Home Assignment - 1

Course Name: Machine Learning & Neural Net

Course code: COMP-5011-FDE

Name: Mehedi Ibtesham Arefin

Student ID: 1132169

Email: arefinm@lakeheadu.ca

Description: The assignment contains the source code in Python for a single layer Neural Network, also known as Perceptron. It uses two additional libraries, numpy and matlibplot. The source code includes five files and two main functions separated for two different tasks:

- Task1.py (First main function for task – 1)
- Task2.py (Second main function for task -2)
- Halfmoon.py (Generates halfmoon data for task – 1)
- Perceptron.py (Perceptron class for creating the Perceptron Model)
- data_banknote_authentication.txt (Dataset for task -2)

Task – 1: The task – 1 uses the files **Task1.py, Perceptron.py** & **Halfmoon.py** to yield testing accuracy and training accuracy. **Task1.py** contains the main function where the Perceptron is initialized. **Perceptron.py** uses the Perceptron class and the data that has been passed while initializing to create a Perceptron instance. It has an initialization function, some helper functions and two main 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 **halfmoon.py** generates data for Task – 1. Below are some outputs of different runs:

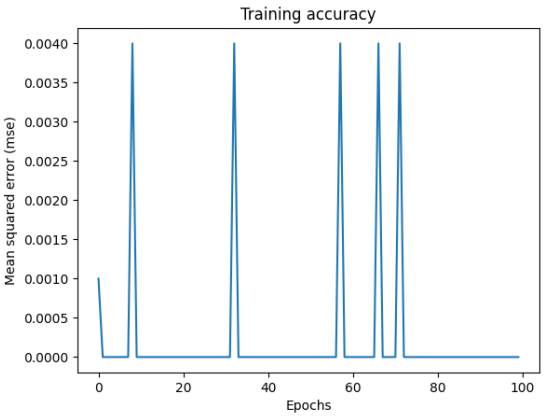

Figure 1: Training accuracy
Total Points trained: 1000
Epochs: 100
Learning rate (eta): 0.01

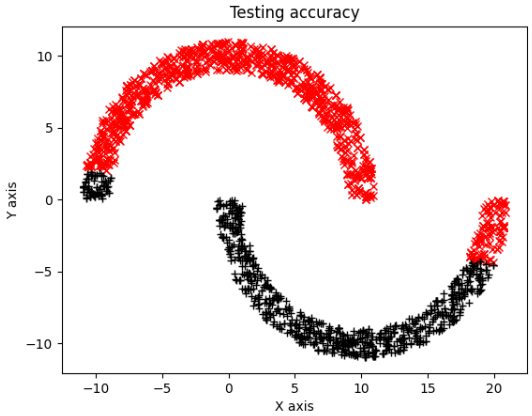

Figure 2: Testing accuracy
Total Points tested: 2000
Total error points: 198
Testing accuracy: 90.10%

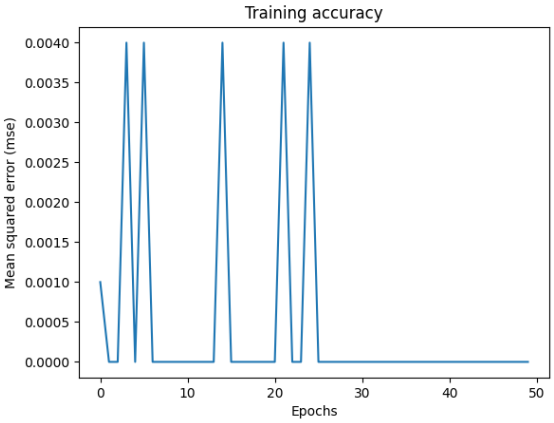

Figure 3: Training accuracy
Total Points trained: 1000
Epochs: 50
Learning rate (eta): 0.1

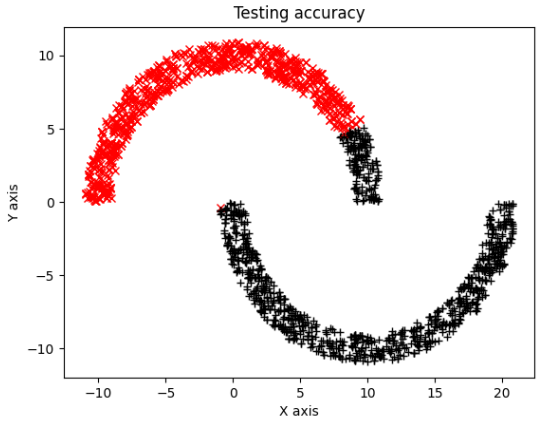

Figure 4: Testing accuracy
Total Points tested: 2000
Total error points: 252
Testing accuracy: 87.4%

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

Task – 2: The task – 2 uses the **Task2.py, Perceptron.py** & **data_banknote_authentication.txt** files to show its testing and training accuracy. **Task2.py** contains the main function which initializes a Perceptron instance as described in Task – 1 but this time with different datasets and settings. The dataset had 1372 samples in total where 30% of it were used as training samples and 70% were allocated for testing.

