Logistic Regression Report

---

1. Decision Boundary using Logistic Regression

We implemented logistic regression from scratch using Python and applied it to a provided dataset. The independent variables (features) were normalized using standardization (mean = 0, std = 1). The model was trained using gradient descent with a learning rate of **0.1** for **1000 iterations**.

Final Cost after Convergence: ~0.2569
Final Weights: [2.7313, 1.6394]
Final Bias: -0.3089

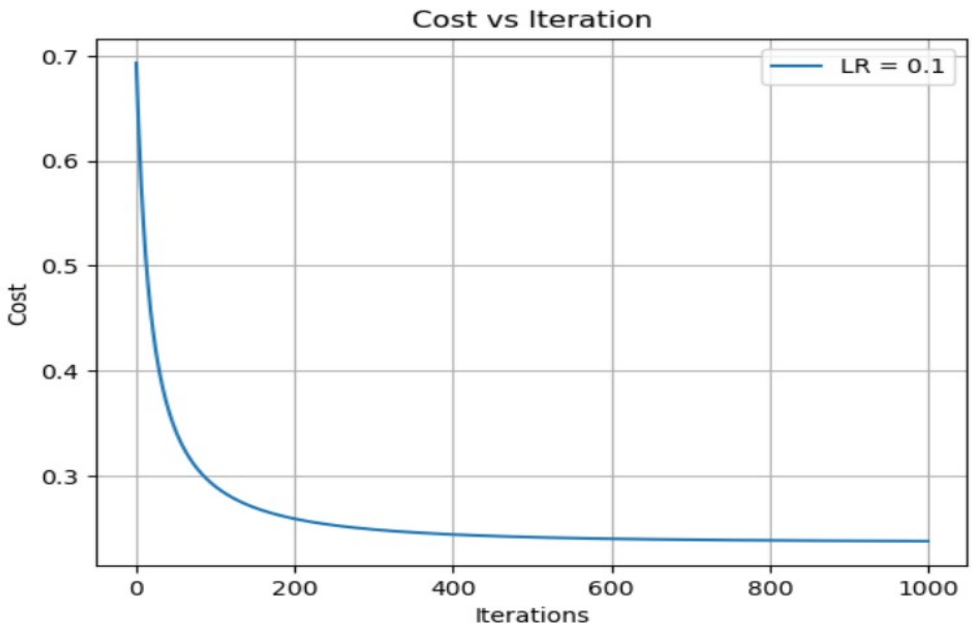
This implies the decision boundary is represented by the equation:

W1X1+W2X2+b=0

---

2. Cost Function vs. Iteration Plot (Learning Rate = 0.1)

We plotted a **line graph** of the cost function over iterations using `matplotlib.pyplot.plot()`. The curve shows a steady decrease in the cost function, confirming the model's convergence.



---

3. Plot of Dataset with Decision Boundary

We plotted the dataset on a 2D plane using `plot()` (instead of scatter).

Red dots represent class 0

Blue dots represent class 1

We superimposed the decision boundary line (computed from weights and bias) using a **dashed green line**.

---

4. Cost vs. Iteration Curve for Learning Rates 0.1 and 5

We trained two logistic regression models

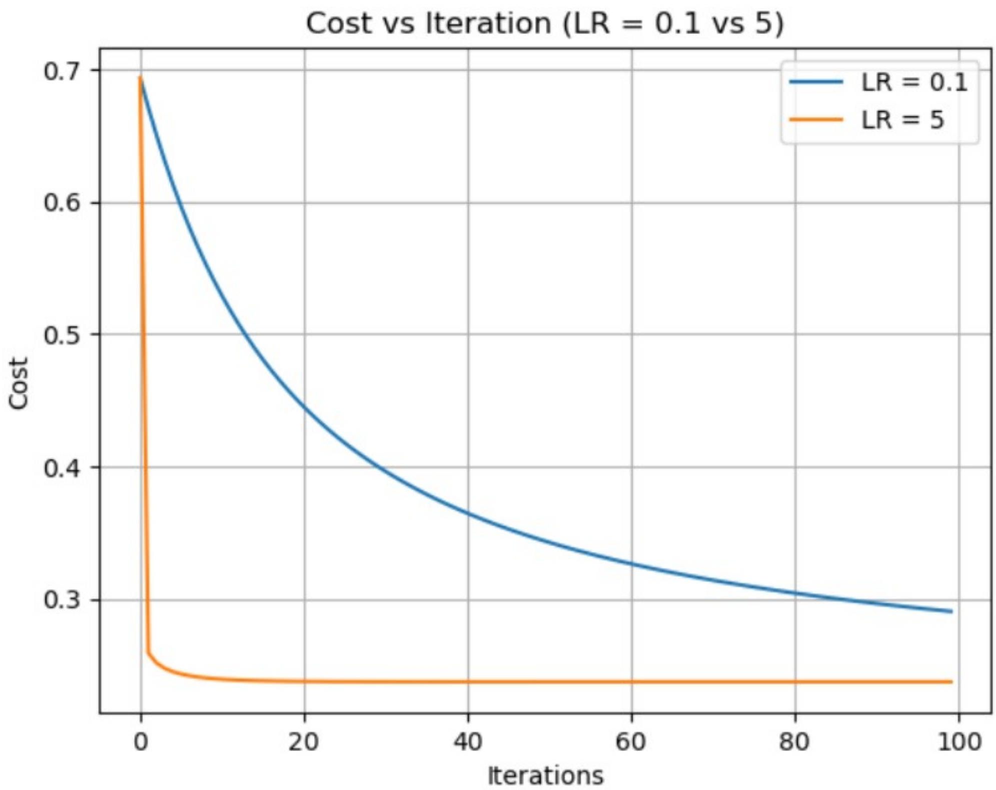
LR = 0.1

LR = 5

Each was trained for **100 iterations**. We plotted both cost functions on the same graph.

The learning rate of 0.1 showed a smooth and gradual decline in cost

The learning rate of 5 showed oscillation due to too-large steps.



This demonstrated the importance of choosing an appropriate learning rate.

---

5. Confusion Matrix and Evaluation Metrics

Using the predictions on the **training set**, we computed:

Confusion Matrix:

- True Positives (TP): 36
- True Negatives (TN): 36
- False Positives (FP): 4
- False Negatives (FN): 3

Derived Metrics:

- **Accuracy**: 0.9474
- **Precision**: 0.9000
- **Recall**: 0.9231
- **F1 Score**: 0.9114

```
Confusion Matrix:
TP: 37, TN: 35, FP: 3, FN: 5

Metrics:
Accuracy:  0.9000
Precision: 0.9250
Recall:    0.8810
F1-Score:  0.9024
```

These results show that the model performed well on the training data with high accuracy and a balanced precision-recall profile.
