LAB # 06

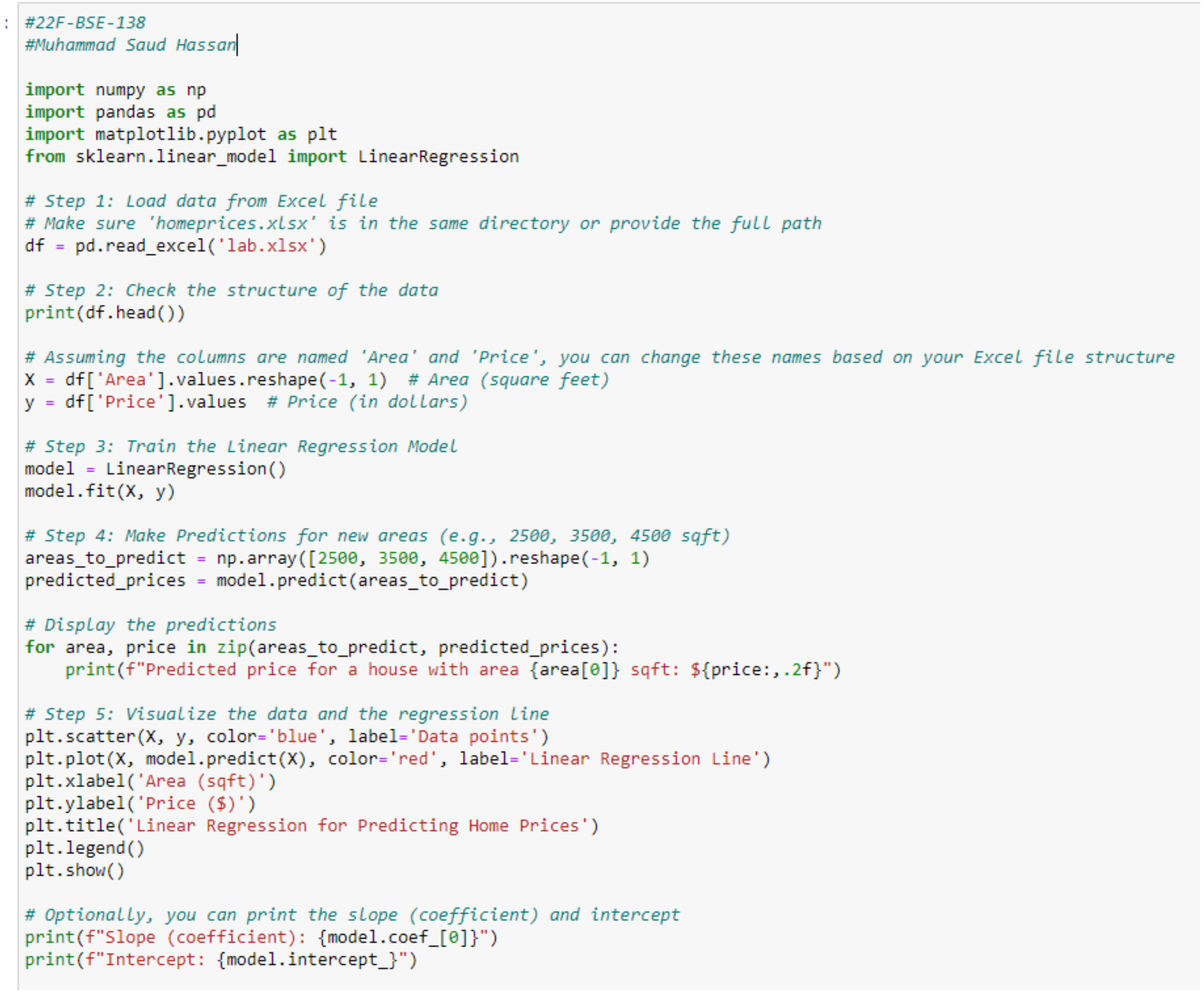
SUPERVISED LEARNING (LINEAR REGRESSION)

OBJECTIVE

Implementing supervised learning, linear regression algorithm for training, testing and classification.

