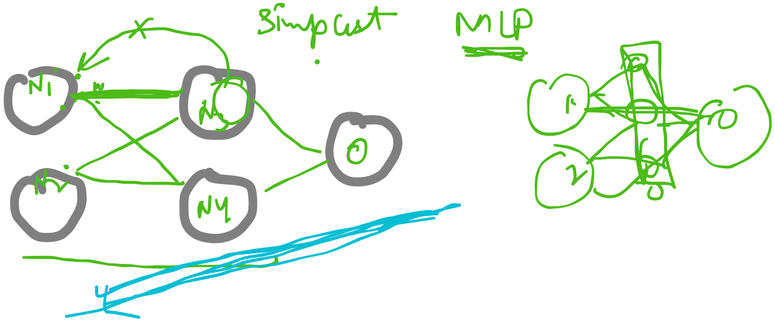
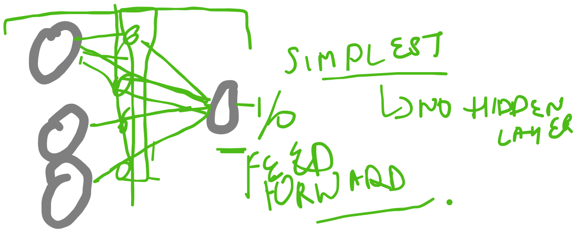
PERCEPTRON

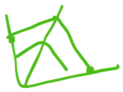
neural networks use concept of perceptron.

FEED FORWARD NEURAL NETWORK



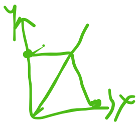
MLP
LV
FFN





LINEAR DATA
→ X & y are linearly related

NON - LINER
→ Text, Images
→ ACTIVATION



PERCEPTRON

- 1 Input layer and 1 output node.
- being perceptron, vey simple, it can't be used in real life

MLP

- Neural networks build upon perceptron.
- uses the functionality of perceptron, but they hidden layers.
- hidden help us understand complex data.

shallow NN

1 hidden layer

dense neural network

1+ hidden layer

TENSORFLOW

- python library

Pandas numpy

API

- Application Programming Interface

- takes input and return the output

-

KERAS:

- High level API

- Can be integrated with many

- one such is tensorflow

Build layer different layers.

x= sum(1, 2)

3

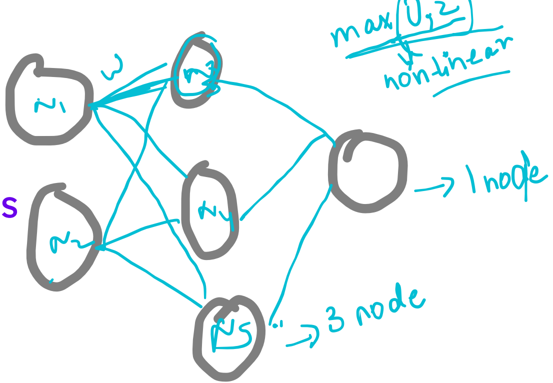
HIGH LEVEL

1+2+...+n

low level scratch

SEQUENCE :

- Data will flow from one layer to the



Dense

- helps to create fully connected layers

- no. of nodes

input_shape

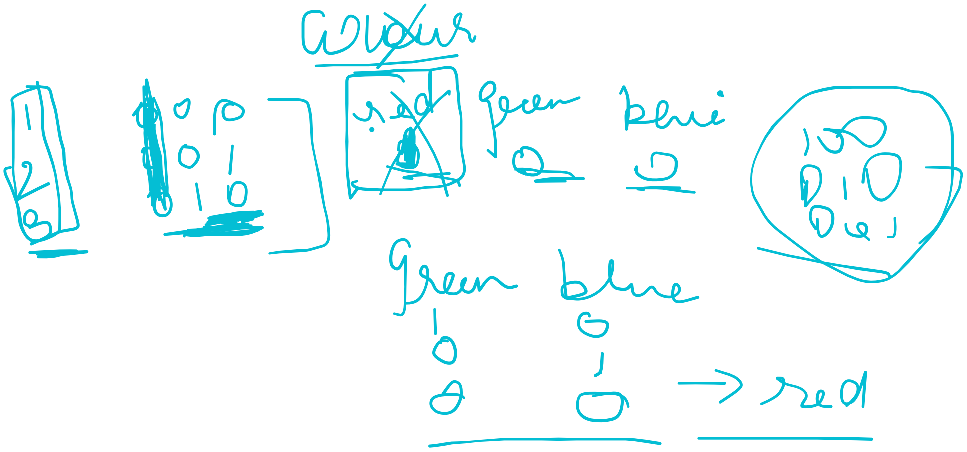
- activation function

--- decides what value should we assign to a node.

