Title: Implementation of a Three Hidden Layer Neural Network for Multi Class Classification

1. Introduction

Neural Networks are powerful machine learning models that are widely used for pattern recognition and classification tasks. While binary classification problems involve only two output classes, many real world applications require classification among multiple categories. This assignment focuses on implementing a Three Hidden Layer Neural Network for Multi Class Classification.

The provided base file (M_1C_neural_network.ipynb) originally supported only binary classification. The main objective of this assignment is to modify the original implementation to support five class classification using appropriate architectural changes, activation functions, loss functions, and evaluation strategies.

The complete system is trained and evaluated on a synthetically generated dataset consisting of five distinct classes.

2. Objectives of the Assignment

The main objectives of this assignment are:

i. To generate a synthetic multi class dataset with five classes
ii. To design and implement a neural network with three hidden layers
iii. To modify the original binary classifier to support multi class classification
iv. To apply Softmax activation and Categorical Cross Entropy loss
v. To train and test the neural network model
vi. To evaluate performance using Accuracy, Precision, Recall, and F1 Score
vii. To analyze the results and discuss observations

3. Dataset Generation

A **synthetic dataset** was created to simulate a real world multi class classification problem. The dataset contains numerical input features and five distinct output classes.

Dataset Properties:

- Number of input features: Multiple numerical features
- Number of classes: 5
- Class labels: 0, 1, 2, 3, 4
- Data distribution: Balanced among all classes

Generation Process:

1. Random numerical values were generated using NumPy.
2. Target labels were assigned based on predefined class conditions.
3. The entire dataset was shuffled randomly.
4. The dataset was divided into:
    - **Training set (80%)**
    - **Testing set (20%)**

This process ensured fair learning and unbiased performance evaluation.

4. Neural Network Architecture

The neural network used in this assignment contains:

Layer	Description
Input Layer	Accepts feature values
Hidden Layer 1	Fully connected
Hidden Layer 2	Fully connected
Hidden Layer 3	Fully connected
Output Layer	5 neurons (one for each class)

Activation Functions Used

- **Hidden Layers:** ReLU (Rectified Linear Unit)
- **Output Layer:** Softmax

Why Softmax?

Softmax converts raw output values into probability scores whose sum equals 1. This makes it ideal for multi class classification.

5. Code Modifications for Multi Class Classification

Several important changes were made to convert the original binary classification model into a multi class classifier.

a. Output Layer Modification

- Original model used 1 output neuron for binary classification.
- Modified model now uses 5 output neurons, one for each class.

This enables the network to predict five different class labels.

b. Activation Function Change

- **Before:** Sigmoid function (for binary output)
- **After:** Softmax function (for multi class output)

c. Loss Function Change

- **Before:** Binary Cross Entropy Loss
- **After:** Categorical Cross Entropy Loss

d. One Hot Encoding

Class labels were transformed into one hot encoded vectors:

- Class 0 → [1, 0, 0, 0, 0]
- Class 1 → [0, 1, 0, 0, 0]
- Class 2 → [0, 0, 1, 0, 0]
- Class 3 → [0, 0, 0, 1, 0]
- Class 4 → [0, 0, 0, 0, 1]

This helps the network compute correct error values during backpropagation.

e. Backpropagation Modification

Backpropagation was updated to handle:

- Multiple output neurons
- Probability based error calculations
- Gradient updates across all five output classes

Weight update rule used:

$$W = W - \alpha \times \nabla W$$

Where:

- W = weight
- α = learning rate
- ∇W = gradient

6. Training and Testing Process

i. Training

During training:

1. Forward propagation computes outputs.
2. Softmax converts outputs into probabilities.
3. Categorical Cross Entropy calculates loss.
4. Backpropagation updates the weights.
5. This process repeats for multiple epochs.

Training continued until the loss stabilized and accuracy improved.

ii. Testing

During testing:

- Model outputs class probabilities
- The predicted class is selected using:

Predicted class = argmax($\hat{y}$)

Test data ensures that the model has not simply memorized the training samples.

7. Performance Evaluation Metrics

The following metrics were computed:

- **Accuracy**
- **Precision**
- **Recall**
- **F1 Score**

Formulas Used

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{F1} = \frac{2\ X\ \text{Precision X Recall}}{\text{Precision} + \text{Recall}}$$

8. Results and Performance Analysis

a. Overall Performance

- The neural network achieved **high classification accuracy**
- Training loss decreased steadily with epochs
- Test accuracy confirmed good generalization

b. Confusion Matrix Analysis

- Most predicted values appeared on the main diagonal
- This indicates correct classification for most samples
- Very few misclassifications were observed

c. ROC Curve Analysis

- Multi class ROC curves were generated using a one vs rest approach
- High Area Under Curve (AUC) values indicate strong class separation

d. Effect of Hyperparameters

Configuration	Behavior
Small hidden layers	Underfitting
Very large hidden layers	Overfitting
Moderate hidden layers	Best performance

Learning rate tuning also improved stability and convergence speed

9. Challenges Faced and Solutions

Problem	Solution
Binary model incompatible with multi class	Output layer expanded to 5 neurons
Sigmoid not suitable	Replaced with Softmax
Loss mismatch	Implemented Categorical Cross Entropy
Training instability	Learning rate optimized
Class confusion	Balanced dataset used

10. Conclusion

This assignment successfully demonstrates the implementation of a Three Hidden Layer Neural Network for Multi Class Classification. The original binary classifier was fully transformed into a five class classifier using Softmax activation and Categorical Cross Entropy loss.

The model achieved strong performance with high training and testing accuracy. The project enhanced understanding of:

- Multi layer neural network architecture
- Forward and backward propagation
- Multi class probability modeling
- Performance evaluation using classification metrics