Double Moon Classification Problem using MLP

Project 1 – TEAM ONE

Abhinav Karthik Sridhar
Master of Science – Electrical Engineering
Arizona State University, USA

Sanjay Kumar Reddy
Master of Science – Electrical Engineering
Arizona State University, USA

Venkata Motupalli
Master of Science – Electrical Engineering
Arizona State University, USA

***Abstract*—The key idea of this project is to have random data points (1000) on the top and bottom moon with the given distance 'd' separating them and have them classify using three neural network cases: Backpropagation, Backpropagation with momentum and Levenberg-Marquardt using multilayer perceptrons.**

I. Introduction

A multilayer perceptron (MLP) is a class of feedforward artificial neural network. An MLP consists of at least three layers of nodes. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

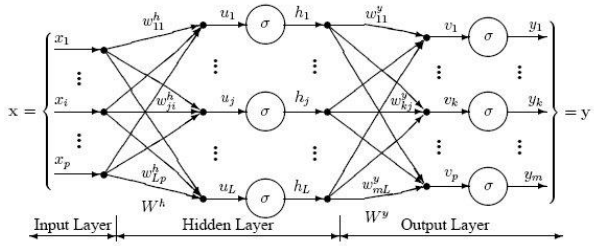


Fig.1 Multi-Layer Perceptron Network

Each MLP has an activation function, number of hidden layers and the number of hidden neurons that are associated with each of the hidden layers and a learning rate associated with the training method. So, we used the three techniques Levenberg-Marquardt, Backpropagation and Backpropagation with momentum to solve the double moon classification problem also for different 'd' (2, -4, -8) units.

II. Implementation and Observations

The double moon classification is to be done with the three methods that are mentioned above. The first thing to do is to bring the double moon shape and plot random data points (1000) in each ring for the various 'd' that we have. The plots of the double moon random data look like the one that is showed below:

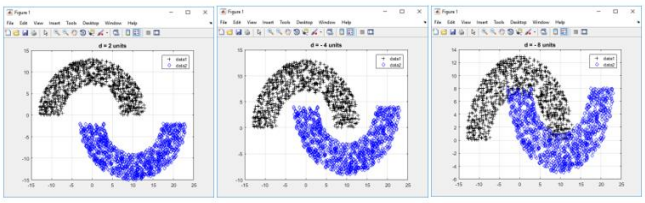


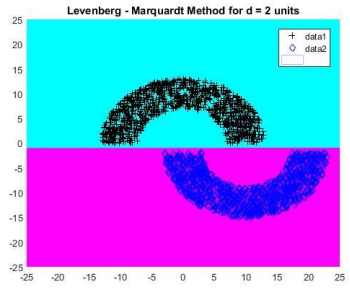
Fig.2 The double moon for different values of 'd'

Now that we have all the data points, it is time to train the data with the input and Target vectors specified for the different methods that we must implement.

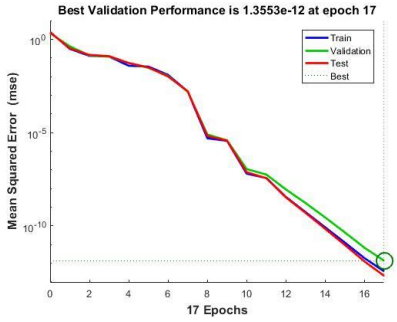
A. Training using Levenberg Marquardt

In this method, the weights and the activation functions are controlled as per the Levenberg-Marquardt activation functions. The Levenberg-Marquardt algorithm, also known as the damped least-squares method, has been designed to work specifically with loss functions which take the form of a sum of squared errors. It works without computing the exact Hessian matrix. Instead, it works with the gradient vector and the Jacobian matrix. After using the '**trainlm**' command in MATLAB to train the data, the classification was done and the results produced for the same are shown below for different values of the 'd'. After numerous attempts in changing the value of the learning rate, the best results were achieved for a=0.01 and the number of epochs for each value of d changes with the usage of "early stopping".

