Lab 15

Multi-class Classification using PyTorch and the Iris Dataset

Objective:
The goal of this lab is to gain hands-on experience in designing, training, and evaluating a neural network for multi-class classification using PyTorch. You will use the classic Iris dataset, which contains measurements of iris flowers from three species. Your task is to build a feedforward neural network that can correctly classify a given flower into its corresponding species based on four numerical features.

Problem Statement:
Design and implement a neural network using PyTorch that classifies instances from the Iris dataset into one of three classes: *Iris Setosa*, *Iris Versicolor*, or *Iris Virginica*. The dataset includes 150 samples, each with four input features: sepal length, sepal width, petal length, and petal width.

You are required to:

1. **Load and Prepare the Data**
    - Use `sklearn.datasets.load_iris()` to load the dataset.
    - Split the dataset into training and testing sets (e.g., 80% train, 20% test).
    - Normalize or standardize features if needed.
2. **Define the Neural Network Architecture**
    - Use PyTorch's `nn.Module` to define the model.
    - The input layer must accept 4 features.
    - Include at least one hidden layer (you may use more).
    - The output layer should have 3 neurons (one for each class).

    - Use ReLU activation for hidden layers and no activation for the final layer (as `CrossEntropyLoss` expects raw logits).

3. **Train the Model**
    - Use `CrossEntropyLoss` as the loss function.
    - Choose an appropriate optimizer (e.g., `torch.optim.Adam`) and learning rate.
    - Train the model over multiple epochs (e.g., 100) and print loss per epoch.

4. **Evaluate the Model**
    - Use the trained model to predict on the test set.
    - Calculate and print the classification accuracy.
    - Display a confusion matrix to visualize the performance.
    - Optionally, show per-class precision, recall, and F1-score.