Q1.

1. **Interpretation**:

   - **Cross-Entropy**: Measures the disparity between the predicted probability distribution and the actual distribution (often one-hot encoded for classification). Lower Cross-Entropy signifies superior model performance as the predicted probabilities closely match the true labels.

   - **Mean Squared Error (MSE)**: Computes the squared discrepancy between predicted continuous values and actual target values. Ideal for regression tasks where prediction involves continuous values (e.g., house price). However, in classification tasks with discrete class labels (e.g., 0 or 1 in binary logistic regression), MSE lacks a direct translation to the model's confidence in class prediction.

2. **Output and Loss Landscape**:

   - **Logistic Regression Output**: Ranges between 0 and 1, representing class probabilities.

   - **Cross-Entropy Loss**: Tailored for the logistic regression output range, penalizing significant deviations from true probabilities, particularly at extremes (0 or 1). Encourages the model to converge towards distinct class boundaries, aiming for a singular optimal classification.

   - **Impact on Training**:

     - **Cross-Entropy Loss**: Drives model adjustment to minimize discrepancies between predicted probabilities and true labels. Promotes learning of effective class boundaries, facilitating clear classification decisions.
     - **Mean Squared Error (MSE)**: Inappropriate for logistic regression output, as it would not penalize small deviations from true labels, potentially resulting in ambiguous predictions around 0.5.

Conclusion:
Cross-Entropy loss is indispensable for logistic regression due to its alignment with model output (probabilities) and training objective (clear classification). By penalizing significant deviations from true labels, particularly at extreme probabilities, Cross-Entropy loss facilitates convergence towards a singular optimal classification, thus enhancing the model's ability to provide definitive predictions.

Q2.

The correct answer is (a) CE, Cross-Entropy loss. Cross-Entropy loss guarantees convex optimization due to its convex nature. While the proof of convexity can be complex, empirically, it's widely used and effective for binary classification tasks with deep neural networks. In contrast, Mean Squared Error (MSE) loss (option (b)) lacks convexity, often leading to multiple local minima. Especially with linear activation functions, MSE may not fully leverage the network's capabilities. Thus, CE loss ensures a convex optimization problem, making it the preferred choice for binary classification tasks with deep neural networks. Options (c) and (d) are incorrect based on this rationale.

Q3.

Code Explanation

This code provides a concise implementation of a neural network classifier for MNIST digit recognition using TensorFlow and Keras.

1. **Data Preparation**:
    - MNIST dataset is loaded and normalized.
2. **Model Architecture**:
    - Sequential model with a flatten layer followed by three dense layers.
    - The dense layers have 128, 64, and 32 units with ReLU, sigmoid, and tanh activations respectively.
    - The output layer has 10 units with softmax activation.
3. **Compilation**:
    - Model is compiled with Adam optimizer and sparse categorical cross-entropy loss.
4. **Training**:
    - Model is trained for 5 epochs on the training data.
5. **Evaluation**:
    - Model's accuracy is evaluated on the test data.

Q4.

Code Explanation

This code trains and evaluates various pre-trained models (AlexNet, VGG-11, and ResNet-18) on a subset of the Street View House Numbers (SVHN) dataset for digit recognition. Here's a breakdown:

Data Loading and Preprocessing:

- Defines transformations to resize images to 64x64 pixels and normalize pixel values.
- Loads the SVHN dataset, both training and test sets, and creates subsets containing 25

Model Initialization:

- Initializes three pre-trained models: AlexNet, VGG-11, and ResNet-18.
- Modifies the output layer of each model to have 10 units for digit classification.

Training:

- Trains each model for one epoch (due to computational constraints).
- Uses cross-entropy loss and stochastic gradient descent (SGD) optimizer with a higher learning rate for faster convergence.
- Prints the average loss per epoch during training.

Evaluation:

- Evaluates each trained model on the test set.
- Computes the accuracy of each model on the test set and prints the results.

The code demonstrates the process of fine-tuning pre-trained models for a specific task (digit recognition) on a subset of the SVHN dataset. It emphasizes the importance of transfer learning, enabling the use of pre-trained models to achieve decent accuracy even with limited training data.