- d = 2 units



(a)

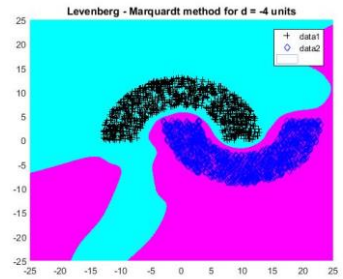


(b)

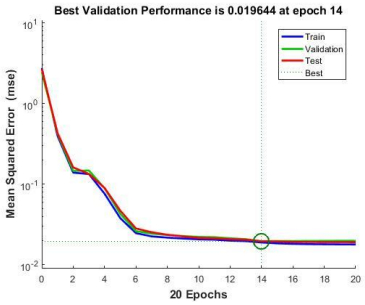
(c)

Fig 3 (a) Classification obtained (b) Confusion matrix which yields 100% classification (c) Learning rate curves with #epochs = 17

- d = -4 units



(a)

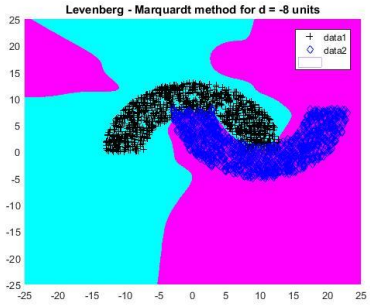


(b)

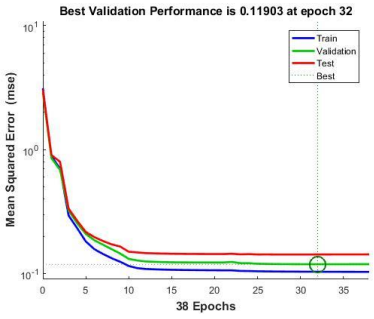
(c)

Fig 4 (a) Classification obtained (b) Confusion matrix which yields 99.7% classification (c) Learning rate curves with #epochs = 14

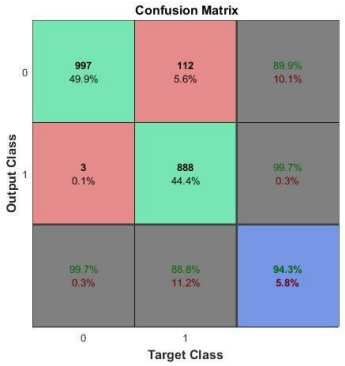
- d= -8 units



(a)



(b)



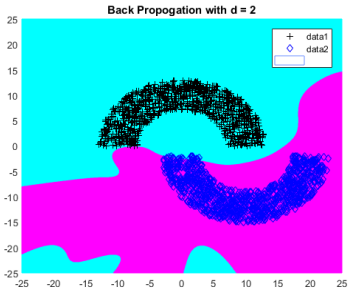
(c)

Fig 5 (a) Classification obtained (b) Confusion matrix which yields 94.3% classification (c) Learning rate curves with #epochs = 32

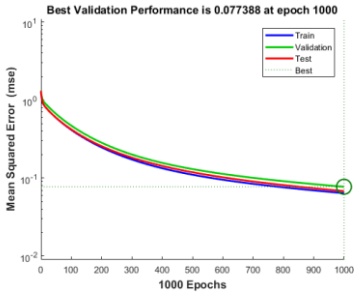
B. Training using Backpropagation algorithm

Backpropagation is a method used in artificial neural networks to calculate the error contribution of each neuron after a batch of data (in image recognition, multiple images) is processed. This is used by an enveloping optimization algorithm to adjust the weight of each neuron, completing the learning process for that case.

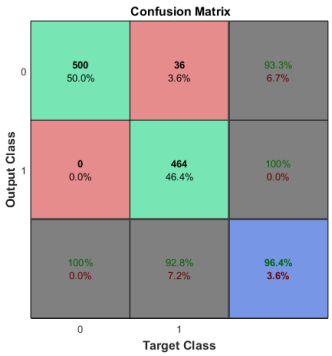
- d = 2 units



(a)



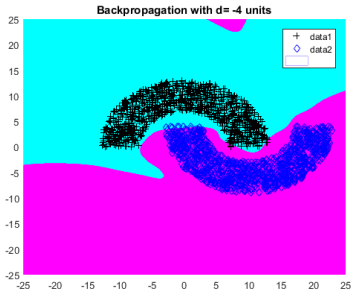
(b)



