Heaven's Light is Our Guide

RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

Computer Science & Engineering

Course No: CSE 4204

Sessional based on CSE 4203

Name of the Experiment

Design and implementation of Multi-layer Neural Networks algorithm (i.e., Back-propagation learning neural networks algorithm)

Submitted by

Tushar Das
Roll: 1803108
Dept. of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Submitted to

Rizoan Toufiq
Assistant Professor
Dept. of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Contents

1 Multi-layer Perceptron Algorithm

1.1 Initialization

```
Initialize weights:
$W^{(1)} \leftarrow$ random values
$W^{(2)} \leftarrow$ random values
Set learning rate:
lr $\leftarrow$ chosen value
```

1.2 Training

```
For each epoch:
  For each training sample $(X, T_p)$ :
    **Forward Pass:**   $H_{\text{in}}^{(1)} = X \cdot W^{(1)}$
                        $H_{\text{out}}^{(1)} = \sigma(H_{\text{in}}^{(1)})$
                        $O_{\text{in}} = H_{\text{out}}^{(1)} \cdot W^{(2)}$
                        $O_{\text{out}} = \sigma(O_{\text{in}})$

    **Backward Pass:**  $\text{Ep} = T_p - O_{\text{out}}$
                        $\text{Del}_O = \text{Ep} \cdot \sigma'(O_{\text{out}})$
                        $\text{H\_err} = \text{Del}_O \cdot (W^{(2)})^T$
                        $\text{Del}_H = \text{H\_err} \cdot \sigma'(H_{\text{out}}^{(1)})$

    **Weight Update:**  $W^{(2)} \leftarrow W^{(2)} + \text{lr} \cdot H_{\text{out}}^{(1)} \cdot \text{Del}_O$
                        $W^{(1)} \leftarrow W^{(1)} + \text{lr} \cdot X \cdot \text{Del}_H$
```

2 The Abalone Dataset

The Abalone dataset [2] is a widely used dataset in machine learning and statistics. It contains measurements and other attributes of abalone, a type of marine mollusk. The dataset has the following characteristics:

- **Features:** The dataset includes 8 physical and geometrical measurements such as Length, Diameter, Height, Whole Weight, Shucked Weight, Viscera Weight, and Shell Weight. Additionally, there is a Sex attribute representing the gender of the abalones.
- **Target Variable:** The target variable is 'Rings,' which is a numerical value indicating the age of the abalones. This variable is crucial for age prediction.
- **Size:** The dataset consists of a total of 4,177 instances.
- **Data Types:** The dataset contains a mixture of numerical and categorical data types, making it suitable for various data preprocessing techniques.

2.1 Exploratory Data Analysis & Preprocessing

- **Data Overview:** Loaded the Abalone dataset using Pandas and displayed the first few rows to get an initial sense of the data structure. Computed summary statistics (mean, standard deviation, etc.) for numeric features using 'describe()' to gain insights into the central tendency and spread of the data.

- **Correlation Analysis:** Created a correlation matrix and visualized it using a heatmap to understand the relationships between different features. This helps identify potential multicollinearity and the impact of features on the target variable.

