```
# Name:- Rohit Kumar
# Roll_No:- 25901334
# Model:- Linear Regression
```

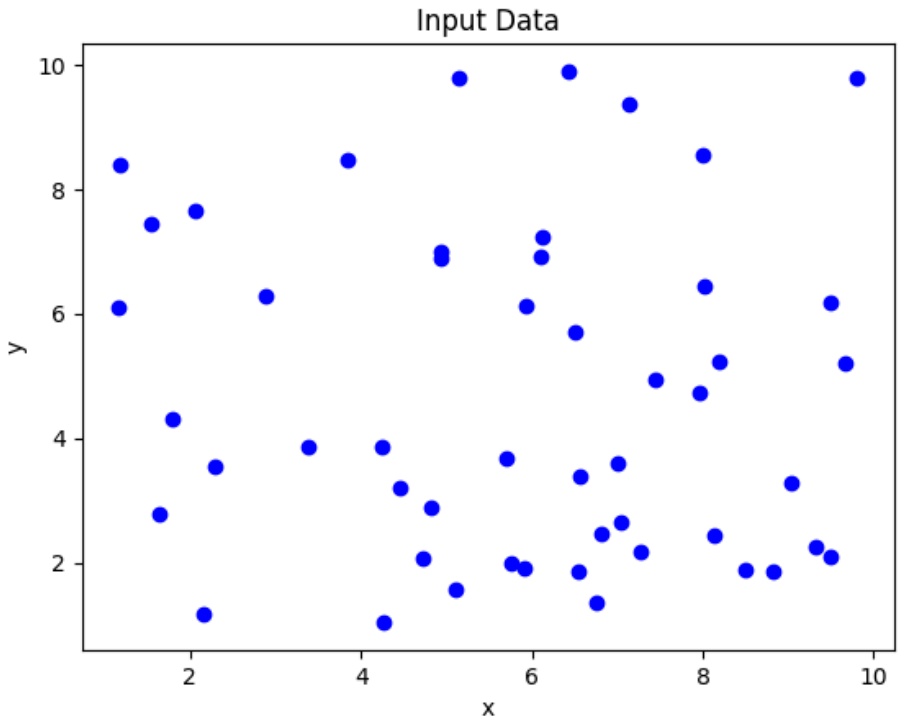
```
import numpy as np
import matplotlib.pyplot as plt
```

```
np.random.seed(0)  # For reproducibility

# Generate 50 random x values between 1 and 10 (can be float)
x = np.random.uniform(1, 10, size=50)

# Generate 50 random y values between 1 and 10 (independent random)
y = np.random.uniform(1, 10, size=50)
```

```
plt.scatter(x, y, color='blue')
plt.title("Input Data")
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```




```
X = np.column_stack((np.ones(x.shape[0]), x))  # Add bias column
Y = y.reshape(-1,1)
```

```
# Step 5a: X^T X
XtX = X.T @ X
print("X^T X:\n", XtX)

# Step 5b: inverse of X^T X
inv_XtX = np.linalg.inv(XtX)
print("Inverse of X^T X:\n", inv_XtX)

# Step 5c: X^T Y
XtY = X.T @ Y
print("X^T Y:\n", XtY)
```

```
# Step 5d: compute theta
theta = inv_XtX @ XtY
print("Theta (intercept, slope):", theta.ravel())
```

```
X^T X:
 [[  50.          292.08430322]
 [ 292.08430322 2006.48595104]]
Inverse of X^T X:
 [[ 0.1336672  -0.01945794]
 [-0.01945794  0.00333088]]
X^T Y:
 [[ 233.43015234]
 [1335.4825582 ]]
Theta (intercept, slope): [ 5.21621008 -0.09374126]
```

