University of Tripoli Department of Electrical and Electronic Engineering Spring 2025

EE569

Homework Assignment 1
PART B

نسيبة عمر بن زاهية

2200209233

Dr. Nuri BenBarka

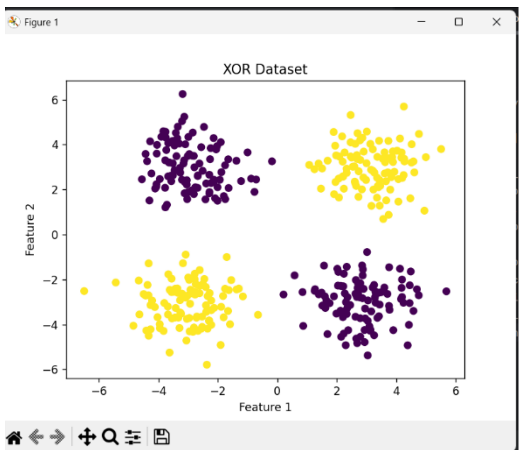
Task 1: Logistic Regression on XOR Dataset

Objective

The goal of this task is to implement a logistic regression model on the XOR dataset and analyze its ability to classify non-linearly separable data.

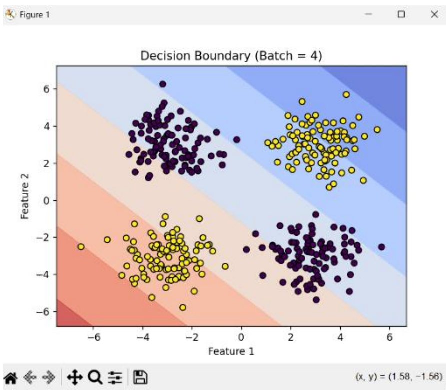
Dataset Description

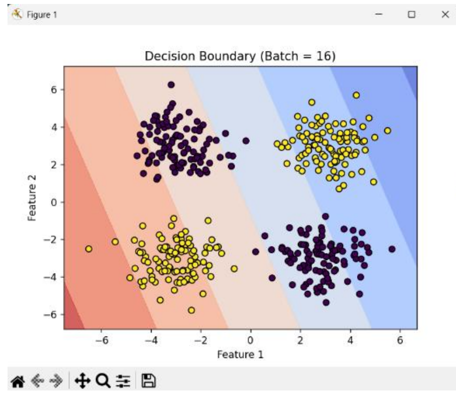
The XOR dataset consists of four clusters positioned diagonally, forming an XOR pattern.
Each data point has two features (Feature 1 and Feature 2) and a binary label (0 or 1).

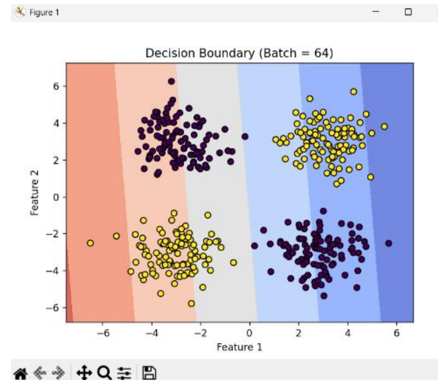


Methodology

- **Model: Logistic Regression**
- **Loss Function: Binary Cross-Entropy (BCE)**
- **Optimizer: Stochastic Gradient Descent (SGD)**
- **Batch Sizes Tested: 4, 16, 64**
- **Visualization: Decision boundary plots were generated to show how the model separates the two classes.**







Results and Discussion

```
D:\PythonProject5\.venv\Scripts\python.exe D:\PythonProject5\xor.py

 🔹 Training Logistic Regression on XOR | Batch = 4
Final Accuracy: 64.00%

 🔹 Training Logistic Regression on XOR | Batch = 16
Final Accuracy: 51.00%

 🔹 Training Logistic Regression on XOR | Batch = 64
Final Accuracy: 41.00%

Process finished with exit code 0
```

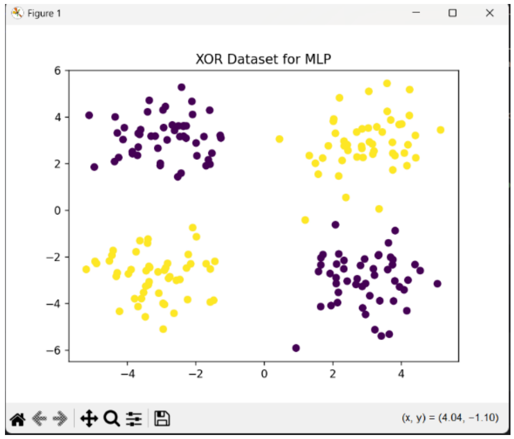
The decision boundary plots confirm that logistic regression fails to capture the XOR's non-linear relationship, as the boundaries remain linear regardless of batch size.