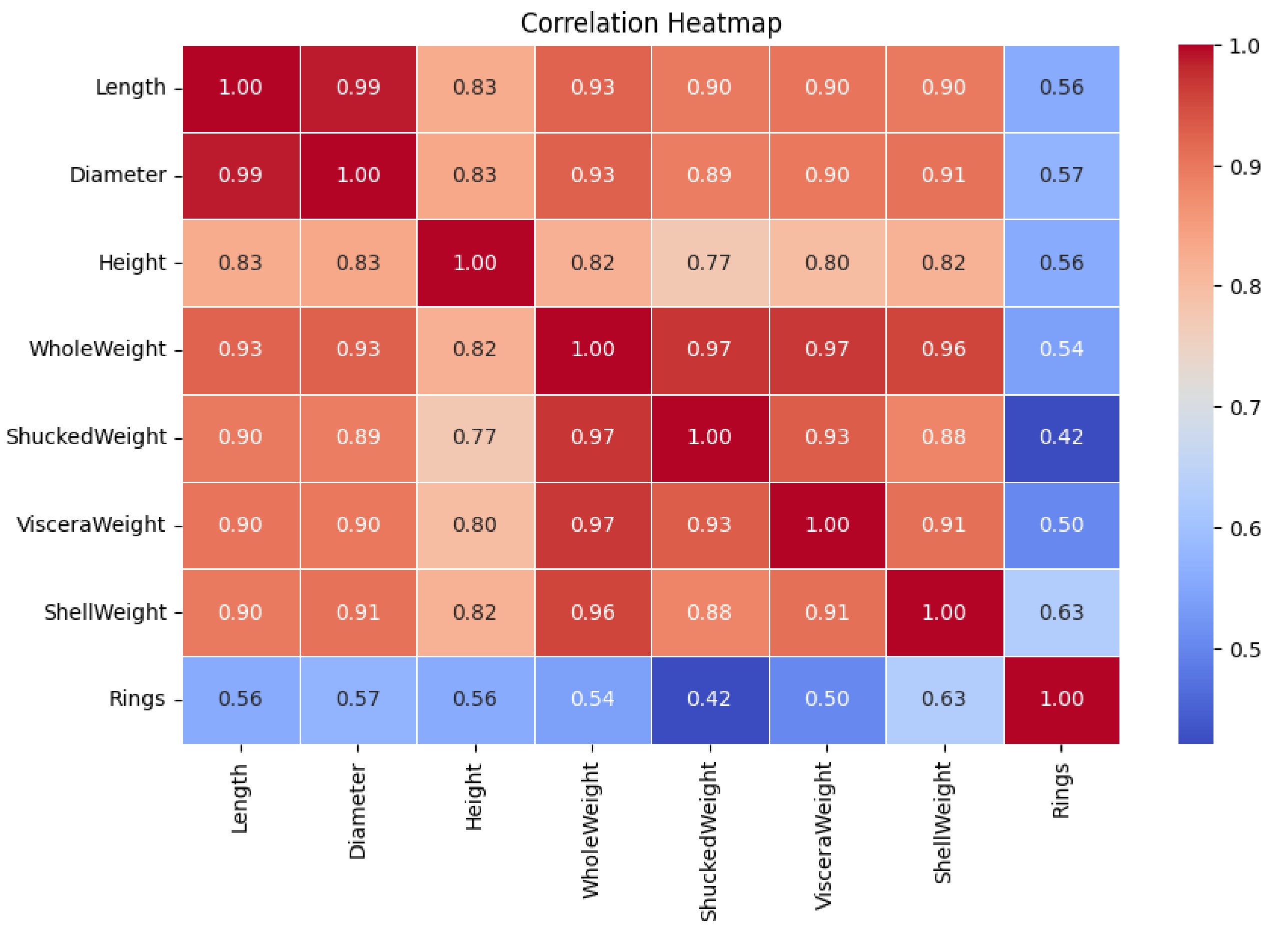

Figure 1: Correlation Heatmap of Features

- **Class Removal & Histogram Analysis:** Plotted a histogram of the 'Rings' feature to understand the distribution and frequency of different ring values. This is especially important for understanding the distribution of the target variable. Kept only the top 5 classes in the 'Rings' feature. This is often done to simplify the classification task and focus on the most prevalent classes.