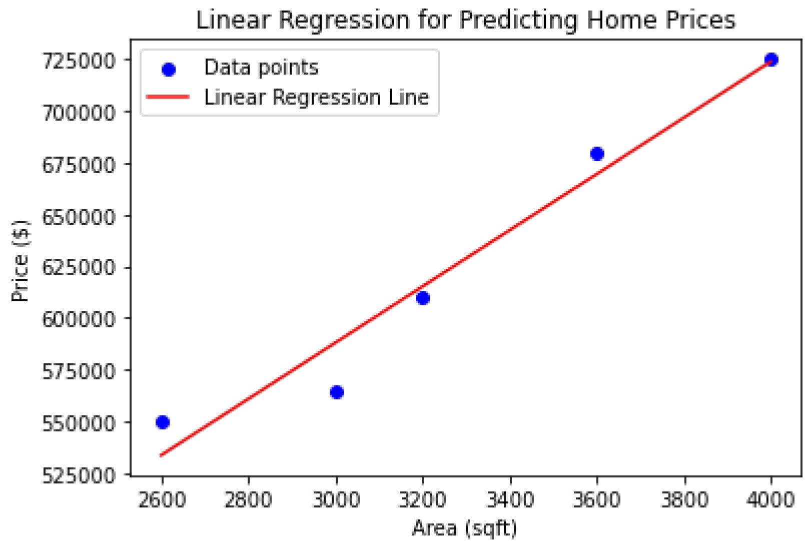
LAB TASK:-

1.implement linear regression algorithm on above dataset predict price of home with areas in the dataset by using (homeprices.csv).



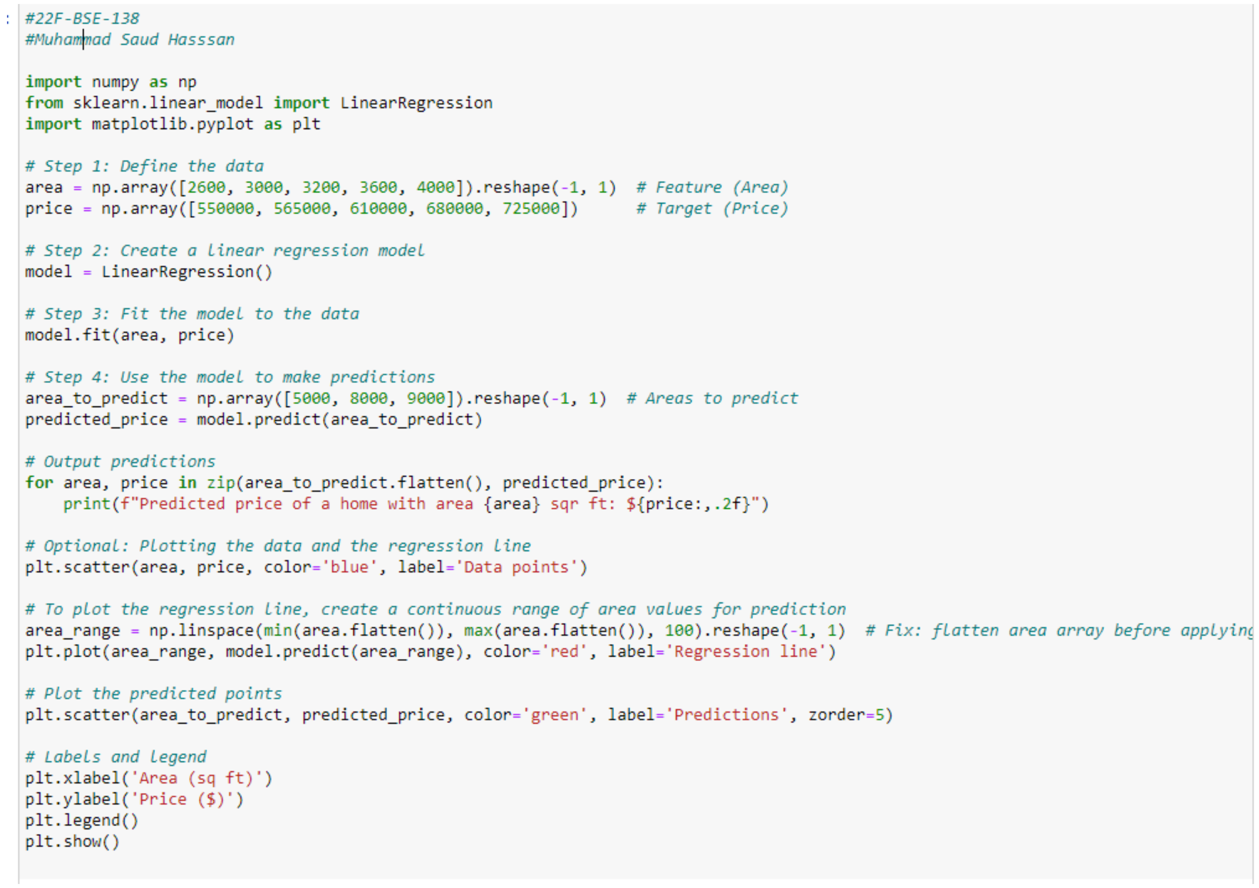
```python
#22F-BSE-138
#Muhammad Saud Hassan

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Step 1: Load data from Excel file
# Make sure 'homeprices.xlsx' is in the same directory or provide the full path
df = pd.read_excel('lab.xlsx')

# Step 2: Check the structure of the data
print(df.head())

# Assuming the columns are named 'Area' and 'Price', you can change these names based on your Excel file structure
X = df['Area'].values.reshape(-1, 1)  # Area (square feet)
y = df['Price'].values  # Price (in dollars)

# Step 3: Train the Linear Regression Model
model = LinearRegression()
model.fit(X, y)

# Step 4: Make Predictions for new areas (e.g., 2500, 3500, 4500 sqft)
areas_to_predict = np.array([2500, 3500, 4500]).reshape(-1, 1)
predicted_prices = model.predict(areas_to_predict)

# Display the predictions
for area, price in zip(areas_to_predict, predicted_prices):
    print(f"Predicted price for a house with area {area[0]} sqft: ${price:,.2f}")

# Step 5: Visualize the data and the regression line
plt.scatter(X, y, color='blue', label='Data points')
plt.plot(X, model.predict(X), color='red', label='Linear Regression Line')
plt.xlabel('Area (sqft)')
plt.ylabel('Price ($)')
plt.title('Linear Regression for Predicting Home Prices')
plt.legend()
plt.show()

# Optionally, you can print the slope (coefficient) and intercept
print(f"Slope (coefficient): {model.coef_[0]}")
print(f"Intercept: {model.intercept_}")
```

```
   Area   Price
0  2600  550000
1  3000  565000
2  3200  610000
3  3600  680000
4  4000  725000
Predicted price for a house with area 2500 sqft: $520,085.62
Predicted price for a house with area 3500 sqft: $655,873.29
Predicted price for a house with area 4500 sqft: $791,660.96
```



