```
import kagglehub

# Download latest version
path = kagglehub.dataset_download("yasserh/walmart-dataset")

print("Path to dataset files:", path)
```

```
Downloading from https://www.kaggle.com/api/v1/datasets/download/yasserh/walmart-data
100%|██████████| 122k/122k [00:00<00:00, 430kB/s]Extracting files...
Path to dataset files: /root/.cache/kagglehub/datasets/yasserh/walmart-dataset/versio
```

```
!pip install pandas numpy scikit-learn matplotlib seaborn
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (2.0.
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/di
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packa
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packag
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-package
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-pack
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (f
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error


from google.colab import files
uploaded = files.upload()

# Load the dataset
import io
df = pd.read_csv(io.BytesIO(uploaded['Walmart.csv']))
```

```
# Display the first few rows
print(df.head())
```

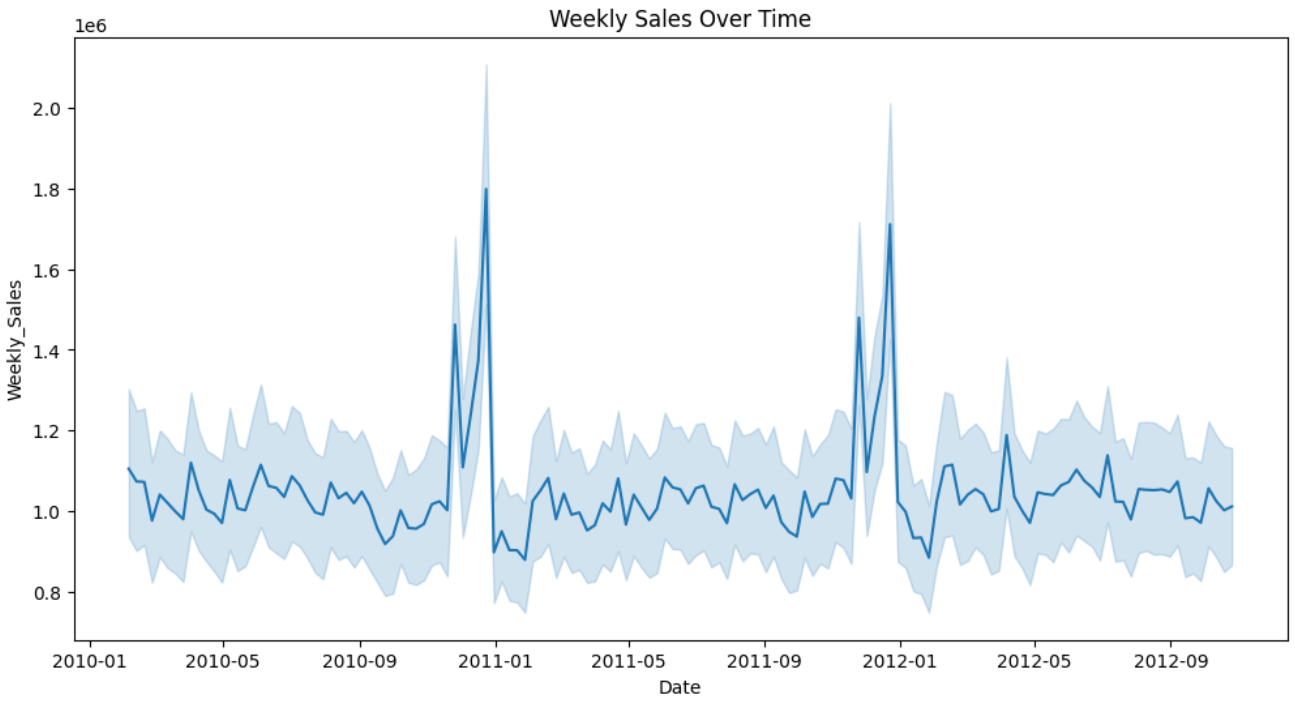
```
Browse... Walmart.csv
Walmart.csv(application/vnd.ms-excel) - 363732 bytes, last modified: n/a - 100% done
Saving Walmart.csv to Walmart.csv
   Store        Date  Weekly_Sales  Holiday_Flag  Temperature  Fuel_Price  \
0      1  05-02-2010    1643690.90             0        42.31       2.572
1      1  12-02-2010    1641957.44             1        38.51       2.548
2      1  19-02-2010    1611968.17             0        39.93       2.514
3      1  26-02-2010    1409727.59             0        46.63       2.561
4      1  05-03-2010    1554806.68             0        46.50       2.625

          CPI  Unemployment
0  211.096358         8.106
1  211.242170         8.106
2  211.289143         8.106
3  211.319643         8.106
4  211.350143         8.106
```

