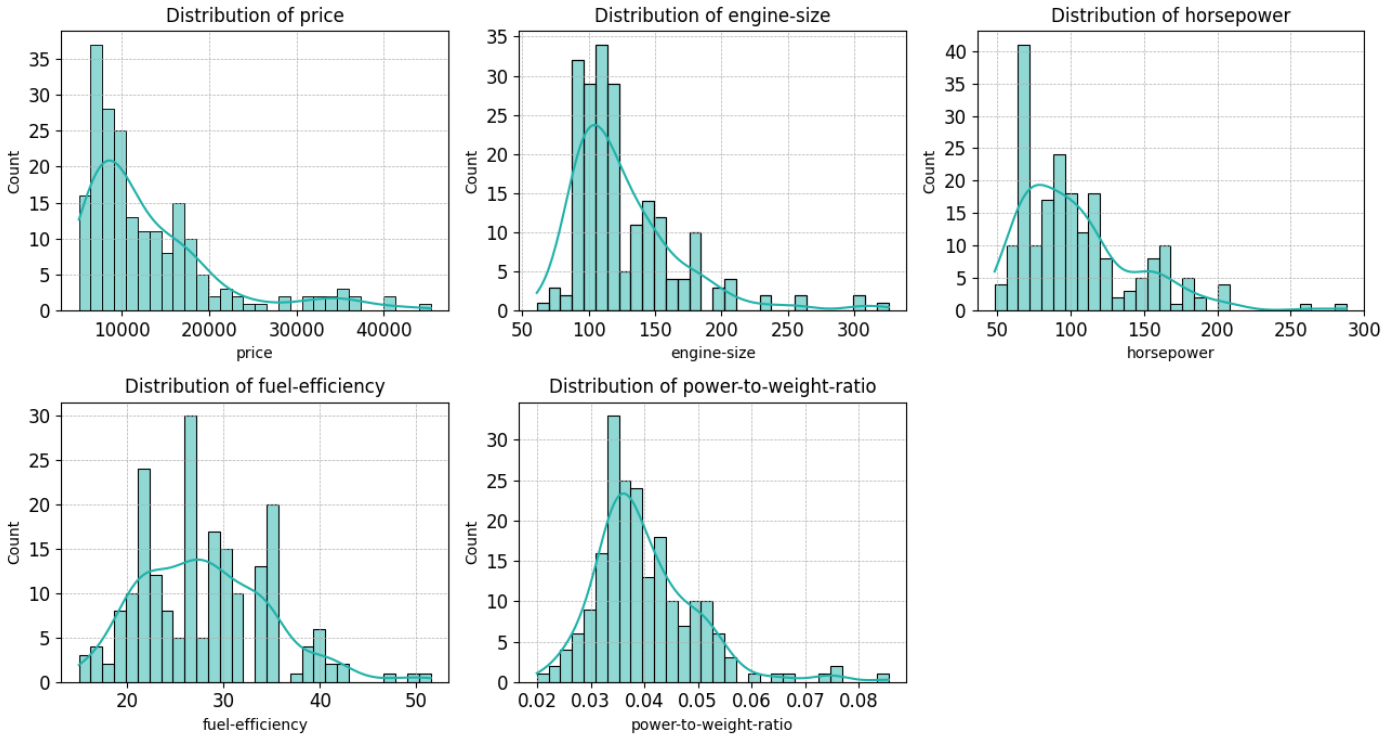


The heatmap provides a visual representation of the correlation matrix, allows to easily see the correlations between different features in the dataset. Here are a few observations from the heatmap:

- Horsepower and Price: There's a positive correlation between horsepower and price, which aligns with our scatter plot analysis. This suggests that cars with higher horsepower tend to be more expensive.
- Fuel Efficiency and City-MPG/Highway-MPG: As expected, there's a strong positive correlation between fuel efficiency and both city and highway miles per gallon. This makes sense, as our fuel efficiency feature is calculated as the average of city and highway MPG.
- Engine Size and Curb Weight: There's a strong positive correlation between engine size and curb weight. This indicates that cars with larger engines tend to be heavier.
- Fuel Efficiency and Engine Size/Curb Weight/Horsepower: These pairs show negative correlations. This suggests that cars with larger engines, heavier weights, or higher horsepower tend to have lower fuel efficiency.

Examine the Distributions of Key Variables:
To examine the distributions of variables in the dataset, I can create histograms. It is an estimate of the probability distribution of a continuous variable. Let's create histograms for a few variables in our dataset: 'price', 'horsepower', 'engine-size', 'fuel-efficiency', and 'power-to-weight-ratio'.

```
# List of key variables to visualize
key_vars = [
    'price', 'engine-size', 'horsepower',
    'fuel-efficiency', 'power-to-weight-ratio']

# Plotting histograms for the key variables
plt.figure(figsize=(13, 7))
for i, col in enumerate(key_vars, 1):
    plt.subplot(2, 3, i)
    sns.histplot(data[col], bins=30, kde=True, color='lightseagreen')
    plt.grid(True, which='both', linestyle='--', linewidth=0.5)
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)
    plt.title(f'Distribution of {col}')
    plt.tight_layout()
plt.show()
```



