# Peeking into the black box: Neural Networks

"Neural Networks" and "Deep Learning" have become quite the buzzwords nowadays, even within more traditional industries like insurance. However, ask any Actuary what's the most commonly used model he/she uses, and it is unlikely you'll hear any answer other than GLMs or GBMs.

## What is a Neural Network?

A neural network, as cool as it sounds, is nothing more than a statistical model. Much like its older brother - The simple linear regressor, the neural network also uses parameters and optimization to construct a line-of-best-fit for whatever data it is being presented. Unlike its old brother however, it is modelled from the way biological neural networks in our brains process information.

## Example: Foo Insurance Ltd

We find ourselves back at Foo Insurance Ltd with a rather interesting problem. Esther, our actuarial manager has some data that has been cleaned and normalized from the company's motor book. She wants to build a model to predict if a policyholder is likely to incur an accident claim based on the age of the car, and the policyholder himself/herself. She has heard a lot about neural networks and wants to test the model’s capabilities on this problem.

INSERT DATASET SCREENSHOT

INSERT DATASET GRAPH

As you can tell, the dataset isn't linearly separable, which is the case in most situations. Although 2 features are a little over-simplified, it will serve the purpose for this article and extends to higher dimensions.

## First Iteration – Single Layered Network

Excited about playing with this new model, Esther proceeds to build a very simple network that looks something like this:

INSERT NODE CHART HERE

Recall that the formula for a neural network output on this dataset (without any bias terms) with 1 layer would look like:

Now, both outputs will give us some kind of probability or likelihood that the response is either 1 or 0. For example, if $o\_1$ gave an output of 0.6 and $o\_2$ gave 0.4, the particular data point would then be classified as green instead of grey.

By utilizing some good old high school linear algebra, recall that we can construct a line with this system of 2 equations, particularly:

This equation represents a line which is also known as a decision boundary and is what separates a policyholder which incurs a claim and one that does not.

INSERT GIF HERE

Looks like this first iteration does not really do a good job in properly classifying the 2 types of policyholders. Note that with the current network architecture, the boundary is restricted to rotating on its own axis. Wouldn’t it be useful if we could shift this line up and down?

## Second Iteration – Single Layered Network with Bias Terms

Esther now recalls the concepts learnt in Simple Linear Regression, where an intercept is included in the model to give predictions some minimum value where the response turns out to be 0. She then proceeds to add an intercept (also known as a bias term) to the 2 initial equations, which gets:

## Third Iteration –Network with Hidden Layer

Even though it is a little better now, remember similar to real life, the dataset here is not linearly separable and our current network has is only capable of producing a straight line to differentiate the 2 categories.

Will adding more layers of neurons give the model non-linear capabilities? Well, going back to high school mathematics yet again, recall that a linear combination of 2 linear equations still produces something linear!

## Fourth Iteration – Multi Layered Network with Activation Functions