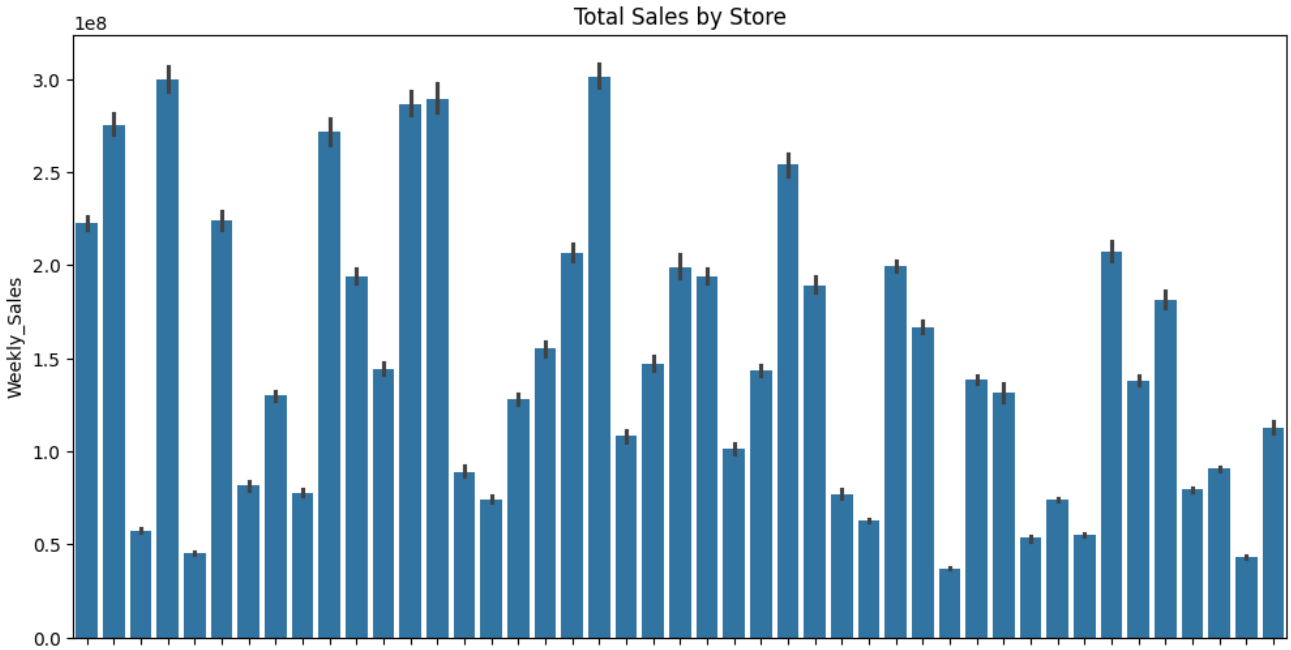
```
# Convert 'Date' to datetime with the correct format
df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y')

# Extract time-based features
df['year'] = df['Date'].dt.year
df['month'] = df['Date'].dt.month
df['day'] = df['Date'].dt.day
df['week_of_year'] = df['Date'].dt.isocalendar().week

# Check for missing values
print(df.isnull().sum())
```

