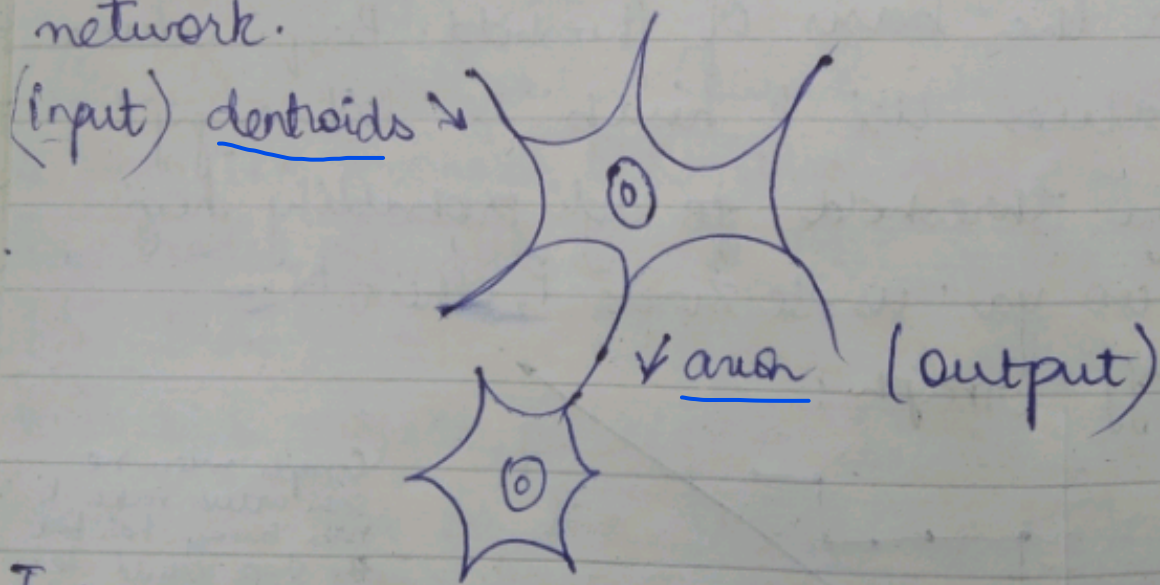
Data Overfitting or models , Data Underfitting or models.

* Every classification model will have same accuracy and other majors.

8/11/21 Class: 10

* Artificial Neural Network : (ANN)

• Neural Network : mimics human neutrons network.



Takes input from sensors (input may be multi
process it and give output (output may

(ie multiple neuron)

→ Input layer :

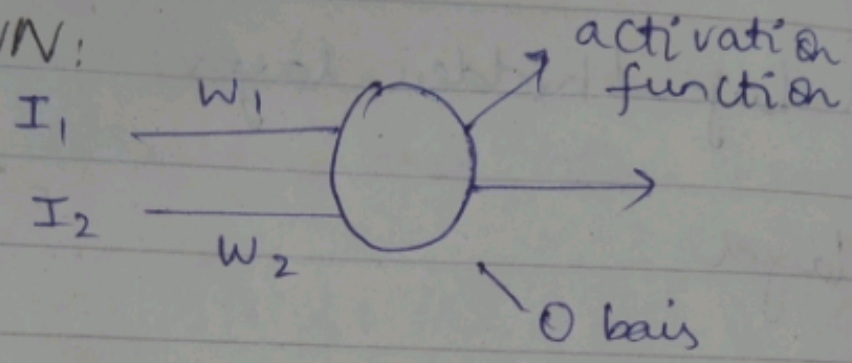
→ Internal layer / hidden layer :

→ Output layer :

* Deep learning : No. of hidden layer use.
→ increases time complexity
→ computing
→ no. of hidden layer is the depth of deep learning.
* Drawback : Interpretation of working is not understandable.
* Advantages :
→ How to give input to model should be known
→ Interpretation of output is needed.
* Conditions : Data should be numeric

→ Data should be normalize (-1,1)
→ Other than these data neural network will not work.