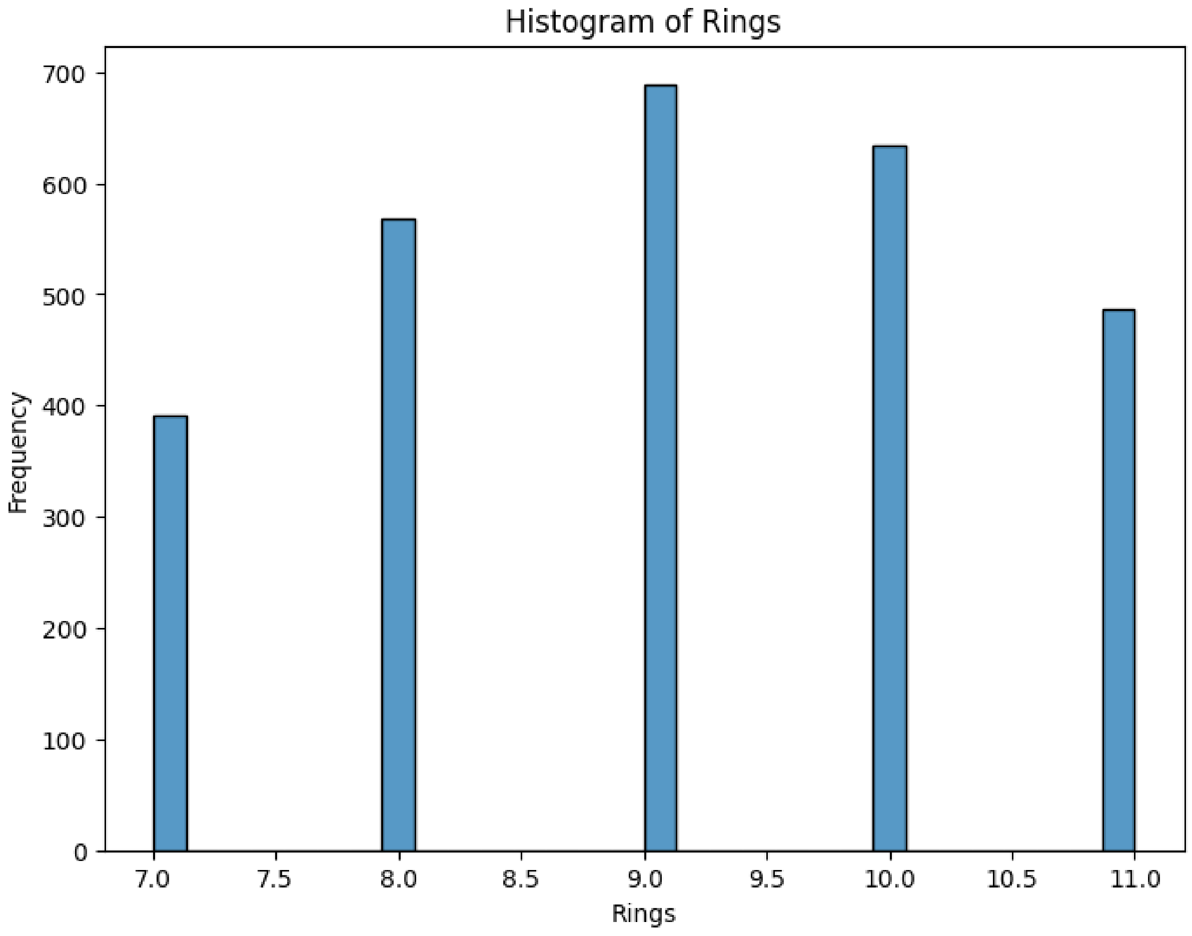

Figure 2: Histogram of the Top 5 Classes

- **Encoding Categorical Features:** Encoded the 'Sex' column to numeric values using Label Encoding. Normalized the values to a range between 0 and 1.
- **Standardization:** Standardization is a preprocessing step that transforms the data so that it has a mean of 0 and a standard deviation of 1. This is important when features in the dataset have different scales, as it helps machine learning algorithms converge faster and makes them less sensitive to the scale of input features.
- **Feature Selection:** Removed unnecessary columns ('Diameter', 'ShuckedWeight') from the dataset, as correlation of these columns are high with 'Height' , and 'Weight'. Feature selection is often performed to improve model performance and reduce complexity.

2.2 Training and Test Dataset Ratio

- **One-Hot Encoding:** This is important for neural networks, especially when dealing with classification problems, as it helps the model understand and interpret categorical data. Neural networks generally work well with numerical data, and one-hot encoding is a way to represent categorical information in a format suitable for training. Each class gets its own binary column, and the network can learn the relationships between these classes during training. In case of Abalone [2] Dataset, the reduced class number is 5.

So, there are 5 category classes in the ‘Ring’ column. This was one hot encoded.

$$\text{One-Hot Encoded } y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- **Training & Testing Ratio:** For model development and evaluation, it is common to split the dataset into training and test sets. The recommended ratio is typically 80/20 or 70/30. In this analysis, we have divided the dataset into a training set and a test set with an 80/20 split. This means that 80% of the data is used for training the MLP classifier, while the remaining 20% is reserved for testing and evaluating the classifier’s performance.

3 Multilayer Perceptron Model

3.1 Sigmoid Activation Function

Used in the hidden and output layers. Converts the weighted sum of inputs into values between 0 and 1, suitable for binary classification problems.

3.2 Weight Initialization

Randomly initialized weights for connections between input-hidden and hidden-output layers. Adjusted during training for improved model performance.

3.3 Forward Pass

- Calculates H_{in} with the dot product of inputs and weights ($X \cdot W_1$).
- Applies the sigmoid activation function to obtain H_{out}.
- Calculates O_{in} with the dot product of H_{out} and weights ($H_{\text{out}} \cdot W_2$).
- Applies sigmoid activation to obtain O_{out}.