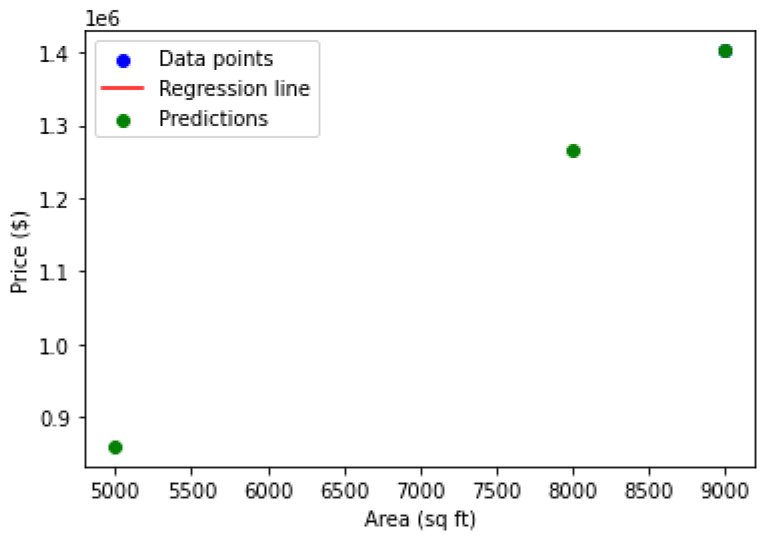
```
Slope (coefficient): 135.78767123287673
Intercept: 180616.43835616432
```