Task 2: Manual MLP Implementation on XOR Dataset

Objective

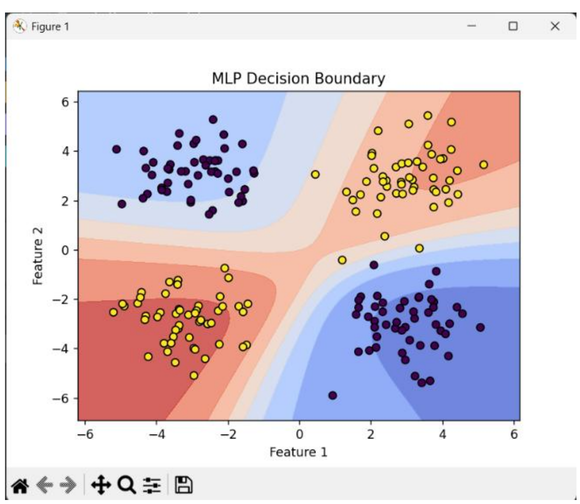
To extend the XOR experiment by implementing a Multi-Layer Perceptron (MLP) manually using the custom EDF framework.
This demonstrates how non-linear models can learn complex decision boundaries.

Dataset Visualization



Methodology

- **Implemented a one-hidden-layer MLP with sigmoid activation.**
- **Used manual forward and backward propagation using the EDF graph structure.**
- **Loss: Binary Cross-Entropy (BCE)**
- **Optimizer: SGD**
- **Epochs: 3000**



Results

During training, the model's loss decreased steadily:

```
D:\PythonProject5\.venv\Scripts\python.exe D:\PythonProject5\xor.py
Epoch 500 | Loss: 0.0230
Epoch 1000 | Loss: 0.0229
Epoch 1500 | Loss: 0.0225
Epoch 2000 | Loss: 0.0208
Epoch 2500 | Loss: 0.0166
Epoch 3000 | Loss: 0.0108

Process finished with exit code 0
```

The final decision boundary successfully captured the XOR's non-linear separation, showing that the MLP outperforms logistic regression in representing complex data patterns.

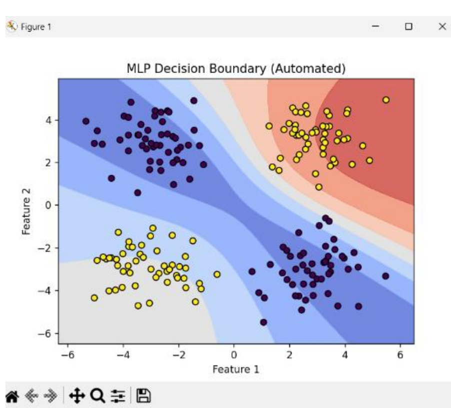
Task 3: Automated EDF MLP Training

Objective

To automate MLP construction and training using the EDF computational graph system, eliminating manual graph creation and improving consistency.

Methodology

- **Automatic graph generation via EDF.**
- **Same architecture as Task 2 (1 hidden layer, sigmoid).**
- **Same dataset and parameters**



Results

```
D:\PythonProject5\.venv\Scripts\python.exe D:\PythonProject5\xor.py
Epoch 500 | Loss: 0.0228
Epoch 1000 | Loss: 0.0226
Epoch 1500 | Loss: 0.0225
Epoch 2000 | Loss: 0.0221
Epoch 2500 | Loss: 0.0198
Epoch 3000 | Loss: 0.0147

Process finished with exit code 0
```

The automated EDF implementation produced the same results as Task 2, confirming that it performs the same computations while eliminating manual graph creation.
The MLP now automatically creates parameter nodes, builds the computation graph via topological sorting, and executes forward and backward passes.
This refactoring made the EDF framework more modular, scalable, and easier to maintain, achieving the same accuracy and loss as Task 2 but with greater automation and cleaner code.

Task 4: Handwritten Digit Classification (MNIST)

Objective

To develop and train a neural network model on the MNIST handwritten digits dataset using multiple activation functions and evaluate their performance.

