ROLL NO:231501112 NAME:NISHA S

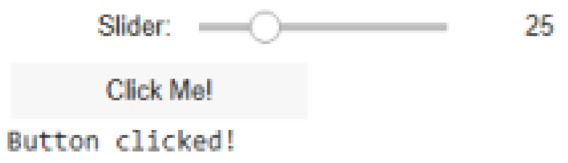
1.SETTING UP THE PYTHON ENVIRONMENT AND LIBRARIES-JUYPTER NOTEBOOK

Create a new notebook for Python
Write and execute Python code
Create new cells for code and Markdown
Demonstrate the application of Jupyter Widgets, Jupyter AI

PROGRAM:

```
import ipywidgets as widgets
from IPython.display import display
slider = widgets.IntSlider(description='Slider:', min=0, max=100,
value=25)
display(slider)
button = widgets.Button(description="Click Me!")
display(button)
def on_button_click(b):
 print("Button clicked!")

button.on_click(on_button_click)
```

OUTPUT:



```
Button clicked!
```

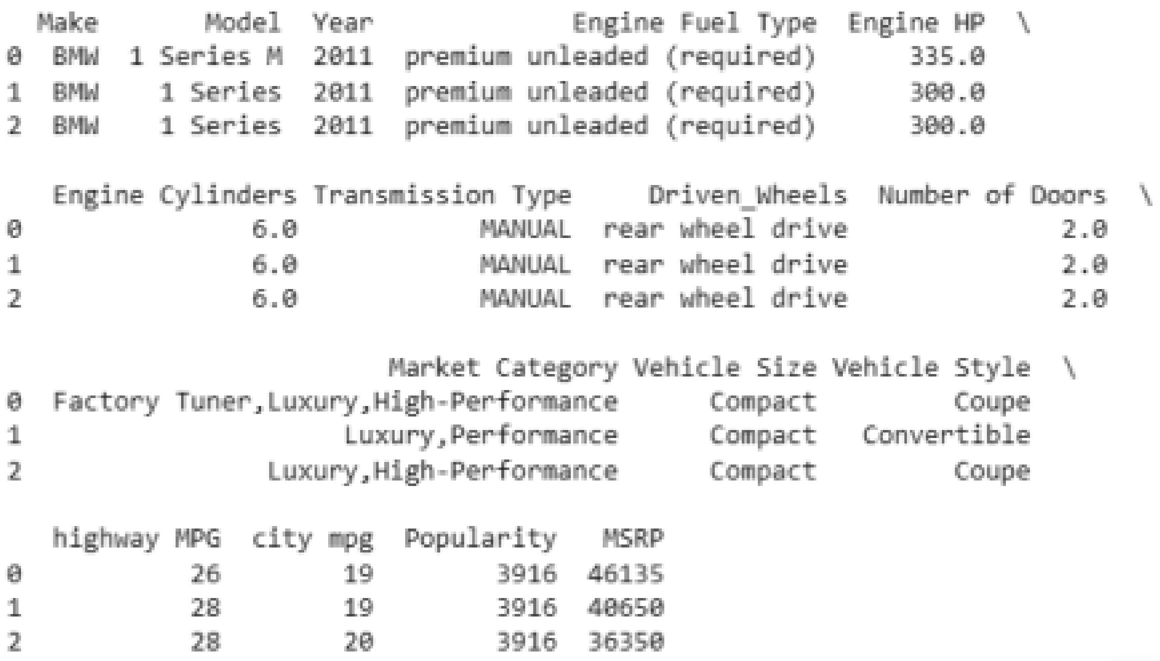
2.EDA-Data Import and Export

Importing data from CSV, Excel, SQL databases, and web scraping
Handling different data formats
Export a DataFrame to an Excel file

program:

```
import numpy as np
import pandas as pd
csv_data = pd.read_csv('/content/data - data.csv')
print(csv_data.head(3))
```

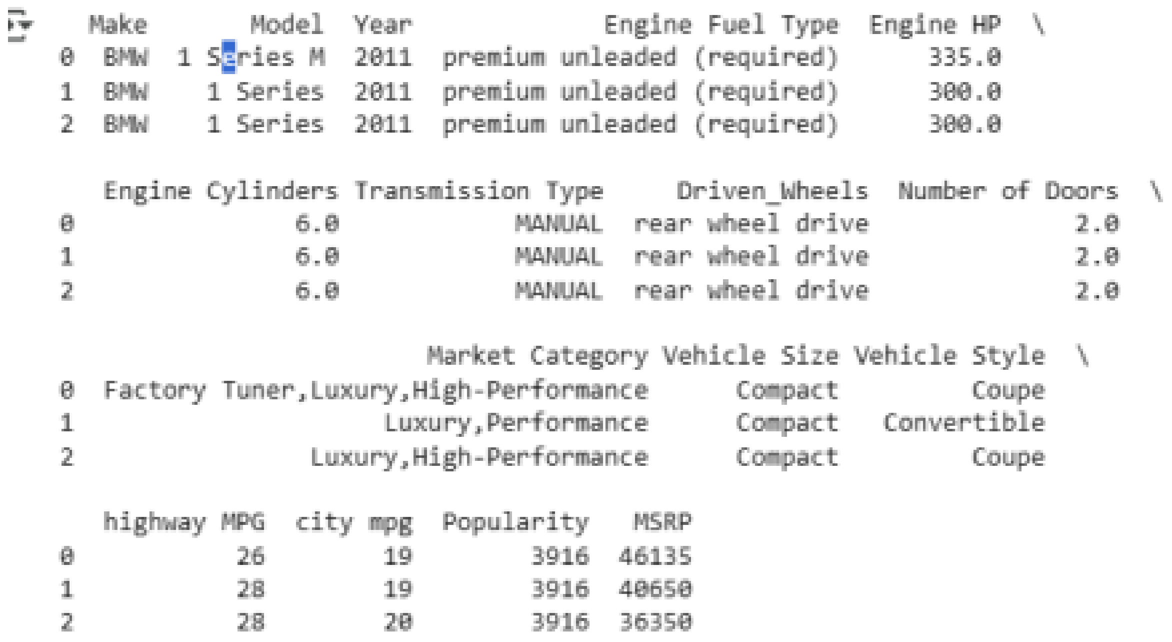
OUTPUT:



```
  Make       Model  Year             Engine Fuel Type  Engine HP  \
0  BMW  1 Series M  2011  premium unleaded (required)      335.0
1  BMW    1 Series  2011  premium unleaded (required)      300.0
2  BMW    1 Series  2011  premium unleaded (required)      300.0

   Engine Cylinders Transmission Type     Driven_Wheels  Number of Doors  \
0               6.0            MANUAL  rear wheel drive              2.0
1               6.0            MANUAL  rear wheel drive              2.0
2               6.0            MANUAL  rear wheel drive              2.0

                         Market Category Vehicle Size Vehicle Style  \
0  Factory Tuner,Luxury,High-Performance      Compact         Coupe
1                     Luxury,Performance      Compact   Convertible
2                Luxury,High-Performance      Compact         Coupe

   highway MPG  city mpg  Popularity   MSRP
0           26        19        3916  46135
1           28        19        3916  40650
2           28        20        3916  36350
```

```
excel_data = pd.read_excel('/content/data.xlsx')
print(excel_data.head(3))
```

