

231501104

Mounesh Kumaran K R

EX NO: 1 Setting up the Python environment and libraries- Jupyter 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

```
import ipywidgets as widgets
from IPython.display import display

slider = widgets.IntSlider(description='Slider:', min=0, max=100,
value=25)

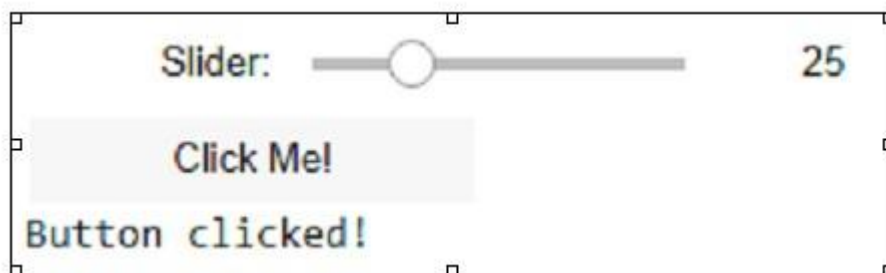
display(slider)

button = widgets.Button(description="Click Me!")

display(button)

def on_button_click(b):
    print("Button clicked!")

button.on_click(on_button_click)
```



Ex NO: 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.

```
import pandas as pd
```

```
df_csv = pd.read_csv('/content/data.csv')
```

```
df_csv.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market	Category	Vehicle Size	Vehicle Style	highwa MP
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High- Performance		Compact	Coupe	2
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Convertible	2
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High- Performance		Compact	Coupe	2
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Coupe	2
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury		Compact	Convertible	2

```
df_excel = pd.read_excel('/content/data.xlsx')
```

```
df_excel.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market	Category	Vehicle Size	Vehicle Style	highwa MP
0	BMW	Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Tuner,Luxury,High- Performance	Factory	Compact	Coupe	2
1	BMW	Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Convertible	2
2	BMW	Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High- Performance		Compact	Coupe	2
3	BMW	Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Coupe	2
4	BMW	Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0		Luxury	Compact	Convertible	2

```
import sqlite3
```

```
conn = sqlite3.connect(':memory:')
```

```
df.to_sql('data_table', conn, index=False, if_exists='replace')
```

```
[9] import sqlite3
     # Create an in-memory SQLite database
     conn = sqlite3.connect(':memory:')

     # Save DataFrame as SQL table
     df.to_sql('data_table', conn, index=False, if_exists='replace')
```

⇒ 11914

```
query = "SELECT * FROM data_table LIMIT 5;"
```

```
result = pd.read_sql_query(query, conn)
```

```
result
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market	Category	Vehicle Size	Vehicle Style	highway MP
0	BMW	Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance		Compact	Coupe	2
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Convertible	2
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance		Compact	Coupe	2
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance		Compact	Coupe	2
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury		Compact	Convertible	2

```
df.to_html('data.htm', index=False)
```

```
df_scraped = pd.read_html('data.htm')[0]
```

```
print(df_scraped.head())
```

	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	

EX NO: 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).**

```
import pandas as pd

df=pd.read_csv('/content/data.csv')