3.4 Backward Pass

- Calculates E_p as the difference between true target T_p and predicted output O_{out}.
- Calculates Δ_{O} using the sigmoid derivative.
- Propagates error back to hidden layer (E_{hidden}) using transposed weights (W_2^T).
- Calculates Δ_{hidden} using sigmoid derivative.
- Updates weights (W_1 and W_2) using calculated deltas and learning rate.

3.5 Training

- Iterates through training data for a specified number of epochs (75).
- Learning rate was 0.0005
- Calls forward and backward pass functions to update weights and minimize error.
- Optionally records loss at regular intervals.

3.6 Accuracy & Loss Calculation

3.6.1 Accuracy Calculation

The accuracy is calculated by comparing the predicted labels to the true labels on a separate test dataset. For the current model, the accuracy is approximately 31%.

3.6.2 Loss Calculation

The loss is measured using Mean Squared Error, quantifying the difference between predicted and true outputs. In this case, the loss is around 20%.

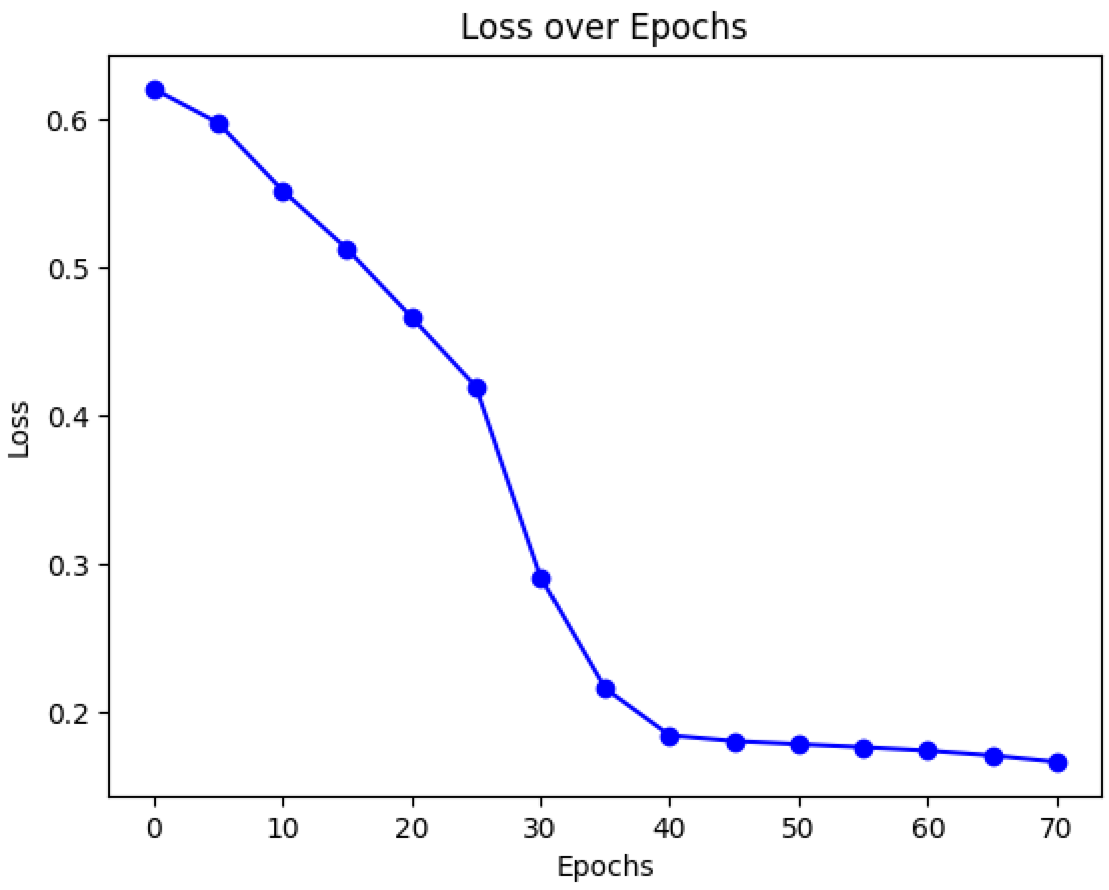

Figure 3: Loss vs Epochs

$$\text{Loss} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where n is the number of samples, y_i is the true label, and $\hat{y}_i$ is the predicted label for the i-th sample.

