Feed Forward Neural Networks

Andrew Wang

Background Content

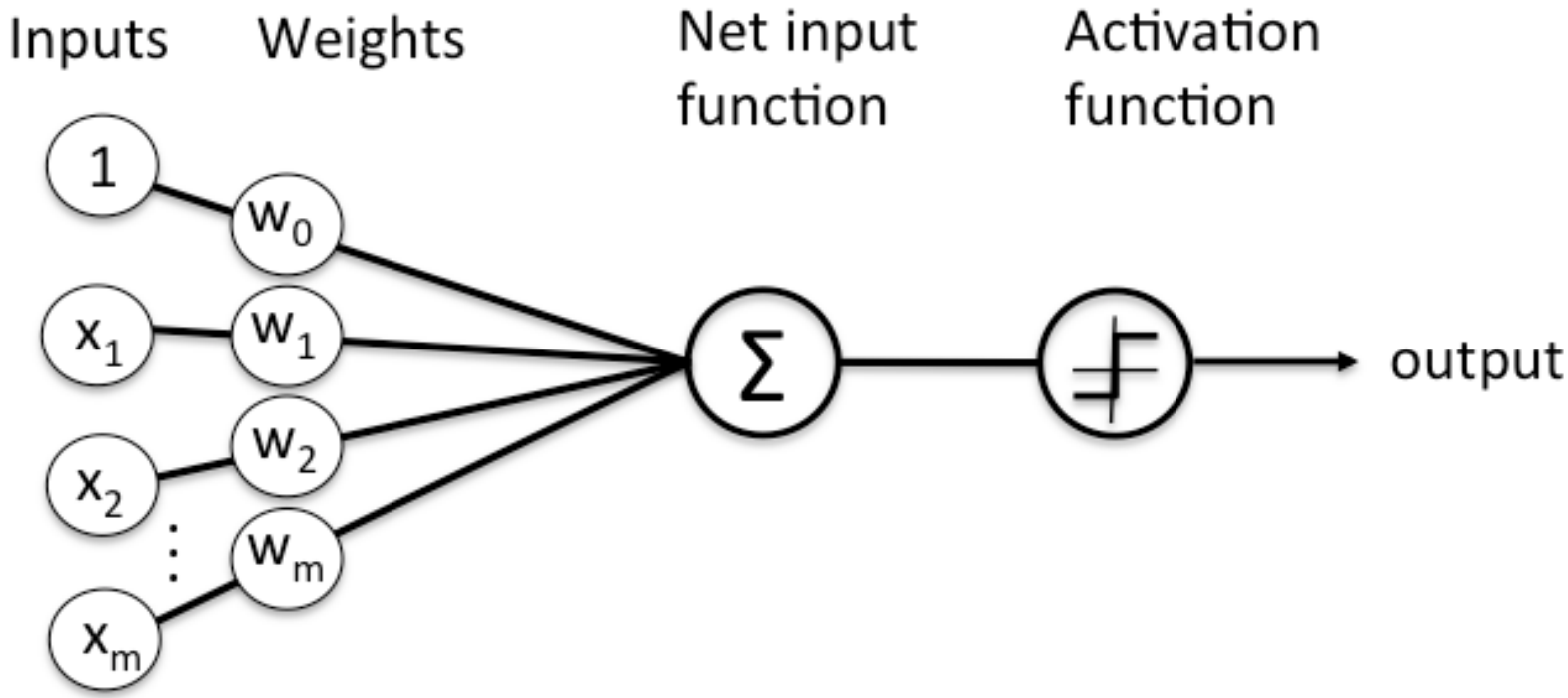
What is a neural network and what is a node?