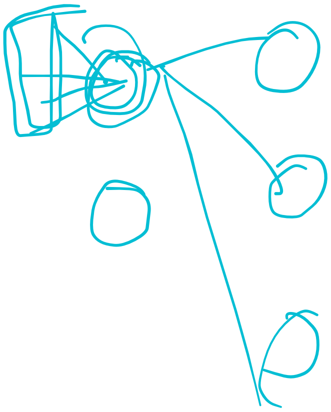
red

green

blue



neural network models expect numpy array to be input neural network models expect numpy array to be input



tensor

- datatype in tensorflow
- store data

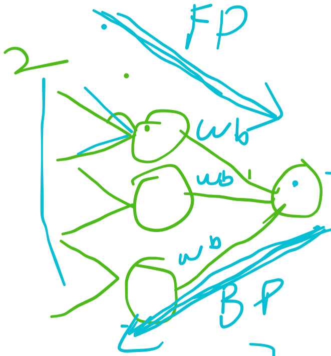
PARAM = WEIGHTS

$\left.\begin{array}{l} F = \boxed{22} \\ N = 32 \end{array}\right]$ $\quad \underline{22} \times \underline{32} = \underline{704}$

$\underline{704} + \underline{32} = 736$

$\boxed{= 736}$

$\Sigma = \underline{w_n x_n + b} \rightarrow$ Bias



70 50

20 = error

N = 20

= EPOCH

Input of output node = $\underline{20}$ $\boxed{1}$

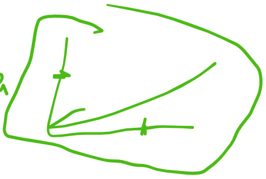
1 bias

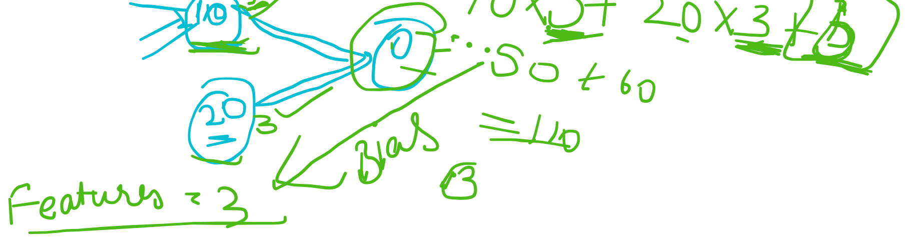
$\underline{20 + 1 = 21}$

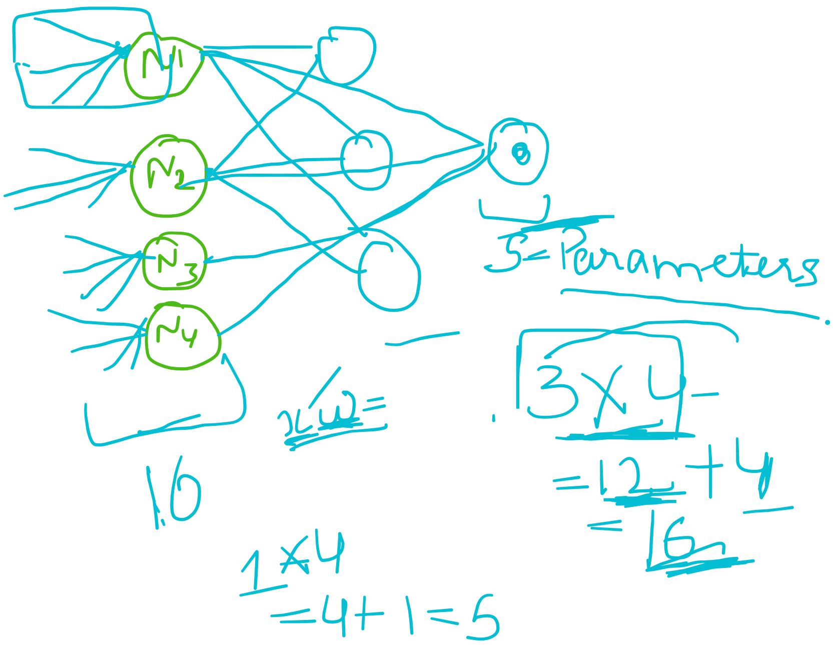
Bias

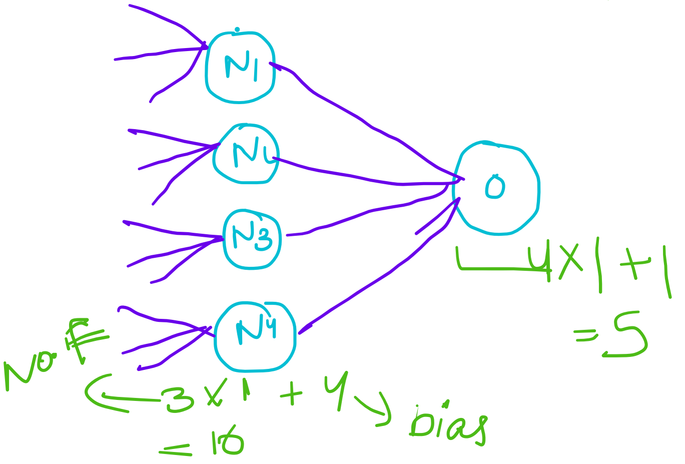
bias non linearity in NN.

NLP
Big Data









every node has 1 bais
every node has weight associated with it

parameter = weight+bias

Param of Input layer =
[3×1]×4 =
where 4 is no. of nodes