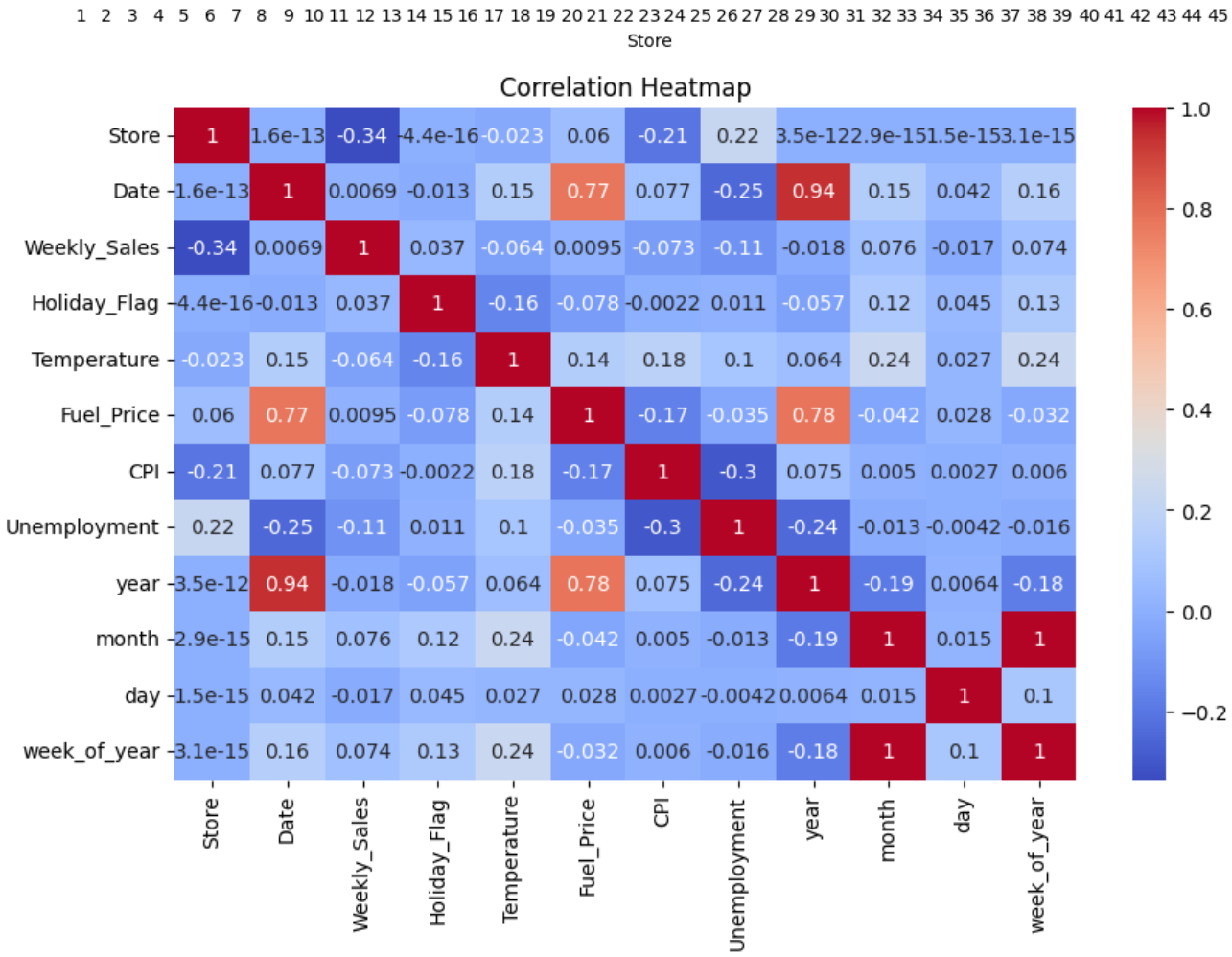
```
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
year            0
month           0
day             0
week_of_year    0
dtype: int64
```

```
# Plot weekly sales over time
plt.figure(figsize=(12, 6))
sns.lineplot(x='Date', y='Weekly_Sales', data=df)
```

```
plt.title('Weekly Sales Over Time')
plt.show()

# Plot total sales by store
plt.figure(figsize=(12, 6))
sns.barplot(x='Store', y='Weekly_Sales', data=df, estimator=np.sum)
plt.title('Total Sales by Store')
plt.show()

# Correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



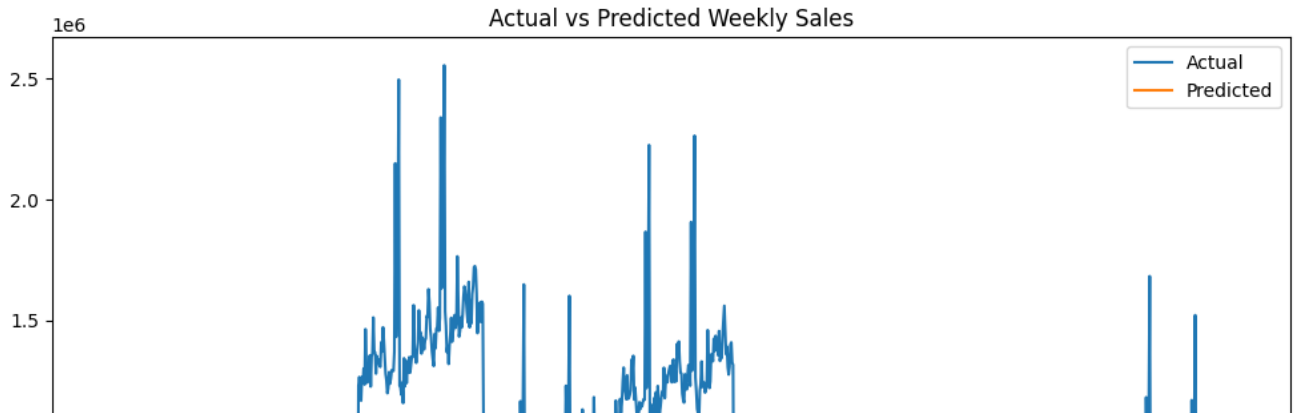


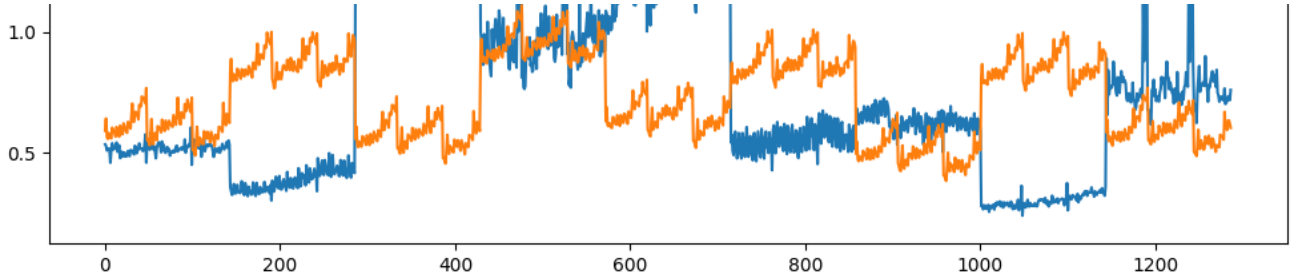

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
Store
Correlation Heatmap
Store 1 1.6e-13 -0.34 -4.4e-16 -0.023 0.06 -0.21 0.22 3.5e-12 2.9e-15 1.5e-15 3.1e-15
Date 1.6e-13 1 0.0069 -0.013 0.15 0.77 0.077 -0.25 0.94 0.15 0.042 0.16
