Assignment 1B: XOR Classification & MNIST Digit Recognition with Neural Networks Deep learning- EE569

Mohammed Ali Mayouf - 2190207878

Supervisor Dr. Nuri Benbarka

November 12, 2025

1 Introduction:

This assignment is devoted to the refactoring and optimization of the code developed in the previous task and then adapting it to solve two different problems: the XOR logic problem and MNIST handwritten digit classification. It covers the generation of synthetic data for the XOR problem, an implementation of an MLP architecture, and refactoring the codebase so as to handle the construction of a computational graph and trainable parameters automatically. On top of that, a neural network model was implemented and trained that can classify MNIST digits with high accuracy. The results of each task are recorded separately, and the complete Python implementation is available in the GitHub repository. For easier tracking, each task corresponds to a separate commit in the repository.

Task 1: XOR Problem:

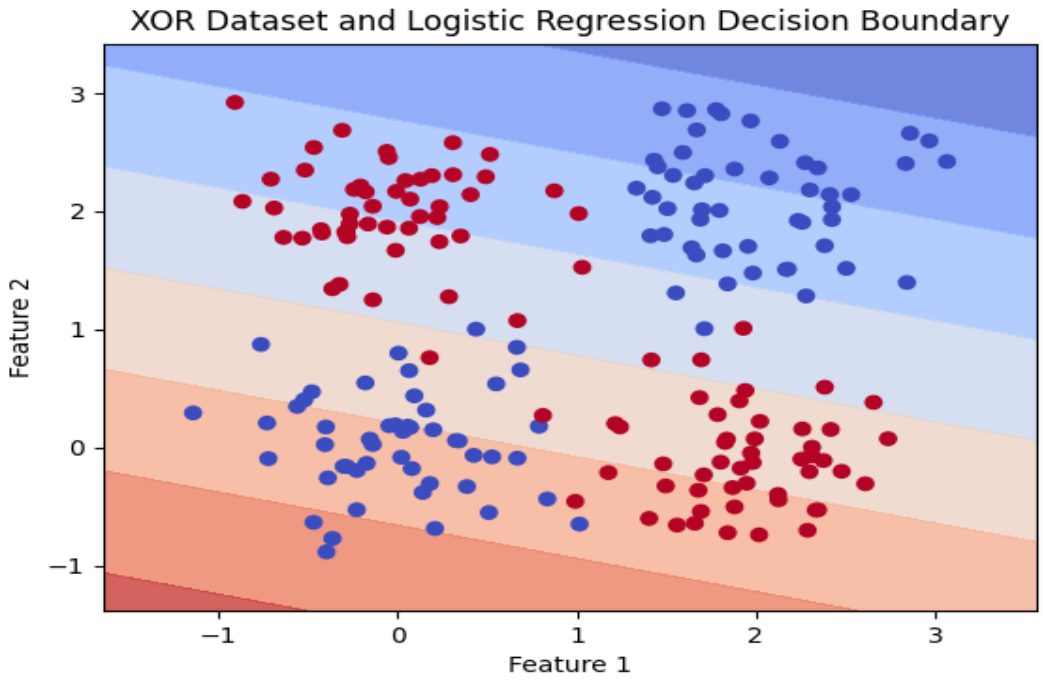
1.1 Task Description:

In this task, a generation of 2 Gaussian distribution classes taking the shape of an XOR problem was generated.

1.2 Results:

We successfully created the dataset as shown in Figure 1, and we tested it using linear logistic regression.

It is clear that we know that XOR cannot be solved using a linear solution, and this method was not successful, as the accuracy after testing was approximately 40 to 50 percent, However, here I noticed that the result was approximately 54 percent.



1.3 implementation:

GitHub commit for task 1

Task 2: Multi-Layer Perceptron:

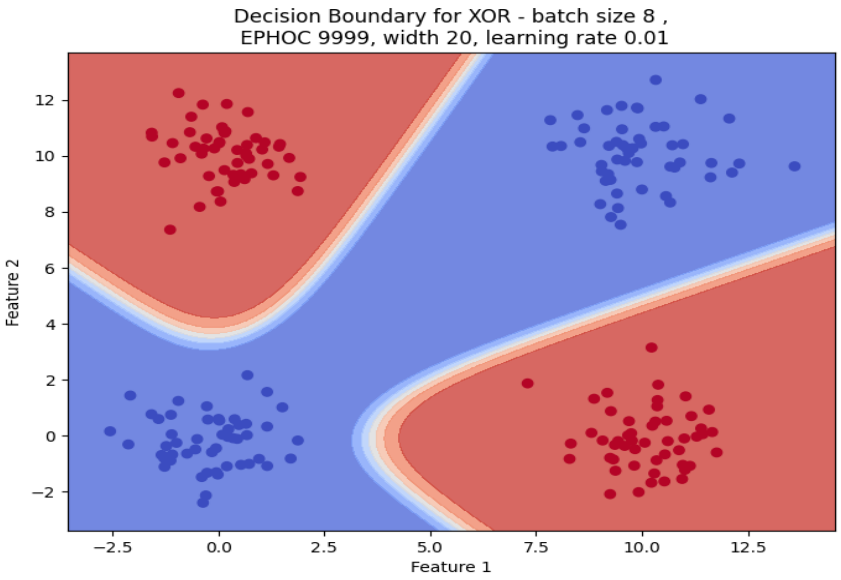
2.1 Task Description:

In this task, a multi-layer perceptron (MLP) model is implemented using the previously implemented Linear class, it includes two hidden layers (depth = 2), with width of 20 sigmoid activation function. The model is trained and tested on the neurons, and uses the XOR dataset generated in Task 1.

2.1 Results:

The multilayer perceptron (MLP) was successfully implemented and tested, achieving an accuracy of 95-100%.

as shown in Figure 2. We conducted several tests in which we tested the model with different learning rates and methods to determinist performance and how it handles several mathematical equations in several models.



2.3 implementation:

GitHub commit for task 1&2.

Task 3: Code Refactoring and Automation:

3.1 Description:

In this task the code was refactored for better maintainability and to insure scalability.

3.2 Results:

We have successfully completed the creation of parameters, graphs, and trainable models, and implemented a recursive function using the idea of topological sorting. As shown in the implementation.

3.3 implementation:

GitHub commit for task 1&2.

Task 4: Handwritten Digit Classification using MNIST Dataset:

4.1 Description:

In this task, a simple MNIST dataset of handwritten digits was used to train and test a model. A neural network was developed to recognize the handwritten digits.

4.2 Results:

The model was successfully trained with stable computations, achieving an accuracy of over 96% after 50 epochs. The confusion matrix in Figure 3 presents detailed results of the model's performance.

4.3 implementation:

GitHub commit for task 4.