```
y_pred = X @ theta
print("Predicted y values:", y_pred.ravel())
```

```
Predicted y values: [4.65945057 4.51908402 4.61393461 4.66276647 4.76504339 4.57754644
 4.75328901 4.37010546 4.30945413 4.79897018 4.45451307 4.67625531
 4.64322588 4.34156943 5.06253773 5.04896032 5.10541113 4.42001129
 4.46596024 4.38846447 4.29683654 4.44824161 4.73313189 4.46395869
 5.02268407 4.58258577 5.00152575 4.32547869 4.68220033 4.77263041
 4.89927082 4.46927002 4.73762784 4.64289737 5.1066164  4.60138743
 4.60606118 4.60197926 4.32625558 4.54723655 4.81916229 4.75375747
 4.53389735 5.07165831 4.55993683 4.55667084 4.94497507 5.01369739
 4.85635095 4.81561645]
```

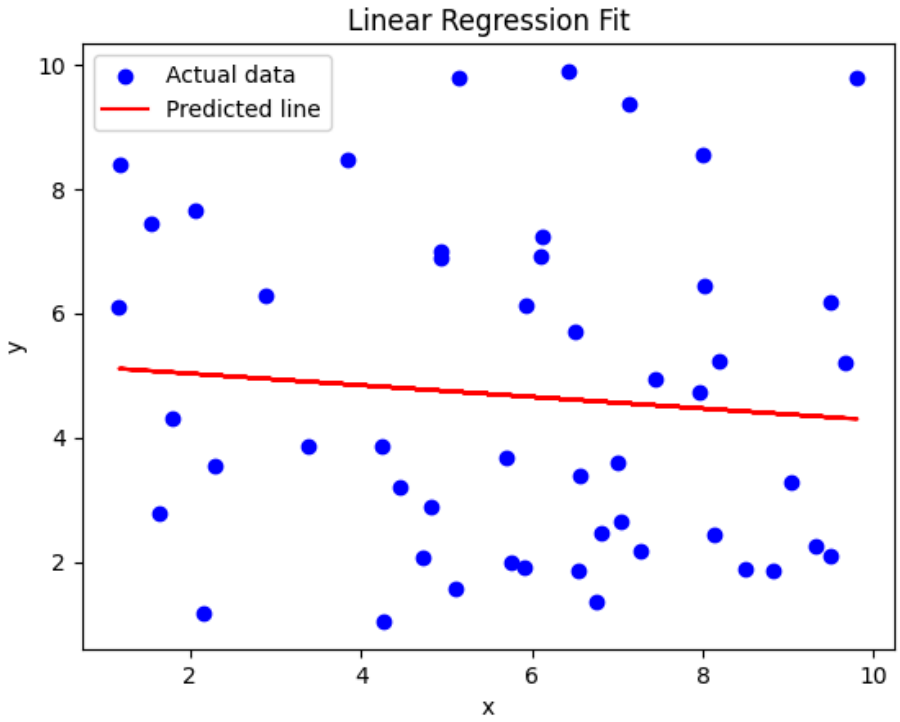
```
# Residuals
residuals = y - y_pred.ravel()
print("Residuals:", residuals)

# Mean Squared Error
mse = np.mean(residuals**2)
print("Mean Squared Error:", mse)

# R^2
ss_res = np.sum(residuals**2)
ss_tot = np.sum((y - np.mean(y))**2)
r2 = 1 - (ss_res/ss_tot)
print("R^2 Score:", r2)
```

```
Residuals: [ 1.47232036  0.4283296   5.28142993 -2.74436318 -1.88515259 -2.12576078
  2.12468592 -1.09048104  0.88734282 -1.59913986 -2.02378681 -2.68287904
  2.26374042 -2.09792287 -2.29329647 -0.73043379  3.28352793 -2.54609981
  4.07554393 -2.5235788   5.49129865  0.7696192   5.0577179   1.97965098
  2.63068814 -3.22989564 -1.45626309 -2.24370964 -1.01693855 -2.70408094
 -1.03742221  0.25909693 -3.16030037  2.58935171  0.99279669 -1.21288201
  1.1031713  -2.75651467  1.85726288  4.81642923 -0.95204172  2.25293595
 -2.34771659  2.37528653 -0.95528199 -1.90794859  1.33364134 -3.83272947
  3.60410932 -3.77335717]
Mean Squared Error: 6.7198378621084
R^2 Score: 0.007790716044710266
```

```
plt.scatter(x, y, color='blue', label='Actual data')
plt.plot(x, y_pred, color='red', label='Predicted line')
plt.title("Linear Regression Fit")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.show()
```



Start coding or generate with AI.