From:

https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

A single-layer neural network can only be used to represent linearly separable functions. This means very simple problems where, say, the two classes in a classification problem can be neatly separated by a line. If your problem is relatively simple, perhaps a single layer network would be sufficient.

In fact, there is a theoretical finding by Lippmann in the 1987 paper “[An introduction to computing with neural nets](https://ieeexplore.ieee.org/abstract/document/1165576/)” that shows that an MLP with two hidden layers is sufficient for creating classification regions of any desired shape. This is instructive, although it should be noted that no indication of how many nodes to use in each layer or how to learn the weights is given.

A further theoretical finding and proof has shown that MLPs are [universal approximators](https://en.wikipedia.org/wiki/Universal_approximation_theorem). That with one hidden layer, an MLP can approximate any function that we require.

*Specifically, the universal approximation theorem states that a feedforward network with a linear output layer and at least one hidden layer with any “squashing” activation function (such as the logistic sigmoid activation function) can approximate any Borel measurable function from one finite-dimensional space to another with any desired non-zero amount of error, provided that the network is given enough hidden units.*

— Page 198, [Deep Learning](https://amzn.to/2IXzUIY), 2016.

This is an often-cited theoretical finding and there is a ton of literature on it. In practice, we again have no idea how many nodes to use in the single hidden layer for a given problem nor how to learn or set their weights effectively. Further, many counterexamples have been presented of functions that cannot directly be learned via a single one-hidden-layer MLP or require an infinite number of nodes.

Even for those functions that can be learned via a sufficiently large one-hidden-layer MLP, it can be more efficient to learn it with two (or more) hidden layers.

*Since a single sufficiently large hidden layer is adequate for approximation of most functions, why would anyone ever use more? One reason hangs on the words “sufficiently large”. Although a single hidden layer is optimal for some functions, there are others for which a single-hidden-layer-solution is very inefficient compared to solutions with more layers.*

— Page 38, [Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks](https://amzn.to/2vhyW8j), 1999.

**1) Experimentation**

In general, when I’m asked how many layers and nodes to use for an MLP, I often reply:

*I don’t know. Use systematic experimentation to discover what works best for your specific dataset.*

I still stand by this answer.

In general, you cannot analytically calculate the number of layers or the number of nodes to use per layer in an artificial neural network to address a specific real-world predictive modeling problem.

The number of layers and the number of nodes in each layer are model hyperparameters that you must specify.

You are likely to be the first person to attempt to address your specific problem with a neural network. No one has solved it before you. Therefore, no one can tell you the answer of how to configure the network.

You must discover the answer using a robust test harness and controlled experiments. For example, see the post:

* [How to Evaluate the Skill of Deep Learning Models](https://machinelearningmastery.com/evaluate-skill-deep-learning-models/)

Regardless of the heuristics you might encounter, all answers will come back to the need for careful experimentation to see what works best for your specific dataset.