1 NN from scratch!

In this exercise the objective is implementation of a neural network from scratch in Python (You can only use NumPy library). A sample notebook is included and you must complete the empty parts of the notebook. To make a proper implementation, keep in mind the big picture here:

1. Feed input data into the neural network.
2. The data flows from layer to layer until you have the output.
3. Once you have the output, you can calculate the error which is a scalar.
4. Finally you can adjust a given parameter (weight or bias) by subtracting the derivative of the error with respect to the parameter itself.
5. Iterate through that process.

Notice that you need to be able to modify your final network to solve a problem, So you must implement each layer separately. Every layer, independent of its type, consists of input, output, forward propagation method and back propagation method. The base class of a layer defined in the sample notebook, you implement a Linear Fully Connected layer(FC layer) inherited from this class. After the linear part you need to add a non-linearity in your network, So, you must implement a Activation Layer inherited from Layer class and consists of a activation function and its derivative.
To calculate the error of your network you need to implement a loss function and its derivative. At the end you need to implement a Network class to create a neural network akin the first picture.

1.1 Regression

Use your implementation to create a neural network for regression task. The regression dataset consists of 3526 samples with 15 features, given as two files, 'x.csv' and 'y.csv'.

1.2 Classification

The same as previous part, use your implementation to create a neural network for classification task. The classification dataset consists of 9834 samples with 17 features and 6 classes, given as two files, 'features.csv' and 'labels.csv'.

1.3 Evaluation

Create the same neural networks of two previous parts using PyTorch library and compare the efficiency of them with your implementations.

2 Bank Marketing Campaign Subscriptions

Design and tune a neural network using PyTorch library to do the tasks below.

2.1 Tasks

- Build effective neural network that can predict the probability of a client subscribing to the bank's product.
- We should note that, even though we are talking about calculating probabilities, you will create classifier neural network - meaning that the final output of your models will be a binary result indicating if the client subscribed ('yes') to the product or not ('no'). Of course applying EDA on the dataset is highly welcomed.
- To evaluate your model, it's highly preferable to calculate the precision and recall score and besides the ROC AUC score.