Soft Computing Assignment-2

Manas Gaur, Wright State University 03/04/2017

1 Question 2

Updating the synaptic weights in backpropagation is done as

$$\Delta w = \eta \Delta x$$

when $\Delta x \to 0$, this becomes δx, hence

$$\Delta w = \eta \delta x$$

where η is the learning rate and δx is the the gradient.

The gradient is calculated as

$$\delta_j = \phi^{'}(v) \sum_k \delta_k w_{kj}$$

Based on the above equations, updating synaptic weights depends on first derivative of activation function. Considering the activation function to be an odd function, because odd function boosts the learning speed. Example of an odd function is sigmoid function. It is represented as

$$\phi(v) = a \tanh bv$$

where

$$a \tanh bv = \frac{a}{1 + e^{-bv}}$$

In the above equation, v is the induced local field for feed forward nerual network. It is calculated as

$$v_j = \sum w_{ji} x_i + bias$$

where x is the input to the neural network and w is the weight of the neural network synaptic weights. If we threshold (saturate) the output values to low or high value, the local field **v** will be controlled. This makes the first order derivative of **v** to become zero in equation δ_j. Hence with the decrease in gradient, the change in weights will also decrease. As a result the network will stop updating weights. Therefore, it is essential to control the output values to some high or low threshold. In order to prevent controlling of output values by thresholding, we can change the activation function such that the gradient does not reach zero or close to zero. This can be done by increasing the slope of the curve and decreasing the value of **b**.

2 Question 3

In order to make neural network learn efficiently we need to provide better rate of learning. According to the equation $\Delta w = \eta \Delta x$, learning rate directly affects the procedure of updating weights in the neural network. If the synaptic weights are increased indefinitely this will make the network oscillatory. By introducing momentum we balance the weights update with learning rate. The modified general delta rule for synaptic weights in the neural network for layer **l** is

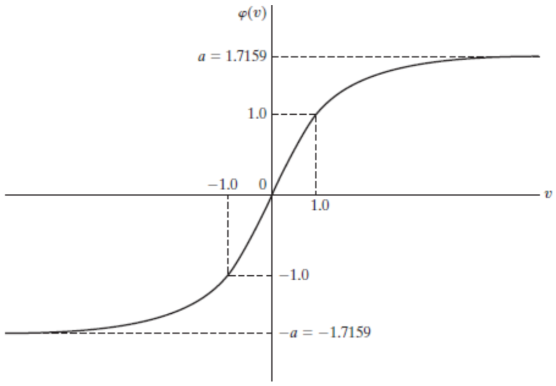

Figure 1: Sigmoid Function

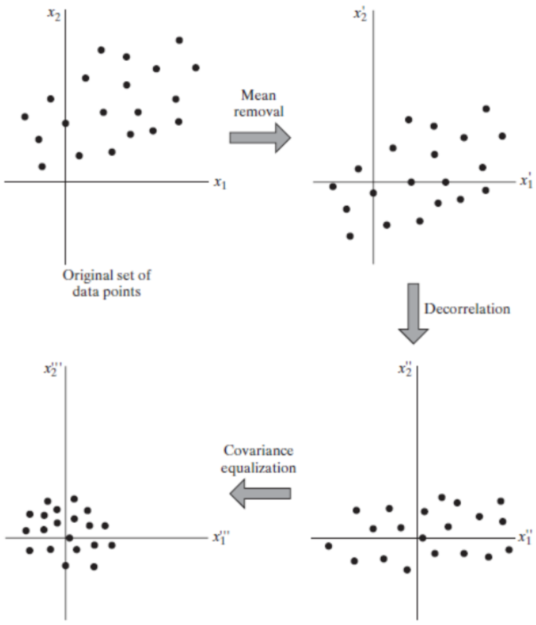

Figure 2: Steps for Preconditioning

$$w_{ji}{}^{l}(n+1) = w_{ji}{}^{l}(n) + \alpha[\Delta w_{ji}{}^{l}(n-1)] + \eta \delta_j{}^{l}(n) y_i{}^{(l-1)}(n)$$

The learning momentum parameter aids in solving the problem of unstable network or oscillatory network that arise while training a data set.