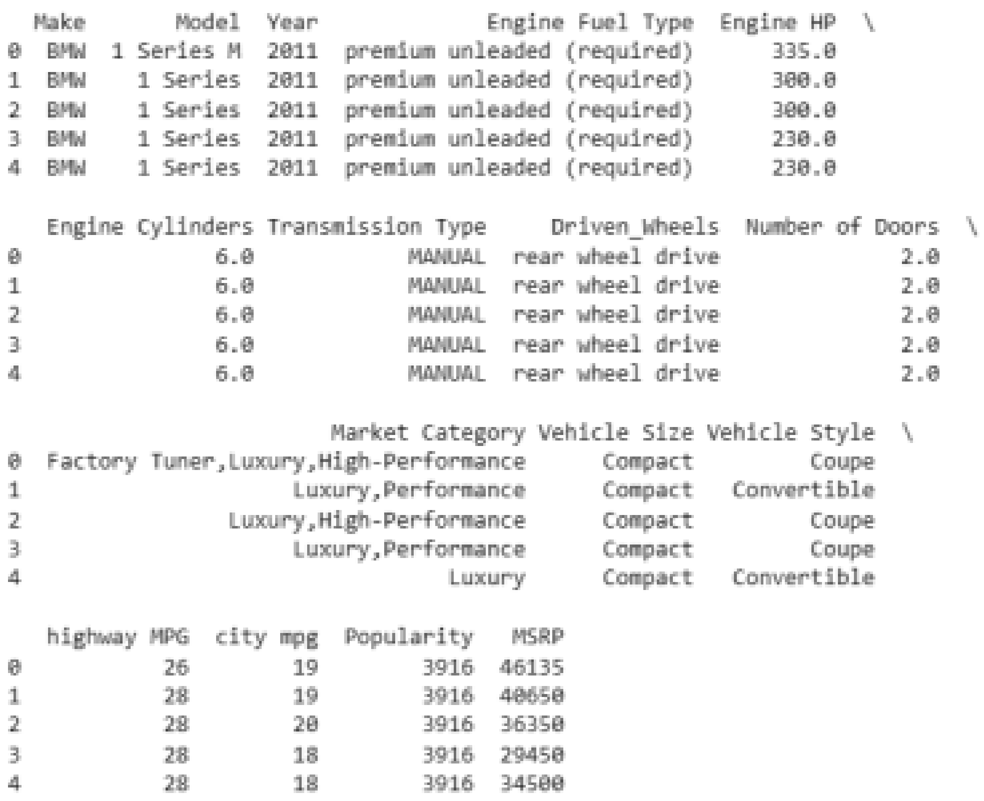
OUTPUT:



```
   Make       Model  Year             Engine Fuel Type  Engine HP  \
0   BMW  1 Series M  2011  premium unleaded (required)      335.0
1   BMW    1 Series  2011  premium unleaded (required)      300.0
2   BMW    1 Series  2011  premium unleaded (required)      300.0

   Engine Cylinders Transmission Type     Driven_Wheels  Number of Doors  \
0               6.0            MANUAL  rear wheel drive              2.0
1               6.0            MANUAL  rear wheel drive              2.0
2               6.0            MANUAL  rear wheel drive              2.0

                        Market Category Vehicle Size Vehicle Style  \
0  Factory Tuner,Luxury,High-Performance      Compact         Coupe
1                     Luxury,Performance      Compact   Convertible
2                Luxury,High-Performance      Compact         Coupe

   highway MPG  city mpg  Popularity   MSRP
0           26        19        3916  46135
1           28        19        3916  40650
2           28        20        3916  36350
```

```
csv_data.to_html('data.htm', index=False)
df_scraped = pd.read_html('data.htm')[0]
print(df_scraped.head())
```

OUTPUT:

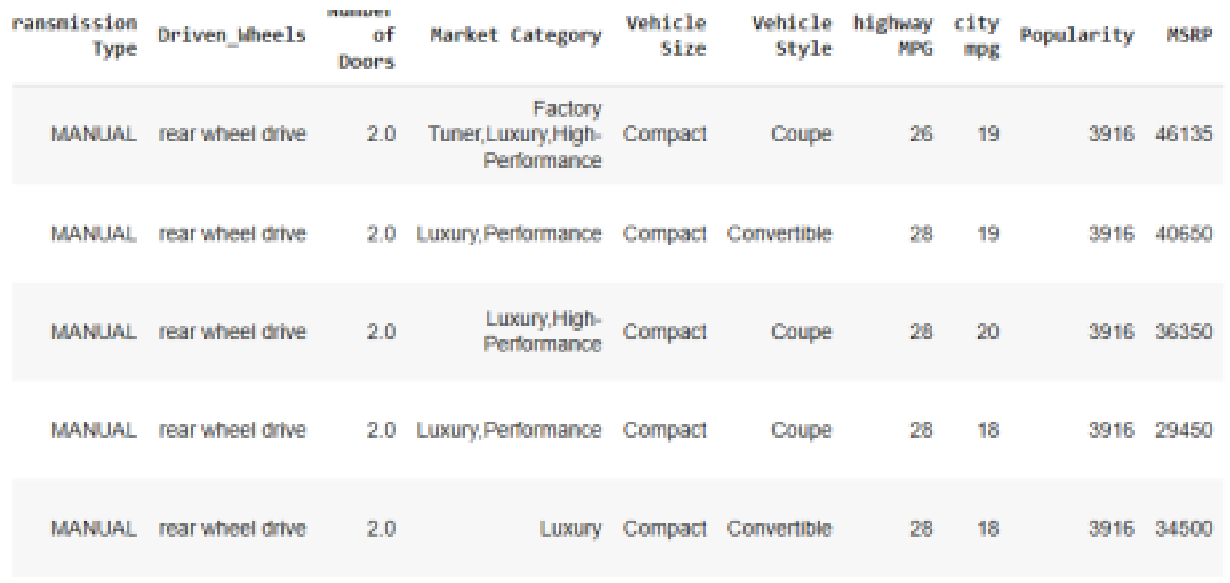


```
  Make       Model  Year             Engine Fuel Type  Engine HP  \
0  BMW  1 Series M  2011  premium unleaded (required)      335.0
1  BMW    1 Series  2011  premium unleaded (required)      300.0
2  BMW    1 Series  2011  premium unleaded (required)      300.0
3  BMW    1 Series  2011  premium unleaded (required)      230.0
4  BMW    1 Series  2011  premium unleaded (required)      230.0

  Engine Cylinders Transmission Type     Driven_Wheels  Number of Doors  \
0              6.0            MANUAL  rear wheel drive              2.0
1              6.0            MANUAL  rear wheel drive              2.0
2              6.0            MANUAL  rear wheel drive              2.0
3              6.0            MANUAL  rear wheel drive              2.0
4              6.0            MANUAL  rear wheel drive              2.0

                        Market Category Vehicle Size Vehicle Style  \
0  Factory Tuner,Luxury,High-Performance      Compact         Coupe
1                     Luxury,Performance      Compact   Convertible
2                Luxury,High-Performance      Compact         Coupe
3                     Luxury,Performance      Compact         Coupe
4                                 Luxury      Compact   Convertible

  highway MPG  city mpg  Popularity   MSRP
0          26        19        3916  46135
1          28        19        3916  40650
2          28        20        3916  36350
3          28        18        3916  29450
4          28        18        3916  34500
```

```
import sqlite3
conn = sqlite3.connect(':memory:')
csv_data.to_sql('data_table', conn, index=False,
if_exists='replace') query = "SELECT * FROM data_table LIMIT 5;"
result = pd.read_sql_query(query, conn)
```

```
result
```

OUTPUT:



ransmission Type	Driven_Wheels	of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popularity	MSRP
MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	26	19	3916	46135
MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	28	19	3916	40650
MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	28	20	3916	36350
MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	28	18	3916	29450
MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	28	18	3916	34500

