**Problem Statement**

The aim of this task is to develop a feedforward neural network model along with backpropagation algorithm and apply gradient descent (and its variants) for a classification problem. Additionally, we will utilize wandb.ai to keep a record of our experiments and their results.

You can access the task instructions [here](https://wandb.ai/cs6910_2023/A1/reports/CS6910-Assignment-1--VmlldzozNTI2MDc5).

**Installation**

Install wandb:

Install Numpy:

Install Keras:

**Question 1**

Approch:

1. Read input data and created an empty dictionary to store class-wise data.
2. Iterated through the input data and populated the dictionary with images belonging to each class as representative.
3. Initialized an empty list to store the selected images from each class of data.
4. For each class in the dictionary, selected the first image and added it to the list of selected images.
5. Visualized the selected images using library Matplotlib for both Train data and Test data.
6. Integrated Wandb by initializing a new run, logging the selected images And pasted those wandb images into wandb report.

The code for question 1 is available at the Assignment1\_model.pynb file

**Question 2**

Approch:

1. For one-hot encoding of categorical variables class named "OneHotEncoder\_from\_scratch" is created and labels are converted into one hot matrix
2. Class Feedforward defined that implementes a feedforward neural network for classification tasks of images
3. The network is initialized with hyperparameters such as the number of epochs, the number of hidden layers, the size of each hidden layer, the learning rate, and the activation function.
4. The weights and biases of the network are initialized using either random or Xavier initialization.
5. The network has methods for computing the forward activation, gradient of the activation, and softmax function.
6. The forward\_pass method computes the output of the network given an input X and the current weights and biases. It prints outputs as probability distribution over the 10 classes.
7. You can see the code is enough flexible to change number of hidden layers and correspondingly repsectlively number of neurons in each hidden layers. Just changing values when calling class will give changes. Both number of hidden layers and neurons in each layer of hidden layers can be changed

The code for question 1 is available at the Assignment1\_model.pynb file

**Question 3**

Approch:

1. Addition to last question backpropogation algorithm added, that performs backpropagation to calculate gradients for each layer of the neural network.
2. The method first performs a forward pass using the input data and current weights. It then computes the derivative of the loss (For both Cross entropy and Mean square error) with respect to the final activation layer. Using this derivative, the method iteratively computes the derivatives for each previous layer, propagating backwards through the network.
3. L2 Regularisation is added in backprop only to not to overfit model
4. Finally, it returns the derivative of the loss with respect to the input layer activations.
5. The fit method takes input parameters such as X and Y training data, X\_val and Y\_val validation data, and several hyperparameters such as the optimization algorithm, learning rate, etc.
6. The method trains the neural network for the given number of epochs using a loop and prints the train loss, train accuracy, validation loss, and validation accuracy every five epochs.
7. The method implements four optimization algorithms (SGD, Momentum, RMSProp, Adam) and updates the weights and biases of the neural network in each epoch accordingly.
8. It prints the training and validation loss and accuracy at each epoch.

The code for question 1 is available at the Assignment1\_model.pynb file

**Question 7**

Approch:

1. The approach to finding the best machine learning model involves experimenting with different hyperparameters and configurations to find the best combination.
2. Once the best model is identified, its accuracy and other relevant metrics should be reported.
3. Implementation of a function to calculate confusion matrix done.
4. Parameter for best model:

