Dataset Loading and Understanding

```
import pandas as pd
import numpy as np
df = pd.read_csv('/content/Salary_dataset.csv')
print(df)
```

```
    Unnamed: 0  YearsExperience    Salary
0            0              1.2   39344.0
1            1              1.4   46206.0
2            2              1.6   37732.0
3            3              2.1   43526.0
4            4              2.3   39892.0
5            5              3.0   56643.0
6            6              3.1   60151.0
7            7              3.3   54446.0
8            8              3.3   64446.0
9            9              3.8   57190.0
10          10              4.0   63219.0
11          11              4.1   55795.0
12          12              4.1   56958.0
13          13              4.2   57082.0
14          14              4.6   61112.0
15          15              5.0   67939.0
16          16              5.2   66030.0
17          17              5.4   83089.0
18          18              6.0   81364.0
19          19              6.1   93941.0
20          20              6.9   91739.0
21          21              7.2   98274.0
22          22              8.0  101303.0
23          23              8.3  113813.0
24          24              8.8  109432.0
25          25              9.1  105583.0
26          26              9.6  116970.0
27          27              9.7  112636.0
28          28             10.4  122392.0
29          29             10.6  121873.0
```

Display first five rows

```
df.head()
```

	Unnamed: 0	YearsExperience	Salary
0	0	1.2	39344.0
1	1	1.4	46206.0
2	2	1.6	37732.0
3	3	2.1	43526.0
4	4	2.3	39892.0

Next steps: Generate code with df New interactive sheet

```
df.tail()
```

	Unnamed: 0	YearsExperience	Salary
25	25	9.1	105583.0
26	26	9.6	116970.0
27	27	9.7	112636.0
28	28	10.4	122392.0
29	29	10.6	121873.0

Identify input and output variables.

```
YearsExperience = df[['YearsExperience']]
Salary = df['Salary']

print("Input variable (YearsExperience):")
print(YearsExperience.head())
print("\nOutput variable (Salary):")
print(Salary.head())
```

```
Input variable (YearsExperience):
   YearsExperience
0              1.2
1              1.4
2              1.6
3              2.1
4              2.3

Output variable (Salary):
0    39344.0
1    46206.0
2    37732.0
3    43526.0
4    39892.0
Name: Salary, dtype: float64
```

Linear Regression using Scikit-learn

Split data into training and testing sets

```
X = df[['YearsExperience']]   # Independent variable
y = df['Salary']              # Dependent variable
# Split into 80% training and 20% testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Import LinearRegression from sklearn

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

Train the model using fit()

```
# Create model
model = LinearRegression()
# Train model
model.fit(X_train, y_train)
```

```
▾ LinearRegression  ⓘ ?
LinearRegression()
```

Predict output values.

```
# Predict salaries for test set
y_pred = model.predict(X_test)
# Compare predictions with actual values
print("Predicted salaries:", y_pred)
print("Actual salaries:", list(y_test))
```

```
Predicted salaries: [115791.21011287  71499.27809463 102597.86866063  75268.80422384
  55478.79204548  60190.69970699]
Actual salaries: [112636.0, 67939.0, 113813.0, 83089.0, 64446.0, 57190.0]
```

Performance Evaluation

Calculate Mean Squared Error, Calculate R-squared value.

```
from sklearn.metrics import mean_squared_error, r2_score
# Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
# Calculate R-squared value
r2 = r2_score(y_test, y_pred)
print("R-squared:", r2)
```

```
Mean Squared Error: 49830096.855908394
R-squared: 0.9024461774180497
```

Compare slope and intercept values.

```
print("Slope (Coefficient):", model.coef_)
print("Intercept:", model.intercept_)
```

Compare prediction results

```
print("Predicted salaries:", y_pred)
print("Actual salaries:", list(y_test))
```

```
Predicted salaries: [115791.21011287  71499.27809463 102597.86866063  75268.80422384
  55478.79204548  60190.69970699]
Actual salaries: [112636.0, 67939.0, 113813.0, 83089.0, 64446.0, 57190.0]
```

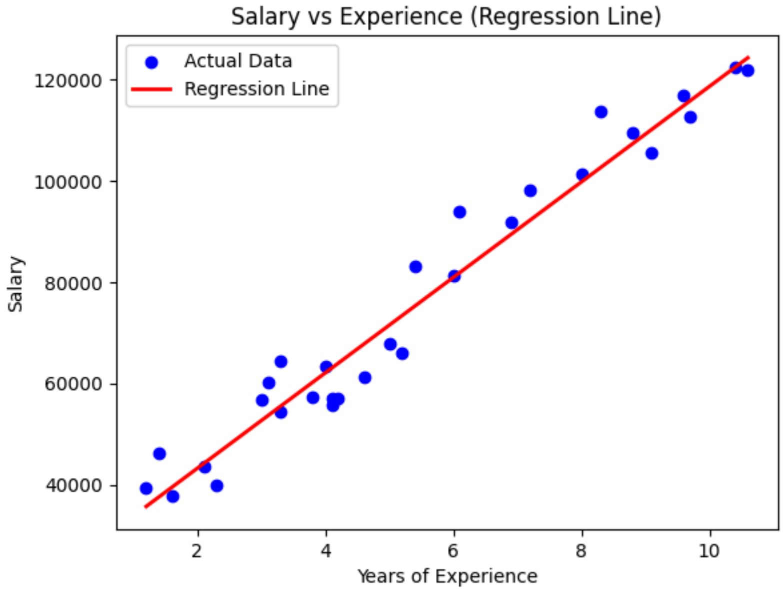
Compare error values

```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MSE:", mse)
print("R²:", r2)
```

```
MSE: 49830096.855908394
R²: 0.9024461774180497
```

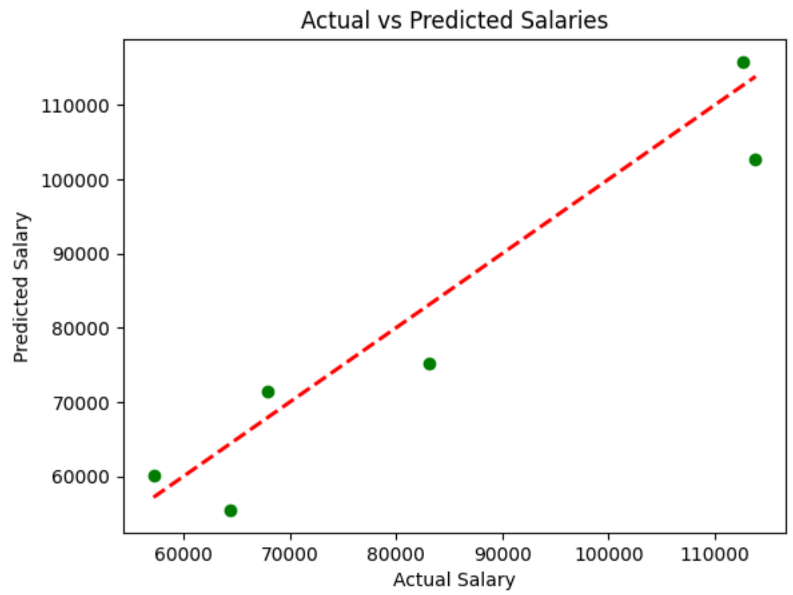
Plot regression line using Scikit-learn

```
import matplotlib.pyplot as plt
# Scatter plot of actual data
plt.scatter(X, y, color='blue', label='Actual Data')
# Regression line
plt.plot(X, model.predict(X), color='red', linewidth=2, label='Regression Line')
plt.xlabel("Years of Experience")
plt.ylabel("Salary")
plt.title("Salary vs Experience (Regression Line)")
plt.legend()
plt.show()
```



Plot actual vs predicted values.

```
plt.scatter(y_test, y_pred, color='green')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', linewidth=2)
plt.xlabel("Actual Salary")
plt.ylabel("Predicted Salary")
plt.title("Actual vs Predicted Salaries")
plt.show()
```

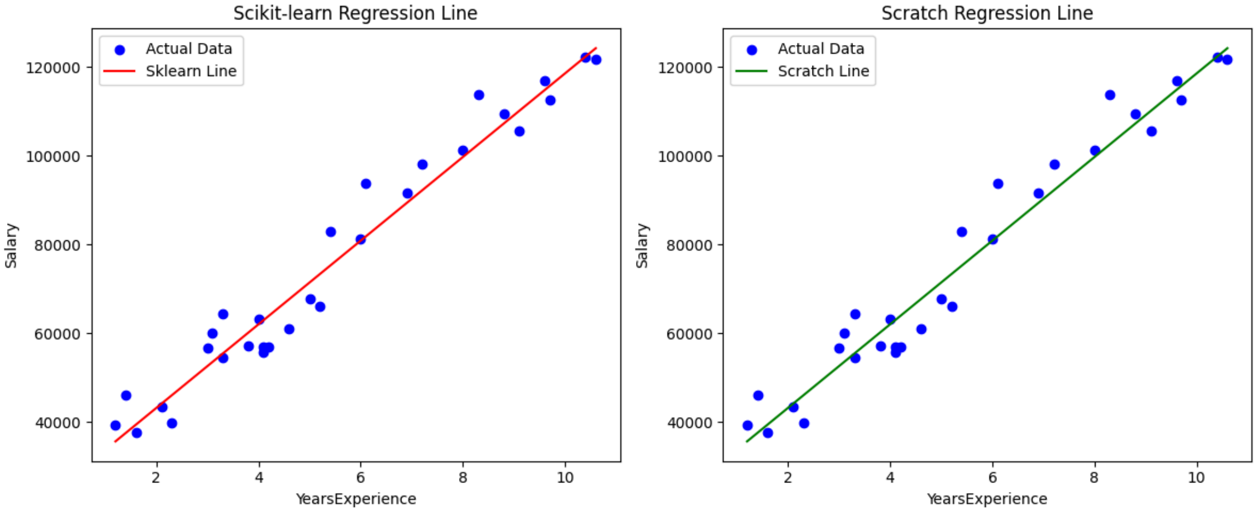


Compare plots with scratch implementation

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

# 1. Load dataset
data = pd.read_csv("Salary_dataset.csv")
X = data[['YearsExperience']]
y = data['Salary']

# 2. Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 3. Scikit-learn implementation
model = LinearRegression()
model.fit(X_train, y_train)
y_pred_sklearn = model.predict(X_test)

# Coefficients from sklearn
print("Scikit-learn Slope:", model.coef_[0])
print("Scikit-learn Intercept:", model.intercept_)

# 4. Scratch implementation (manual math)
x_mean = X_train['YearsExperience'].mean()
y_mean = y_train.mean()

# slope (m)
m = np.sum((X_train['YearsExperience'] - x_mean) * (y_train - y_mean)) / \
    np.sum((X_train['YearsExperience'] - x_mean)**2)

# intercept (c)
c = y_mean - m * x_mean

print("Scratch Slope:", m)
print("Scratch Intercept:", c)

# Predictions using scratch formula
y_pred_scratch = m * X_test['YearsExperience'] + c

# 5. Plot comparison
plt.figure(figsize=(12,5))

# Left: Scikit-learn regression line
plt.subplot(1,2,1)
plt.scatter(X, y, color='blue', label='Actual Data')
plt.plot(X, model.predict(X), color='red', label='Sklearn Line')
plt.title("Scikit-learn Regression Line")
plt.xlabel("YearsExperience")
plt.ylabel("Salary")
plt.legend()

# Right: Scratch regression line
```

```
plt.subplot(1,2,2)
plt.scatter(X, y, color='blue', label='Actual Data')
plt.plot(X, m*X['YearsExperience'] + c, color='green', label='Scratch Line')
plt.title("Scratch Regression Line")
plt.xlabel("YearsExperience")
plt.ylabel("Salary")
plt.legend()

plt.tight_layout()
plt.show()

# 6. Compare predictions
print("First 5 predictions (Scikit-learn):", y_pred_sklearn[:5])
print("First 5 predictions (Scratch):", y_pred_scratch[:5])
```

```
Scikit-learn Slope: 9423.815323030976
Scikit-learn Intercept: 24380.201479473704
Scratch Slope: 9423.815323030978
Scratch Intercept: 24380.201479473697
```



```
First 5 predictions (Scikit-learn): [115791.21011287  71499.27809463 102597.86866063  75268.80422384
  55478.79204548]
First 5 predictions (Scratch): 27    115791.210113
15     71499.278095
23    102597.868661
17     75268.804224
8      55478.792045
Name: YearsExperience, dtype: float64
```