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

print(df.info())
```

```

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

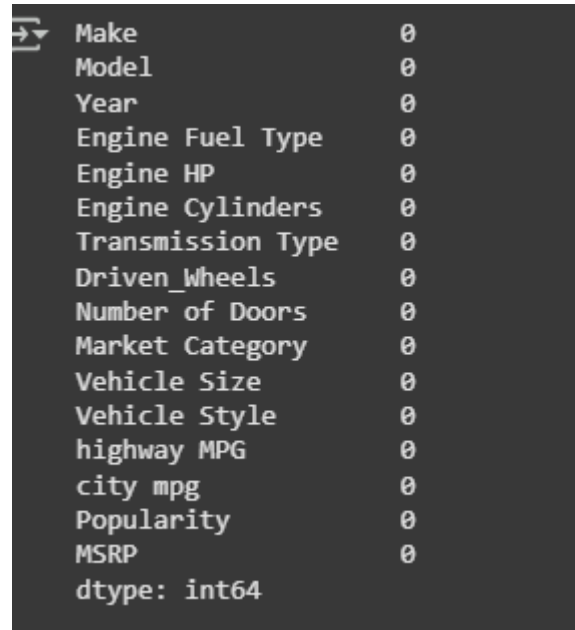
```



```

df.dropna(inplace=True)
print(df.isnull().sum())

```



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)
```

```
df['Make'] = df['Make'].str.title()
```

```
df['Model'] = df['Model'].str.title()
```

```
df['Transmission Type'] = df['Transmission Type'].astype('category')
```

```
df['Driven_Wheels'] = df['Driven_Wheels'].astype('category')
```

```
df['Vehicle Size'] = df['Vehicle Size'].astype('category')
```

```
df['Vehicle Style'] = df['Vehicle Style'].astype('category')
```

```
print(df.dtypes)
```

```
Make          object
Model         object
Year          int64
Engine Fuel Type  object
Engine HP      float64
Engine Cylinders float64
Transmission Type category
Driven_Wheels  category
Number of Doors float64
Market Category object
Vehicle Size   category
Vehicle Style  category
highway MPG    int64
city mpg       int64
Popularity     int64
MSRP           int64
dtype: object
```

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler

numeric_cols = ['Engine HP', 'Engine Cylinders', 'highway MPG', 'city mpg', 'Popularity',
'MSRP']

scaler = StandardScaler()

df[numeric_cols] = scaler.fit_transform(df[numeric_cols])

# Min-Max Scaling (optional alternative)

# minmax = MinMaxScaler()

# df[numeric_cols] = minmax.fit_transform(df[numeric_cols])

df.to_csv('cleaned_dataset.csv', index=False)
```

EX NO: 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)

```
import pandas as pd
```

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

```
print(df.head())
```

```
print("Rows:", df.shape[0], "Columns:", df.shape[1])
```

```
print(df.dtypes)
```

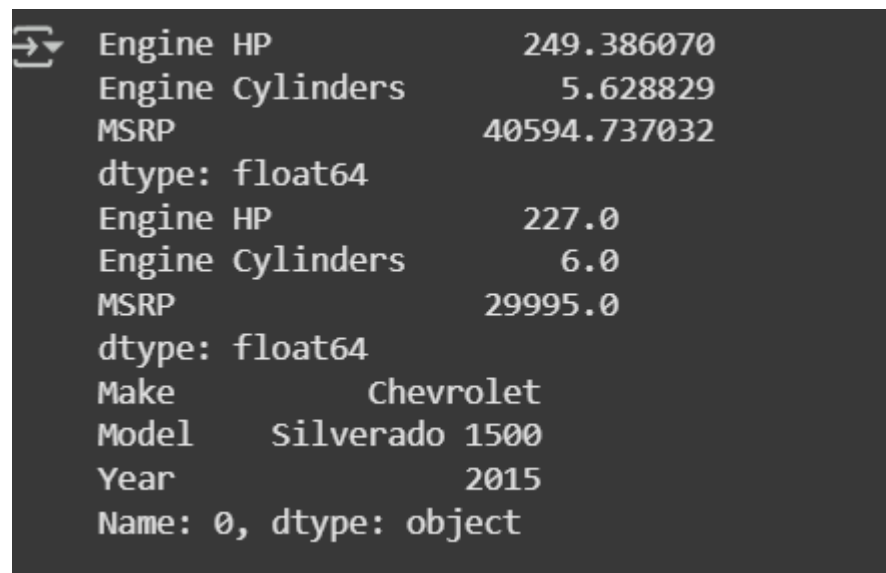
```
print(df.isnull().sum())
```

```
print(df.describe())
```

	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			

```
car_after_2015 = df[df['Year'] > 2015]
high_hp_cars = df[df['Engine HP'] > 300]
selected_columns = df[['Make', 'Model', 'MSRP']]
luxury_cars = df[df['Market Category'].str.contains('Luxury', na=False)]
```

```
print(df[['Engine HP', 'Engine Cylinders', 'MSRP']].mean())
print(df[['Engine HP', 'Engine Cylinders', 'MSRP']].median())
print(df[['Make', 'Model', 'Year']].mode().iloc[0])
```



```
Engine HP      249.386070
Engine Cylinders  5.628829
MSRP           40594.737032
dtype: float64
Engine HP      227.0
Engine Cylinders  6.0
MSRP           29995.0
dtype: float64
Make           Chevrolet
Model          Silverado 1500
Year           2015
Name: 0, dtype: object
```

```
range_values = df[['Engine HP', 'Engine Cylinders', 'MSRP']].max() - df[['Engine HP', 'Engine Cylinders', 'MSRP']].min()
print("Range:\n", range_values)
print("Variance:\n", df[['Engine HP', 'Engine Cylinders', 'MSRP']].var())
print("Standard Deviation:\n", df[['Engine HP', 'Engine Cylinders', 'MSRP']].std())
```

```
Range:
  Engine HP          946.0
  Engine Cylinders    16.0
  MSRP               2063902.0
  dtype: float64

