Home Assignment – 2 (Part - 1)

Course Name: Machine Learning & Neural Net

Course code: COMP-5011-FDE

Name: Mehedi Ibtesham Arefin

Student ID: 1132169

Email: arefinm@lakeheadu.ca

Overview: The assignment contains the source code in Python for a single layer Neural Network in LMS (Least Mean Squared) method. It uses two additional libraries, numpy and matlibplot. The source code includes three files.

- Main.py (Second main function for task -2)
- LMS.py (LMS class for creating the LMS Model)
- data_banknote_authentication.txt (Dataset)

Improvements: As per the feedback received from the first assignment, this time the total dataset has been shuffled and then separated as training_data and testing_data to avoid overlapping.

Dataset: The dataset has been downloaded from: *https://archive.ics.uci.edu/ml/datasets/banknote+authentication*. Data were extracted from images that were taken from genuine and forged banknote-like specimens. Wavelet Transform tool were used to extract features from images. The dataset consists of 5 attributes.

Description: The task uses the files **Main.py** & **LMS.py** to yield testing accuracy and training accuracy and **data_banknote_authentication.txt** as the dataset. **Main.py** contains the main function where the LMS is initialized. **LMS.py** uses the LMS class and the data that has been passed while initializing to create a LMS instance. It has an initialization function, some helper functions and two important other functions: fit and predict. The fit function trains the model for a certain number of epochs and training samples that outputs training accuracy and sets the input weight whereas the predict function plots the testing samples according to what it has learnt through the fit function and outputs the testing accuracy. The dataset has 1372 samples in total where 30% of it were used as training samples and 70% were allocated for testing.
Below are some outputs of different runs:

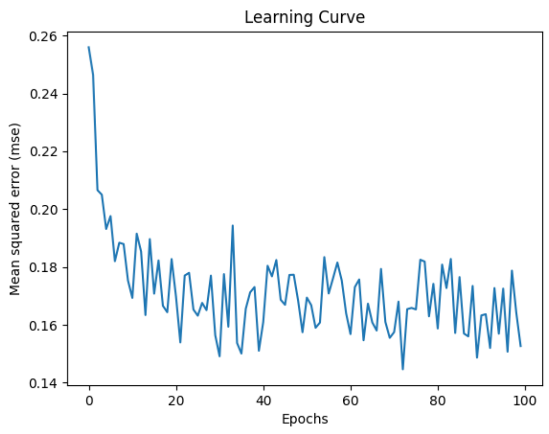

Figure: 1 (Training)

Total Points trained: 411
Epochs: 100
Learning rate (eta): 0.1 > 0.01 > 0.001
Training time: 0.22 seconds

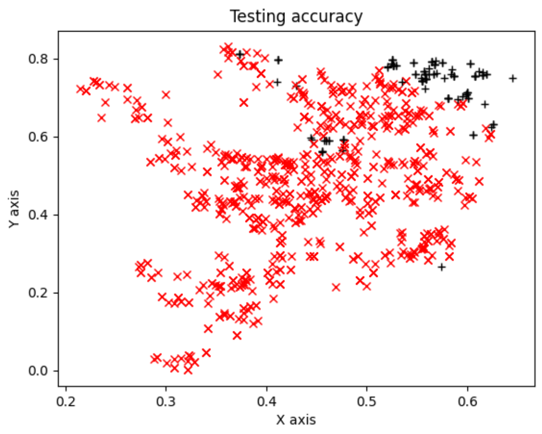

Figure: 2 (Testing)

Total Points tested: 961
Total error points: 545
Testing accuracy: 43.29%
Testing time: 0.59 seconds

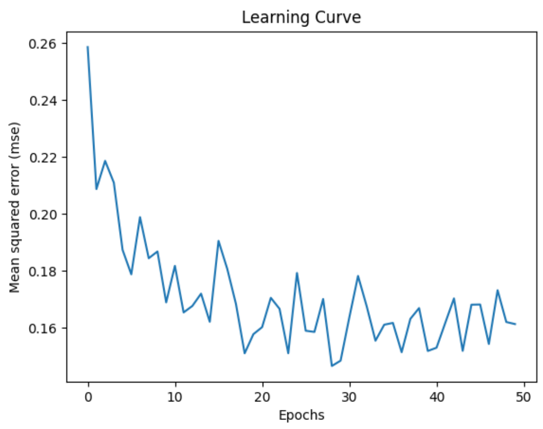

Figure: 2 (Training)

Total Points trained: 411
Epochs: 50
Learning rate (eta): 0.1 > 0.01 > 0.001
Training time: 0.14 seconds

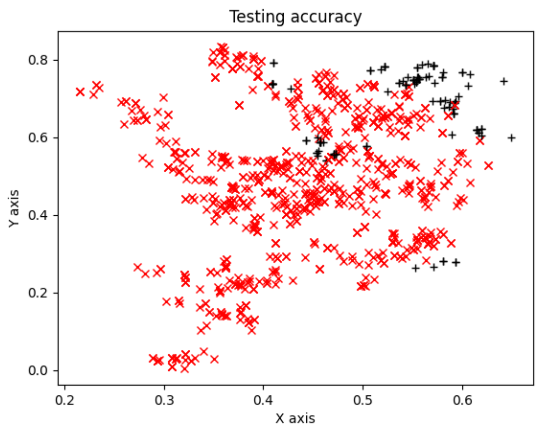

Figure: 4 (Testing)

Total Points tested: 961
Total error points: 563
Testing accuracy: 41.42%
Testing time: 0.66 seconds

Please note that, the data is consistently shuffled after iteration of each epoch. Therefore, the results will vary and each run will yield different training and testing accuracy.

Observations: Although the error rate for training was decreasing consistently, the testing accuracy was still low. Which further illustrates the shortcomings of single layer neural networks.

Discussion: The Perceptron is almost identical to the LMS algorithm. However, there are some key differences, such as:

- The perceptron rule is nonlinear, in contrast to the LMS rule, which is linear.
- While LMS can be used with either analog or binary desired responses, Perceptron can only be used with binary desired responses.