Labwork 3: Logistic Regression

May 6, 2025

1 Introduction

Logistic Regression: Logistic regression is a statistical method used for binary classification problems. It models the probability that a given input belongs to a particular class using a logistic function. Unlike linear regression, logistic regression predicts a probability value between 0 and 1, which can then be thresholded to classify the input.

In this lab, we implemented a logistic regression algorithm using gradient descent to minimize the loss function. The loss function used is the Binary Cross-Entropy (BCE), which measures the difference between the predicted probabilities and the actual class labels.

2 Implementation

The implementation of logistic regression in this lab uses the gradient descent optimization algorithm to iteratively update the weights w_0, w_1, and w_2 to minimize the loss function.

The logistic regression model predicts the probability of the positive class for a given input (x_1, x_2) as follows:

$$\hat{y} = \sigma(w_1 x_1 + w_2 x_2 + w_0)$$

where $\sigma(z)$ is the sigmoid function defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

The Binary Cross-Entropy loss function for a single data point (x_1, x_2, y) is defined as:

$$L = -\left[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right]$$

The Focal Loss function is a modification of the BCE loss that focuses on hard-to-classify examples. It is defined as:

$$L = -\alpha(1 - \hat{y})^{\gamma} \log(\hat{y})$$

where α is a balancing factor and γ is a focusing parameter.
The weights are updated iteratively using the learning rate r:

$$w_0 \leftarrow w_0 - r\frac{\partial L}{\partial w_0}$$
$$w_1 \leftarrow w_1 - r\frac{\partial L}{\partial w_1}$$
$$w_2 \leftarrow w_2 - r\frac{\partial L}{\partial w_2}$$

The algorithm stops after a fixed number of iterations or when the loss converges below a specified threshold.

3 Evaluation

The program was tested with a dataset loaded from a CSV file. The user provides the learning rate r and initial values for w_0, w_1, and w_2. The program outputs the updated weights, predicted probabilities, and the loss for each iteration.

For example, testing with a dataset containing loan approval data, the program iteratively adjusts the weights to minimize the loss. Below is an example of the output:

```
1th iteration:
w0: 0.25, w1: 2.25, w2: 3.0, yi_pred: 0.5 loss: 0.6931471805599453
...
Final w0: -4.749999991371084,
Final w1: 17.049999978427714,
Final w2: 79.49999996548434,
Final loss: 18.134129583172598
```

4 Conclusion

In this labwork, we implemented a logistic regression algorithm using gradient descent. The algorithm successfully minimized the Binary Cross-Entropy loss function and found the optimal weights for the given dataset.

An important observation is that the choice of learning rate r significantly affects the convergence speed and stability of the algorithm. Proper tuning of r is essential to achieve optimal results without overshooting or slow convergence.