2. Implement linear regression using table 1 in such a way that the
Predict price of a home with area = 5000 sqr ft
Predict price of a home with area = 8000 sqr ft
Predict price of a home with area = 9000 sqr ft



```
#22F-BSE-138
#Muhammad Saud Hasssan

import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Step 1: Define the data
area = np.array([2600, 3000, 3200, 3600, 4000]).reshape(-1, 1)  # Feature (Area)
price = np.array([550000, 565000, 610000, 680000, 725000])      # Target (Price)

# Step 2: Create a linear regression model
model = LinearRegression()

# Step 3: Fit the model to the data
model.fit(area, price)

# Step 4: Use the model to make predictions
area_to_predict = np.array([5000, 8000, 9000]).reshape(-1, 1)  # Areas to predict
predicted_price = model.predict(area_to_predict)

# Output predictions
for area, price in zip(area_to_predict.flatten(), predicted_price):
    print(f"Predicted price of a home with area {area} sqr ft: ${price:,.2f}")

# Optional: Plotting the data and the regression line
plt.scatter(area, price, color='blue', label='Data points')

# To plot the regression line, create a continuous range of area values for prediction
area_range = np.linspace(min(area.flatten()), max(area.flatten()), 100).reshape(-1, 1)  # Fix: flatten area array before applying
plt.plot(area_range, model.predict(area_range), color='red', label='Regression line')

# Plot the predicted points
plt.scatter(area_to_predict, predicted_price, color='green', label='Predictions', zorder=5)

# Labels and Legend
plt.xlabel('Area (sq ft)')
plt.ylabel('Price ($)')
plt.legend()
plt.show()
```

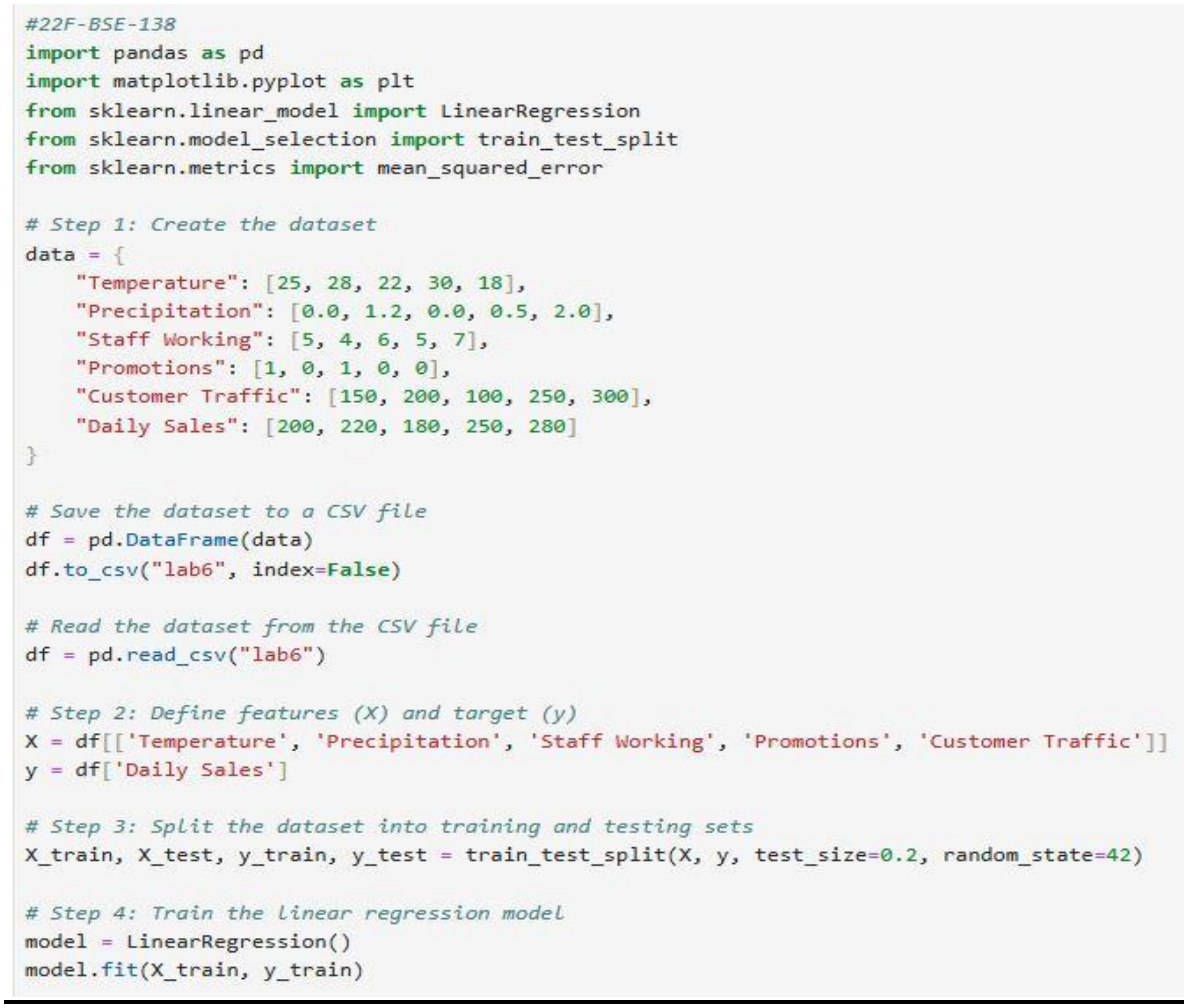
```
Predicted price of a home with area 5000 sqr ft: $859,554.79
Predicted price of a home with area 8000 sqr ft: $1,266,917.81
Predicted price of a home with area 9000 sqr ft: $1,402,705.48
```



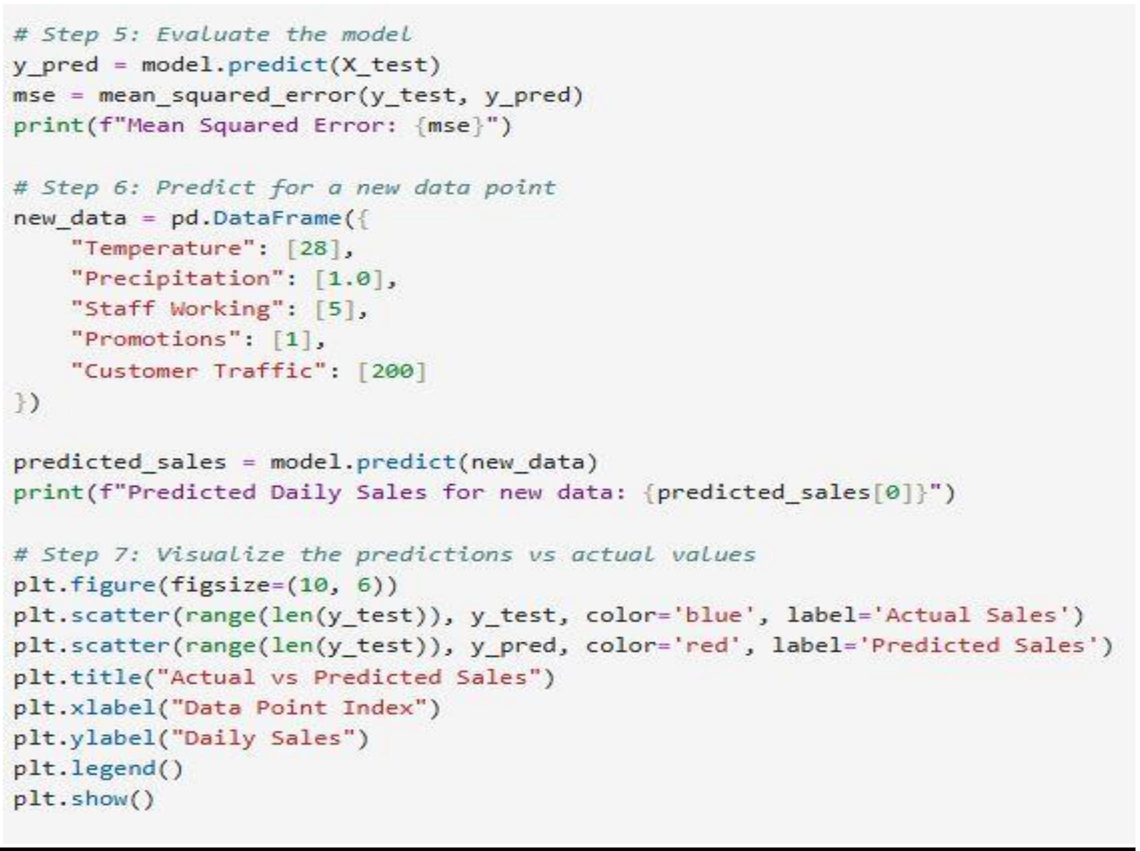
HOME TASK:-

temperature	precipitation	staff working	promotions	customer traffic	daily sales
25	0	5	1	150	200
28	1.2	4	0	200	220
22	0	6	1	100	180
30	0.5	5	0	250	250
18	2	7	0	300	280

1. Using Linear Regression, train a model on the given dataset to predict daily sales based on features like temperature, precipitation, staff working, promotions, and customer traffic. Also, predict the daily sales for a new data point where temperature = 28, precipitation = 1.0, staff working = 5, promotions = 1, and customer traffic = 200.

