**Sample MidTerm Questions**

1. Answer as True or False and justify your answer:
2. A perceptron will converge for a linearly separable problem

T – as perceptron can learn any linearly separable function.

1. Perceptron can learn X-NOR Gate

F – X-NOR gate is similar to XOR gate and it is not a linearly separable problem.

1. The Backpropagation method is mainly based on Gradient Descent

F – Yes BP uses gradient descent BUT its main idea is to propagate errors back.

1. Neural nets –



1. **Consider a neural net with a step function threshold. Suppose that you *multiply* all weights and thresholds by a constant. Will the behavior of the network change?**
2. Back Propagation of Errors (use the network shown in #2. Assume tk = [.7, .8] and sigmoid.

**Ans.** Ignore all the values within the circles in the output layer [the input values in the input layer remain same as in (a) i.e. 0.5 (left input neuron) and 1.0 (right input neuron). The same is true for hidden layer before using the sigmoid] as we are now going to calculate new values with targets as 0.7 for the left out put neuron and 0.8 for the right output neuron.

**Now, let’s first do the forward pass:**

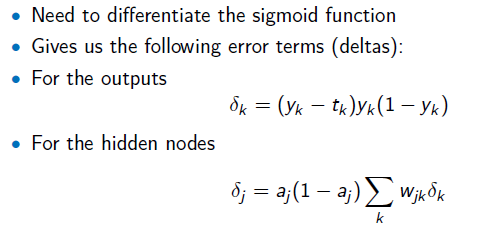
Sum of inputs at the left hidden neuron = 0.5 x -1 + 1x0 = -0.5. Now, we need to use the sigmoid function to get the final output of left hidden neuron –

* Left hidden neuron -> 1/(1 + exp(-x) = 1/(1 + exp[-(-0.5)] = 1/ (1 + 1.60) = 0.384
* For 2nd neuron in the hidden layer, we have, 1/(1 + exp(-x) = 1/(1 + exp(-2) = 1/ (1 + .13) = 0.88
* For the 3rd neuron in the hidden layer, 1/(1+ exp (0)) ] = 0.50
* Hence, net input to the output neuron (left) = 2x0.384 + (-0.5x0.88) +0.5 = 0.768 – 0.44 + 0.5 = 0.828. Now, we take the sigmoid -> yk (left output neuron) = 1/(1 + exp(-.828)) = 0.69 and Target, Tk for this neuron is 0.7 as given.
* Now do the same for the right output neuron -> -2x0.384 + 0.88x1 + 0.5 x weight (0.5) = 0.362. Now take the sigmoid – 1/(1 + exp(-0.362) = 0.59. The Tk is 0.8 as given.

**Now, let’s first do the backward pass: [Consider just the left hidden layer neuron.**

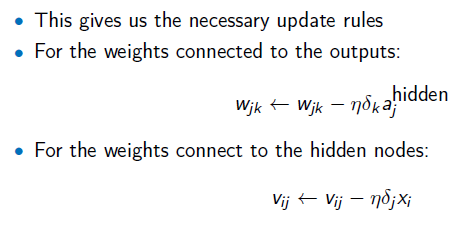
First, we need to calculate delta at the out put layer neuron. For left out put neuron, we have

Equations to be used are:



* Delta for left output neuron = (yk =0.69 – Tk = 0.7)x(0.69)(1-0.69) = -0.01x0.69x0.31 = -0.002
* For the right output neuron, it is (0.59 – 0.8)(0.59)(1-0.59) = -0.05
* Now, we can calculate back propagated errors to the hidden layer units. Let’s just do it for the left hidden layer unit:
* Using delta j equation shown above, we have, aj(1-aj). (delta k at left output layer x weight = 2 + delta k at right output layer x weight = -2. Note aj (as found before) is 0.384.
* 0.384(1-0.384).[(-0.002x2) + (-0.05 x -2)] = **0.0236**

Now we can use the weight update equation:



So, for the output layer weigh (between left output neuron and left hidden layer neuron), we have

New Wjk = 2 – eta (0.25).( **0.0236** = delta k at left output layer).(0.384 = ajhidden)

= 2 - 0.0059 = 1.9941. Similar approach applies for all other weights.

1. SVM

A close up of text on a white background

Description automatically generated

1. Why Machine Learning?

Ans: Key reasons are:

1. Algorithms do not exist for many data driven applications (e.g. auto-driven cars).
2. Algorithms may exist for some complex applications but they will take longer time to run and will be inefficient or less accurate.
3. For large data (e.g. Big Data), the size of the data is too large and even if some algorithms may exist, it will be inefficient to use those.
4. Automation of programming is very important to save time, improve accuracy and more. Automatically configuring / reconfiguring, or replacing some blocks that might be learned by an ML method are good list for automation using ML.
5. Dimensionality Reduction – Concept (LDA, PCA)
6. Some key issues: training, validation, testing, learning parameters, VC Dimension
7. Q & A