CSCI -B555: Extra Credit Programming Assignment

Due on Tuesday, April 25, 2017

Donald Williamson

Sushmita Sivaprasad

Problem 1

The aim of the Neural Network algorithm that has been implemented here is to be able to predict the output for a given set of inputs by using a supervised learning approach. The type of neural network algorithm implemented here is a 2 layer perceptron back-propagation algorithm. The truth table for the 4 bit parity probrlem is as shown in the figure where $'x_1'x_2'x_3'and'x_4'$ are the 4 binary inputs and 'd' represents the desired output.

x_1	x_2	x_3	x_4	d
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Figure 1: Truth table showing inputs and desired output

IMPLEMENTATION

A 2 layer back-propagation algorithm is implemented here where we have 4 input nodes , 4 hidden nodes and one output node.

The input layer, hidden layer and the output layers are taken as a matrix where I have initialized the size of the matrix as these are pre-determined as per our requirement. The initial weights given are randomized with a value between 0 and 1 to avoid major variation in weights.

Forward Propagation

- The forward propagation algorithm is implemented where the activation potential is determined as a dot product of the Input matrix and the weight vector which gives a sum of the products of the Input and the Weights for each of the input layers.

- The activation potential is fed into the sigmoid function which is used to calculate the activation function for the hidden layer. The sigmoid function gives us output values between 0 and 1 rather than returning binary values which makes it easier to calculate approximate values of the updated weights.
- The predicted output value is to be calculated next , for which the activation potential is calculated on the activation function and the final predicted value is calculated by feeding the activation potential into the sigmoid function.

Backward Propagation

The backward propagation algorithm involves using either the stochastic gradient descent or the batch gradient descent to find the global minima in the error variable for the chosen value of the weights. A derivative is taken of the cost function with respect to the weights and the weights are updated accordingly.

Cost Function

This gives us the difference between the desired output value d and the predicted output value ypred. The cost function is the sum of the squared values of the error in the prediction. Our aim here is to train the network in such a manner so that we can minimize the cost function. The cost function is given by the variable 'J'.

Derivative of Cost Function

The partial derivative of the Cost Function J is taken w.r.t the Weight between the input and the hidden layer and the hidden layer and the output layer , which is used to calculate the batch gradient descent.

Updating the Weights

The error value in the weights is added/subtracted to the randomized weight values and multiplied with the learning rate.

Computing the Convergence

A sequence of learning rate is given as a value varying from 0.05 to 0.5 with an increment of 0.05. The results obtained are as below where I have a convergence for every value of the learning rate.

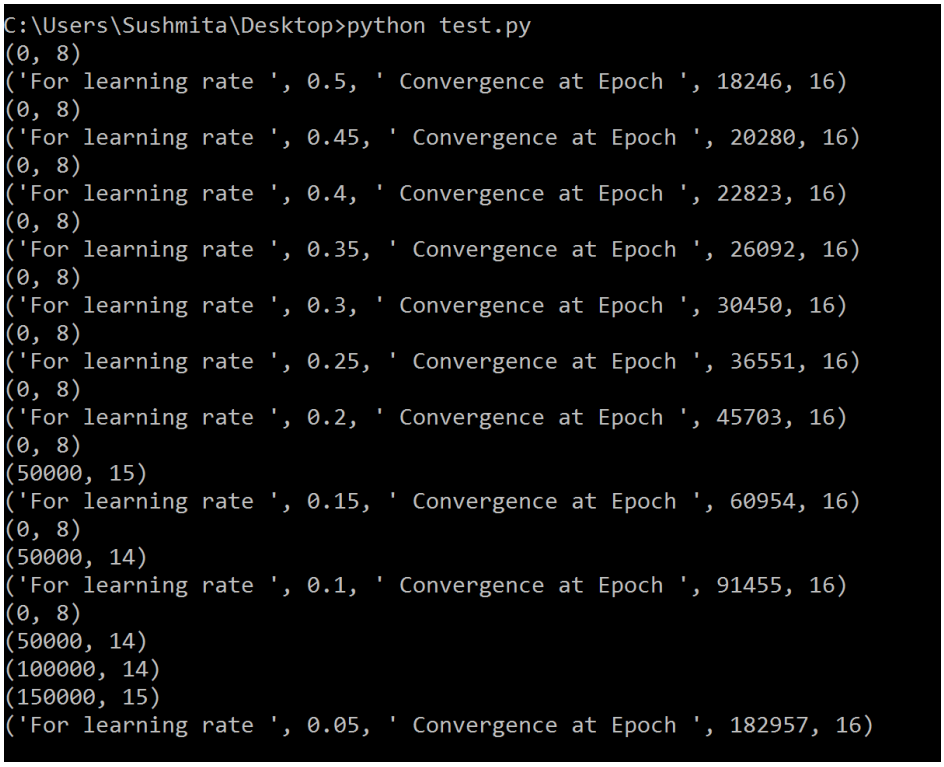

Figure 2: Testing the Implementation of the algorithm for the given learning rate

Momentum

The momentum function with a value of α as 0.9 and the value is computed against all the learning rates . The figure below shows the convergence rate with very few epochs

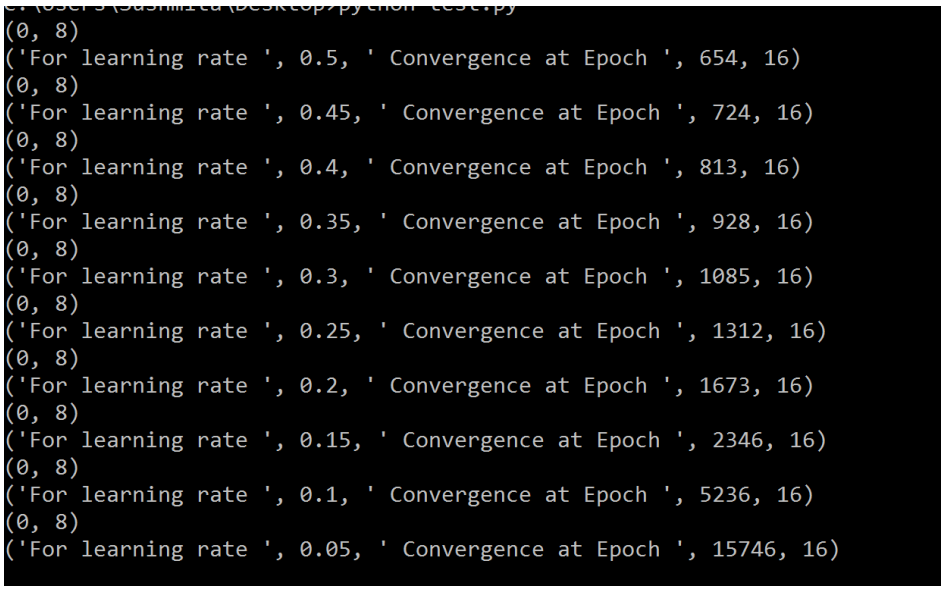

Figure 3: Reduction in the epochs with introduction of momentum