3.7 Model Evaluation

The observed accuracy of 31% and loss of 20% may indicate that the model is not performing optimally. This combination of metrics suggests the model may be overfitting or experiencing difficulty in learning complex patterns. This is because there was no hyperparameter tuning was experimented. This also can occur due to the dataset.

4 MLP for XOR Problem

The XOR problem is a classic example that showcases the limitations of a single-layer perceptron. A single-layer perceptron cannot learn the XOR function due to its linear nature. To overcome this limitation, MLP can be used. The above MLP model is applied to solve the Xor Problem and got an accuracy of 100%.

4.1 Problem Overview

The XOR problem involves binary inputs (0 or 1) and a binary output. The task is to learn a mapping from the input pairs to the correct output. The XOR function outputs 1 only when the inputs are different ($0 \oplus 1$ or $1 \oplus 0$).

4.2 Truth Table

X_1	X_2	XOR Output
0	0	0
0	1	1
1	0	1
1	1	0

4.3 Training Results

After training the model for a sufficient number of epochs (10000), the learning rate is 0.1. it is observed that the MLP successfully learns the XOR function with an accuracy of 100%. The trained model can accurately predict the XOR output for new input pairs.

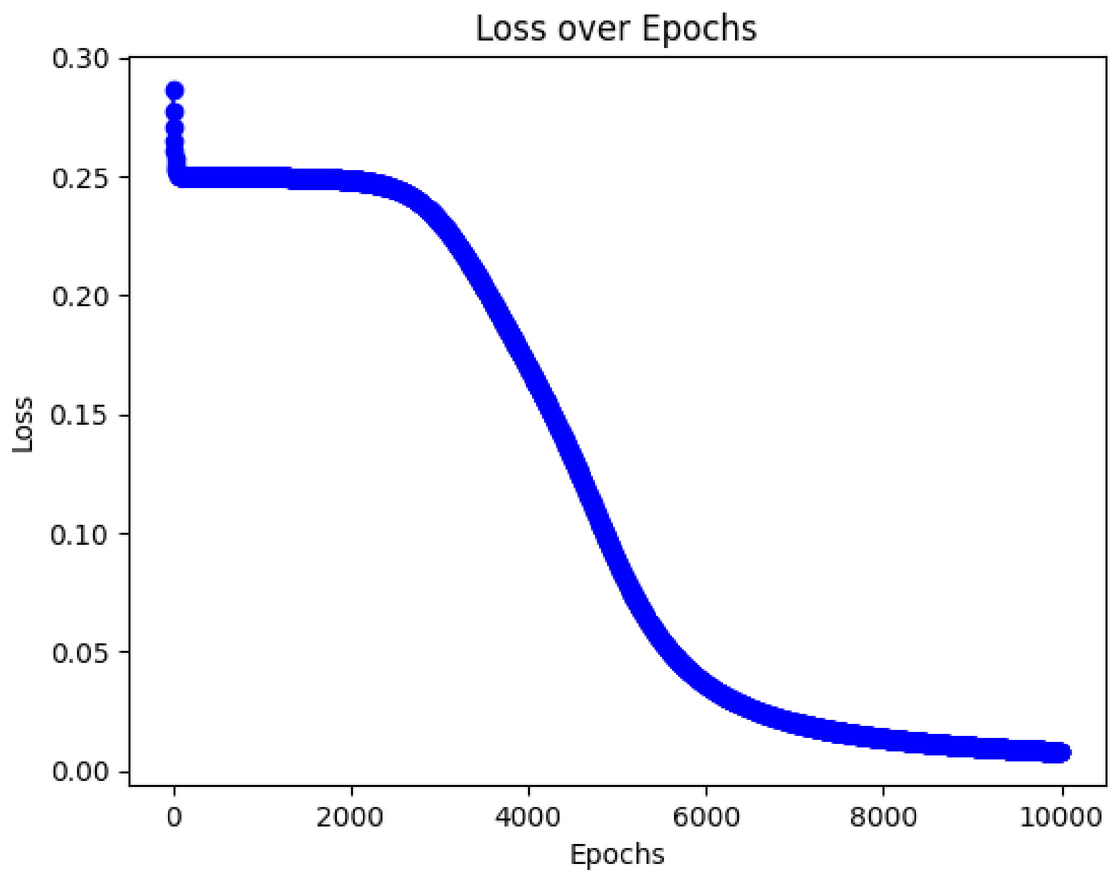

Figure 4: Loss vs Epochs

Output

```
Input: [0 0], Target: 0, Predicted Output: 0.11364221229949091
Input: [0 1], Target: 1, Predicted Output: 0.9168284304708115
Input: [1 0], Target: 1, Predicted Output: 0.9103556972228541
Input: [1 1], Target: 0, Predicted Output: 0.06726768343497751
Accuracy: 100.0%
```