Weekly_Sales -0.34 0.0069 1 0.037 -0.064 0.0095 -0.073 -0.11 -0.018 0.076 -0.017 0.074
Holiday_Flag -4.4e-16 -0.013 0.037 1 -0.16 -0.078 -0.0022 0.011 -0.057 0.12 0.045 0.13
Temperature -0.023 0.15 -0.064 -0.16 1 0.14 0.18 0.1 0.064 0.24 0.027 0.24
Fuel_Price 0.06 0.77 0.0095 -0.078 0.14 1 -0.17 -0.035 0.78 -0.042 0.028 -0.032
CPI -0.21 0.077 -0.073 -0.0022 0.18 -0.17 1 -0.3 0.075 0.005 0.0027 0.006
Unemployment 0.22 -0.25 -0.11 0.011 0.1 -0.035 -0.3 1 -0.24 -0.013 -0.0042 -0.016
year 3.5e-12 0.94 -0.018 -0.057 0.064 0.78 0.075 -0.24 1 -0.19 0.0064 -0.18
month 2.9e-15 0.15 0.076 0.12 0.24 -0.042 0.005 -0.013 -0.19 1 0.015 1
day 1.5e-15 0.042 -0.017 0.045 0.027 0.028 0.0027 -0.0042 0.0064 0.015 1 0.1
week_of_year 3.1e-15 0.16 0.074 0.13 0.24 -0.032 0.006 -0.016 -0.18 1 0.1 1
Store
Date
Weekly_Sales
Holiday_Flag
Temperature
Fuel_Price
CPI
Unemployment
year
month
day
week_of_year
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