Variance:
  Engine HP          1.192286e+04
  Engine Cylinders    3.170392e+00
  MSRP               3.613104e+09
  dtype: float64

Standard Deviation:
  Engine HP          109.191870
  Engine Cylinders    1.780559
  MSRP               60109.103604
  dtype: float64
```

```
import matplotlib.pyplot as plt
```

```
# Histogram of Engine HP
```

```
df['Engine HP'].dropna().hist(bins=20)
```

```
plt.title("Distribution of Engine HP")
```

```
plt.xlabel("Engine HP")
```

```
plt.ylabel("Frequency")
```

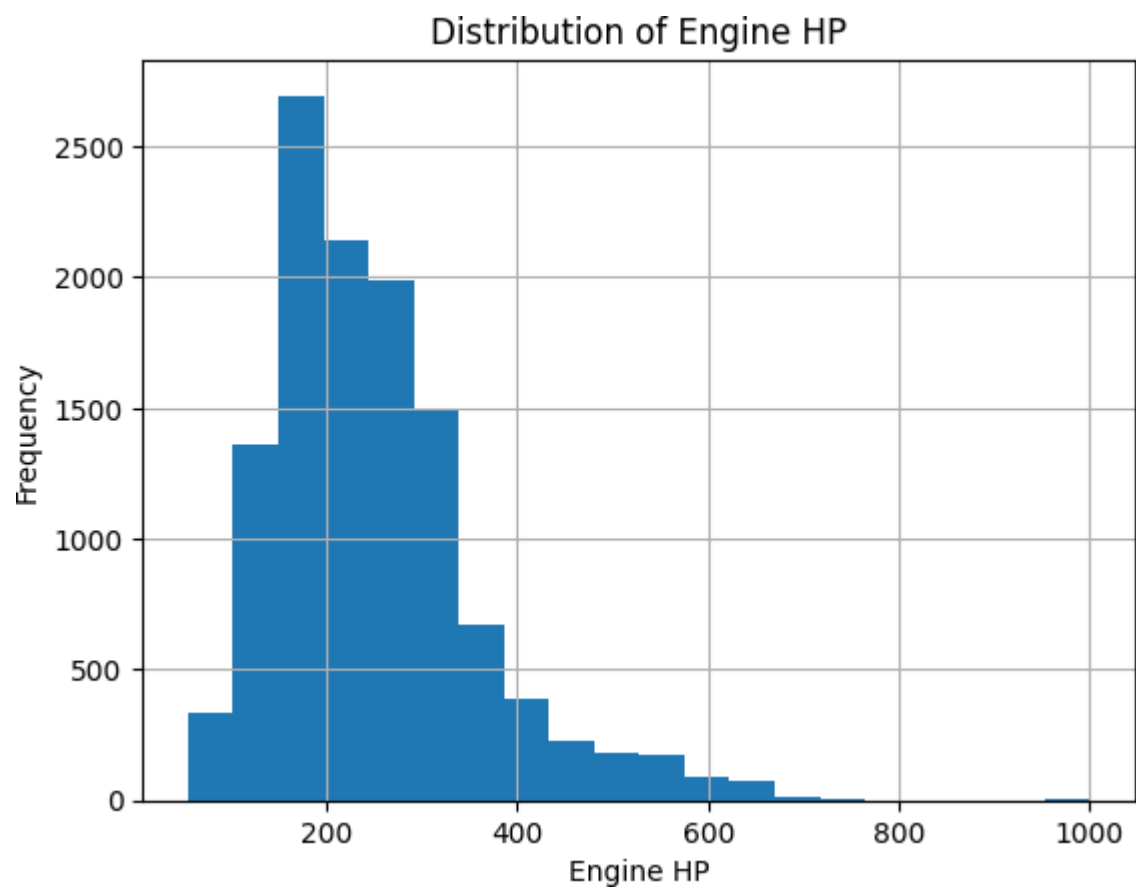
```
plt.show()
```

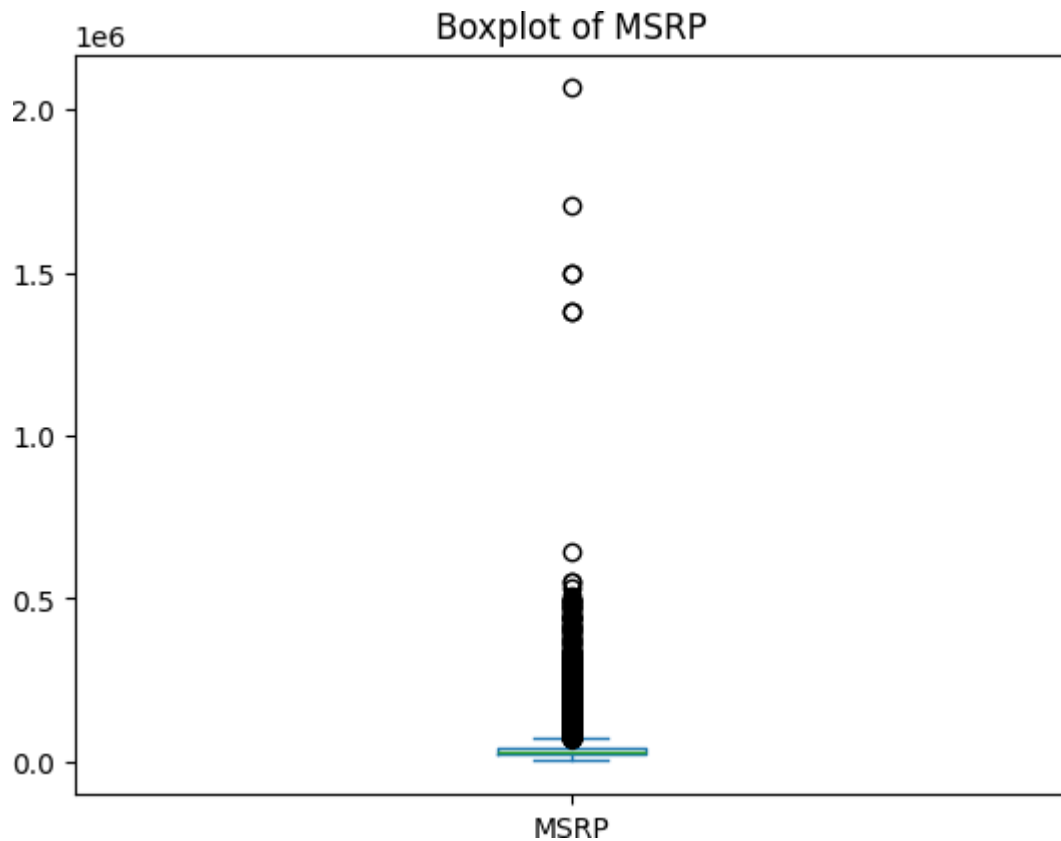
```
# Boxplot of MSRP
```