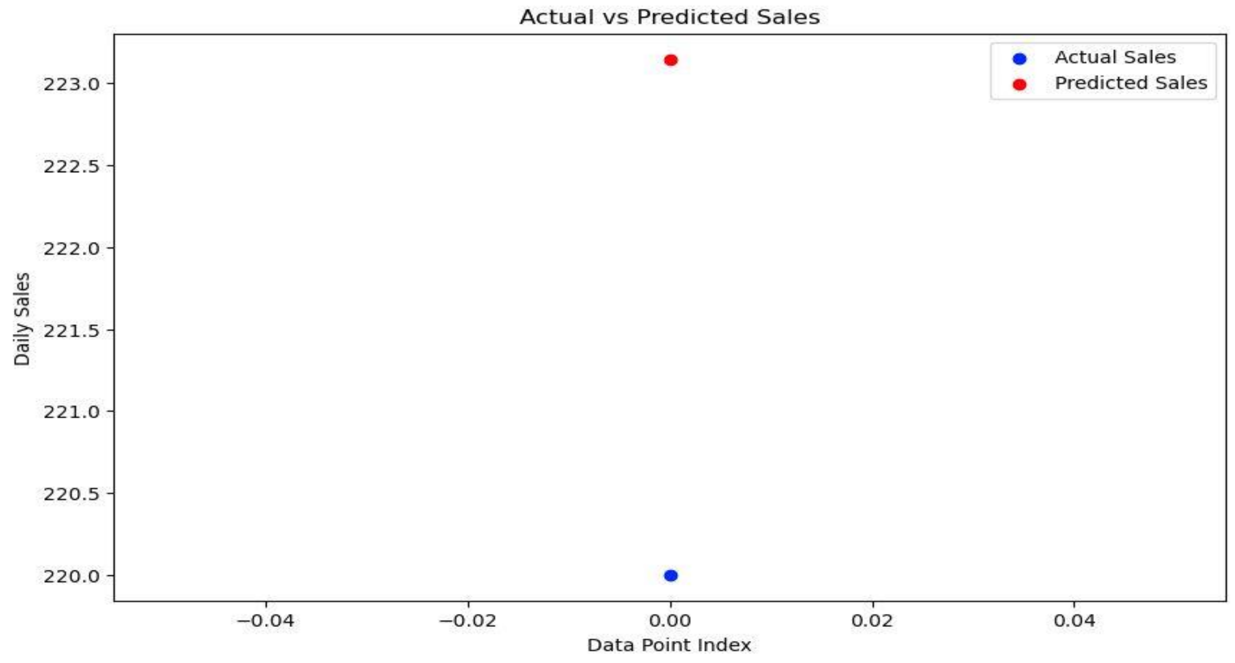


```python
#22F-BSE-138
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Step 1: Create the dataset
data = {
    "Temperature": [25, 28, 22, 30, 18],
    "Precipitation": [0.0, 1.2, 0.0, 0.5, 2.0],
    "Staff Working": [5, 4, 6, 5, 7],
    "Promotions": [1, 0, 1, 0, 0],
    "Customer Traffic": [150, 200, 100, 250, 300],
    "Daily Sales": [200, 220, 180, 250, 280]
}

# Save the dataset to a CSV file
df = pd.DataFrame(data)
