CSE343/ECE343: Machine Learning
Assignment-3(Perceptron, MLP, SVM)
Max Marks: 25 (Programming: 15, Theory: 10) **Due Date**: 10/11/2024, 11:59 PM

Instructions

- Keep collaborations at high-level discussions. Copying/Plagiarism will be dealt with strictly.
- Late submission penalty: As per course policy.
- Your submission should be a single zip file **202xxxx_HW3.zip** (Where *2020xxx* is your roll number). Include **all the files (code and report with theory questions)** arranged with proper names. A single **.pdf report** explaining your codes with results, relevant graphs, visualization and solution to theory questions should be there. The structure of submission should follow:

  202xxxx_HW3
  |– code_rollno.py/.ipynb
  |– report_rollno.pdf
  |– (All other files for submission)
- Anything not in the report will **not** be graded.
- Remember to **turn in** after uploading on Google Classroom. No excuses or issues would be taken regarding this after the deadline.
- Start the assignment early. Resolve all your doubts from TAs in their office hours at least **two days before the deadline.**
- Your code should be neat and well-commented.
- **You have to do either Section B or C.**
- **Section A is mandatory.**

---

1. (10 points) **Section A (Theoretical)**

   (a) (3 points) Consider a regression problem where you have an MLP with one hidden layer (ReLU activation) and a linear output (refer to attached Fig 1). Train the network using mean squared error loss. Given a dataset with inputs [1, 2, 3] and corresponding targets [3, 4, 5], perform a single training iteration and update the weights. Assume appropriate initial weights and biases and a learning rate of 0.01.

   (b) (4 points) You have the following dataset:

   (a) Are the points linearly separable? Support your answer by plotting the points.

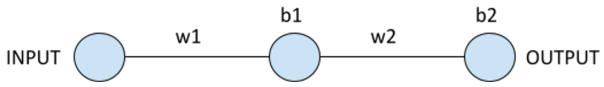

Figure 1: Figure for Question 1.a

Class	x_1	x_2	**Label**
+	0	0	+
+	1	0	+
+	0	1	+
-	1	1	-
-	2	2	-
-	2	0	-

Table 1: Figure for Question 1.b

(b) Find out the weight vector corresponding to the maximum margin hyperplane. Also find the support vectors present.

(c) (3 points) Consider the dataset with features x_1 and x_2 and 2 classes $y = -1$ and $y = +1$.
Training Dataset:

Sample No.	x_1	x_2	y
1	1	2	+1
2	2	3	+1
3	3	3	-1
4	4	1	-1

Table 2: Table for Question 1.c

The SVM formulates a decision boundary of format: w.x + b = 0 Given the values: w1 = 1, w2 = -1, b = 0 Solve the following parts:

(a) Calculate the margin of the classifier.

(b) Identify the support vectors. (given samples 1 to 4).

(c) Predict the class of a new point : x1=1, x2=3.

2. (15 points) **Section B (Scratch Implementation)**

1. (5.5 points) Implement a class named **NeuralNetwork** with the following parameters during initialization:

(a) N: Number of layers in the network.

(b) A list of size N specifying the number of neurons in each layer.

(c) lr: Learning rate

(d) Activation function (same activation function is used in all layers of the network except the last layer).

(e) Weight initialization function.
(f) Number of epochs.
(g) Batch size.

The NeuralNetwork class should also implement the following functions:

(a) fit(X, Y): trains a model on input data X and labels Y.
(b) predict(X): gives the prediction for input X.
(c) predict_proba(X): gives the class-wise probability for input X.
(d) score(X, Y): gives the accuracy of the trained model on input X and labels Y.

2. (2 points) Implement the following activation functions (along with their gradient functions): sigmoid, tanh, ReLU, Leaky ReLU and softmax (only used in the last layer).
3. (1.5 points) Implement the following weight initialization functions: zero init, random init, and normal init (Normal(0, 1)). Choose appropriate scaling factors.
4. (6 points) Train the implemented network on the MNIST dataset. Perform appropriate preprocessing and use a 80:10:10 train-validation-test split. Use the following configurations for training the network:

   (a) Number of hidden layers = 4.
   (b) Layer sizes = [256,128,64,32].
   (c) Number of epochs = 100 (can be less if computation is taking too long).
   (d) Batch size = 128 (or any other appropriate batch size if taking too long).
   (e) Learning rate = 2e-5.

   Plot training loss vs. epochs and validation loss vs. epochs for each activation function and weight initialization function and report your findings in the report (such as which function combination performed the best and where did it perform suboptimally).
   Also, save all the 12 trained models as .pkl files. You will be asked to run them during the demo to reproduce your results on the test set.

OR

3. (15 points) **Section C (Algorithm implementation using packages)** For this question, you would need to download the **Fashion-MNIST dataset**. It contains a train.csv and a test.csv. You need to take the first 8000 images from the train data and the first 2000 from the test data. These will be your training and testing splits.

   1. (1 point) Perform appropriate preprocessing on the data (for eg: normalization) and visualize any 10 samples from the test dataset.
   2. (4 points) Train a MLP Classifier from sklearn's neural network module on the training dataset. The network should have 3 layers of size [128, 64, 32], should be trained for 100 iterations using an 'adam' solver with a batch size of 128 and learning rate of 2e-5. Train it using all the 4 activation functions i.e. 'logistic',

'tanh', 'relu' and 'identity'. For each activation function, plot the training loss vs epochs and validation loss vs epochs curves and comment on which activation function gave the best performance on the test set in the report.

3. (3 points) Perform grid search using the best activation function from part 2 to find the best hyperparameters (eg: solver, learning rate, batch size) for the MLP classifier and report them in the report.

4. (4 points) For this part, you need to train a MLPRegressor from sklearn's neural network module on a regeneration task:

   (a) This means you will need to design a 5 layer neural network with layer sizes following the format: [c, b, a, b, c] where c > b > a.

   (b) By regeneration task, it means that you will try to regenerate the input image using your designed neural network and plot the training and validation losses per epoch to see if your model is training correctly.

   (c) Train 2 neural networks on the task above. One using a 'relu' activation and the other using the 'identity' activation function. Set the solver as adam and use a constant learning rate of 2e-5.

   (d) Post training both the networks, visualize the generations for the 10 test samples you visualized in part 1 and describe your observations in the report. (4 points, 1 for each part).

5. (3 points) Lastly, from the two neural networks trained above extract the feature vector of size 'a' for the train and test data samples. Using this vector as your new set of image features, train two new smaller MLP Classifiers with 2 layers, each of size 'a' on the training dataset and report accuracy metrics for both these classifiers. Train it for 200 iterations with the same solver and learning rate as part 2.

   Contrast this with the MLP Classifier you trained in part 2 and report possible reasons why this method still gives you a decent classifier?