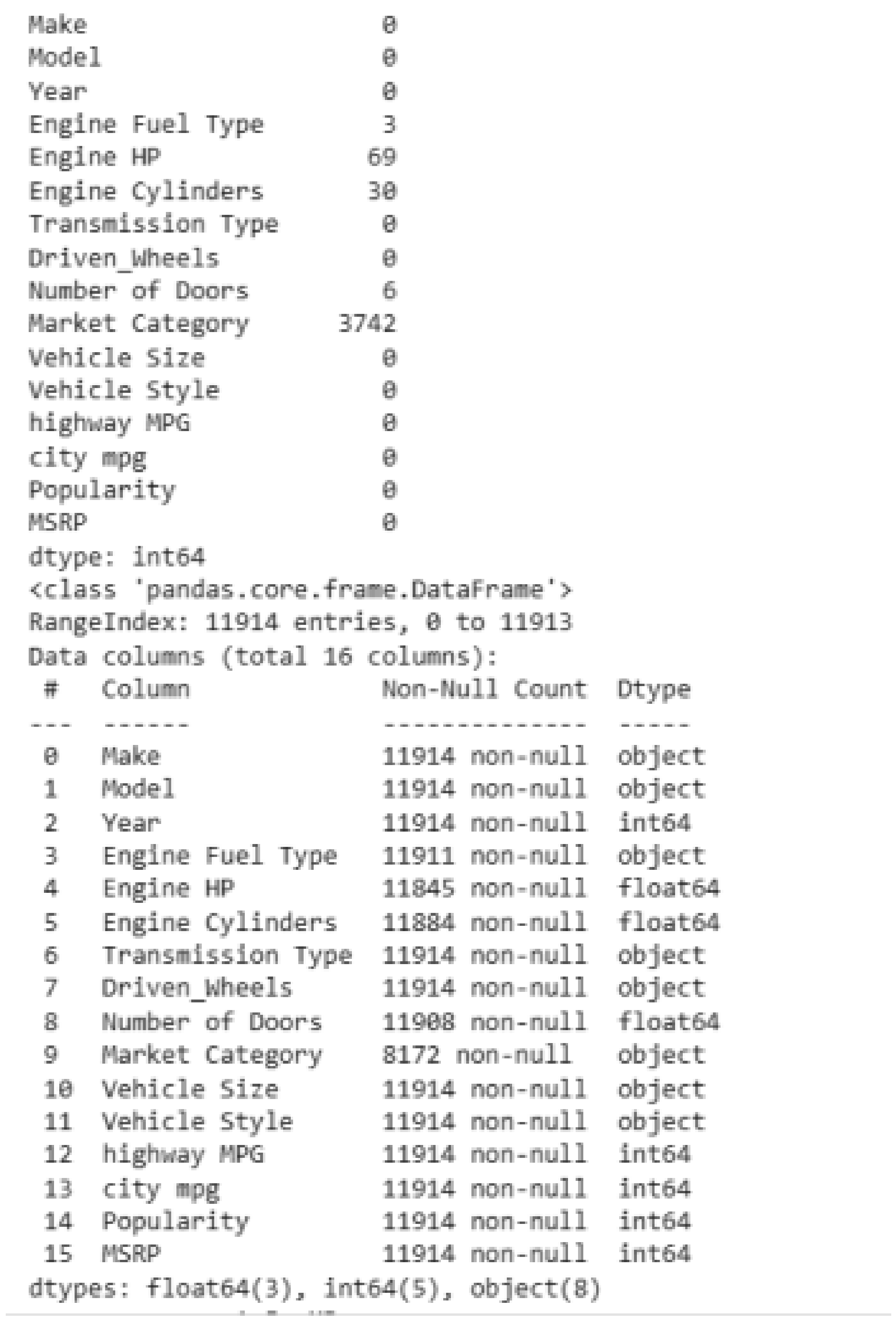
3.EDA-Data Cleaning

Handling missing values detection, filling, and dropping
Removing duplicates and unnecessary
data Data type conversion and ensuring consistency Normalize data (e.g., standardization, min-max scaling).

PROGRAM:

```
import pandas as pd
df = pd.read_csv('/content/data - data.csv')
print(df.isnull().sum())
print(df.info())
```

OUTPUT:



```
Make                   0
Model                  0
Year                   0
Engine Fuel Type       3
Engine HP             69
Engine Cylinders      30
Transmission Type      0
Driven_Wheels          0
Number of Doors        6
Market Category     3742
Vehicle Size           0
Vehicle Style          0
highway MPG            0
city mpg               0
Popularity             0
MSRP                   0
dtype: int64
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Make               11914 non-null  object
 1   Model              11914 non-null  object
 2   Year               11914 non-null  int64
 3   Engine Fuel Type   11911 non-null  object
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object
 7   Driven_Wheels      11914 non-null  object
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object
 10  Vehicle Size       11914 non-null  object
 11  Vehicle Style      11914 non-null  object
 12  highway MPG        11914 non-null  int64
 13  city mpg           11914 non-null  int64
 14  Popularity         11914 non-null  int64
 15  MSRP               11914 non-null  int64
dtypes: float64(3), int64(5), object(8)
```

```
df.columns = df.columns.str.strip()
print(df.columns.tolist())
```

OUTPUT:

```
['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP', 'Engine
Cylinders', 'Transmission Type', 'Driven_Wheels', 'Number of Doors',
'Market Category', 'Vehicle Size', 'Vehicle Style', 'highway MPG', 'city
mpg', 'Popularity', 'MSRP']
```

```
df['Engine HP'] = df['Engine HP'].fillna(df['Engine HP'].mean())
df['Engine Cylinders'] = df['Engine Cylinders'].fillna(df['Engine
Cylinders'].mean())
df['Number of Doors'] = df['Number of Doors'].fillna(df['Number of
Doors'].mean())
df['Engine Fuel Type'] = df['Engine Fuel Type'].fillna(df['Engine Fuel
Type'].mode()[0])
df['Market Category'] = df['Market Category'].fillna(df['Market
Category'].mode()[0])

df.dropna(inplace=True)


print(df.isnull().sum())
```

OUTPUT:

```
 Make 0
Model 0
Year 0
Engine Fuel Type 0
Engine HP 0
Engine Cylinders 0
Transmission Type 0
Driven_Wheels 0
Number of Doors 0
Market Category 0
Vehicle Size 0
Vehicle Style 0
highway MPG 0
city mpg 0
Popularity 0
MSRP 0
dtype: int64
df.drop_duplicates(inplace=True)
```

```
columns_to_numeric = ['Engine HP', 'Engine Cylinders', 'Number of Doors',
 'highway MPG', 'city mpg', 'MSRP']
df[columns_to_numeric] = df[columns_to_numeric].apply(pd.to_numeric,
errors='coerce')
df['Engine Fuel Type'] = df['Engine Fuel Type'].str.lower().str.strip()
df['Transmission Type'] = df['Transmission
Type'].str.upper().str.strip() df['Driven_Wheels'] =
df['Driven_Wheels'].str.lower().str.strip() print(df.dtypes)
print(df[['Engine Fuel Type', 'Transmission Type',
 'Driven_Wheels']].head())
```

OUTPUT:

```
Make object
Model object
Year int64
Engine Fuel Type object
Engine HP float64
Engine Cylinders float64
Transmission Type object
Driven_Wheels object
Number of Doors float64
Market Category object
Vehicle Size object
Vehicle Style object
highway MPG int64
city mpg int64
Popularity int64
MSRP int64
dtype: object
  Engine  Fuel  Type  Transmission  Type  Driven_Wheels  0  premium
unleaded  (required)  MANUAL  rear  wheel  drive  1  premium  unleaded
(required) MANUAL rear wheel drive 2 premium unleaded (required)
MANUAL rear wheel drive 3 premium unleaded (required) MANUAL rear
wheel drive 4 premium unleaded (required) MANUAL rear wheel drive
```

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler
numeric_cols = ['Engine HP', 'Engine Cylinders', 'Number of Doors',
'highway MPG', 'city mpg', 'MSRP']
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric,
errors='coerce') df_clean = df.dropna(subset=numeric_cols)
min_max_scaler = MinMaxScaler()
df_clean[numeric_cols] =
min_max_scaler.fit_transform(df_clean[numeric_cols])
print(df_clean[numeric_cols].head())
```

OUTPUT:

```
Engine HP Engine Cylinders Number of Doors highway MPG city mpg \ 0
0.295983 0.375 0.0 0.040936 0.092308  1 0.258985 0.375 0.0 0.046784
0.092308  2 0.258985 0.375 0.0 0.046784 0.100000  3 0.184989 0.375 0.0
0.046784 0.084615  4 0.184989 0.375 0.0 0.046784 0.084615

