Neural Networks

A **Neural Network (NN)** is a computational model inspired by the human brain. It consists of **layers of neurons** that learn patterns from data. Neural networks are widely used in **deep learning** for tasks like image recognition, NLP, and reinforcement learning.

🎯 How Neural Networks Learn

Neural networks learn using **backpropagation** and **gradient descent**:

1. **Forward Propagation** → Data flows through the network, making predictions.
2. **Loss Calculation** → Measures how far predictions are from actual values.
3. **Backpropagation** → Computes gradients of the loss w.r.t. weights using the chain rule.
4. **Gradient Descent** → Updates weights using optimization techniques like **SGD, Adam, or RMSprop**.

🏗️ Common Types of Neural Networks

🔷 **Feedforward Neural Network (FNN)** – Basic type; used for structured data.

🔷 **Convolutional Neural Network (CNN)** – Designed for image processing.

🔷 **Recurrent Neural Network (RNN)** – Used for sequential data (e.g., time series, NLP).

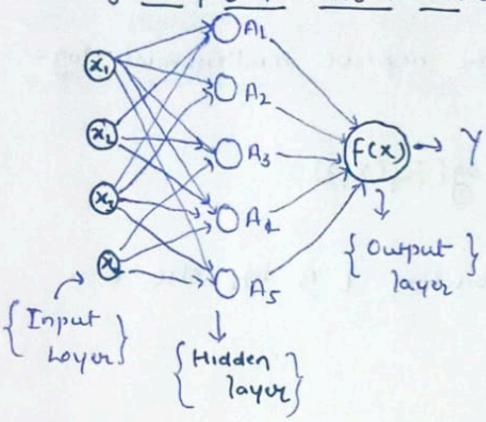
🔷 **Transformers** – Modern NLP models (e.g., BERT, GPT).

🔷 **Autoencoders** – Used for dimensionality reduction and anomaly detection.

Day - 9 Neural Networks

→ A neural network is a machine learning program, or model that makes decision in a manner similar to the human brain, by using processes that mimic the way biological neurons work together to identify phenomena, weigh options and arrive at conclusion.

- Single Layer Neural Network



$$f(x) = \beta_0 + \sum_{k=1}^{K} \beta_k h_k(x)$$

$$\Rightarrow \beta_0 + \sum_{k=1}^{K} \beta_k \left(w_{k0} + \sum_{j=1}^{p} w_{kj} x_i\right)$$

$$A_k = h_k(x) = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} x_j\right)$$

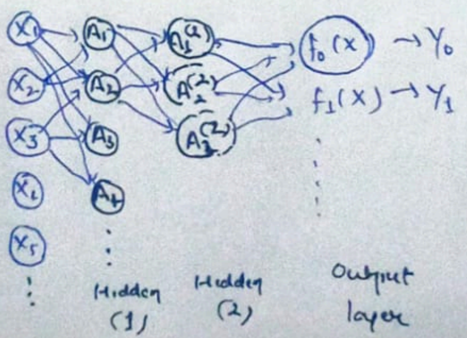
are called activation in the hidden layer

$g(z)$ → activation function

popular are { sigmoid and rectified linear }

- Activation functions in hidden layers are typically non linear, otherwise the model collapses to the linear model.
- So the activations are like derived features - non-linear transformation of the linear combinations of the features
- The model is fit by minimizing $\sum_{i=1}^{n} (y_i - f(x_i))^2$
  (e.g) for regression (loss)

- Complex Neural Network



Details of Output layer

- Let $Z_m = \beta_{m0} + \sum_{l=1}^{k_2} \beta_{ml} A_l^{(2)}$, $m = 0, 1, \ldots, 9$ be 10 linear combination of activation at second layer
- Output activation function encodes the softmax function

$$f_m(x) = \Pr(Y = m | X) = \frac{e^{Z_m}}{\sum_{l=0}^{9} e^{Z_l}}$$

- We fit the model by minimizing the negative multinomial log-likelihood (or cross-entropy)

$$- \sum_{i=1}^{n} \sum_{m=0}^{9} y_{im} \log(f_m(x_i)).$$

- y_{im} is 1 if true class for observation i is m, else 0 - i.e one-hot encoded.