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.

Below are some outputs of different runs:

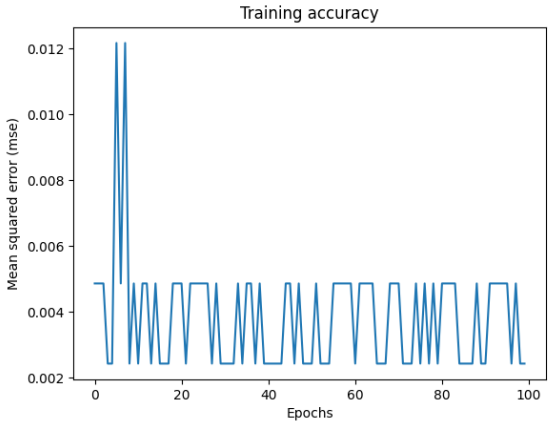

Figure 5: Testing accuracy
Total points trained: 411
Epochs: 100
Learning rate (eta): 0.01

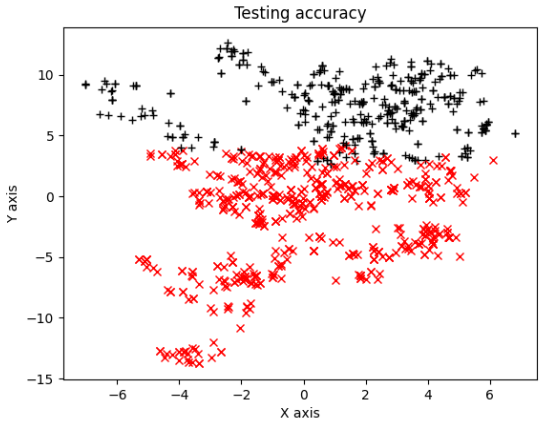

Figure 6: Training accuracy
Total points tested: 961
Total error points: 621
Testing accuracy: 35.38%

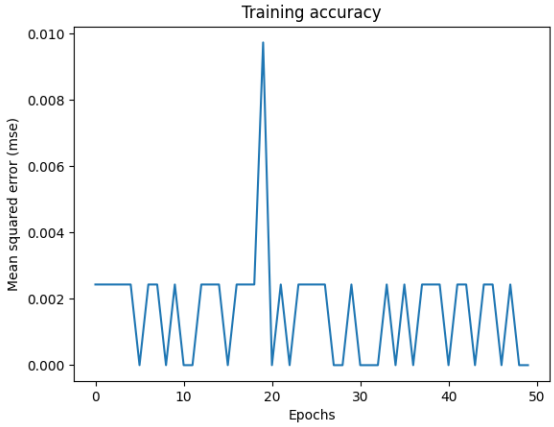

Figure 7: Testing accuracy
Total points trained: 411
Epochs: 50
Learning rate (eta): 0.1

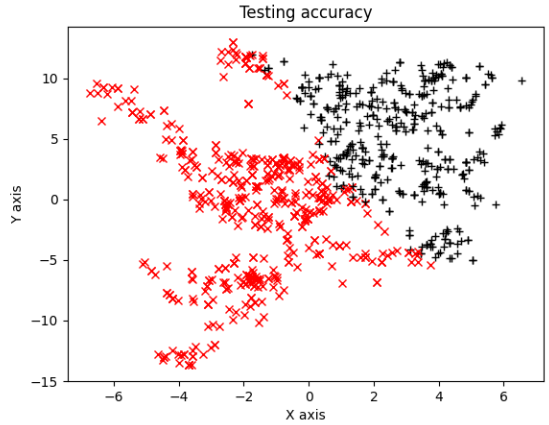

Figure 8: Training accuracy
Total points tested: 961
Total error points: 571
Testing accuracy: 40.58%

Conclusion: To recapitulate, the Perceptron performs well when it comes to binary classification problems as shown in figure 1-4. However, when the dataset has more than two classes the Perceptron's performance declines as shown in figure 5-8.