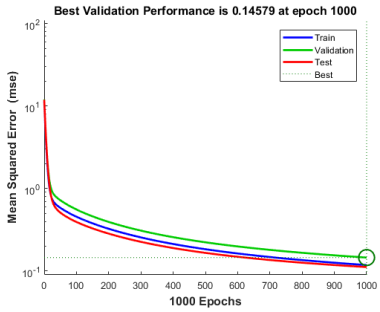
(c)

Fig 6 (a) Classification obtained (b) Confusion matrix which yields 96.4% classification (c) Learning rate curves with #epochs = 1000

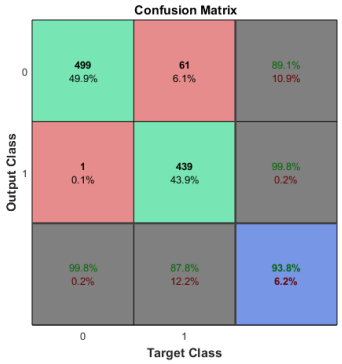
- d = -4 units



(a)



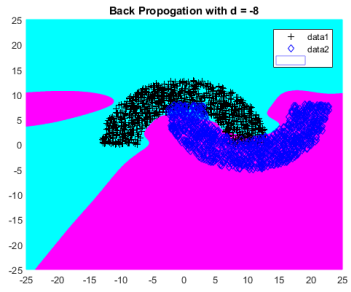
(b)



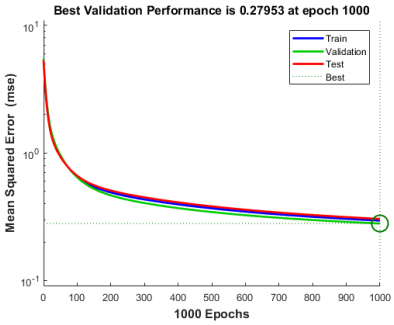
(c)

Fig 7 (a) Classification obtained (b) Confusion matrix which yields 93.8% classification (c) Learning rate curves with #epochs = 1000

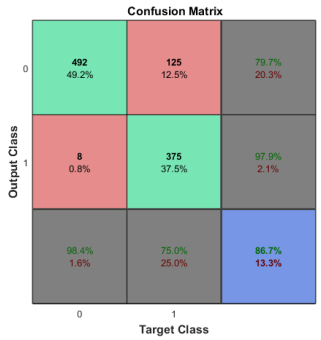
- d = -8 units



(a)



(b)



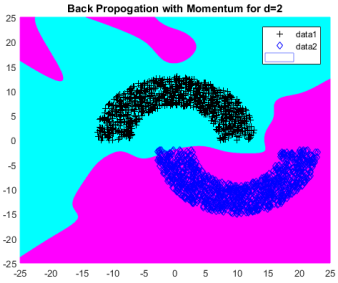
(c)

Fig 8 (a) Classification obtained (b) Confusion matrix which yields 86.7% classification (c) Learning rate curves with #epochs = 1000

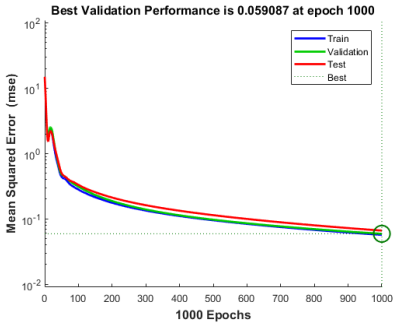
C. Training using Backpropagation with momentum

Gradient descent with momentum, implemented by **traingdm**, allows a network to respond not only to the local gradient, but also to contemporary trends in the error surface. Acting like a lowpass filter, momentum allows the network to ignore small features in the error surface. Without momentum, a network can get stuck in a shallow local minimum. With momentum, a network can slide through such a minimum.