- Set of algorithms (modelled after human brain) designed to recognize patterns
  - Input data is numbers
  - Output data is class (ex: spam vs not spam) or continuous range of numbers (ex: 0.1, 0.153, etc.)
- Neural network typically comprised of several “layers” of “nodes”
- Hierarchy
  - Node = neuron = perceptron
    - Where computation/math happens
    - Algorithm
    - Computer imitation of human brain’s neuron
  - Layer = group of neurons organized together
  - Neural network = bunch of layers of nodes organized and connected in various ways

More on what a node is

- Each node has 4 parts
  - Input values
  - Weights and bias
    - Weights = array of numbers that helps control magnitude and sign of output based on input
    - Bias = number that controls when activation function is activated
      - **Similar to “b” in y = mx + b -> a number that helps you to fit data better/shift output up+down**
  - Net sum
  - Activation function “F”
    - Function (ex: sigmoid, tanh, etc) that introduces non linearity to data
      - **Allows neural network to better simulate brain**
    - Without activation function, neural network would just be 1 big linear classifier
- How does node work? How is output calculated from input array of numbers?
  - Output = Activation_Function {(Input x Weights) + Bias}

Visualization of node (aka perceptron)



Picture from: https://wiki.pathmind.com/neural-network

Why do we need weights and biases pt1

- Imagine we want to use neuron to predict car price (in $) based on:
  - Year car was made
  - # Miles driven
- We would expect
  - Higher year (more recent car)-> Higher price -> weight for year is (+)
  - Higher # miles driven (more used car) -> Lower price -> weight for price is (-)

http://webcache.googleusercontent.com/search?q=cache:vj72OfznF58J:https://medium.com/fintechexplained/neural-networks-bias-and-weights-10b53e6285da&hl=en&gl=us&strip=1&vwsrc=0

Why do we need weights and biases pt2

- Recall neuron made of multiple parts
  - Input values: Assume it is a 1 by 2 array = [year, # miles driven]
  - Weights array: Let =

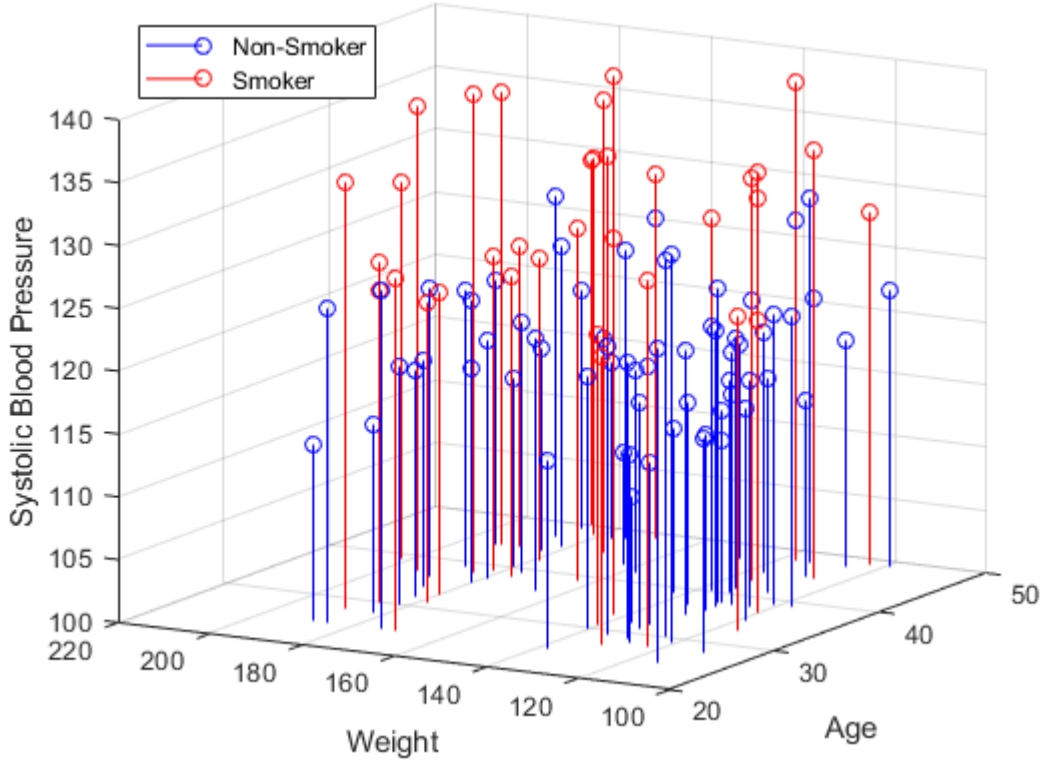
[weight for year,

weight for miles driven]

