1. Multilinear Regression on Data1.csv

Data1.csv:

```
plaintext
Copy code
Y,X1,X2,X3
20,10,30,40
50,20,40,50
60,30,50,60
80,40,60,70
```

Code

```
python
Copy code
import numpy as np
import pandas as pd

# Load the data
data = pd.read_csv("Data1.csv")

# Extract X and Y
X = data[['X1', 'X2', 'X3']].values
Y = data['Y'].values

# Add a column of ones to X for the intercept term
X = np.column_stack((np.ones(X.shape[0]), X))

# Calculate coefficients using the Normal Equation: B = (X'X)^(-1)X'Y
coefficients = np.linalg.inv(X.T @ X) @ X.T @ Y
print("Coefficients:", coefficients)

# Predict Y for the given values
new_data = np.array([[1, 50, 70, 80], [1, 30, 40, 50]])
predicted_Y = new_data @ coefficients
print("Predicted Y values:", predicted_Y)
```

---

2. Multiple Linear Regression with Preprocessing

Code

```
python
Copy code
# Data
data = {
    "Area": [2600, 3000, 3200, 3600, 4000],
    "Bedrooms": [3, 4, np.nan, 3, 5],
    "Age": [20, 15, 18, 30, 8],
    "Price": [550000, 565000, 610000, 595000, 760000]
}

df = pd.DataFrame(data)

# Preprocess data: Fill missing values with mean
df['Bedrooms'] = df['Bedrooms'].fillna(df['Bedrooms'].mean())

# Extract X and Y
X = df[['Area', 'Bedrooms', 'Age']].values
Y = df['Price'].values
```

```
# Add a column of ones to X for the intercept term
X = np.column_stack((np.ones(X.shape[0]), X))

# Calculate coefficients using the Normal Equation: B = (X'X)^(-1)X'Y
coefficients = np.linalg.inv(X.T @ X) @ X.T @ Y
print("Coefficients:", coefficients)

# Predict prices for new data
new_data = np.array([[1, 3000, 3, 40], [1, 2500, 4, 5]])
predicted_prices = new_data @ coefficients
print("Predicted Prices:", predicted_prices)
```

---

3. Code for Bike Sharing Multiple Linear Regression

```
python
Copy code
import numpy as np
import pandas as pd

# Load the dataset
data = pd.read_csv("bike_sharing.csv")  # Replace with your dataset filename

# Display the first few rows to understand the structure
print(data.head())

# Assuming the relevant columns are selected as features and target
# Replace 'target_column' with the actual target column name
features = ['feature1', 'feature2', 'feature3']  # Replace with your actual
feature names
target = 'target_column'  # Replace with your actual target column name

# Extract X (features) and Y (target)
X = data[features].values
Y = data[target].values

# Handle missing values (if any) by filling them with the mean of the column
X = np.nan_to_num(X, nan=np.nanmean(X, axis=0))

# Add a column of ones to X for the intercept term
X = np.column_stack((np.ones(X.shape[0]), X))

# Calculate coefficients using the Normal Equation: B = (X'X)^(-1)X'Y
coefficients = np.linalg.inv(X.T @ X) @ X.T @ Y
print("Coefficients:", coefficients)

# Predict Y values for a test dataset (replace with actual test data)
test_data = np.array([
    [1, value1, value2, value3],  # Replace with actual values
    [1, value4, value5, value6]   # Replace with actual values
])
predicted_Y = test_data @ coefficients
print("Predicted Y values:", predicted_Y)
```