EDA DIGITAL ASSIGNMENT

BISWAYAN MANDAL - 21BDS0024

Performing Exploration on given auto.csv dataset

1. Loading the Dataset

We use the pandas library to read the dataset from a CSV file into a DataFrame. This allows us to explore and manipulate the data efficiently. We also check the dataset's size (number of rows and columns), column names, data types, a sample of the first few rows, and the count of missing values.

---

2. Checking for Duplicates

After loading the data, we check for duplicate rows using the duplicated() method. Duplicate rows can distort analysis and models, so if any are found, they are removed using drop_duplicates().

---

3. Summary Statistics

We use the describe() method to get key statistical metrics (mean, standard deviation, min, max, and percentiles) for all numerical columns. This gives an initial understanding of the data distribution and highlights potential anomalies like very large or small values.

---

4. Univariate Analysis

Univariate analysis focuses on one feature at a time:

- We use histograms to visualize the distribution of each numerical column.
- These histograms help identify the central tendency (mean or median), spread (variance or standard deviation), and shape (e.g., skewness) of the data.
- For example, the mpg histogram shows how car fuel efficiencies are distributed.

5. Bivariate Analysis

Bivariate analysis examines relationships between two variables:

- Scatterplots are used to visualize how one numerical feature (e.g., weight) relates to another (e.g., mpg).
- This helps uncover trends like positive, negative, or no correlation.
- For example, weight and mpg might show a negative correlation (heavier cars typically have lower mileage).

6. Multivariate Analysis

Multivariate analysis examines relationships among three or more variables:

- A **correlation heatmap** is used to see how strongly numerical variables are related to one another.
- Before calculating correlations, non-numeric columns (e.g., name) are excluded because they cannot be correlated numerically.
- The heatmap visually represents correlations using color intensity. Strong positive correlations are marked with dark shades (close to 1), while strong negative correlations (close to -1) are marked with the opposite.

7. Handling Errors (ValueError Issue)

The correlation calculation failed initially because the dataset included non-numeric columns like name, which cannot be processed for numerical correlations. To fix this:

- We selected only numeric columns using select_dtypes() before calculating correlations.
- This ensures that the correlation matrix contains valid numerical data only.

In [1]:

```
import pandas as pd

# Load the dataset
file_path = 'C:/smth/Auto.csv'  # Replace with the correct path
data = pd.read_csv(file_path)

# Display basic information
print("Shape of the dataset:", data.shape)
print("\nColumns in the dataset:\n", data.columns)
print("\nData types:\n", data.dtypes)
print("\nFirst few rows of the dataset:\n", data.head())
print("\nMissing values:\n", data.isnull().sum())
```

```
Shape of the dataset: (392, 9)

Columns in the dataset:
 Index(['mpg', 'cylinders', 'displacement', 'horsepower', 'weight',
       'acceleration', 'year', 'origin', 'name'],
      dtype='object')

Data types:
 mpg             float64
cylinders         int64
displacement    float64
horsepower        int64
weight            int64
acceleration    float64
year              int64
origin            int64
name             object
dtype: object

First few rows of the dataset:
     mpg  cylinders  displacement  horsepower  weight  acceleration  year  \
0  18.0          8         307.0         130    3504          12.0    70
1  15.0          8         350.0         165    3693          11.5    70
2  18.0          8         318.0         150    3436          11.0    70
3  16.0          8         304.0         150    3433          12.0    70
4  17.0          8         302.0         140    3449          10.5    70

   origin                       name
0       1  chevrolet chevelle malibu
1       1          buick skylark 320
2       1         plymouth satellite
3       1              amc rebel sst
4       1                ford torino

Missing values:
 mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
year            0
origin          0
name            0
dtype: int64
```

In [2]:

```
# Summary statistics
print("\nSummary Statistics:\n", data.describe())

# Check and handle duplicates
duplicates = data.duplicated().sum()
print(f"\nNumber of duplicate rows: {duplicates}")
if duplicates > 0:
    data = data.drop_duplicates()
```

```
Summary Statistics:
              mpg   cylinders  displacement  horsepower       weight  \
count  392.000000  392.000000    392.000000  392.000000   392.000000   
mean    23.445918    5.471939    194.411990  104.469388  2977.584184   
std      7.805007    1.705783    104.644004   38.491160   849.402560   
min      9.000000    3.000000     68.000000   46.000000  1613.000000   
25%     17.000000    4.000000    105.000000   75.000000  2225.250000   
50%     22.750000    4.000000    151.000000   93.500000  2803.500000   
75%     29.000000    8.000000    275.750000  126.000000  3614.750000   
max     46.600000    8.000000    455.000000  230.000000  5140.000000   

       acceleration        year      origin  
count    392.000000  392.000000  392.000000  
mean      15.541327   75.979592    1.576531  
std        2.758864    3.683737    0.805518  
min        8.000000   70.000000    1.000000  
25%       13.775000   73.000000    1.000000  
50%       15.500000   76.000000    1.000000  
75%       17.025000   79.000000    2.000000  
max       24.800000   82.000000    3.000000  

Number of duplicate rows: 0
```