  - If we just used input values and weights array, price would be
    - Output = Price = year*weight_for_year + miles_drive*weight_for_miles_driven
    - Let weight_for_year = 2
    - Let weight_for_miles_driven = -1
    - Assume year = 2008, and miles driven = 50,000
    - Price would = 2008*2 - 1*50000 = 4016 – 50000 = - 45,984 dollars
  - As you can see, using weights and bias alone doesn't make sense (data might not fit very well or make sense) -> introduce bias to make better predictions of output based on input
  - Bias: Let bias = 50,008
  - Now, output = price = year*weight_for_year + miles_driven*weight_for_miles_driven + bias
    - Output = price = 2008*2 – 1*50000 + 50008 = $ 4,024 -> makes more sense (to have positive price rather than a negative price)
  - Activation function "F" –ignore for now, look at next slide to understand why we need activation functions

Why do we need activation functions pt 1

- Recall: activation functions introduce non-linearity to data
- If we tried to classify non smokers vs smokers in picture to right with linear classifier (i.e a flat/linear 3-D plane), we wouldn't be able to since the data is NON-LINEAR
- BUT if we used a curved 3-D plane, we WOULD be able to differentiate non-smokers and smokers

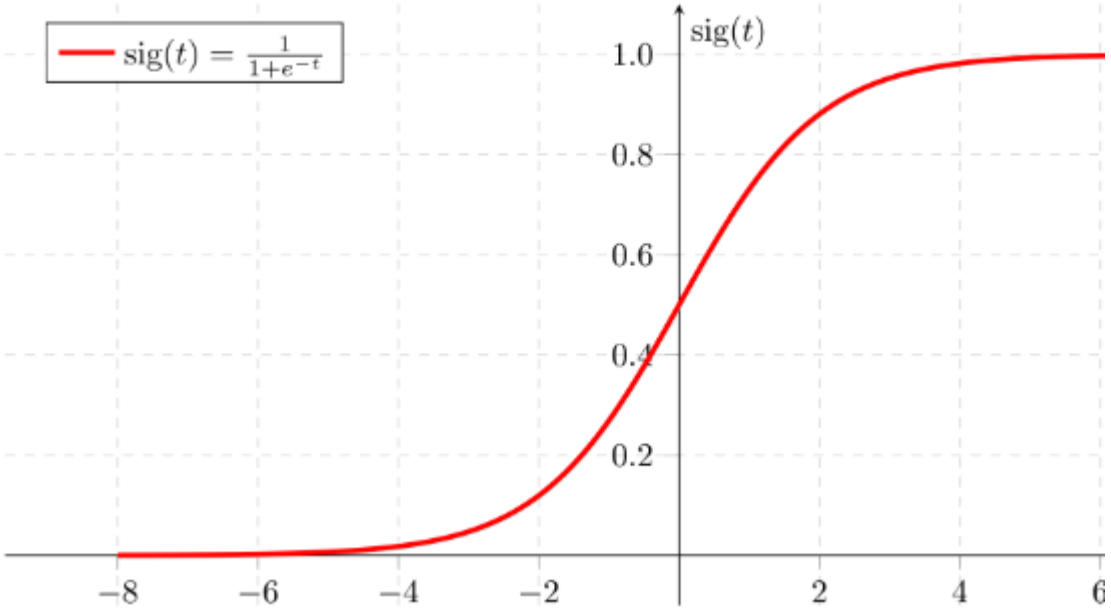


https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-models-84ba9f82c253

Why do we need activation functions pt2

- If a neuron only had inputs, weights, and biases, the output would be:
  - Output = Weight*Input + Bias = wx+b
  - Model has degree of 1 -> identical to linear classifier -> can't classify complex nonlinear data
- BUT if we made the model non-linear by adding activation function at end (output = activation_function(wx+b)), the model would not have degree 1 and would be able to classify non-linear data

Ex of activation function (sigmoid)