df.to_csv("lab6", index=False)

# Read the dataset from the CSV file
df = pd.read_csv("lab6")

# Step 2: Define features (X) and target (y)
X = df[['Temperature', 'Precipitation', 'Staff Working', 'Promotions', 'Customer Traffic']]
y = df['Daily Sales']

# Step 3: Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 4: Train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

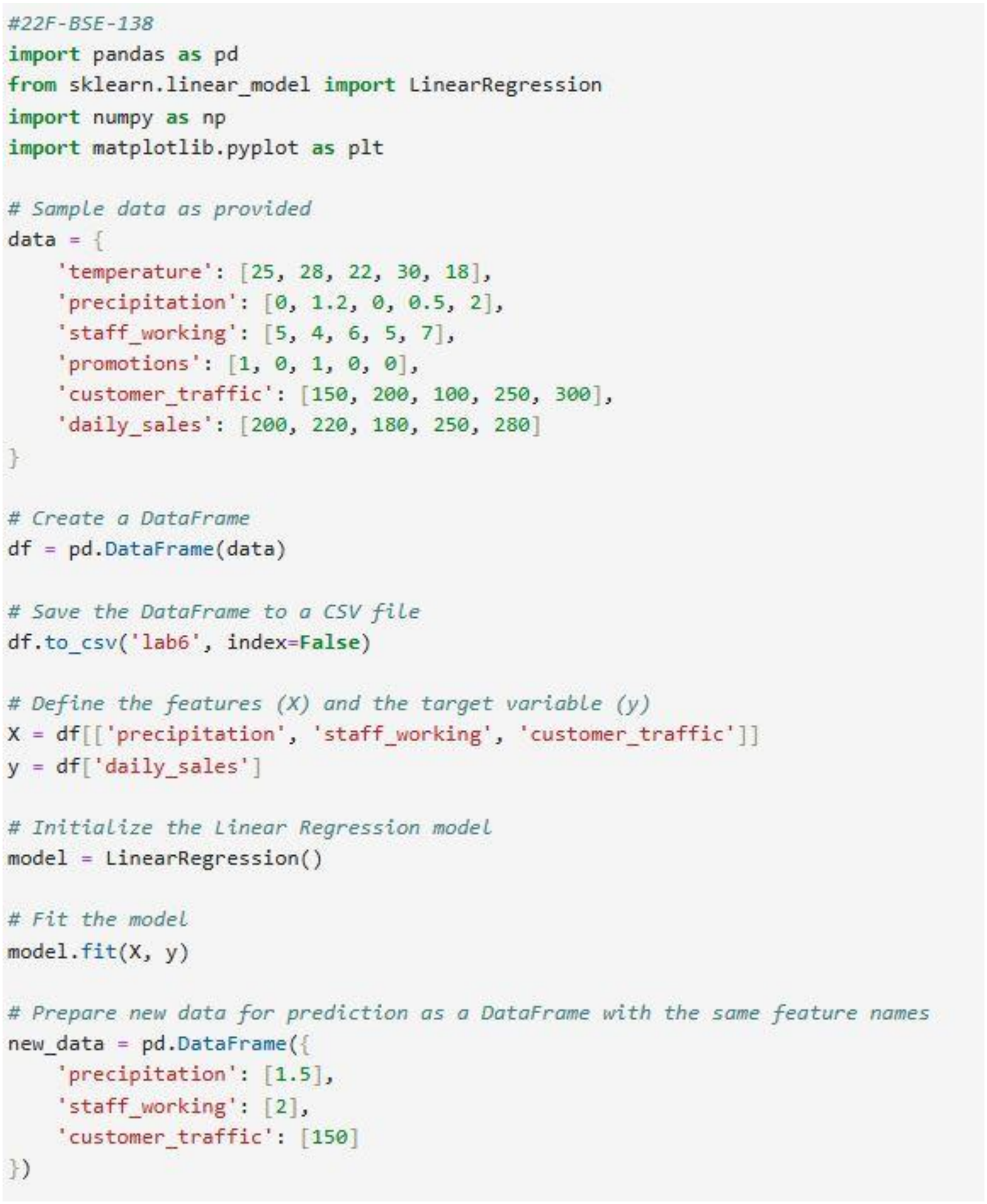


