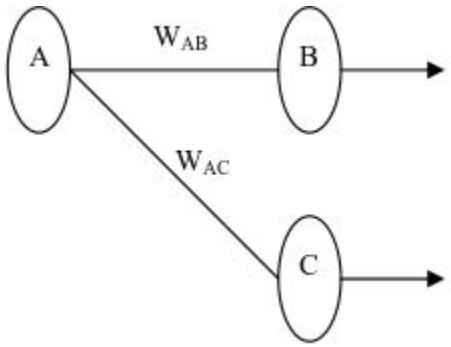
NEURAL NETWORK

Author : Abhishek Mehra and Mridul Birla



Here A is the Hidden Node and B,C are the Output Node

1. First we calculate weighted sum of all the input weights coming to the particular hidden node and apply the sigmoid activation function over it.
2. For the output layer we take weighted sum of hidden layer to output layer to calculate the outputs of the 4 nodes

3. Next work out the error for neuron B. The error is what you want – what you actually get, in other words:
   Error $_B$= Output$_B$(1-Output$_B$)(Target$_B$– Output$_B$)
   The "Output(1-Output)"term is necessary in the equation because of the Sigmoid
   Function.

4. Change the weight. Let W_{+AB} be the new (trained) weight and W_{AB} be the initial weight.
   W_{+AB}= W_{AB}+ (Error$_B$ x Output$_A$)
   We update all the weights in the output layer in this way.

5. Calculate the Errors for the hidden layer neurons. Unlike the output layer we can't
   calculate these directly (because we don't have a Target), so we back Propagate them from the output layer . This is done by taking the Errors from the output neurons and running them back through the weights to get the hidden layer

errors. For ex if neuron A is connected as shown to B and C then we take the errors from B and C to generate an error for A.

$$Error_A = Output_A(1 - Output_A)(Error_B * W_{AB} + Error_C * W_{AC})$$

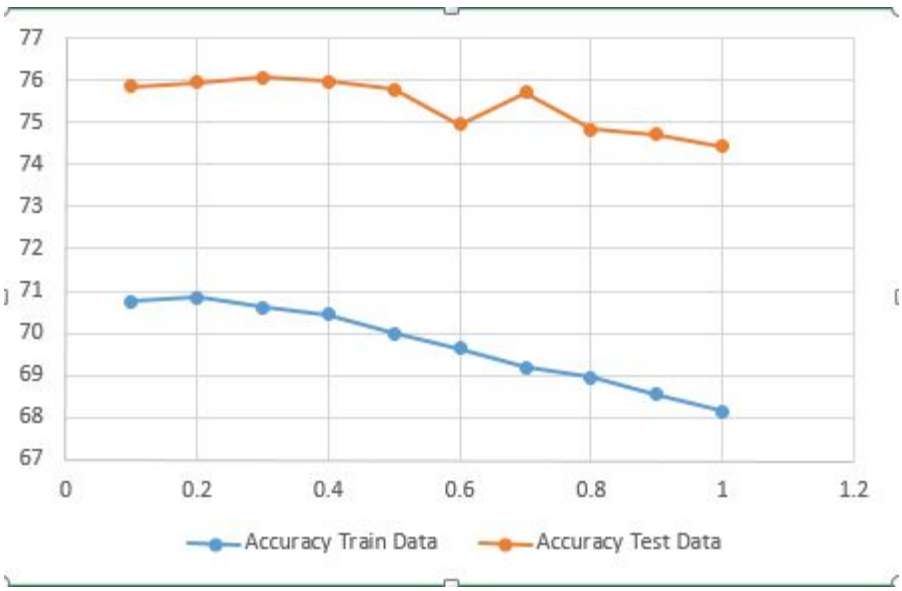
Again, the factor “Output (1 - Output)” is present because of the derivative of sigmoid function. Now we calculate the weight same as step 3

Road Blocks:

1. Initially we had weights randomized between 0 and 1 due to which sigmoid function was giving output as 1 most of the times. Later we changed it to -.1 to .1 which started giving optimal results.

2. To minimize the weight summations we normalized the input data by dividing it by 255.00

Results



Accuracy vs Learning Rate

Confusion Matrix for hidden node count =30 and learning rate=.6:

Train Accuracy: 69.6938554738
Confusion Matrix:

	270	180	90	0
270	6312	1155	1068	709
180	690	6452	1134	968
90	1065	762	6461	956
0	891	1121	687	6545

Test Accuracy: 73.0646871686
Confusion Matrix:

	270	180	90	0
270	184	20	27	13
180	13	173	21	29

90	27	17	169	11
0	19	34	23	16

BEST Algorithm

1. For the best algorithm we have stored the weights of the hidden layer and output layer in a separate file. So whenever neural network runs the updated weights gets stored .

2. We have also taken the the final learning rate as .6 and hidden node count as 30.

3. For running the best algorithm you must first Neural Network.