 MSRP
0 0.021384
1 0.018727
2 0.016643
3 0.013300
4 0.015747
```

4.EDA-Data Inspection and Analysis

Viewing and inspecting DataFrames
Filtering and subsetting data using conditions
Descriptive statistics: measures of central tendency (mean, median, mode) and measures of dispersion (range, variance, standard deviation)

PROGRAM:

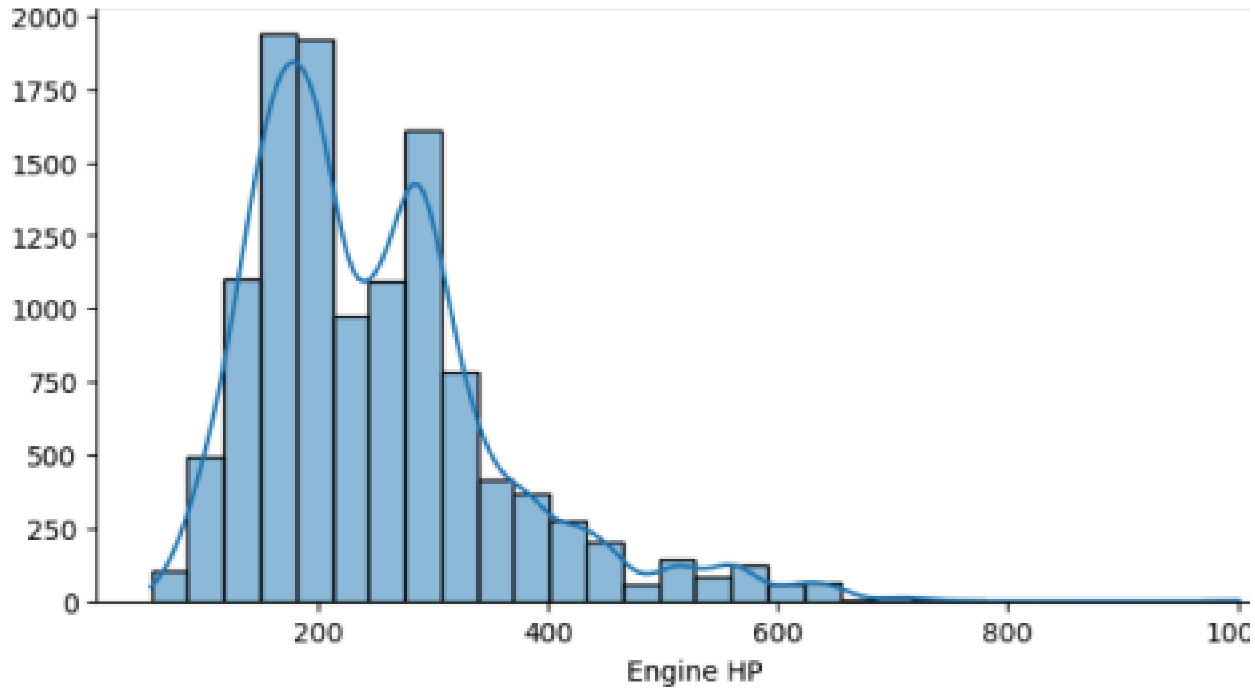
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("data - data.csv")

print("First 5 rows:")
print(df.head())

print("\nShape of dataset:", df.shape)

print("\nColumn names:")
print(df.columns)

print("\nInfo about data types and null values:")
print(df.info())

print("\nSummary statistics:")
print(df.describe())

print("\nRows where MSRP > 50000:")
print(df[df['MSRP'] > 50000].head())

print("\nSelect columns: Engine HP and MSRP")
print(df[['Engine HP', 'MSRP']].head())
print("\nDescriptive statistics for 'Engine HP':")
print(f"Mean: {df['Engine HP'].mean():.2f}")
print(f"Median: {df['Engine HP'].median():.2f}")
print(f"Mode: {df['Engine HP'].mode().values}")

print("\nDescriptive statistics for 'MSRP':")
print(f"Range: {df['MSRP'].max() -
df['MSRP'].min()}") print(f"Variance:
{df['MSRP'].var():.2f}")
print(f"Standard Deviation: {df['MSRP'].std():.2f}")


plt.figure(figsize=(8,4))
sns.histplot(df['Engine HP'].dropna(), kde=True,
bins=30) plt.title('Engine HP Distribution')
plt.xlabel('Engine HP')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(8,4))
sns.boxplot(x=df['MSRP'])
plt.title('Boxplot of MSRP')
plt.xlabel('MSRP')
plt.show()


plt.figure(figsize=(12,8))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm',
fmt=".2f")
plt.title('Correlation Heatmap - Car Dataset')
plt.show()
```

OUTPUT:

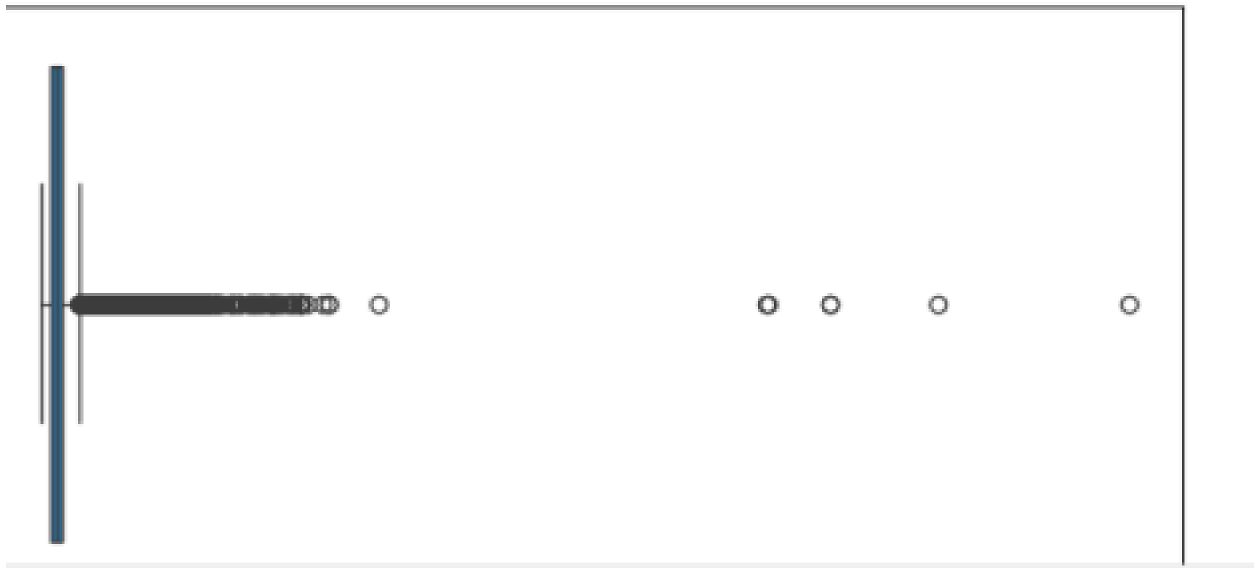
```
First 5 rows:
 Make Model Year Engine Fuel Type Engine HP \ 0 BMW 1 Series M 2011
premium unleaded (required) 335.0  1 BMW 1 Series 2011 premium
unleaded (required) 300.0  2 BMW 1 Series 2011 premium unleaded
(required) 300.0  3 BMW 1 Series 2011 premium unleaded (required)
230.0  4 BMW 1 Series 2011 premium unleaded (required) 230.0

 Engine Cylinders Transmission Type Driven_Wheels Number of Doors  \
0 6.0 MANUAL rear wheel drive 2.0  1 6.0 MANUAL rear wheel drive 2.0  2
6.0 MANUAL rear wheel drive 2.0  3 6.0 MANUAL rear wheel drive 2.0  4
6.0 MANUAL rear wheel drive 2.0

 Market Category Vehicle Size Vehicle Style \ 0 Factory
Tuner,Luxury,High-Performance Compact Coupe  1 Luxury,Performance
Compact Convertible  2 Luxury,High-Performance Compact Coupe  3
Luxury,Performance Compact Coupe  4 Luxury Compact Convertible

 highway MPG city mpg Popularity MSRP
0 26 19 3916 46135
1 28 19 3916 40650
2 28 20 3916 36350
```

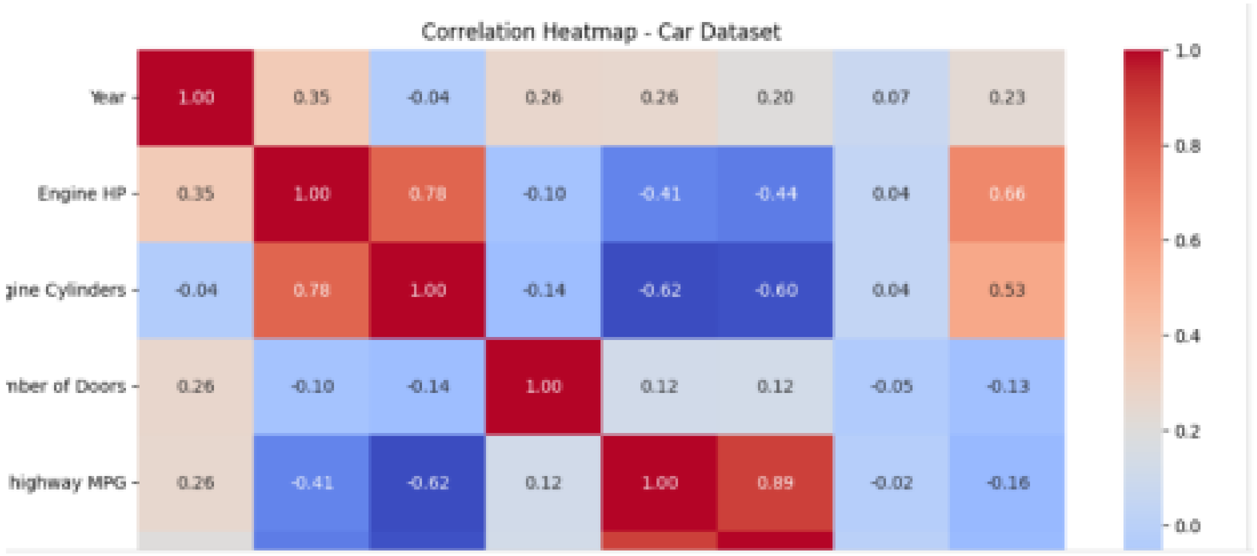
```
3 28 18 3916 29450
4 28 18 3916 34500

Shape of dataset: (11914, 16)

Column names:
Index(['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP',
'Engine Cylinders', 'Transmission Type', 'Driven_Wheels',  'Number of
Doors', 'Market Category', 'Vehicle Size', 'Vehicle  Style',
 'highway MPG', 'city mpg', 'Popularity', 'MSRP'],
dtype='object')

Info about data types and null values:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 # Column Non-Null Count Dtype
--- ------ -------------- -----
 0 Make 11914 non-null object
 1 Model 11914 non-null object
 2 Year 11914 non-null int64
 3 Engine Fuel Type 11911 non-null object
 4 Engine HP 11845 non-null float64
 5 Engine Cylinders 11884 non-null float64
 6 Transmission Type 11914 non-null object
 7 Driven_Wheels 11914 non-null object
 8 Number of Doors 11908 non-null float64
 9 Market Category 8172 non-null object
 10 Vehicle Size 11914 non-null object
 11 Vehicle Style 11914 non-null object
 12 highway MPG 11914 non-null int64
 13 city mpg 11914 non-null int64
 14 Popularity 11914 non-null int64
 15 MSRP 11914 non-null int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
None

Summary statistics:
 Year Engine HP Engine Cylinders Number of Doors \ count 11914.000000
11845.00000 11884.000000 11908.000000  mean 2010.384338 249.38607
5.628829 3.436093  std 7.579740 109.19187 1.780559 0.881315  min
1990.000000 55.00000 0.000000 2.000000  25% 2007.000000 170.00000
4.000000 2.000000  50% 2015.000000 227.00000 6.000000 4.000000  75%
2016.000000 300.00000 6.000000 4.000000  max 2017.000000 1001.00000
16.000000 4.000000

  highway  MPG  city  mpg  Popularity  MSRP   count  11914.000000
11914.000000   11914.000000   1.191400e+04     mean   26.637485
19.733255  1554.911197  4.059474e+04    std  8.863001  8.987798
1441.855347  6.010910e+04    min  12.000000  7.000000  2.000000
2.000000e+03   25%  22.000000  16.000000  549.000000  2.100000e+04
```

```
50%   26.000000   18.000000   1385.000000   2.999500e+04     75%
30.000000 22.000000 2009.000000 4.223125e+04  max 354.000000
137.000000 5657.000000 2.065902e+06

Rows where MSRP > 50000:
 Make Model Year Engine Fuel Type Engine HP \ 49 BMW 2 Series 2016
premium unleaded (required) 320.0  52 BMW 2 Series 2017 premium unleaded
(recommended) 335.0  132 BMW 3 Series 2015 premium unleaded (required)
335.0
294 Ferrari 360 2002 premium unleaded (required) 400.0  295 Ferrari 360
2002 premium unleaded (required) 400.0

 Engine Cylinders Transmission Type Driven_Wheels Number of Doors  \
49  6.0  AUTOMATIC  rear  wheel  drive  2.0   52  6.0  AUTOMATIC  all  wheel  drive
2.0   132  6.0  AUTOMATIC  rear  wheel  drive  4.0   294  8.0  MANUAL  rear  wheel
drive 2.0  295 8.0 MANUAL rear wheel drive 2.0

 Market Category Vehicle Size Vehicle Style \ 49 Factory
Tuner,Luxury,High-Performance Compact Convertible  52 Factory
Tuner,Luxury,High-Performance Compact Convertible  132
Luxury,High-Performance,Hybrid Midsize Sedan  294
Exotic,High-Performance Compact Convertible  295
Exotic,High-Performance Compact Coupe

 highway MPG city mpg Popularity MSRP
49 30 20 3916 50750
52 32 21 3916 51050
132 33 25 3916 50150
294 15 10 2774 160829
295 15 10 2774 140615

Select columns: Engine HP and MSRP
 Engine HP MSRP
0 335.0 46135
1 300.0 40650
2 300.0 36350
3 230.0 29450
4 230.0 34500

Descriptive statistics for 'Engine HP':
Mean: 249.39
Median: 227.00
Mode: [200.]

Descriptive statistics for 'MSRP':
Range: 2063902
Variance: 3613104336.03
Standard Deviation: 60109.10
```