```python
# Step 5: Evaluate the model
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")

# Step 6: Predict for a new data point
new_data = pd.DataFrame({
    "Temperature": [28],
    "Precipitation": [1.0],
    "Staff Working": [5],
    "Promotions": [1],
    "Customer Traffic": [200]
})

predicted_sales = model.predict(new_data)
print(f"Predicted Daily Sales for new data: {predicted_sales[0]}")

# Step 7: Visualize the predictions vs actual values
plt.figure(figsize=(10, 6))
plt.scatter(range(len(y_test)), y_test, color='blue', label='Actual Sales')
plt.scatter(range(len(y_test)), y_pred, color='red', label='Predicted Sales')
plt.title("Actual vs Predicted Sales")
plt.xlabel("Data Point Index")
plt.ylabel("Daily Sales")
plt.legend()
plt.show()
```

```
Mean Squared Error: 9.905078355350518
Predicted Daily Sales for new data: 224.65058703076772
```



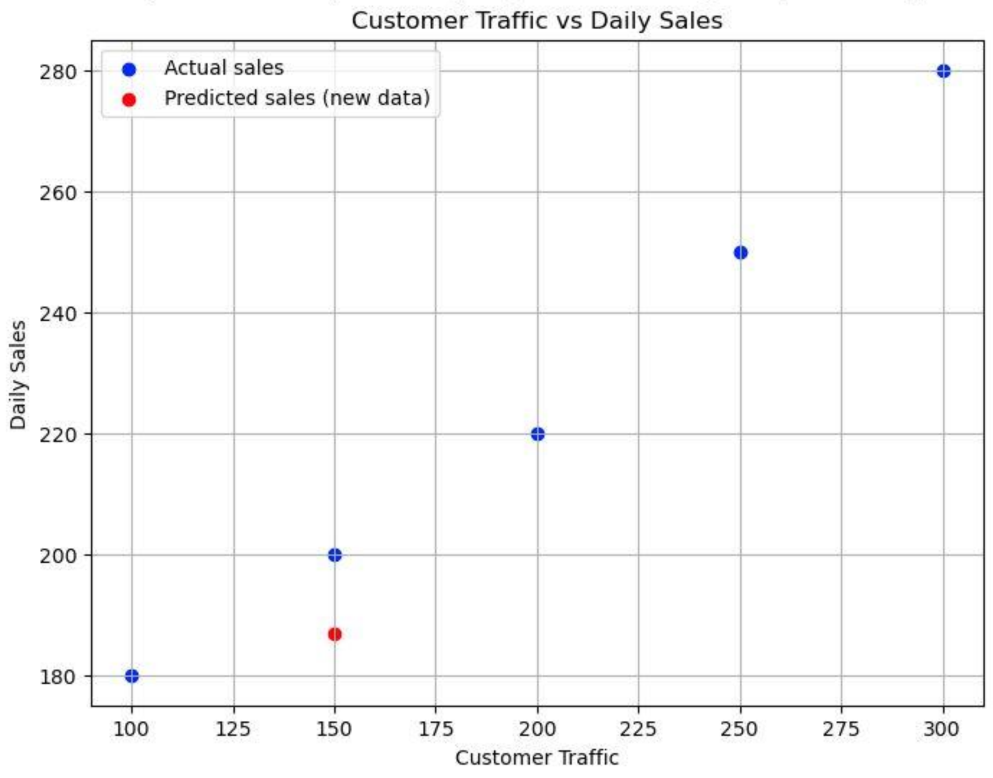
2. Implement linear regression using above mention table in such a way that the **Predict temperature = 32 Predict precipitation = 1.5 Predict staff working=2.**



```
#22F-BSE-138
import pandas as pd
from sklearn.linear_model import LinearRegression
import numpy as np
import matplotlib.pyplot as plt

# Sample data as provided
data = {
    'temperature': [25, 28, 22, 30, 18],
    'precipitation': [0, 1.2, 0, 0.5, 2],
    'staff_working': [5, 4, 6, 5, 7],
    'promotions': [1, 0, 1, 0, 0],
    'customer_traffic': [150, 200, 100, 250, 300],
    'daily_sales': [200, 220, 180, 250, 280]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('lab6', index=False)

# Define the features (X) and the target variable (y)
X = df[['precipitation', 'staff_working', 'customer_traffic']]
y = df['daily_sales']

# Initialize the Linear Regression model
model = LinearRegression()

# Fit the model
model.fit(X, y)

# Prepare new data for prediction as a DataFrame with the same feature names
new_data = pd.DataFrame({
    'precipitation': [1.5],
    'staff_working': [2],
    'customer_traffic': [150]
})
```

```
# Predict the daily sales for the new data
predicted_sales = model.predict(new_data)

# Print the predicted daily sales
print("Predicted daily sales for the input values (precipitation=1.5, staff_working=2, customer_traffic=150):", predicted_sales[0])

# Plotting the graph to visualize the predictions
plt.figure(figsize=(8,6))

# Plot the data points and predictions
plt.scatter(df['customer_traffic'], df['daily_sales'], color='blue', label='Actual sales')
plt.scatter(new_data['customer_traffic'], predicted_sales, color='red', label='Predicted sales (new data)')

plt.title('Customer Traffic vs Daily Sales')
plt.xlabel('Customer Traffic')
plt.ylabel('Daily Sales')
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```

```
Predicted daily sales for the input values (precipitation=1.5, staff_working=2, customer_traffic=150): 186.866048862679
```


Customer Traffic vs Daily Sales
● Actual sales
● Predicted sales (new data)
Daily Sales
Customer Traffic