While making neural network learn using general delta rule, the trough between A to C makes the delta rule based gradient susceptible to oscillation between A-B-C. With the introduction of momentum parameter in modified general delta rule ($w_{ji}{}^{l}(n+1)$), the network learns to skip the trough between A and C. In order to prevent oscillations in neural network learning, a value strictly less than 1 is desirable for learning momentum.

3 Question 4

Preconditioning of inputs by normalization in a way that all inputs are arranged in a dimension such that their variance is centered around zero. This adjustment of input data set is known as preconditioning.

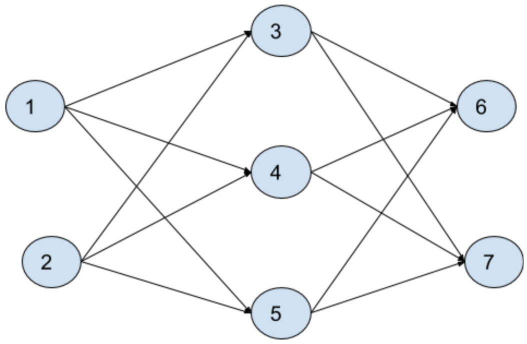

Figure 3: Sigmoid Function

Advantage of preconditioning are :

- Inputs are consistently positive : Synaptic weights of a neuron in the first hidden layer will increase in only positive direction. When the weight vector of the neuron is to change direction, it can so only by zigzagging though error. This will reduce the learning rate.
- Inputs are consistently negative: Synaptic weights of a neuron in the first hidden layer will increase in only negative direction. When the weight vector of the neuron is to change direction, it can so only by zigzagging though error. This will reduce the learning rate.
- If our dataset is more diverse or spread out, it is better to perform preconditioning so that variance will decrease.

The downside of preconditioning of inputs is the decrease in weight update. In order to prevent the downside, we increase the learning rate parameter.

4 Question 5

Since there are 3 classes for 3 patterns in the data, so we will be needing atleast 2 output neurons. The data set is divided into sub datasets of 4 groups which will be needing 3 hidden neurons in the hidden layer.

For instance, in the network shown above in the figure, the activation function be a step function. The output of hidden layer be A, B and C respectively. Output of the hidden layer neurons are :
table here

Hidden Neuron 1 or Neuron 3 :
Class 1 : 1 1
Class 2 : 1 0
Class 3 : 1

Hidden Neuron 2 or Neuron 4:
Class 1 : 0 0

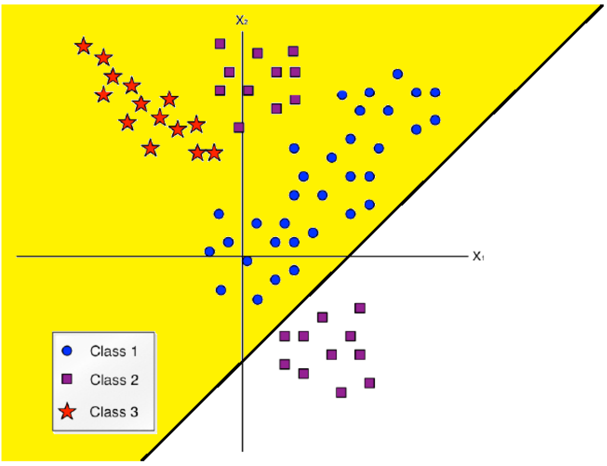

Figure 4: Decision Boundary created by first hidden Neuron

Class 2 : 1 0

Class 3 : 1

Hidden Neuron 3 or Neuron 5 :

Class 1 : 0 1

Class 2 : 1 0

Class 3 : 0

XOR Rules :