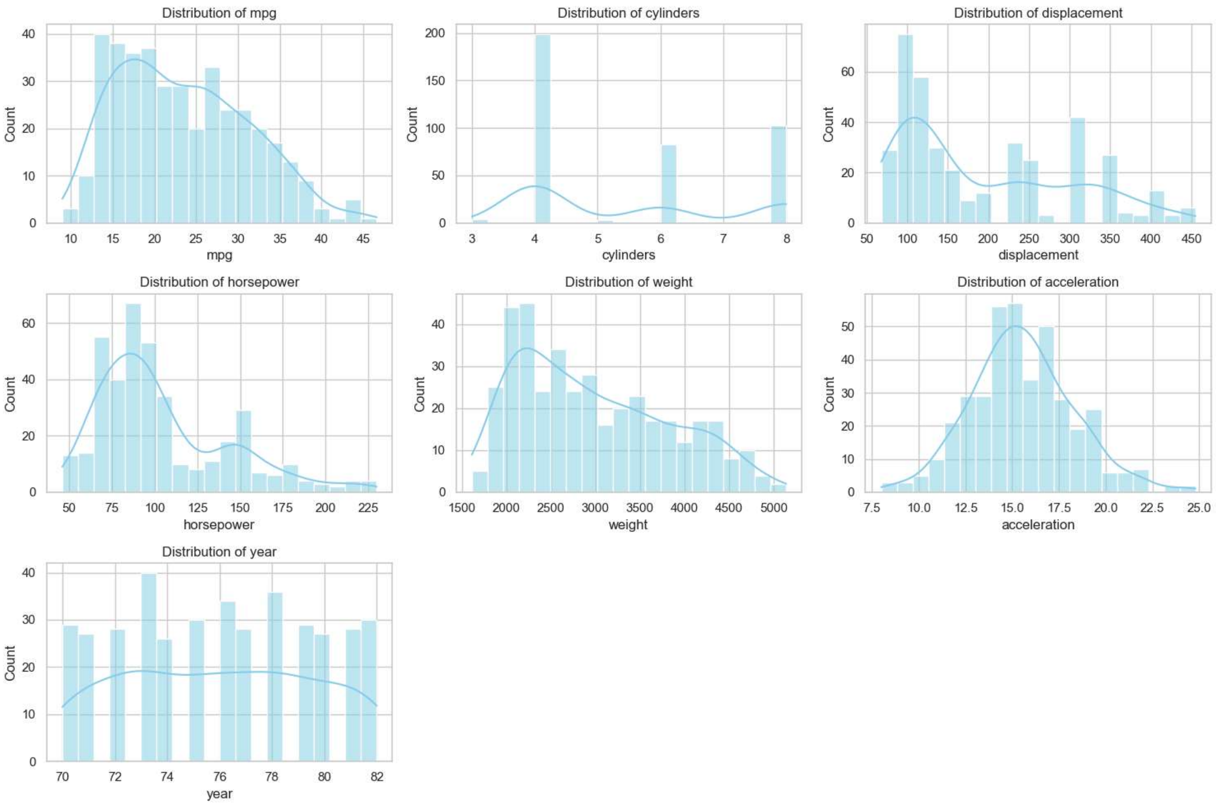
In [3]:

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

sns.set(style="whitegrid")
plt.figure(figsize=(15, 10))

# Plot distributions for all numerical features
numerical_columns = ['mpg', 'cylinders', 'displacement', 'horsepower', 'weight', 'a
for i, column in enumerate(numerical_columns):
    plt.subplot(3, 3, i + 1)
    sns.histplot(data[column], kde=True, bins=20, color='skyblue')
    plt.title(f'Distribution of {column}')