Boxplot of MSRP





EXP 5-EDA-DataVisualization with Matplotlib

Basicplotting: line charts, bar charts, histograms

import matplotlib.pyplot as plt

import numpy as np

# Sample Dataset (Student Scores)

students = ["Alice", "Bob", "Charlie", "David", "Eva"]

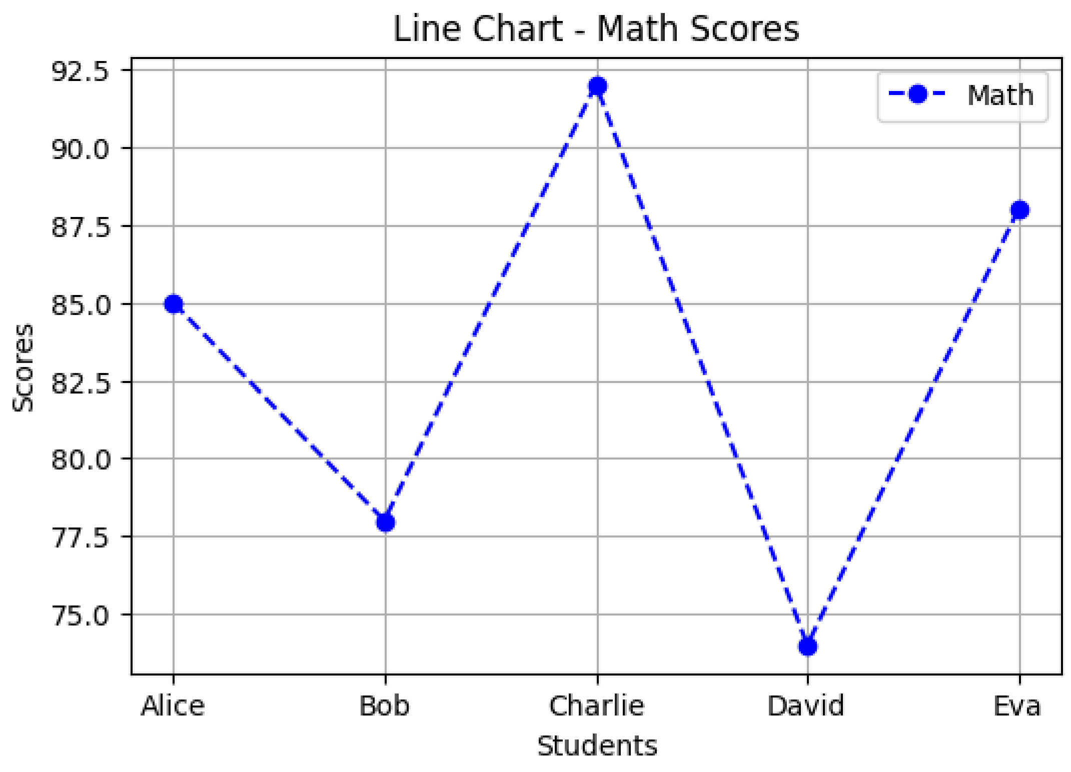
math_scores = [85, 78, 92, 74, 88]