```
# Define features and target
X = df[['Store', 'Holiday_Flag', 'Temperature', 'Fuel_Price', 'CPI', 'Unemployment', 'yea
y = df['Weekly_Sales']

# Split data into training and testing sets
train_size = int(len(df) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]


# Train a Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print(f"MAE: {mae}")
print(f"RMSE: {rmse}")
```

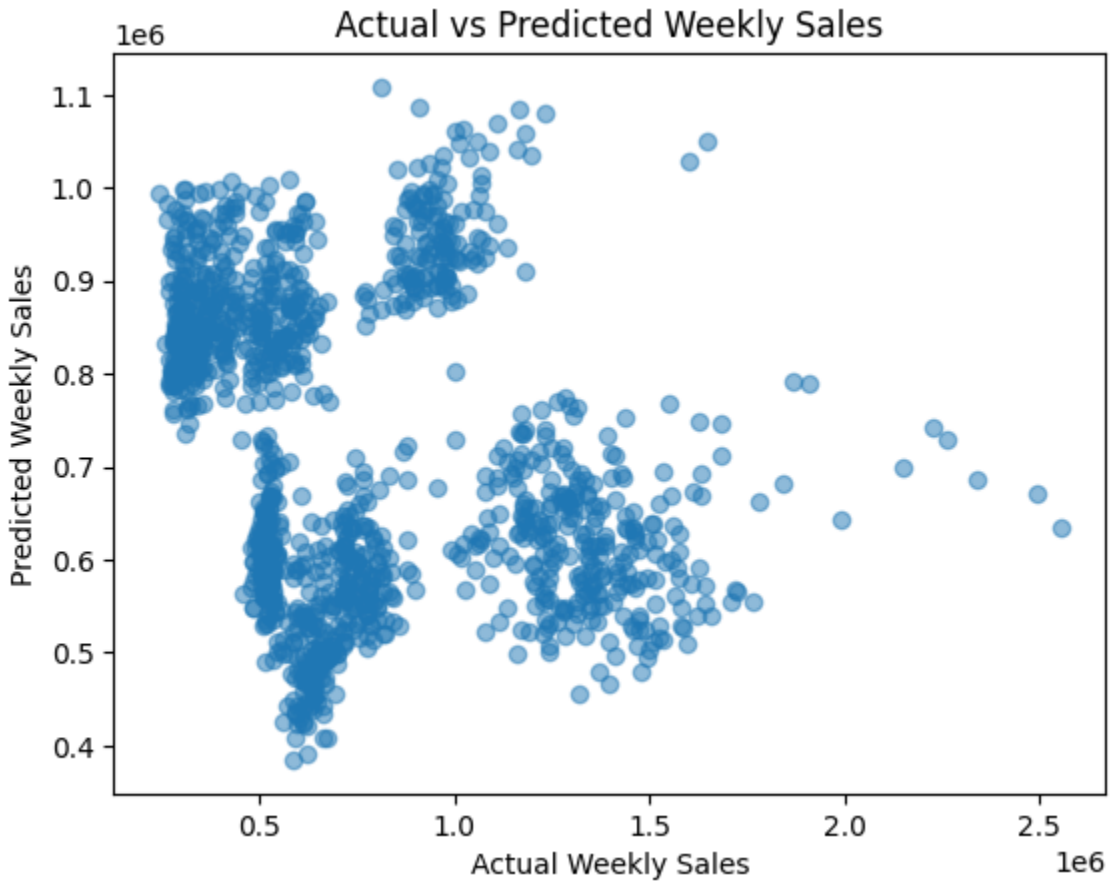
```
MAE: 370142.18418043246
RMSE: 466362.68879954953
```