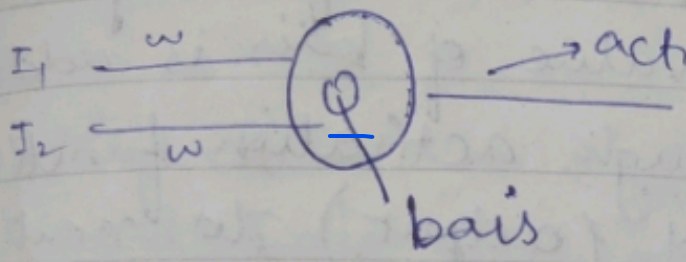
* ANN:



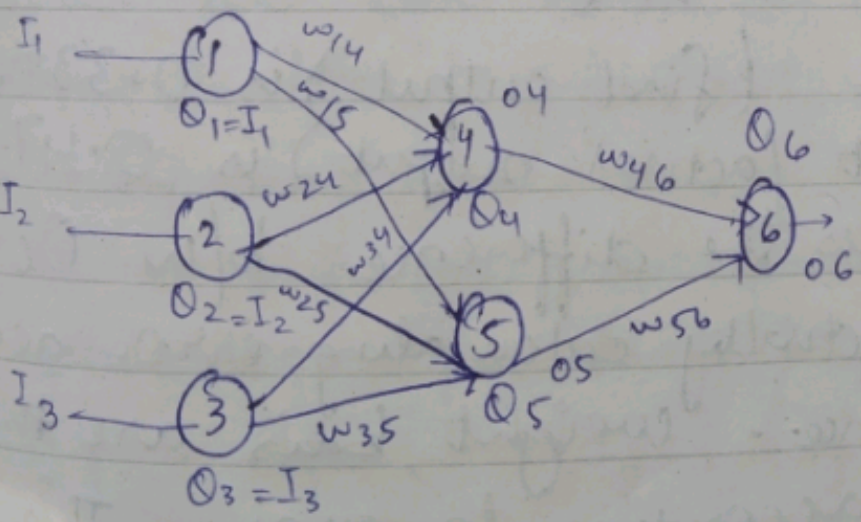
→ give weights to every input.
→ multiple value of input and weights and then sum all values.
→ weight is given by correctors.
→ Activation function is non-linear regress
→ Activation function do this;
→ data should be normalize b/w (-1 to 1) to (0 to 1)

* Simplest Neuron / Single Perception:
→ only 1 neuron is used

- It has capability to give some output



- weight is given according to neuron
- initially weight is provided randomly
- we can only manipulate weights and bais



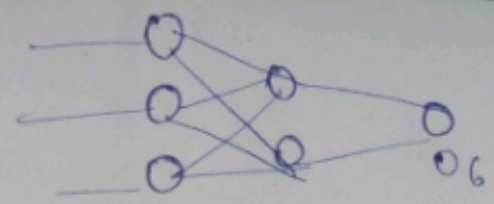
⇒ Multi Layer Perceptor (MLP): If atleast 3 layers are present in model.

Initially simple input is forward to output of its node. weighted sum of every node is summed up and value of bais is added and passed through activation function then value is forward (output) to next layer as input.

$$I_4 = I_1 \times w_{14} + I_2 \times w_{24} + I_3 w_{34} + \theta_4$$

$$O_4 = \frac{1}{1 + e^{-I_4}}$$

Suppose O_6 (final output) is 0.33 and T_6 (Target (actual output) is 0.47 then model (takes the difference b/w O_6 & T_6. This is actually calculating error according its difference. (weight bais are then updated according to error. This is known as back propagation (learning is done on its basis).



← back Propogation

The steps taken from actual starting point of inputs is the final output. → feed foward

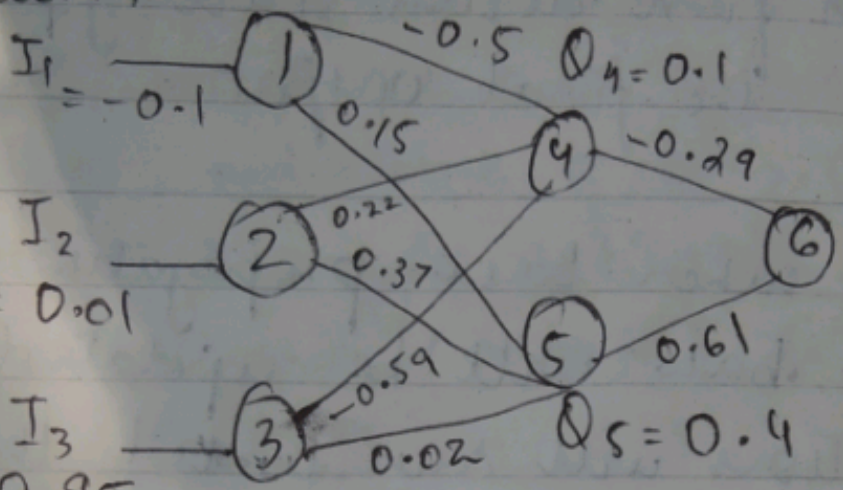
- Every time when back propogation occurs, weights and bais will be updated.
- These iterations will run until the model gives error less then the decided threshold.
- Threshold: Any decided value, the error rate which we can bear in our model b/c nothing is 100% efficient.
- When comparing threshold with error (we neglect -ve sign) but we send actual value of negative sign for back propogation.

