1

In [1]:

import pandas as pd  
import numpy as np  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import r2\_score  
import matplotlib.pyplot as plt

---------------------------------------------------------------------------  
ModuleNotFoundError Traceback (most recent call last)  
Cell In[1], line 5  
 3 from sklearn.linear\_model import LinearRegression  
 4 from sklearn.metrics import r2\_score  
----> 5 import matplotlib.pyplot as plt  
  
ModuleNotFoundError: No module named 'matplotlib'

In [ ]:

data = {  
 'store': [1, 2, 3, 4, 5, 6, 7],  
 'sales': [368, 340, 665, 954, 331, 556, 376],  
 'ad\_cost': [1.7, 1.5, 2.8, 5, 1.3, 2.2, 1.3]  
}

In [ ]:

df = pd.DataFrame(data)  
print(df)

store sales ad\_cost  
0 1 368 1.7  
1 2 340 1.5  
2 3 665 2.8  
3 4 954 5.0  
4 5 331 1.3  
5 6 556 2.2  
6 7 376 1.3

In [ ]:

model = LinearRegression();  
x = df['ad\_cost'].values.reshape(-1, 1)  
y = df['sales']. values  
model.fit(x, y)

Out[ ]:

LinearRegression()

**In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.**  
**On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.**

  LinearRegression[?Documentation for LinearRegression](https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LinearRegression.html)iFitted

LinearRegression()

In [ ]:

predictions = model.predict(x)

In [ ]:

print(f"R2: {r2\_score(y, predictions):.2f}") # if the value is near to 1, then the model is perfectly fitted.

R2: 0.96

In [ ]:

print(f"y = {model.coef\_[0]:.2f}x + {model.intercept\_:.2f}")  
print(f"Coefficient: {model.coef\_[0]:.2f}")

y = 171.47x + 125.83  
Coefficient: 171.47

In [ ]:

%matplotlib inline  
plt.scatter(x, y, color='blue', marker='+', label='Actual Sales')  
plt.plot(x, predictions, color='red', label='Predicted Sales')  
plt.xlabel("Online Advertising Dollars (1000s)")  
plt.ylabel("Monthly E-Commerce Sales (1000s)")  
plt.legend()

Out[ ]:

<matplotlib.legend.Legend at 0x78b05a3347d0>

![No description has been provided for this image](data:image/png;base64;base64,)

In [ ]:

test\_ad\_cost = np.array([1.0, 2.6, 3.1, 4.9]).reshape(-1, 1)  
predictions = model.predict(test\_ad\_cost)  
  
for sale, adv\_cost in zip(predictions, test\_ad\_cost):  
 print(f"For advertising cost of ${adv\_cost[0]}k, we have {sale:.2f}k sales")

For advertising cost of $1.0k, we have 297.30k sales  
For advertising cost of $2.6k, we have 571.65k sales  
For advertising cost of $3.1k, we have 657.38k sales  
For advertising cost of $4.9k, we have 966.02k sales