Introduction to ANN, TensorFlow and Keras

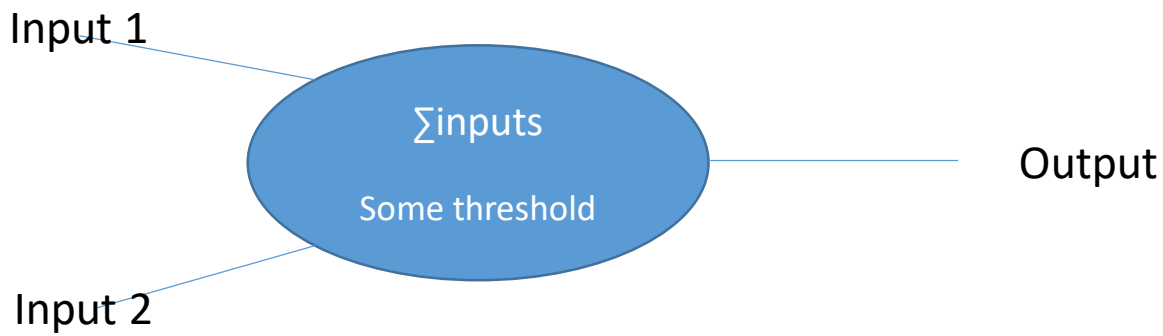
Topics Covered in the week

- Introduction to Artificial Neurons
- McCulloch – Pitts Neuron
- Rosenblatt's Neuron
- Introduction of Artificial Neural Networks
- Introduction to TensorFlow and Keras
- Case Study

Introduction to Artificial Neurons

- The inception of research around the artificial neurons was inspired from the biological neuron.
- The objective was to develop a system that can do tasks on its own.
- Researchers tried to implement logic gates through the artificial neurons in the early research.
- An important work was done by Warren McCulloch and Walter Pitts in this field. They created the McCulloch-Pitts neuron.

McCulloch-Pitts Neuron



- The McCulloch-Pitts neuron had binary inputs and outputs where inputs could be many but the output was just one.
- The input was multiplied with weights and summed up. That result would later be fed to a threshold function/ step function to determine the output.

Rosenblatt's Neuron

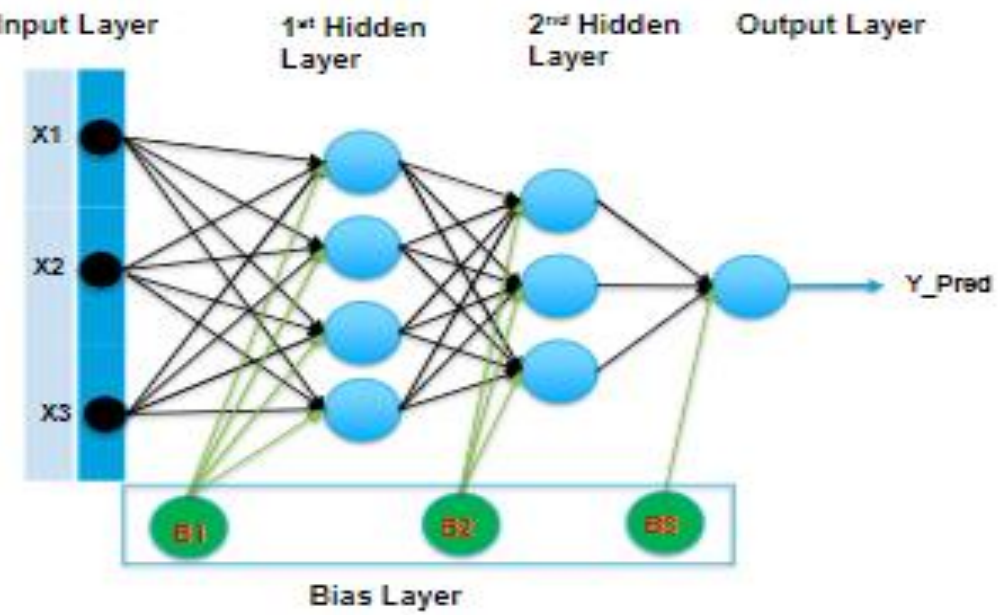
- Rosenblatt's Neurons were an upgrade to the earlier McCulloch-Pitts neuron.
- Introduced different weights for inputs, bias input, captured errors and a learning rule.
- The neuron successfully captured the AND and OR gate w/o the need of manually setting the threshold.
- Thus the concept of learning from data was introduced.

However,

- The neuron could not capture XOR gate. The gates where the targets were not linearly separable in the binary input(2-D) space.
- This led to further research and eventually the development of the ANN.

The Artificial Neural Networks

- It was demonstrated that a single neuron could not capture gates like XOR but a network of gates could do so.
- This gave birth to the ANN.



The basic structure of the ANN

TensorFlow and Keras

- TensorFlow is an end-to-end open source machine learning library by google that will help you develop and train ML and DL models.
- It offers easy model building support and makes development easy.
- Keras is a high-level neural networks API, written in Python.
- Tensorflow.Keras is TensorFlow's implementation of the keras API specification.

Relevant Tensorflow functions

- Tensors - A tensor is a generalization of vectors and matrices to potentially higher dimensions. Internally, TensorFlow represents tensors as n-dimensional arrays of base datatypes. Quoting from the official tensorflow website, A tf.Tensor has the below properties-

  - A data type
  - A shape
    - *Each element in the Tensor has the same data type, and the data type is always known. The shape (that is, the number of dimensions it has and the size of each dimension) might be only partially known.*

TensorFlow and Keras - Contd

Special tensors

- Tf.Variable
- Tf.constant

Tensorflow allows handling mathematical operations quite similar to NumPy . Some important functions –

- Tf.random.uniform – outputs random values from a uniform distribution
- Tf.random.normal – outputs random values from a normal distribution

Similarly, there are other random generators.

TensorFlow and Keras - Contd

Now we list some relevant operators. Note that there are numerous of these and all can't be covered here. Just mentioning a few-

For multiplication on matrices,

- Tf.matmul – for matrix multiplication
- Tf.multiply – for elementwise multiplication
- Tf.GradientTape – for finding the derivative of functions
- Tf.assign – assign values to a variable tensor
- Tf.assign_sub/ Tf.assign_add – sub/add to the given variable tensor.
- Tf.math – for other math operations

Case Study – Linear regression using TF2.0

- In this case study, we will build a very simple linear regression model from scratch and will try to predict car prices for our test set.
- We will use the functions/operators we have discussed as will see how to work with tensorflow for mdel building.
- Note that, in the coming weeks, we will be talking about tf.keras sequential models.

Questions?

Thank You!
Happy Learning!