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



Distribution of mpg

Distribution of cylinders

Distribution of displacement

Distribution of horsepower

Distribution of weight

Distribution of acceleration

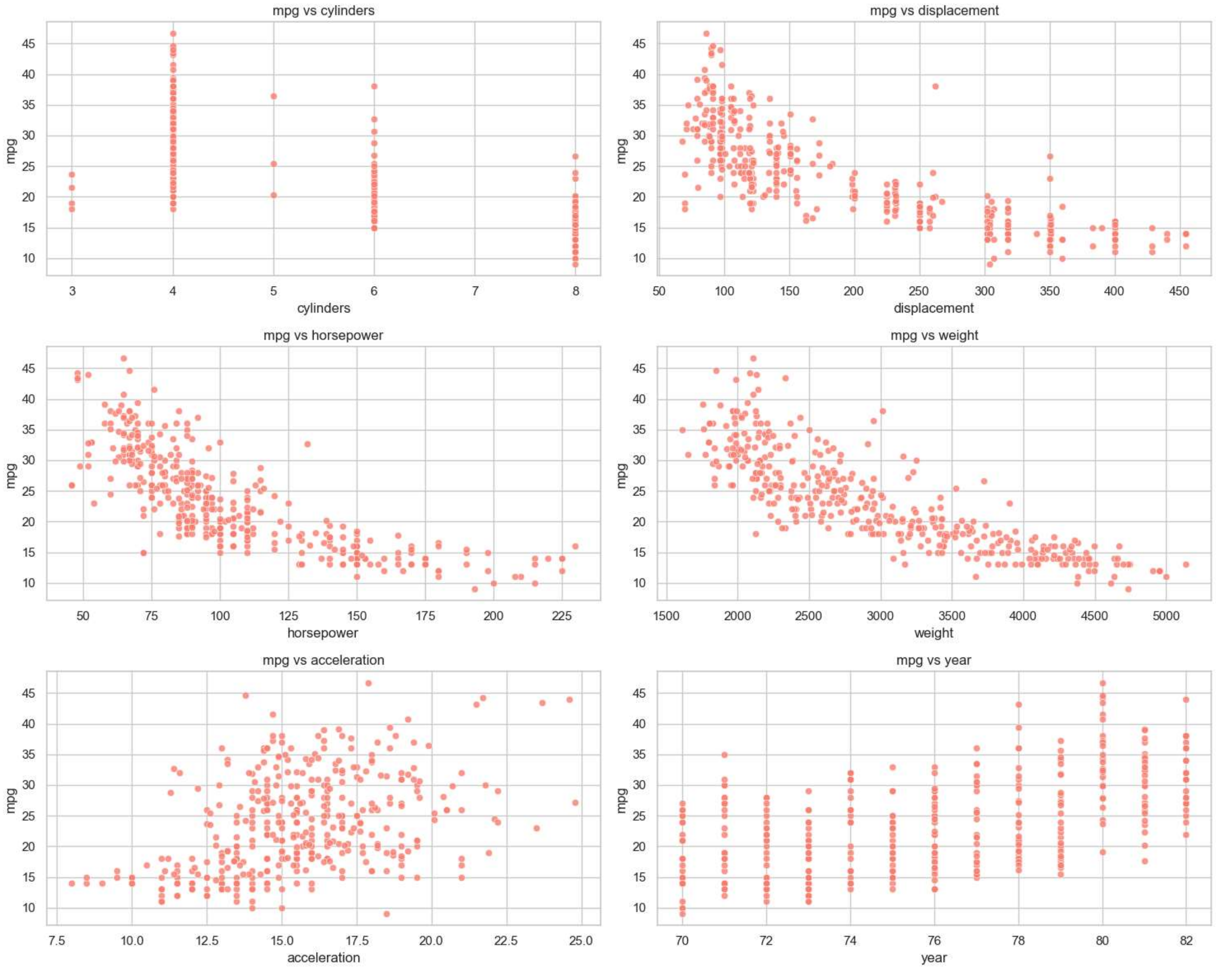
Distribution of year

In [4]:

```
plt.figure(figsize=(15, 12))

# Scatterplots for mpg vs other numerical features
target = 'mpg'
features = ['cylinders', 'displacement', 'horsepower', 'weight', 'acceleration', 'y
for i, feature in enumerate(features):
    plt.subplot(3, 2, i + 1)
    sns.scatterplot(x=data[feature], y=data[target], alpha=0.8, color='salmon')
    plt.title(f'{target} vs {feature}')
    plt.xlabel(feature)
    plt.ylabel(target)

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



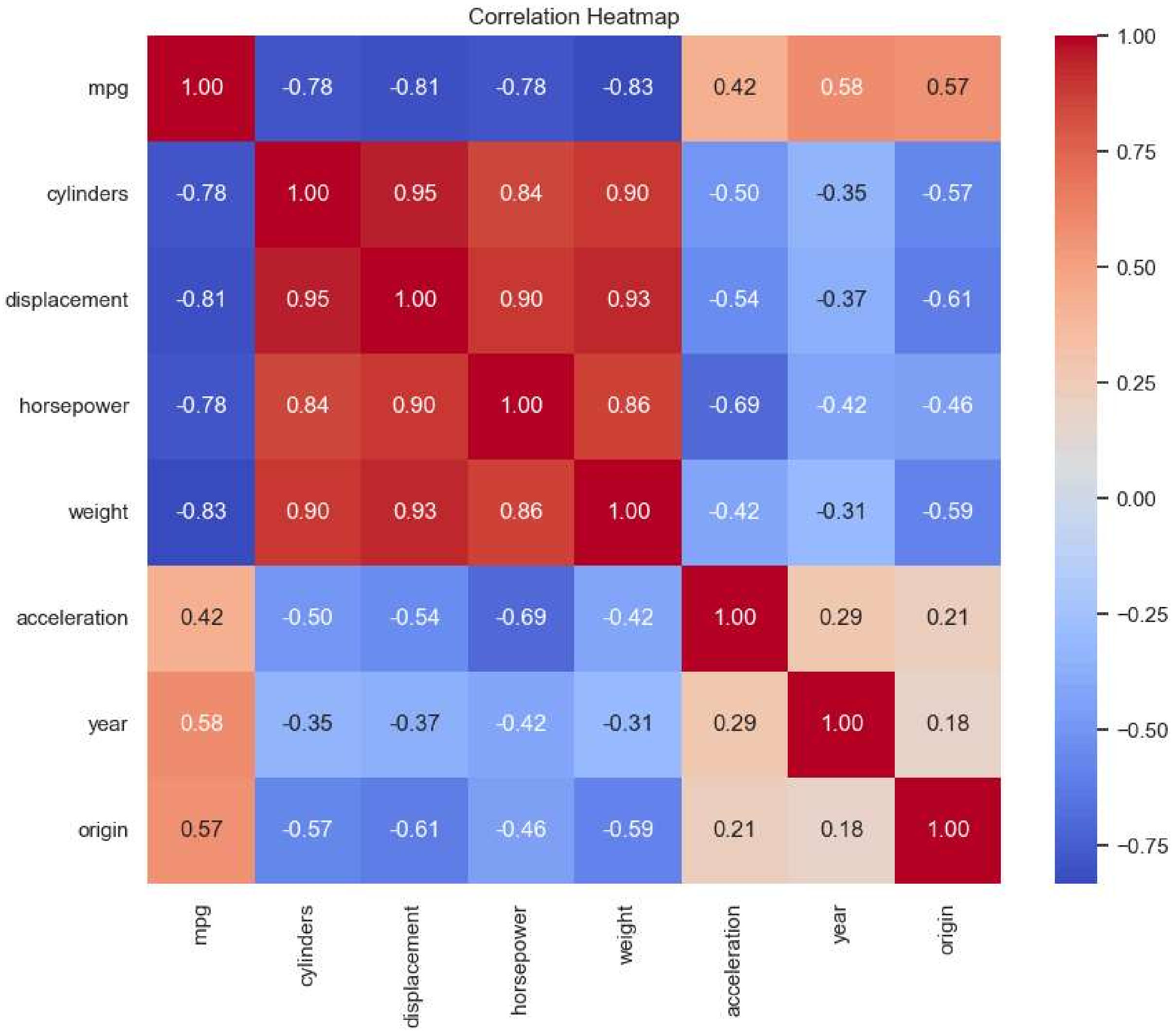
In [6]:

```python
# Correlation heatmap for numerical columns only
plt.figure(figsize=(10, 8))

# Select only numeric columns
numeric_data = data.select_dtypes(include=['float64', 'int64'])

# Calculate the correlation matrix
correlation_matrix = numeric_data.corr()

# Plot the heatmap
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Correlation Heatmap")
plt.show()
```


Correlation Heatmap
mpg 1.00 -0.78 -0.81 -0.78 -0.83 0.42 0.58 0.57
cylinders -0.78 1.00 0.95 0.84 0.90 -0.50 -0.35 -0.57
displacement -0.81 0.95 1.00 0.90 0.93 -0.54 -0.37 -0.61
horsepower -0.78 0.84 0.90 1.00 0.86 -0.69 -0.42 -0.46
weight -0.83 0.90 0.93 0.86 1.00 -0.42 -0.31 -0.59
acceleration 0.42 -0.50 -0.54 -0.69 -0.42 1.00 0.29 0.21
year 0.58 -0.35 -0.37 -0.42 -0.31 0.29 1.00 0.18
origin 0.57 -0.57 -0.61 -0.46 -0.59 0.21 0.18 1.00
mpg
cylinders
displacement
horsepower
weight
acceleration
year
origin
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

```
In [ ]:
```