**Question 1.**

Which loss function, out of Cross Entropy and Mean Squared Error, works best with logistic regression because it guarantees a single best answer (no room for confusion)? Explain why this is important and maybe even show how it affects the model's training process.

**Ans.**

In logistic regression, the Cross Entropy loss function is the preferred choice over Mean Squared Error (MSE). This is because logistic regression is primarily used for binary classification tasks where the goal is to predict probabilities indicating class membership. The Cross Entropy loss, also known as Log Loss, is well-suited for this purpose as it measures the difference between predicted probabilities and true probabilities.

Using Cross Entropy loss ensures that the model learns to assign higher probabilities to correct predictions and lower probabilities to incorrect ones. It achieves this by penalizing confidently wrong predictions more heavily than uncertain ones, leading to a more calibrated model. In contrast, MSE treats logistic regression outputs as continuous values and may not optimize for probabilities directly, potentially resulting in suboptimal performance.

In practical applications like medical diagnosis or fraud detection, accurate probability estimates are crucial for informed decision-making. Therefore, using Cross Entropy loss helps ensure that logistic regression models provide well-calibrated probability estimates, enhancing their reliability and usefulness in real-world scenarios. -

**Question 2:**

For a binary classification task with a deep neural network (containing at least one hidden layer) equipped with linear activation functions, which of the following loss functions guarantees a convex optimization problem? Justify your answer with a formal proof or a clear argument. (a) CE (b) MSE (c) Both (A) and (B) (d) None

**Ans.**

In a Multilayer Perceptron (MLP) where all activation functions are linear, the network essentially reduces to a linear transformation of the input X. In such a scenario, the loss functions play a crucial role in determining the convexity of the optimization problem.

**Mean Squared Error (MSE):** For MSE, the loss function is given by:

$$L_{\text{MSE}} = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

The gradient of the MSE loss function with respect to the parameters is:

$$\nabla^2 L_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} \nabla^2 (y_i - \hat{y}_i)^2$$

Since $(y_i - \hat{y}_i)^2$ is a convex function of $\hat{y}_i$, and the sum of convex functions is convex, the MSE loss function is convex.

**Cross-Entropy Loss (CE):** For CE, the loss function is:

$$L_{\text{CE}} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

Here, $y_i$ factors as 1 only for the correct class index. Therefore, the function becomes:

$$f(x) = -\log(x)$$

Computing the second derivative:

$$\frac{\partial^2 L_{\text{CE}}}{\partial \hat{y}^2} = \frac{1}{\hat{y}^2} > 0$$

This confirms that the CE loss function is convex.

In conclusion, both Mean Squared Error (MSE) and Cross-Entropy Loss (CE) are convex for MLPs with linear activation functions, ensuring a convex optimization landscape.

Question 3.

Ans.

The neural network architecture is defined in the `FeedForwardNN` class, comprising fully connected (dense) layers . Users can customize the architecture by specifying the number of hidden layers and neurons per layer. The output layer, with 10 neurons representing the classes in the MNIST dataset, employs a softmax activation function.

Implementation:

1. Preprocessing: Image preprocessing involves a sequence of transformations using `torchvision.transforms.Compose`. This includes conversion to tensors and normalization using mean and standard deviation.
2. Dataset: The MNIST dataset is loaded using `torchvision.datasets.MNIST`, with the defined transformations applied.
3. Data Loaders: Data loaders for training and testing datasets are created using `torch.utils.data.DataLoader`.
4. Training Loop: Training the model involves a basic loop where batches of data are iterated over. For each batch, a forward pass is executed, followed by loss calculation, backward pass, and weight updates using the Adam optimizer.
5. Evaluation: Model evaluation on the test dataset is performed to ascertain its accuracy.