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

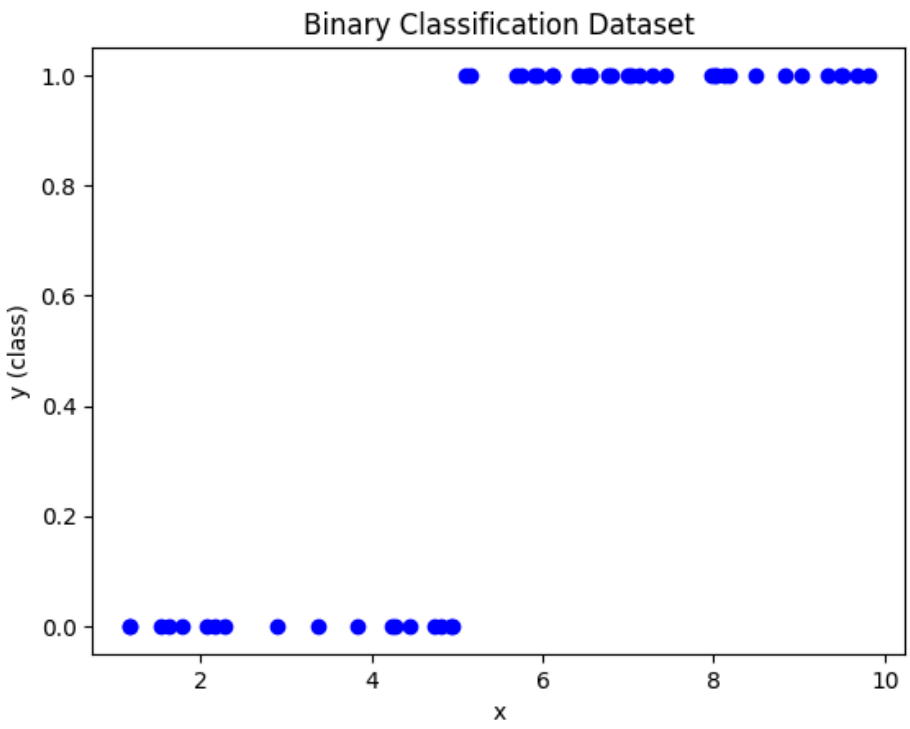
```
import numpy as np
import matplotlib.pyplot as plt
```

```
# Example: 50 points, 2 classes
np.random.seed(0)

# Feature x (1D for simplicity)
x = np.random.uniform(1, 10, 50)

# Binary label y: 0 if x < 5, 1 if x >= 5
y = np.array([0 if xi < 5 else 1 for xi in x])

# Visualize
plt.scatter(x, y, color='blue')
plt.xlabel("x")
plt.ylabel("y (class)")
plt.title("Binary Classification Dataset")
plt.show()
```




```
def sigmoid(z):
    return 1 / (1 + np.exp(-z))
```

```
theta0 = 0.0  # intercept
theta1 = 0.0  # slope
alpha = 0.1   # learning rate
iterations = 1000
```

```
# Reshape x for vectorized computation
X = np.column_stack((np.ones(len(x)), x))  # Add bias column
Y = y.reshape(-1,1)

# Initialize theta vector
theta = np.zeros((2,1))
```

```
# Gradient descent
for i in range(iterations):
    z = X @ theta
    y_hat = sigmoid(z)
    gradient = (1/len(X)) * (X.T @ (y_hat - Y))
    theta -= alpha * gradient

# Final parameters
intercept, slope = theta.ravel()
print(f"Intercept (theta0) = {intercept:.4f}")
print(f"Slope (theta1) = {slope:.4f}")
```

```
Intercept (theta0) = -5.0769
Slope (theta1) = 1.0891
```

```
# Predict probabilities
probabilities = sigmoid(X @ theta)

# Convert to class labels
y_pred = (probabilities >= 0.5).astype(int).ravel()
print("Predicted classes:", y_pred)
```

```
Predicted classes: [1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1 1 0 1 1
 1 1 1 0 1 1 0 1 1 0 0 0 0]
```

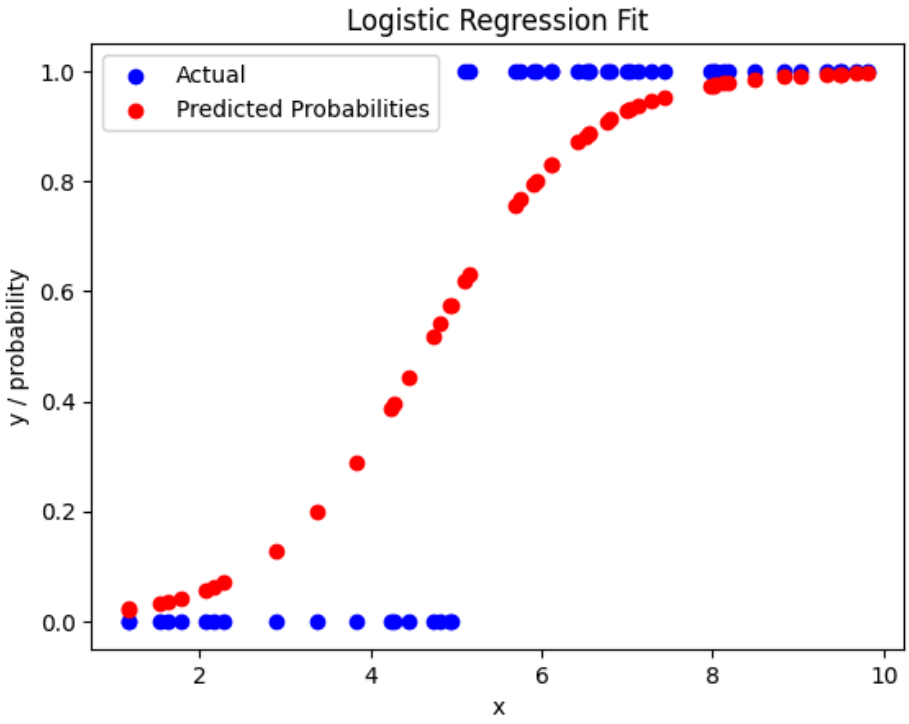
```
# Accuracy
accuracy = np.mean(y_pred == y)
print(f"Accuracy: {accuracy*100:.2f}%")

# Confusion matrix
tp = np.sum((y_pred==1) & (y==1))
tn = np.sum((y_pred==0) & (y==0))
fp = np.sum((y_pred==1) & (y==0))
fn = np.sum((y_pred==0) & (y==1))

print(f"Confusion Matrix:\nTP={tp}, TN={tn}, FP={fp}, FN={fn}")
```

```
Accuracy: 92.00%
Confusion Matrix:
TP=32, TN=14, FP=4, FN=0
```

```
plt.scatter(x, y, color='blue', label='Actual')
plt.scatter(x, probabilities, color='red', label='Predicted Probabilities')
plt.xlabel("x")
plt.ylabel("y / probability")
plt.title("Logistic Regression Fit")
plt.legend()
plt.show()
```



Start coding or generate with AI.