```
# Plot actual vs predicted sales
plt.figure(figsize=(12, 6))
plt.plot(y_test.values, label='Actual')
plt.plot(y_pred, label='Predicted')
plt.legend()
plt.title('Actual vs Predicted Weekly Sales')
plt.show()
```



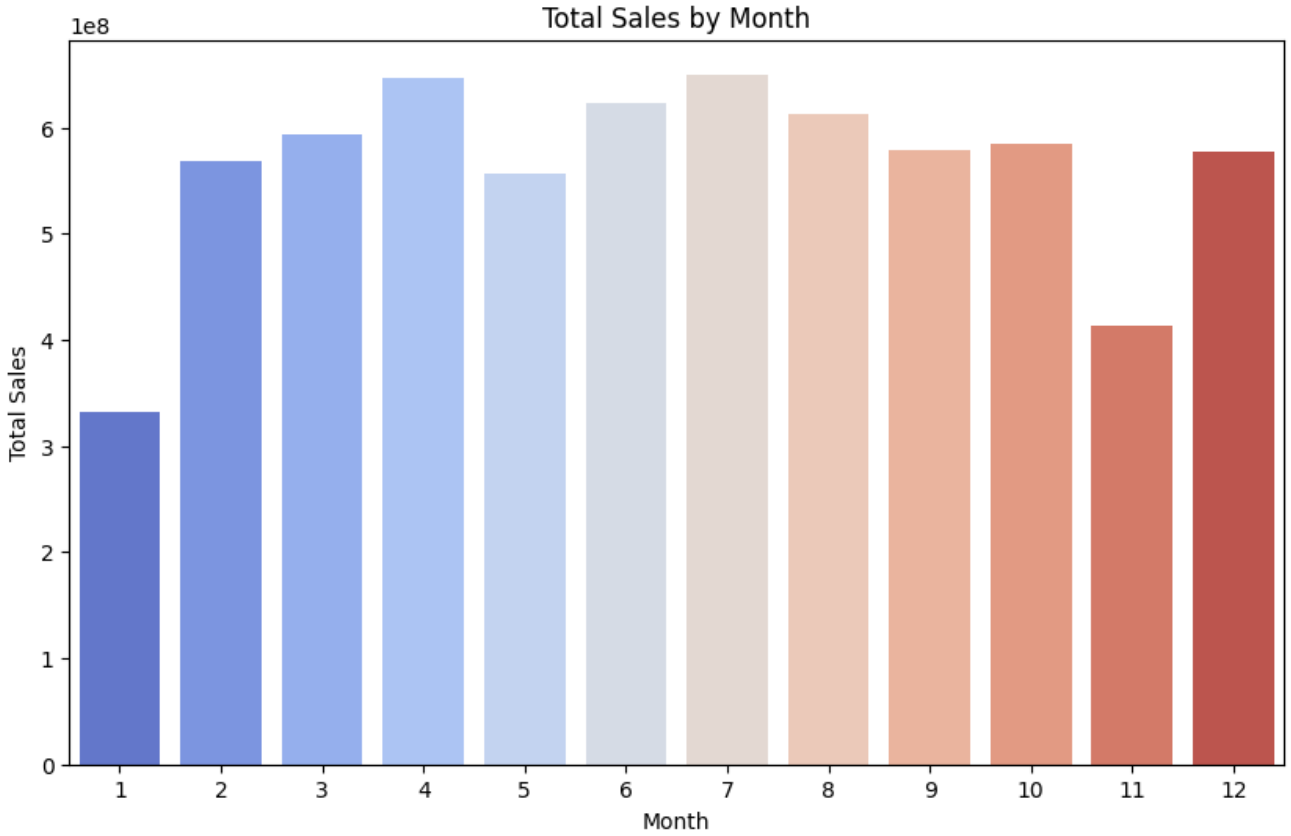



```
# Plot actual vs predicted sales
plt.scatter(y_test, y_pred, alpha=0.5)
plt.xlabel('Actual Weekly Sales')
plt.ylabel('Predicted Weekly Sales')
plt.title('Actual vs Predicted Weekly Sales')
plt.show()
```



```
# Aggregate sales by month
monthly_sales = df.groupby('month')['Weekly_Sales'].sum().reset_index()

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='month', y='Weekly_Sales', data=monthly_sales, palette='coolwarm')
plt.title('Total Sales by Month')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.show()
```

```
<ipython-input-26-7d8a13445dad>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.barplot(x='month', y='Weekly_Sales', data=monthly_sales, palette='coolwarm')
```




```
plt.figure(figsize=(10, 6))
sns.lineplot(x='Date', y='Weekly_Sales', data=df, color='purple')
plt.title('Weekly Sales Trend Over Time')
```

```
plt.xlabel('Date')
plt.ylabel('Weekly Sales')
plt.legend()
plt.show()
```

```
<ipython-input-29-7b318f2c3bb7>:6: UserWarning: No artists with labels found to put i
  plt.legend()
```