```
df['MSRP'].dropna().plot(kind='box')
```

```
plt.title("Boxplot of MSRP")
```

```
plt.show()
```





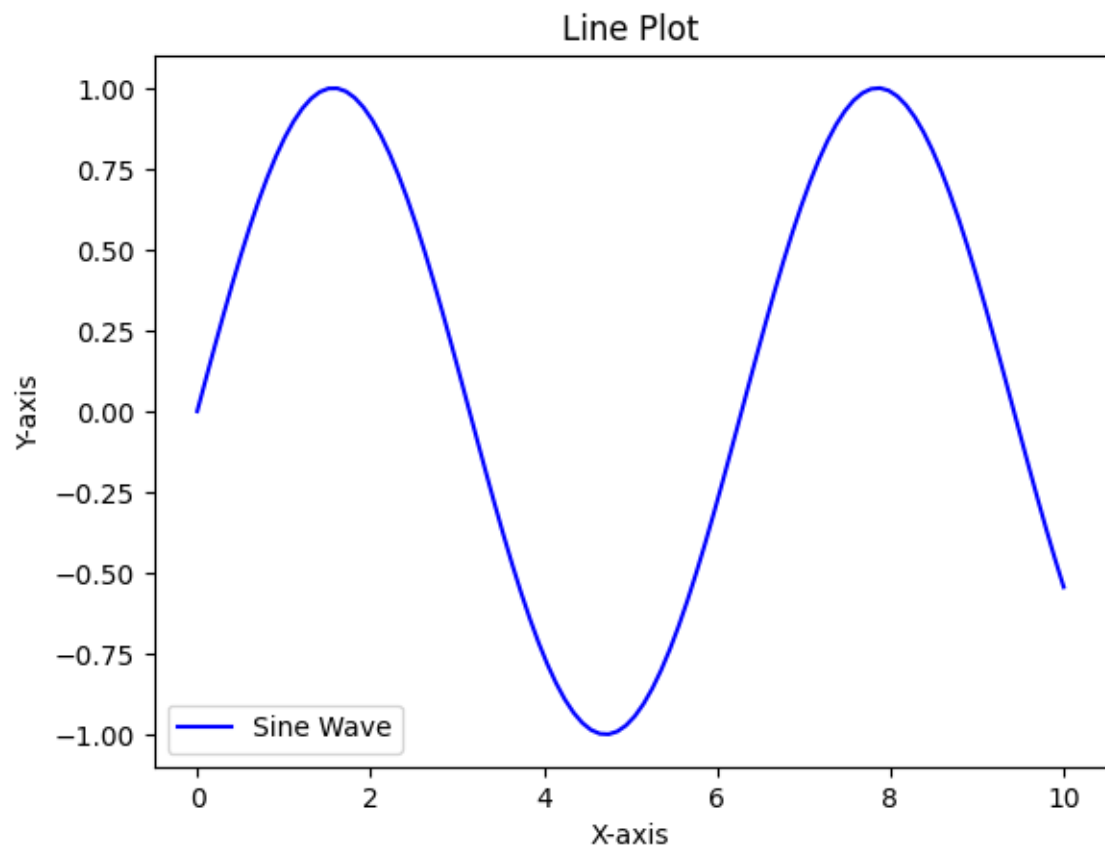
EX NO : 5 EDA – DATA VISUALIZATION

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