What is a neural network

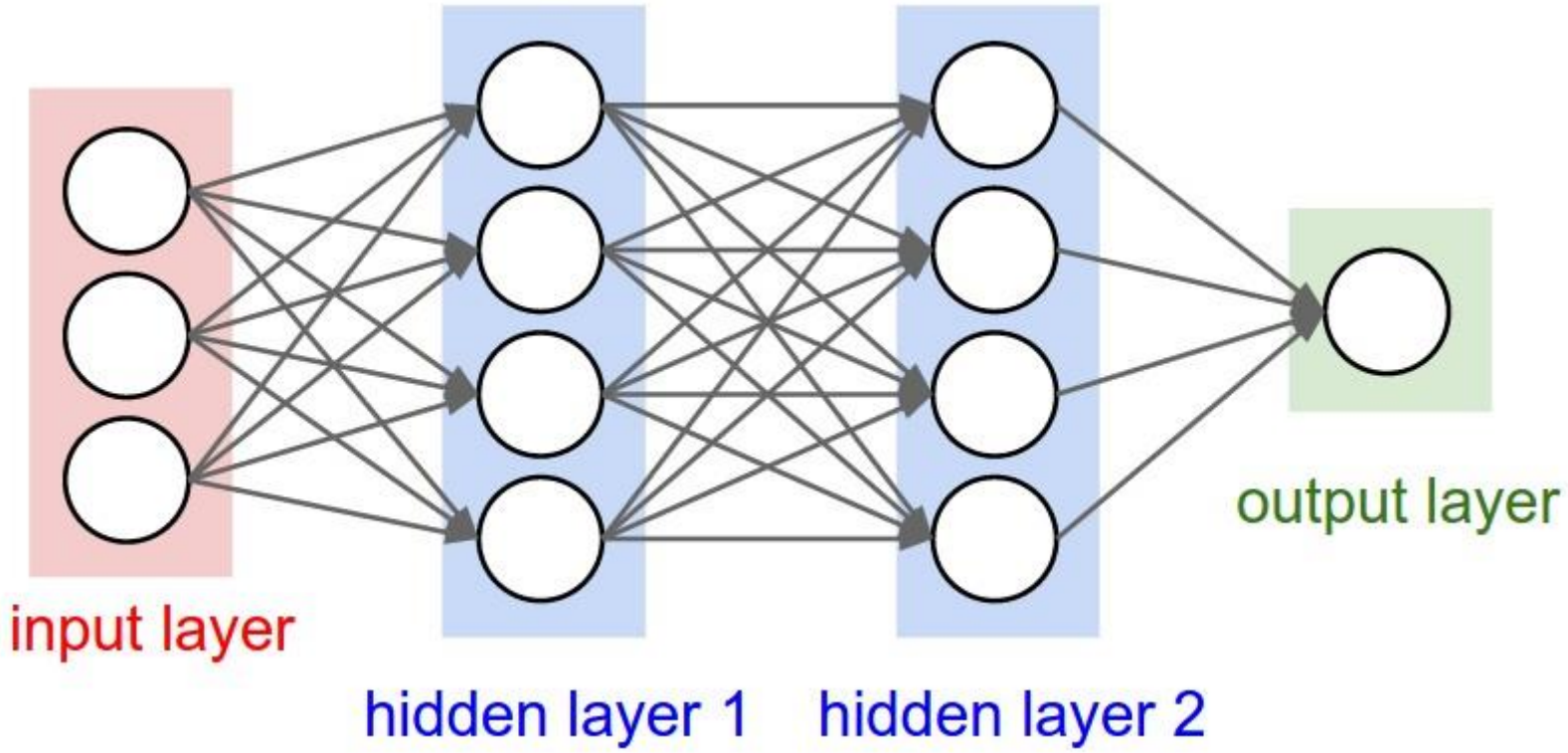
- Recall that a neural network = bunch of organized neurons linked together

What makes a neural network a feed forward neural network?