0 XOR 0 : 0

1 XOR 1 : 0

0 XOR 1 : 1

1 XOR 0 : 1

Hidden Neuron 1 or Neuron 3's characteristics are :

$$w_{13} = -1, w_{23} = 1,$$

$$b_3 = \frac{-3}{2}$$

decision boundary slope (m) is

$$m = \frac{-w_{13}}{w_{23}} = 1$$

Hidden Neuron 2 or Neuron 4's characteristics are :

$$w_{14} = -1, w_{24} = 1,$$

$$b_4 = \frac{-1}{2}$$

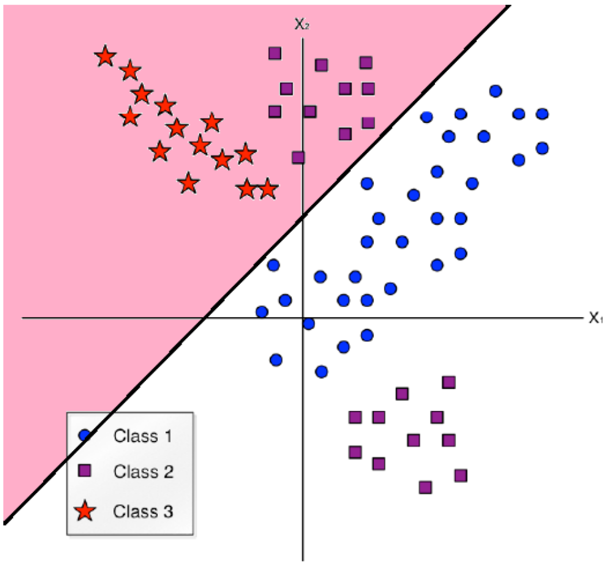

Figure 5: Decision Boundary created by second hidden Neuron

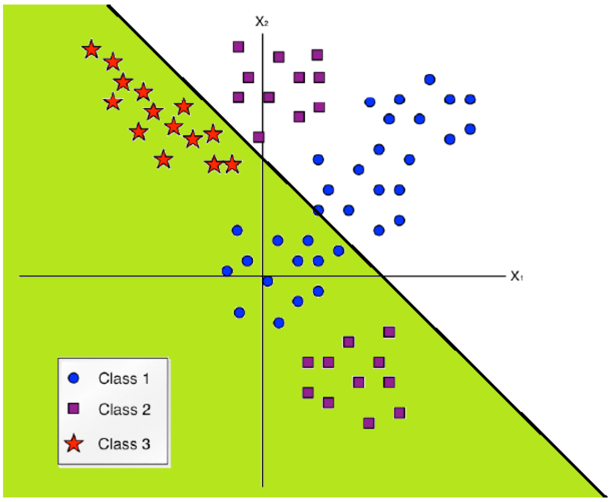

Figure 6: Decision Boundary created by third hidden Neuron

decision boundary slope (m) is

$$m = \frac{-w_{14}}{w_{24}} = 1$$

Hidden Neuron 3 or Neuron 5's characteristics are :

w_{15} = -1, w_{25} = 1,

$$b_5 = \frac{-1}{2}$$

decision boundary slope (m) is

$$m = \frac{-w_{15}}{w_{25}} = -1$$

The output of the network for the XOR function as follows :

Output Neuron 1 or Neuron 6:

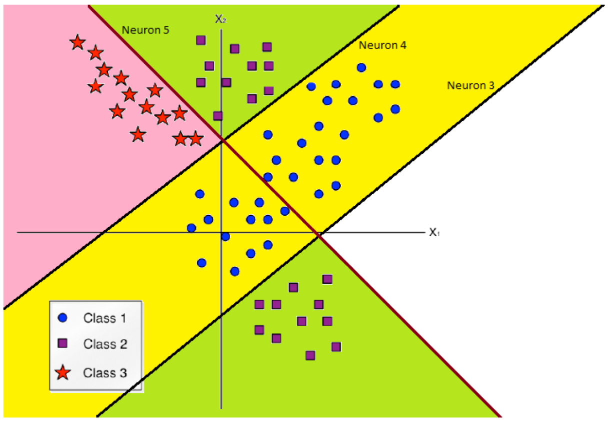

Figure 7: Decision Boundary created by first, second and third hidden Neurons

Class 1 :1

Class 2 :1

Class 3 :0

Output Neuron 2 or Neuron 7:

Class 1 :1

Class 2 :0

Class 3 :1