```
df = pd.read_csv("/content/data.csv") # Replace with your file
df.head()
```

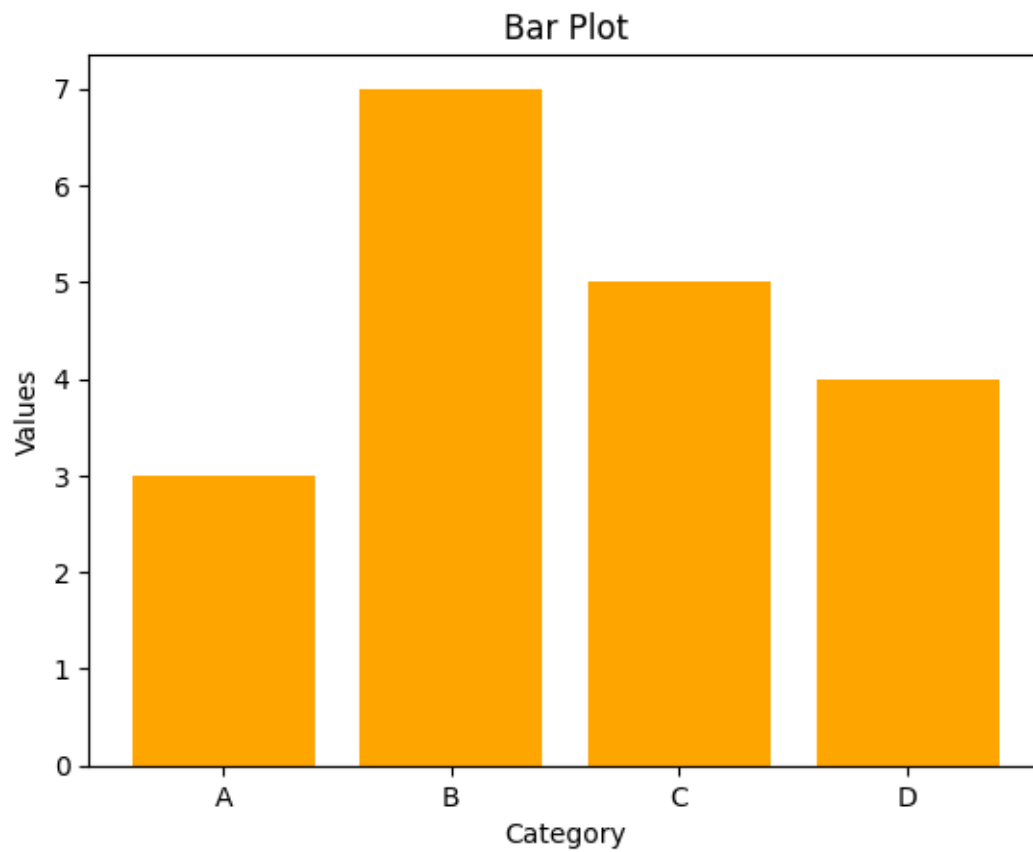
LINE CHART:

```
x = np.linspace(0, 10, 100)
y = np.sin(x)
plt.plot(x, y, color='blue', label='Sine Wave')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Line Plot')
plt.legend()
plt.show()
```



BAR CHART:

```
categories = ['A', 'B', 'C', 'D']
values = [3, 7, 5, 4]
plt.bar(categories, values, color='orange')
plt.xlabel('Category')
plt.ylabel('Values')
plt.title('Bar Plot')
plt.show()
```



HISTOGRAM:

```
data = np.random.randn(1000)
plt.hist(data, bins=20, color='purple', edgecolor='black')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram')
plt.show()
```