1. Perceptrons (nodes) organized by layer
    1. 1st layer – Input
    2. Middle layers – Hidden layers
    3. Last layer – Output
2. Each perceptron in one layer connected to each perceptron in next layer (“info fed forward from one layer to next”)
3. No connection between perceptrons in same layer

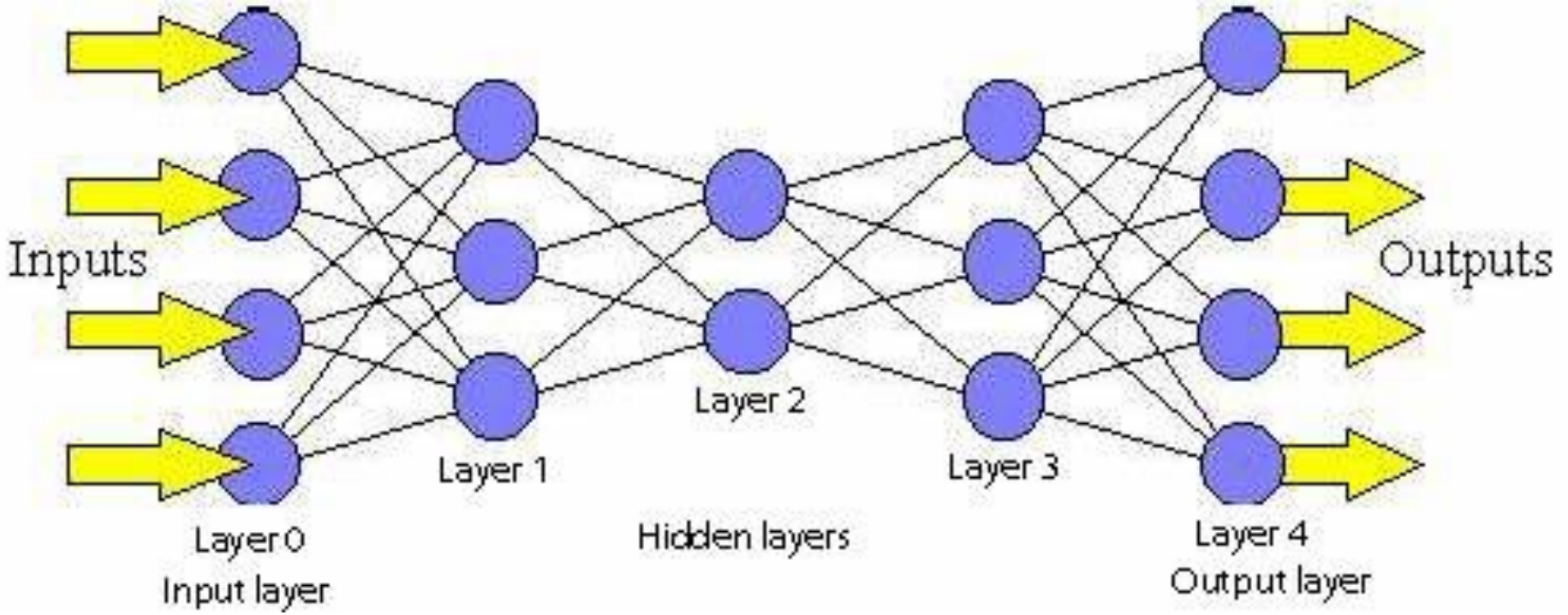
https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html

Picture of typical feed forward neural network



- Picture from: https://stackoverflow.com/questions/35345191/what-is-a-layer-in-a-neural-network