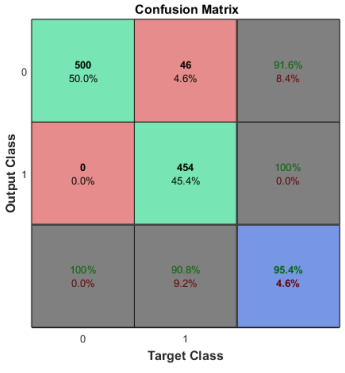
- d = 2 units



(a)



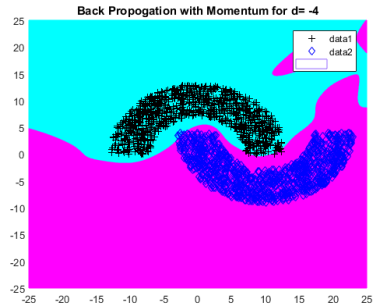
(b)



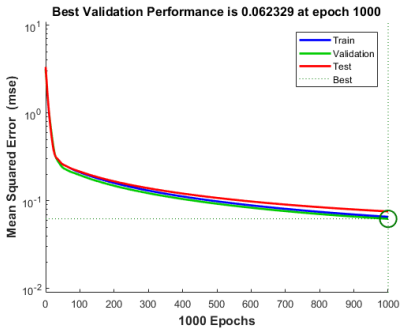
(c)

Fig 9 (a) Classification obtained (b) Confusion matrix which yields 95.4% classification (c) Learning rate curves with #epochs = 1000

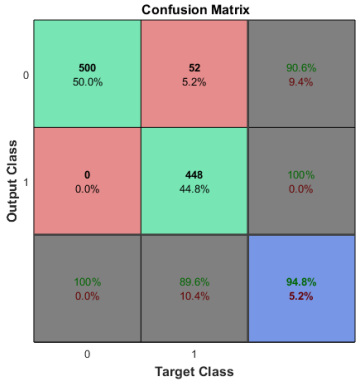
- d = -4 units



(a)



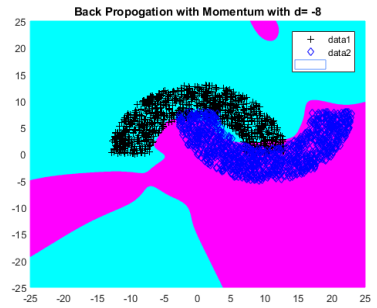
(b)



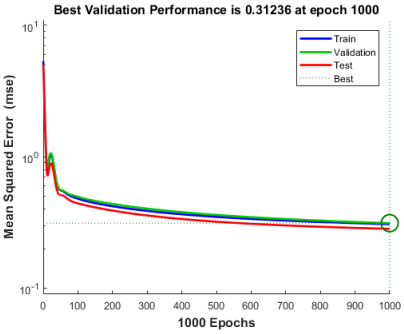
(c)

Fig 10 (a) Classification obtained (b) Confusion matrix which yields 94.8% classification (c) Learning rate curves with #epochs = 1000

- d = -8 units



(a)



(b)

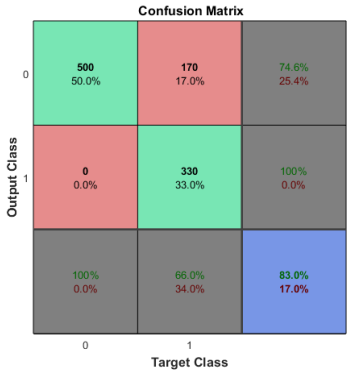


Fig 11 (a) Classification obtained (b) Confusion matrix which yields 83% classification (c) Learning rate curves with #epochs = 1000

III. Results

Various classifications have been obtained for the double moon classification problem with three methods namely – Levenberg marquardt, backpropagation and backpropagation with momentum. It was seen that it is trivial to change certain parameters when the effects are studied. For example, the learning rate parameter, number of hidden neurons in a hidden layer etc changes the results. Certain issues like data pre-processing, weight initialization and stopping criteria were dealt with in this project to get optimum results. Throughout the project the number of hidden layers was kept constant to one hidden layer. After training the neural networks, test data of 500 data points in each ring is taken and then implemented to see the test results that are documented above.

IV. References

[1] “Backpropagation.” Wikipedia, Wikimedia Foundation, 16 Sept. 2017, en.wikipedia.org/wiki/Backpropagation.

[2] “Documentation.” Multilayer Neural Networks and Backpropagation Training - MATLAB & Simulink, Matlab Documentation, ww.mathworks.com/help/nnet/ug/multilayer-neural-networks-and-backpropagation-training.html?requestedDomain.

[3] Eraqi, Hesham. “Simply !” MLP Neural Network with Backpropagation [MATLAB Code], 1 Jan. 2015, heraqi.blogspot.com.eg/2015/11/mlp-neural-network-with-backpropagation.html.