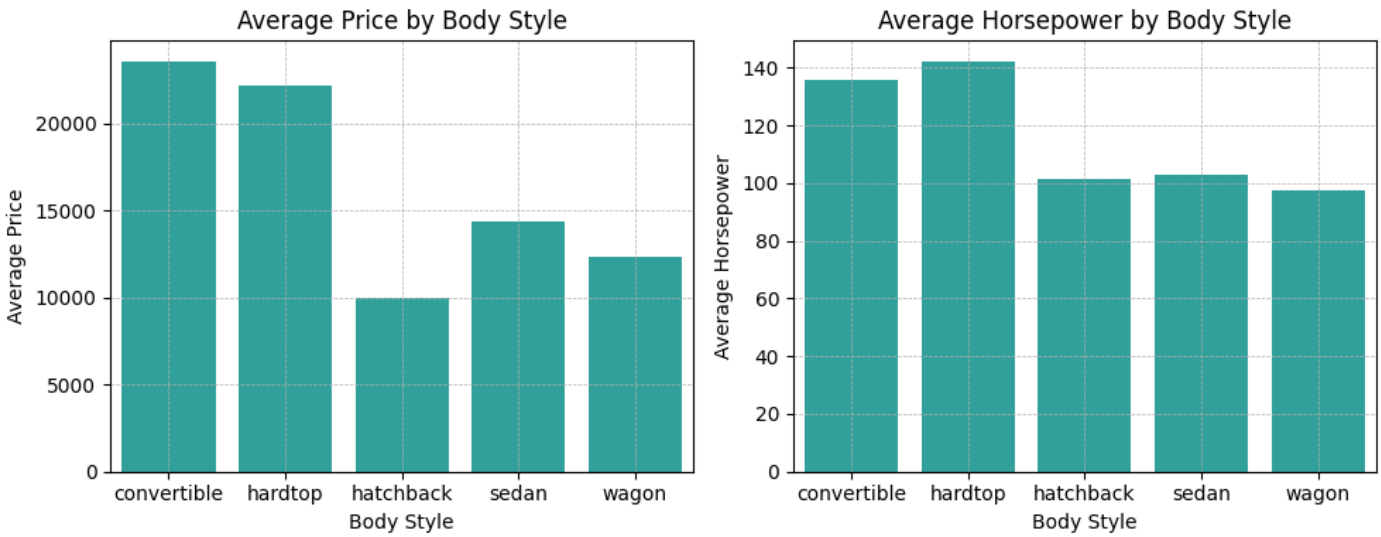
The histograms above provide a visual representation of the distributions of various variables in our dataset:

- Distribution of Price: The histogram reveals that the majority of cars are priced in the lower range, with fewer cars in the high price range. This indicates a right-skewed distribution.
- Distribution of Horsepower: The majority of cars have horsepower in the lower to mid-range. Like the price, the distribution of horsepower is right-skewed.
- Distribution of Engine Size: The engine size of most cars is in the lower to mid-range, and fewer cars have larger engines. This distribution is also right-skewed.
- Distribution of Fuel Efficiency: The histogram shows that the fuel efficiency of cars is approximately normally distributed, with the majority of cars having average fuel efficiency.
- Distribution of Power-to-Weight Ratio: The power-to-weight ratio of cars is approximately normally distributed, with most cars having a moderate power-to-weight ratio.

Examining these distributions helps us understand the characteristics of this dataset. For example, I can see that this dataset contains more cars with lower prices, smaller engines, and lower horsepower.

Segmentation Analysis: Average Price and Horsepower by Body Style:
Segmentation analysis involves dividing the data into different segments and analyzing each of these groups separately. In this case, let's divide the cars into different segments based on the 'body-style' feature, which represents the body type of each car (e.g., sedan, hatchback, etc.). For each body style, we will compute the average price and average horsepower. This will help us understand if there are differences in price and horsepower across different car body styles. Let's create bar plots to visualize the average price and horsepower for each body style. This will allow us to easily compare these metrics across different body styles. We'll use seaborn library to create these plots.

```
# Define the variables to plot
variables = ['price', 'horsepower']
titles = ['Average Price by Body Style', 'Average Horsepower by Body Style']
y_labels = ['Average Price', 'Average Horsepower']

# Grouping by 'body-style' and computing average price and horsepower
segmented = data.groupby('body-style')[['price', 'horsepower']].mean()
segmented = segmented.reset_index()

# Creating the Bar Plot
plt.figure(figsize=(12, 5))
for i, var in enumerate(variables):
    plt.subplot(1, 2, i+1)
    sns.barplot(x='body-style', y=var, data=segmented, color='lightseagreen')
    plt.title(titles[i])
    plt.xlabel('Body Style')
    plt.ylabel(y_labels[i])
    plt.grid(True, which='both', linestyle='--', linewidth=0.5)

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



The bar plots above provide a visual representation of the average price and horsepower for each body style of car. Here are the insights we can draw from these plots:

- Average Price by Body Style: Convertibles and hardtops have the highest average prices, indicating that these types of cars tend to be more expensive. On the other hand, hatchbacks have the lowest average prices, suggesting that they are generally more affordable. Sedans and wagons fall in the middle.
- Average Horsepower by Body Style: Similarly, convertibles and hardtops also have the highest average horsepower, suggesting that these cars are typically more performance-oriented. Hatchbacks, sedans, and wagons have relatively lower average horsepower.

These visualizations clearly illustrate the differences in average price and horsepower across different body styles.

CONCLUSION

In the exploration of the automobile dataset, I unearthed several intriguing insights:

- Price and Performance: Cars with higher horsepower generally command higher prices. This suggests consumers might value power when making purchase decisions.
- Fuel Efficiency: Cars with larger engines tend to be less fuel-efficient. This highlights a trade-off between power and fuel economy.
- Car Body Styles: Convertibles and hardtops, which are typically seen as luxury or sporty options, are priced higher and have more horsepower on average. In contrast, hatchbacks are more budget-friendly and have modest performance attributes.
- Missing Data: We identified and addressed missing data points, ensuring our analysis was based on a complete and robust dataset.
- Feature Engineering: New features like 'power-to-weight-ratio' and 'fuel-efficiency' were introduced, providing fresh perspectives on car performance and economy.

In essence, this analysis underscores the relationship between a car's attributes and its market price, with performance and body style being pivotal factors.