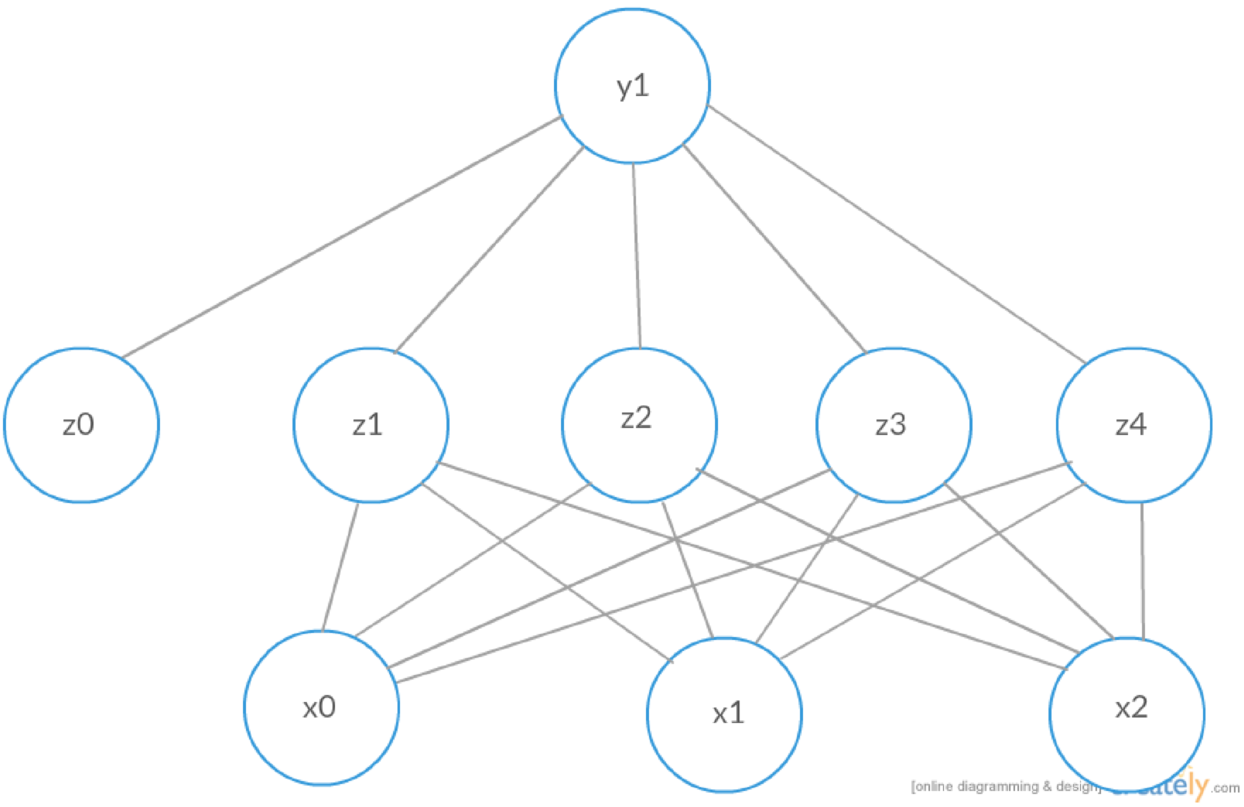
Task 3 Report – s1558717

Finding the weights

For finding the weights for the decision boundaries of my polygon, I found out the linear equations for the four edges of the quadrangle using the data provided.

I then took the coefficients for the constant, x and y values in each equation as the weights from x0, x1 and x2 respectively to each corresponding neuron in the first layer. For the top and right edges of the polygon, I took the equation corresponding to the region inside the polygon with a value greater than equal to zero, and appropriately took the weight for their respective representative neurons in the next layer with positive one. For the bottom and left edges of the polygon, I took the equation corresponding to the region inside the polygon with a value less than zero, and appropriately took the weight for their respective neurons in the next layer with negative one.

As each of the neurons in the first layer gives value 1 when positive, I took a bias of -1 in the first layer to make sure that the data points given are classified inside the region and the overall value of y stays positive when true.



Investigating decision regions

The decision boundaries for sNN_AB() were much harder to obtain as compared to those for hNN_AB().

The sigmoid function uses 1/(1+exp(-a)) where a is the same variable as used in the step function (the variable is basically the data points multiplied by the weights summed with the corresponding bias).

The sigmoid function approaches 1 for very large values of a, and is 0 for very small values of a (as the exp of -a approaches infinity). The step function is however 1 for all positive values of a and 0 for all other values of a.

We have therefore multiplied the sigmoid function by a large value so that it can replicate the step function for binary classification; however the boundaries obtained are erratic and change by a big amount for any change in the weights at the last layer or the number that is multiplied. However, no significant change was observed if the number multiplied was increased from 2000 and beyond.

The decision regions for hNN_AB() are therefore easy to obtain where the region inside the first polygon and outside the second polygon can be weighted positive one, whereas the decision regions for sNN_AB() need to take very large weights (for y tending to one) and are still close to one or close to zero, hence much more difficult to approximate.