FA20-BCE-030 (SAMEEN YASER)

Assignment #2

a) Pseudocode for Simple Linear Regression:

```
function simple_linear_regression(X, y):
    # X: input feature (independent variable)
    # y: target variable (dependent variable)
    n = length(X)  # number of data points
    mean_X = mean(X)
    mean_y = mean(y)
    # Calculate the slope (m) and y-intercept (b) for the line equation y = mx + b
    numerator = 0
    denominator = 0
    for i from 1 to n:
        numerator += (X[i] - mean_X) * (y[i] - mean_y)
        denominator += (X[i] - mean_X)^2
    slope_m = numerator / denominator
    y_intercept_b = mean_y - slope_m * mean_X
    return slope_m, y_intercept_b
function predict(X, slope, intercept):
    # X: input feature for prediction
    return slope * X + intercept
```

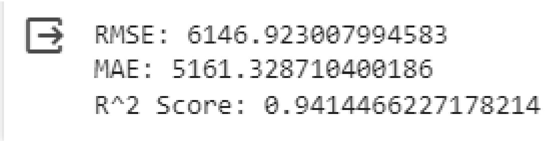
b) Python Program for Simple Linear Regression:

```python
import numpy as np

def simple_linear_regression(X, y):
    mean_X = np.mean(X)
    mean_y = np.mean(y)

    numerator = 0
    denominator = 0

    for i in range(len(X)):
        numerator += (X[i] - mean_X) * (y[i] - mean_y)
        denominator += (X[i] - mean_X) ** 2

    slope_m = numerator / denominator
    intercept_b = mean_y - slope_m * mean_X

    return slope_m, intercept_b

def predict(X, slope, intercept):
    return slope * X + intercept
```

c) **Implement and Evaluate your Model:**

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import pandas as pd

# Load the dataset
data = pd.read_csv('SalaryData.csv')
X = data['YearsExperience'].values
y = data['Salary'].values

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train your simple linear regression model
slope, intercept = simple_linear_regression(X_train, y_train)

# Make predictions on the test set
y_pred = predict(X_test, slope, intercept)

# Evaluate the performance
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"MAE: {mae}")
print(f"R^2 Score: {r2}")
```

Output:

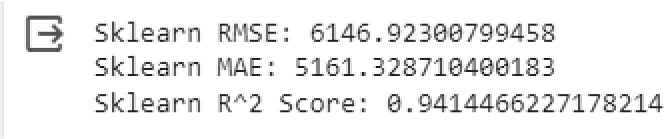


```
RMSE: 6146.923007994583
MAE: 5161.328710400186
R^2 Score: 0.9414466227178214
```

d) Use LinearRegression from sklearn:

```
from sklearn.linear_model import LinearRegression

# Create a Linear Regression model
model = LinearRegression()

# Train the model
X_train = X_train.reshape(-1, 1)  # Reshape to 2D array for sklearn
model.fit(X_train, y_train)

# Make predictions on the test set
X_test = X_test.reshape(-1, 1)  # Reshape to 2D array for sklearn
y_pred_sklearn = model.predict(X_test)

# Evaluate the performance
rmse_sklearn = np.sqrt(mean_squared_error(y_test, y_pred_sklearn))
mae_sklearn = mean_absolute_error(y_test, y_pred_sklearn)
r2_sklearn = r2_score(y_test, y_pred_sklearn)

print(f"Sklearn RMSE: {rmse_sklearn}")
print(f"Sklearn MAE: {mae_sklearn}")
print(f"Sklearn R^2 Score: {r2_sklearn}")
```

Output:

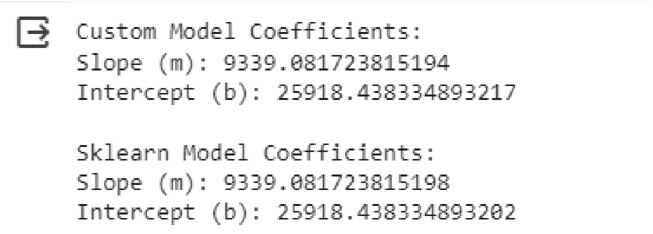


```
Sklearn RMSE: 6146.92300799458
Sklearn MAE: 5161.328710400183
Sklearn R^2 Score: 0.9414466227178214
```

e) Print Coefficients:

```
print("Custom Model Coefficients:")
print(f"Slope (m): {slope}")
print(f"Intercept (b): {intercept}")
print("\nSklearn Model Coefficients:")
print(f"Slope (m): {model.coef_[0]}")
print(f"Intercept (b): {model.intercept_}")
```

Output:

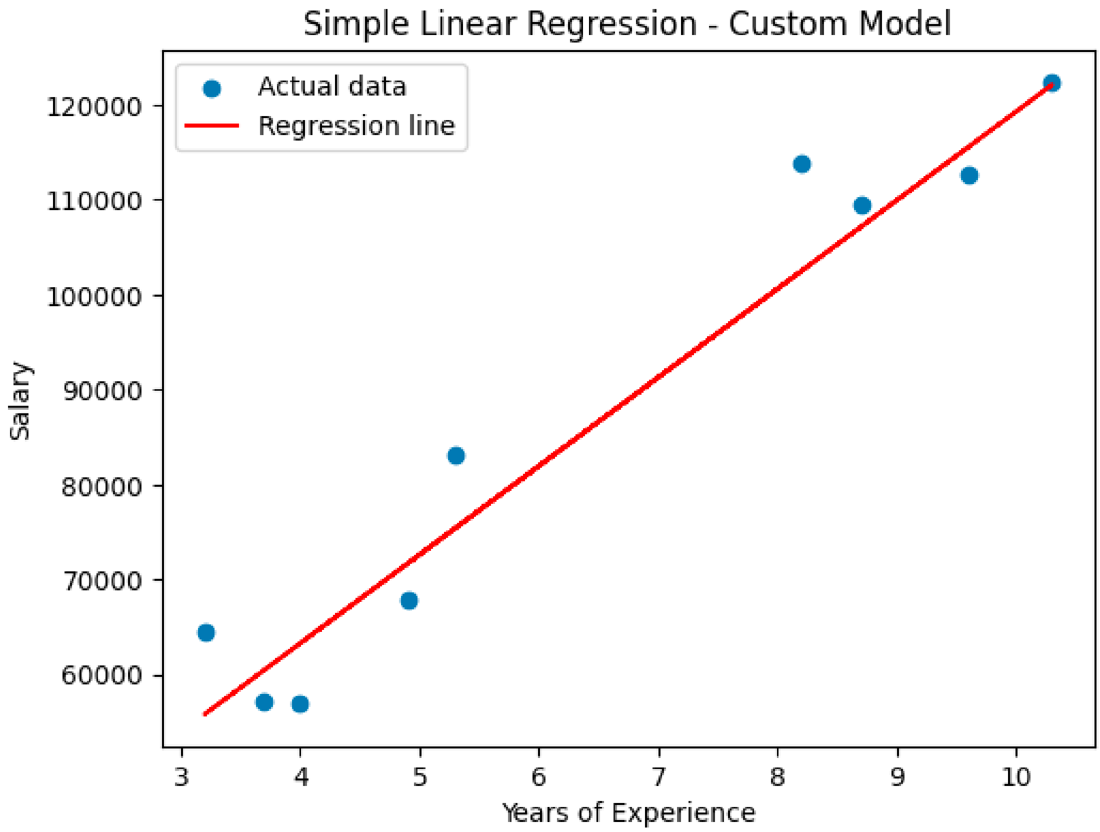


```
Custom Model Coefficients:
Slope (m): 9339.081723815194
Intercept (b): 25918.438334893217

Sklearn Model Coefficients:
Slope (m): 9339.081723815198
Intercept (b): 25918.438334893202
```

f) Plot Scatter Plot and Regression Line for Your Model:

```
import matplotlib.pyplot as plt
# Scatter plot
plt.scatter(X_test, y_test, label='Actual data')
# Regression line
plt.plot(X_test, y_pred, color='red', label='Regression line')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.title('Simple Linear Regression - Custom Model')
plt.legend()
plt.show()
```

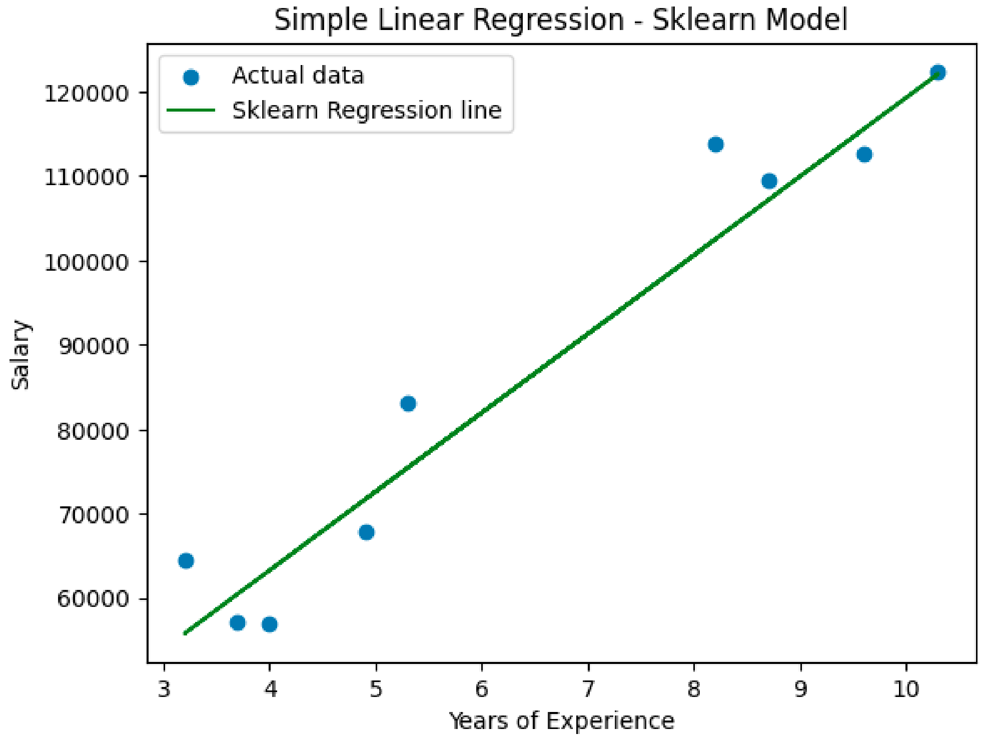
Output:



g) Plot Scatter Plot and Regression Line for Sklearn Model:

```
# Scatter plot
plt.scatter(X_test, y_test, label='Actual data')

# Regression line
plt.plot(X_test, y_pred_sklearn, color='green', label='Sklearn
Regression line')

plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.title('Simple Linear Regression - Sklearn Model')
plt.legend()
plt.show()
```

Output:


Simple Linear Regression - Sklearn Model
● Actual data
Sklearn Regression line
Salary
Years of Experience