

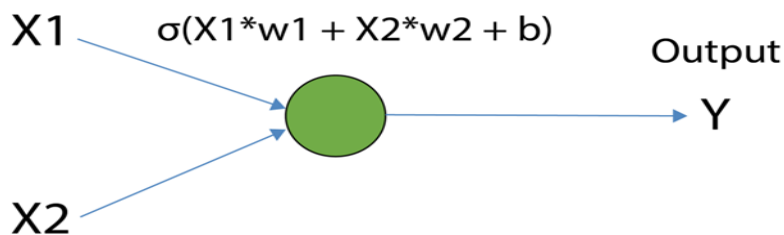
## Assignment B-3

Problem Statement:

Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months. (Customer Churning)

Neural Network Concept:

Input



The image above is called one **node** or **unit** in the deep learning world, which is represented by the green circle. We call the parameters of the model as **coefficients** and **intercepts**. In deep learning models, the parameters are referred to as **weights** ( $w$ ) and **biases** ( $b$ ):

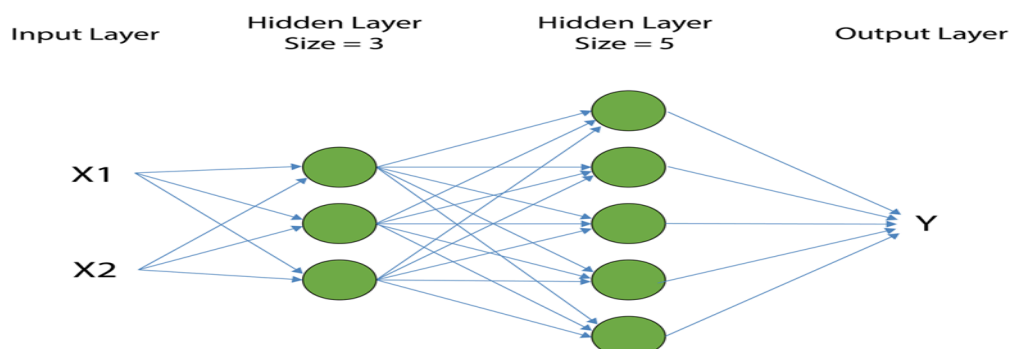
Forward propagation—the propagation of data through the network, multiplying the input values by the weight of each connection for every node, and

Backpropagation—the calculation of the gradient of the loss function with respect to the weights in the matrix,

Gradient descent—the optimization algorithm that's used to find the minimum of the loss functions. An overview of the logistic regression model with two-dimensional input can be seen in the following image.

Activation Function - a nonlinear function is applied to the sum of the weighted inputs and the bias term is used to compute the final output of the node. Ex Sigmoid, tanh, ReLU

It is also possible to build multi-layer neural networks by stacking multiple layers of processing nodes after one another, as shown in the following image.



**Hyper parameters-** The parameters that are required to be selected by the developer are called hyperparameters and include parameters such as the number of layers and the number of nodes in each layer.

**Loss Function-** When learning the optimal parameters (weights and biases) of a model, we need to define a function to measure error. This function is called the **loss function** and it provides us with a measure of how different network-predicted outputs are from the real outputs in the dataset.

Ex.- • `mean_squared_error.`, `mean_absolute_error.`, `mean_absolute_percentage_error.`, `binary_crossentropy`, `categorical_crossentropy`

During the training process, we keep changing the model parameters until the minimum difference between the model-predicted outputs and the real outputs is reached. This is called an **optimization process**.

`batch_size` -this argument determines the number of data examples to be included at each iteration of the optimization algorithm. `batch_size=None` is equivalent to the standard version of gradient descent, which uses the entire dataset in each iteration.

`epoch` - this argument determines how many times the optimization algorithm passes through the entire training dataset before it stops.

**Keras :** Keras runs on top of open source machine libraries like TensorFlow or Cognitive Toolkit (CNTK). TensorFlow is the most famous symbolic math library used for creating neural networks and deep learning models. TensorFlow is very flexible and the primary benefit is distributed computing. CNTK is deep learning framework developed by Microsoft. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. TensorFlow is very powerful library but difficult to understand for creating neural networks.

Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models

Database Used:

<https://www.kaggle.com/datasets/barelydedicated/bank-customer-churn-modeling>

Python : Colab, spider or similar platform

YT Ref: <https://www.youtube.com/watch?v=p5FydQTHtP0>

Code (As attached) & Graphs (wherever applicable) -----

Metrics used for performance measurement: \_\_\_\_\_

**Conclusion:** NN are self-learning models based on input data. NN model can be single layer or multi-layered. By using Keras libraries multilayer NN can be developed easily. Once

trained NN can produce accurate result with partial data. NN are used to solve complicated engineering problems. Their training require more time as compared to other models of ML.