4.4 Visualizing The Decision Boundary

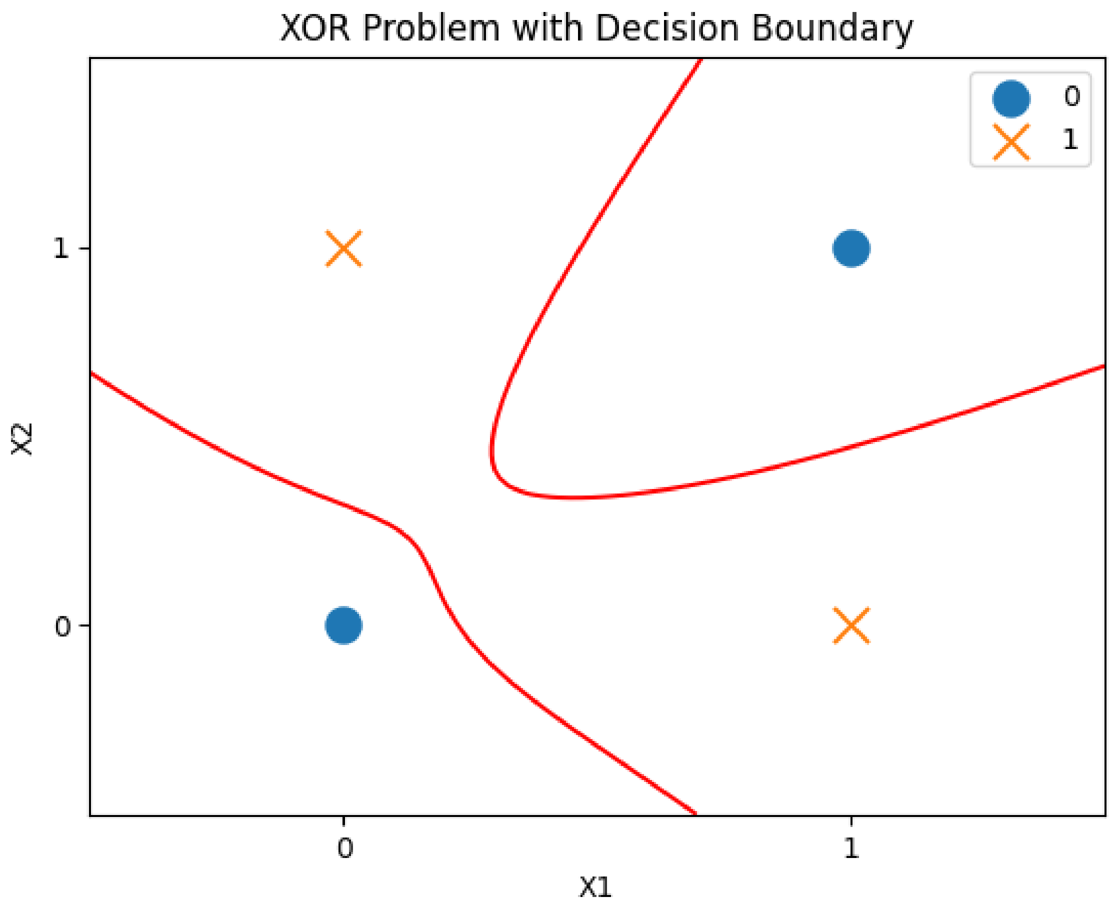

Figure 5: Xor Problem with Decision Boundary

5 Conclusion

- Explored dataset using EDA techniques to understand its structure and relationships.
- Developed a Multi-Layer Neural Network (MLP) [3] with appropriate architecture.
- Trained the MLP model using specified parameters (epochs, learning rate) on the dataset. Accuracy was 31%.
- Successfully solved the XOR problem, achieving a 100% accuracy rate.

References

[1] Source Code “https://shorturl.at/vwGO5”

[2] Abalone Dataset “https://rb.gy/2ov6cr”

[3] Neural Computing: An Introduction - R Beale and T Jackson