Assignment: Building and Evaluating a Fully Connected Neural Network

Objective

The goal of this assignment is to develop proficiency in constructing, training, and evaluating a fully connected neural network (also known as a multilayer perceptron) for a classification task. You will:

- Select an appropriate dataset.
- Preprocess the data.
- Design and train a neural network model.
- Visualize the training process and results.
- Evaluate the model using metrics like the confusion matrix and other relevant evaluation metrics.

Instructions

1. Dataset Selection

- Choose of the the following datasets:
    - **MNIST Handwritten Digits**: Classify images of handwritten digits (0-9).
    - **Iris Dataset**: Classify iris plants into three species based on flower measurements.
    - **CIFAR-10**: Classify images into 10 different categories like airplanes, cars, birds, etc.

2. Data Preprocessing

- **Load the Data**: Import the dataset using appropriate libraries (e.g., pandas, NumPy).
- Clean the Data:

    - Handle missing values (if any).
    - Remove or correct outliers.
- Feature Engineering:
    - Encode categorical variables using one-hot encoding or label encoding.
    - Normalize or standardize numerical features to improve model performance.
- Split the Data:
    - Divide the dataset into training and testing sets (e.g., 80% training, 20% testing).

3. Model Design and Training

- Build the Neural Network:
    - Use PyTorch.
    - Design a fully connected neural network architecture:
        - Decide the number of hidden layers and neurons.
        - Choose activation functions (e.g., ReLU, sigmoid, softmax).
- Compile the Model:
    - Select an appropriate loss function (e.g., categorical cross-entropy for multi-class classification).
    - Choose an optimizer (e.g., Adam, SGD).
    - Specify metrics to monitor (e.g., accuracy).
- Train the Model:
    - Set hyperparameters like batch size and number of epochs.
    - Include validation during training if desired.

4. Visualization

- Plot Training Metrics:
    - Visualize training and validation loss over epochs.
    - Visualize training and validation accuracy over epochs.

5. Model Evaluation

- Performance Metrics:

- **Confusion Matrix**: Create a confusion matrix to evaluate classification performance across classes.
- **Accuracy**: Measure the overall correctness.
- **Precision, Recall, F1-Score**: Compute these metrics for a more detailed performance analysis.
- **ROC Curve and AUC** (if applicable): Plot and interpret the Receiver Operating Characteristic curve.