$$Err_j = O_j (1 - O_j)(T_j - O_j)$$

According to our example:

$$Err_6 = O_6 (1 - O_6)(T_6 - O_6)$$

Question:



T = 0.5

O1 = -0.1

O2 = 0.01

O3 = 0.95

$I_4 = -0.1 \times -0.5 + 0.01 \times 0.22 + 0.95 \times$
$+ 0.1$

$= -0.4082$

$O_4 = \frac{1}{1 + e^{I_4}}$

$= \frac{1}{1.664845}$

$= -0.1 \times 0.15 + 0.01 \times 0.37 + 0.95 \times 0.02 + 0.4$

$I_5 = 0.4077$

$I_4 = -0.4083$ $\quad O_4 = 0.399$

$I_5 = 0.4077$ $\quad O_5 = 0.6005$

$I_6 = 0.5805$ $\quad O_6 = 0.6411$

$Err_6 = |-0.0246|.$

Class : 11

23/11. Class : 12

* No of neutrons on a layer are our choice.
* No. of neutrons on input layer : N-attribute
* Data should be normalized i.e b/w 1 to -1 or 0 to 1, then ultimately the output will be normalized.
For eg: Income = 100,000 so normalize it in decimal values i.e ÷ by 100,000 (max value of the column income)
* Binary attribute ko 2 columns/attributes mein divide kardengy. When the data is nominal so jitni possible domains hongi utne mein attribute break kardengy.
For eg: A column have values of valid and invalid will be divided into 2 columns, 1 of attribute having valid values and 1 of invalid
This is also called binary coding.
* Same happens on ordinal data.
* Input layers par no. of neutrons increase

* Supervised learning.

Output column is 1. (No. of neurons in output layer depends on the no. of classes / prediction.)
If for class I, target is 1 then for all other classes target will be 0.

* Due to the activation function learning is possible if error is never 0 due to activation function.

* Har neuron k upper we have applied symbolic regression.

* Activation function makes the data complex which results in generating better results; otherwise linearly data ko leta rehta ho.

=> * RNN: Loops are involved i.e output becomes next input. Hidden layer par recurrency hoti hai. (Recurrent Neural Network)

* Image Processing is mostly done on CNN

* Image ko nominal kar k bhejty hien. It has no. of pixels and each pixel is di

into 3 values of R, G, B or in grayscale 2 value 1 or 0 (i.e black & white)

* Text / Image ke features nikaly jaty hien or selection ke baad learning hoti hai.

* LSTM is extension of RNN. (Long Short Term Memory)

⇒ Application : OCR (Optical Character Recognition)

• Extraction of Text from image.

• for eg: Capture no. plate of overspeeding ca

* Image is pre-processed i.e visible karna i quality bartana, feature nikalna etc is pre-processing.

* Learn karna hai ke output of 52 mien se kisi class mien classify krdien. (52 for small & letters).

* If image is greyscale then no. of neuro in input layer will be no. of total pixe

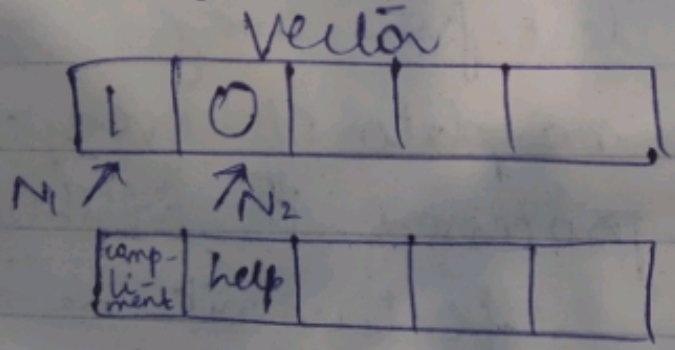
25 x 25 = 625 pixels

* ANN is good for parallel processing.
* Normals laptops / PC's are not good enough for image processing.

Sentiment Analysis:

Conversion of text to gain output that tells it is +ve, -ve or neutral.

* Vector size will be the ~~size~~ no of words in the dictionary.



* Drawback of NN is interpretation.