Questions

1. **Advantages of hidden layers**

Hidden layers provide the discrimination necessary to be able to separate your training data. When you add layers, you increase the dimensional complexity of the data you can learn. Every time you add a layer, you change the shape of the discriminator.

Increasing the number of neurons will allow you to decrease your training error but it also reduces the amount of generalization.

1. **Disadvantages of having more hidden layers – hyperparameter**

Increase the dimensional complexity will increase the computing time and therefore affect performance. In other words, it may need longer training time.

1. **How can you represent more than two outputs?**

Having additional output nodes

1. **Can a NN overfit?** YES

If you train for too long, the model will start to overfit and learn patterns from the training data that don’t generalize to the test data.

To prevent overfitting, the best solution is to use more training data. A model trained on more data will naturally generalize better. When that is no longer possible, the next best solution is to use techniques like regularisation.

The most common ways to prevent overfitting in neural networks:

* Get more training data.
* Reduce the capacity of the network. Reduce the size of the model, i.e. the number of learnable parameters in the model (which is determined by the number of layers and the number of units per layer).
* Add weight regularization.
* Add dropout. Applied to a layer, consists of randomly “dropping out” (i.e. set to zero) a number of output features of the layer during training.

How you can prevent it:

* Change network complexity by changing the network structure (number of weights).
* Change network complexity by changing the network parameters (values of weights).

1. **Why Multilayer Perceptrons**

A Multilayer Perceptron can be used to represent convex regions. This means that in effect, they can learn to draw shapes around examples in some high-dimensional space that can separate and classify them, overcoming the limitation of linear separability.

Neural Networks

The human brain it is considered as a highly complex, non-linear and parallel information-processing system. These three operations can be replicated with a computer system using artificial Neural Networks. These are a set of algorithms, modelled loosely after the human brain, that are designed to recognise patterns.

Perceptrons

They are the simplest NNs, developed in 50s and 60s.

A single perceptron can only be used to implement linearly separable functions.

Components:

* External input links
* Internal input link -> bias
* Threshold
* One output link

NN = neuralnet(Formula~., Data, hidden = 0 , threshold = 0.001, stepmax = 1e+06, linear.output = FALSE)

* Formula = a symbolic description of the model to be fitted.
* Data = a data frame containing the variables specified in formula
* Threshold (umbral) = It determines, based on the inputs, whether the perceptron fires or not. Basically, the perceptron takes all of the weighted input values and adds them together. If the sum is above or equal to this value (called the threshold) then the perceptron fires and takes the activated value (typically 1); otherwise it takes the deactivated value (typically -1).
* Stepmax = the maximum steps for the training of the neural network. Reaching this maximum leads to a stop of the neural network's training process.
* Linear.output = This is used to specify whether we want to do regression linear.output=TRUE or classification linear.output=FALSE

Actually, you'll just set threshold when you aren't using bias. Otherwise, the threshold is 0.

Normalise the data

The goal of normalisation is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

KNN

Looks for K number of neighbours using one of the distance metrics Euclidian, Correlation and Mahalanobis.

* K high more computational expensive and accurate
* K low less computational expensive and accurate