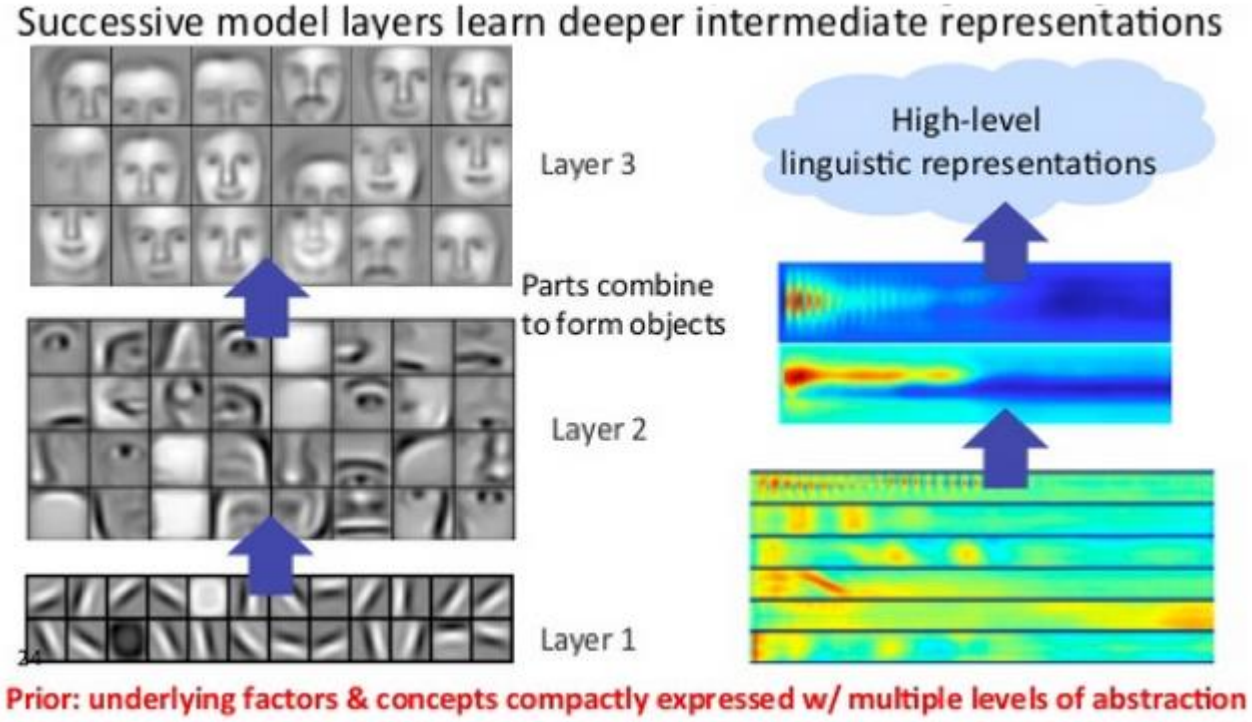
Another example of feed forward neural network



https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html

Why do we need hidden layers?

- We need at least 2 layers (input and output layer), but we can also have hidden layers -> why?
- Adding multiple hidden layers allows us neural network to learn higher level features in later layers from lower level features in earlier layers



https://wiki.pathmind.com/neural-network

How do neural networks “learn”?

- “learning” – refers to adjusting weights and biases in layers/neurons to achieve better performance
  - Better performance = better predicting outputs based on input
- In order to learn:
  - Need error function to quantify difference between prediction and true/correct result
    - Loss function – quantifies difference for single training example
    - Cost function – average of loss function over entire training set
  - Need to adjust weights and biases based on error, to reduce error
    - Back propagation – algorithm that uses gradients and partial derivatives to adjust weights and biases in order to reduce loss (achieve higher accuracy)

WAIT - How are layers, neurons, weights, biases, initialized?

- Random Initialization - Generally weights initialized with random numbers and biases initialized with 0
- Other initialization techniques:
  - “He” initialization
  - “Xavier” initialization

For more info: https://towardsdatascience.com/weight-initialization-techniques-in-neural-networks-26c649eb3b78

General Structure of Feed Forward NN Code

- Explain and walk through MNIST code in PyTorch

Works Cited and links for more details

1. https://wiki.pathmind.com/neural-network
2. https://stackoverflow.com/questions/35345191/what-is-a-layer-in-a-neural-network
3. https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html
4. http://webcache.googleusercontent.com/search?q=cache:vj72OfznF58J:https://medium.com/fintechexplained/neural-networks-bias-and-weights-10b53e6285da&hl=en&gl=us&strip=1&vwsrc=0
    1. https://medium.com/fintechexplained/neural-networks-bias-and-weights-10b53e6285da
5. https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-models-84ba9f82c253
6. https://towardsdatascience.com/weight-initialization-techniques-in-neural-networks-26c649eb3b78
7. https://stats.stackexchange.com/questions/179026/objective-function-cost-function-loss-function-are-they-the-same-thing#:~:text=The%20loss%20function%20computes%20the,of%20the%20entire%20training%20set.
8. http://neuralnetworksanddeeplearning.com/chap2.html