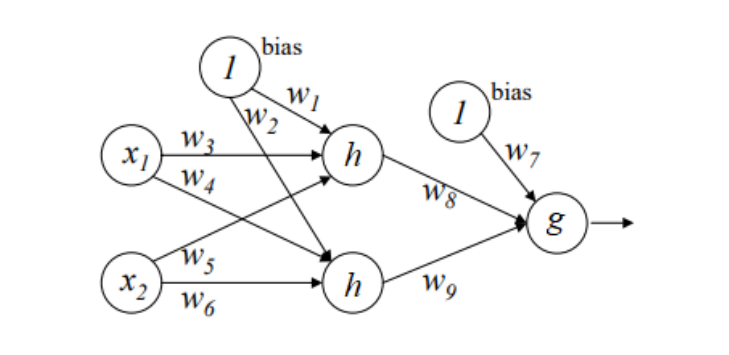
Obraz zawierający tekst

Opis wygenerowany automatycznie

Exercise 34



1. We have that and , we can then calculate the output in terms of , and .
2. The classification boundary will be then

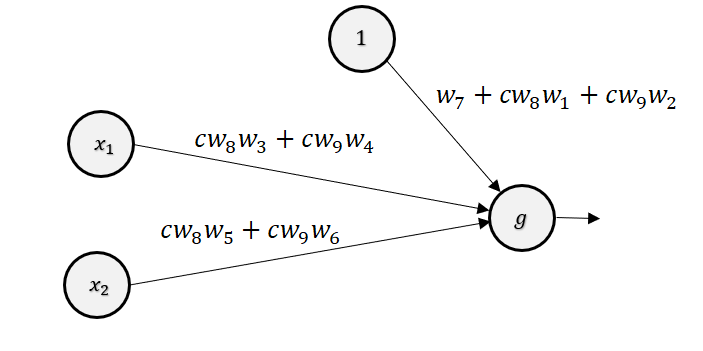
The decision boundary in terms of output and is linear in this case.

⋅

1. Cross-entropy can be used as a loss function, and it will looks like:

where is the ground truth vector, is the estimate and is the inner product.

Exercise 35



Can any multi-layered neural net with linear activation functions at hidden layers be represented as a neural net without any hidden layer?

Yes. If linear activation functions are used for all the hidden units, output from hidden units will be written as linear combination of input features. Since these intermediate output serves as input for the final output layer, we can always find an equivalent neural network which does not have any hidden layer as seen in Exercise 34.

Exercise 36

1. Becouse softmax function is a vector-to-vector transformation, it’s derivative is a Jacobian matrix. Jacobian matrix has a row for each output element and a column for each input element .

We have then diagonal entry of row . We compute the derivative of the ‘th output, , with respect to its ‘th input, :

And off-diagonal row entries of row . We compute the derivative of the ‘th output, , with respect to it’s ‘th input, , where :

The form of the off-diagonals tells us that the Jacobian of softmax is a symmetric matrix. And the full form will look like this:

1. If :

If :

have different forms of probability function, we have then different forms of derivative

Exercise 37

If :

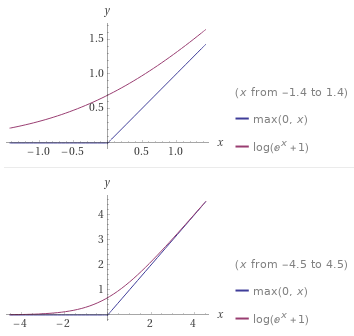
If :

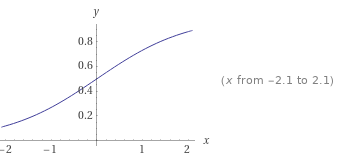
Exercise 38

1. Derivative of is:

And for function will be undefined, becouse left and right derivative values are not equal.

- Softplus





1. Saturated function is a function that has been mathematically "smoothed."

Firstly, an output close to 0 or 1 does not imply “high confidence”. It just means that the output is close to 0 or 1. The interpretation of this scenario depends on the problem and also the relative position of the neuron in the network. For example, a saturated neuron in the output layer renders the entire network useless. Such a network will always throw 1 (or 0) as it's output irrespective of the input sample. Secondly, saturation is only a problem in sigmoidal or tanh activations, not in linear ones.

The biggest problem with saturation, as you rightly pointed out, is the gradient falling to 0. The moment that happens, the learning process stops and the weights stop updating with the iterations. This is how gradient descent works. There needs to be a non zero gradient for the weights to be updated. An algorithm like backpropagation essentially employs gradient descent for updating (learning) the weights of the network. If the gradient is 0, the network won't “learn”.