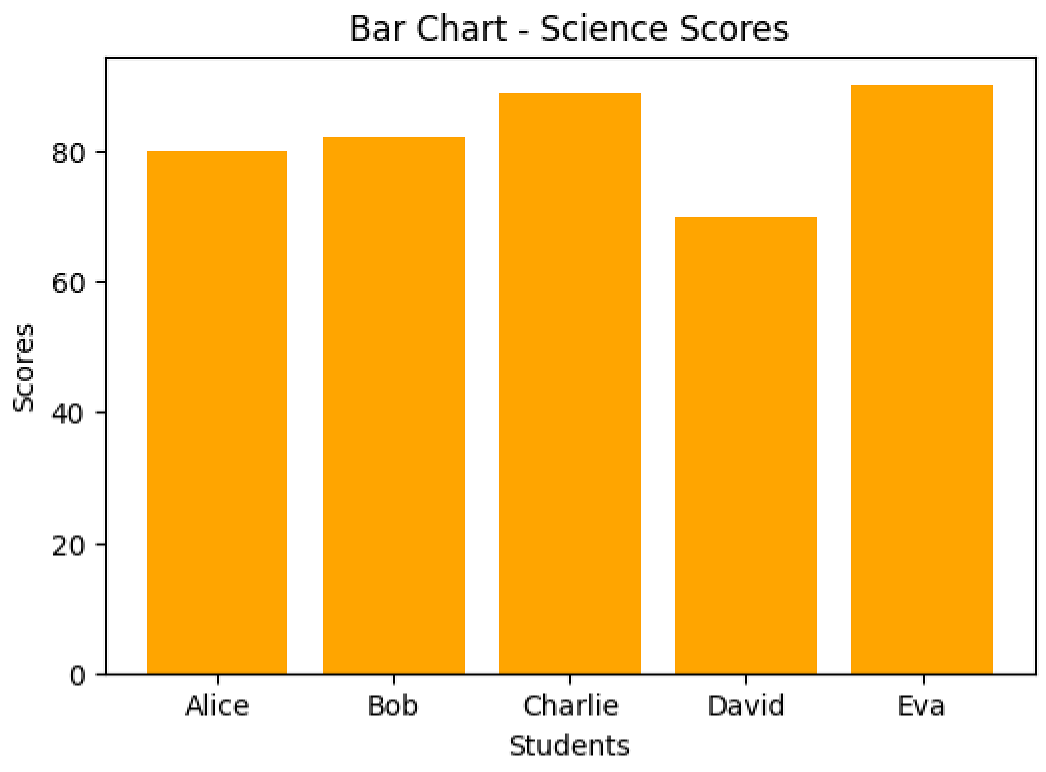
science_scores = [80, 82, 89, 70, 90]

english_scores = [78, 85, 84, 76, 86]

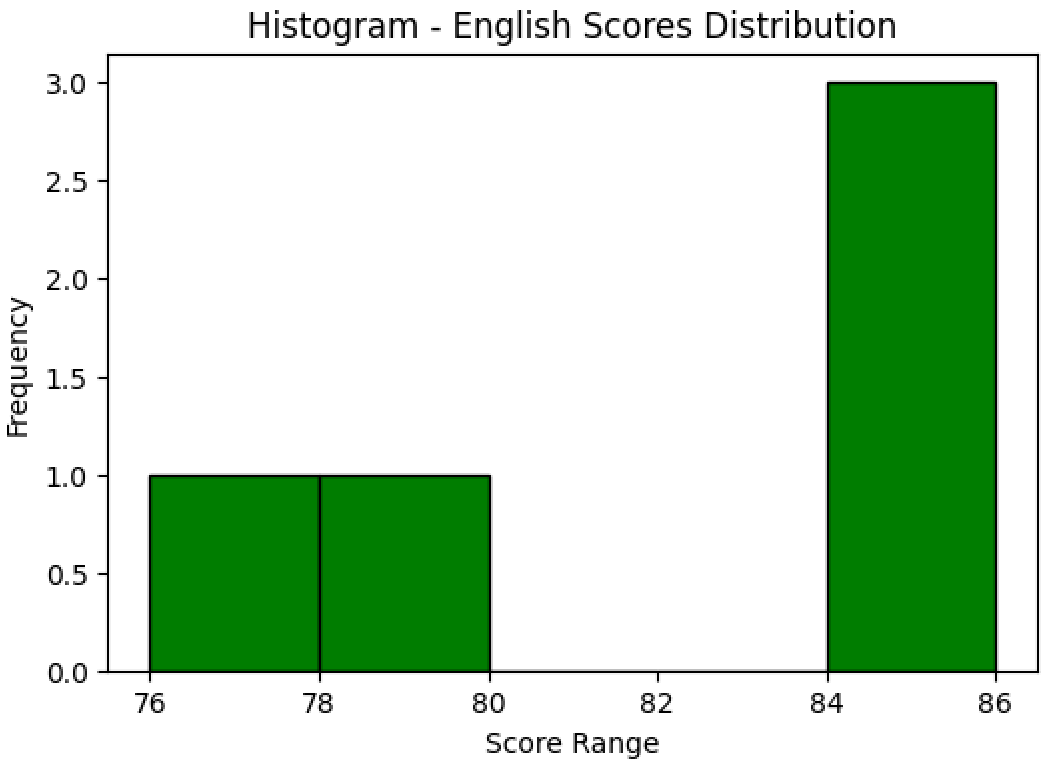
```
# 1. Line Chart (Trend of Math Scores)

plt.figure(figsize=(6,4))

plt.plot(students, math_scores, marker='o', linestyle='--', color='blue',

label="Math")

plt.title("Line Chart - Math Scores")

plt.xlabel("Students")

plt.ylabel("Scores")

plt.legend()

plt.grid(True)

plt.show()

# 2. Bar Chart (Comparison of Science Scores)

plt.figure(figsize=(6,4))

plt.bar(students, science_scores, color='orange')

plt.title("Bar Chart - Science Scores")
plt.xlabel("Students")

plt.ylabel("Scores")

plt.show()

# 3. Histogram (Distribution of English Scores)

plt.figure(figsize=(6,4))

plt.hist(english_scores, bins=5, color='green', edgecolor='black')

plt.title("Histogram - English Scores Distribution")

plt.xlabel("Score Range")

plt.ylabel("Frequency")

plt.show()
```

OUTPUT:


Line Chart - Math Scores
Math
Scores
92.5
90.0
87.5
85.0
82.5
80.0
77.5
75.0
Alice
Bob
Charlie
David
Eva
Students


Bar Chart - Science Scores
Scores
80
60
40
20
0
Alice
Bob
Charlie
David
Eva
Students

Students


Histogram - English Scores Distribution
Frequency
Score Range
0.0
0.5
1.0
1.5
2.0
2.5
3.0
76
78
80
82
84
86