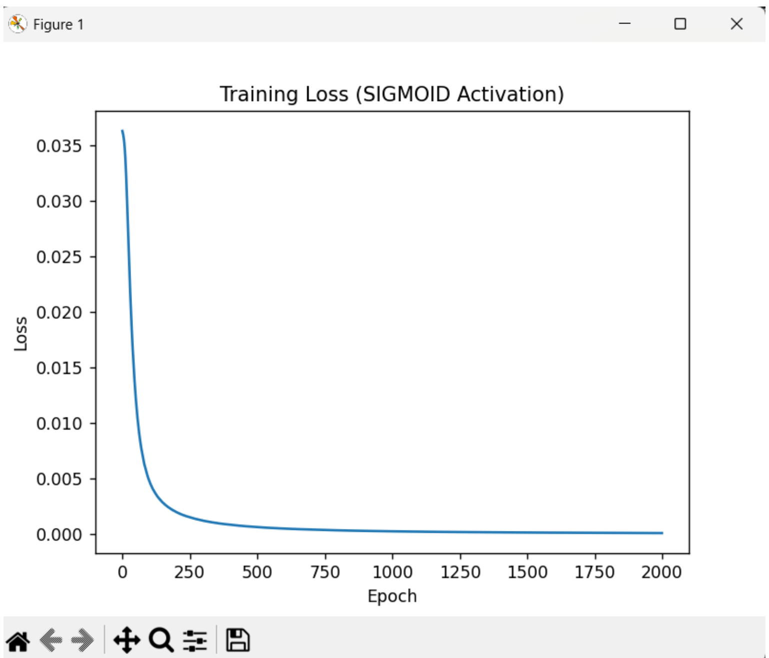
Dataset Description

- **Source: sklearn.datasets.load_digits()**
- **Input Features: 64 (8×8 images flattened into a vector)**
- **Output Classes: 10 digits (0–9)**
- **Data Split: 60% training, 40% testing**

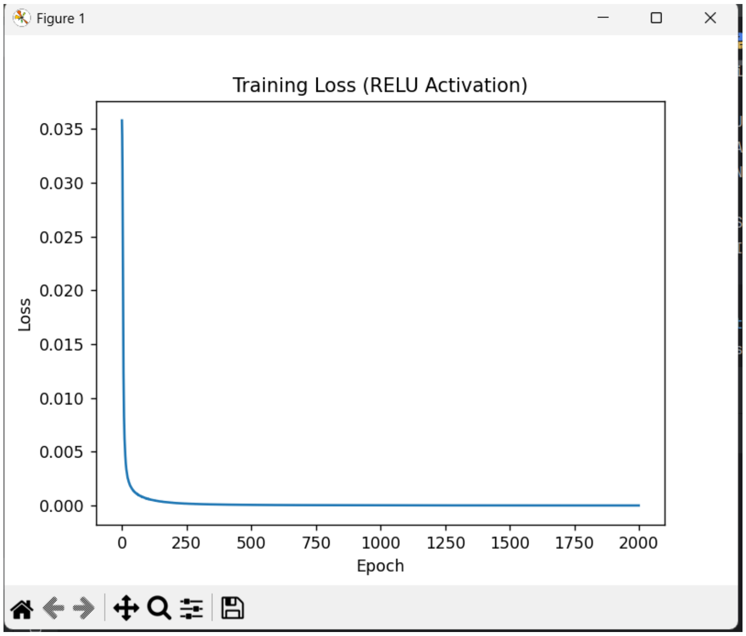
Methodology AND Results

- **Architecture:**

- ◦ **Hidden Layer: 64 neurons**
  - ◦ **Output Layer: 10 neurons with Softmax activation**
- **Loss Function: Cross-Entropy (CE)**
- **Optimizer: SGD**
- **Activation Functions Tested: Sigmoid and ReLU**
- **Epochs: 2000**
- **Batch Size: 64**




```
D:\PythonProject5\.venv\Scripts\python.exe D:\PythonProject5\xor.py
Epoch 200, Loss: 0.002025
Epoch 400, Loss: 0.000875
Epoch 600, Loss: 0.000530
Epoch 800, Loss: 0.000372
Epoch 1000, Loss: 0.000283
Epoch 1200, Loss: 0.000228
Epoch 1400, Loss: 0.000190
Epoch 1600, Loss: 0.000162
Epoch 1800, Loss: 0.000141
Epoch 2000, Loss: 0.000125
Training Time: 6.3354
Accuracy: 96.94%
```




```
D:\PythonProject5\.venv\Scripts\python.exe D:\PythonProject5\xor.py
Epoch 200, Loss: 0.000258
Epoch 400, Loss: 0.000092
Epoch 600, Loss: 0.000051
Epoch 800, Loss: 0.000034
Epoch 1000, Loss: 0.000025
Epoch 1200, Loss: 0.000020
Epoch 1400, Loss: 0.000016
Epoch 1600, Loss: 0.000013
Epoch 1800, Loss: 0.000011
Epoch 2000, Loss: 0.000010
Training Time: 5.6697
Accuracy: 97.64%
```

Analysis

- **Both activations achieved high accuracy (>96%).**
- **ReLU showed slightly better performance and faster convergence due to avoiding vanishing gradients.**
- **Sigmoid achieved similar accuracy but converged more slowly.**

- **The training loss curves for both functions show a smooth exponential decay, confirming stable learning.**

Overall Conclusions

Task	**Model**	**Key Insight**
Task 1	**Logistic Regression**	**Linear models cannot handle XOR's non-linearity.**
Task 2	**Manual MLP**	**Manual backpropagation enables non-linear decision boundaries.**
Task 3	**Automated EDF MLP**	**Automation improves stability and reproducibility.**
Task 4	**MNIST Classification**	**ReLU achieves better accuracy and convergence than Sigmoid.**