Intro to

Deep Learning

Neil Gogte

KMCE | DL 2024-25

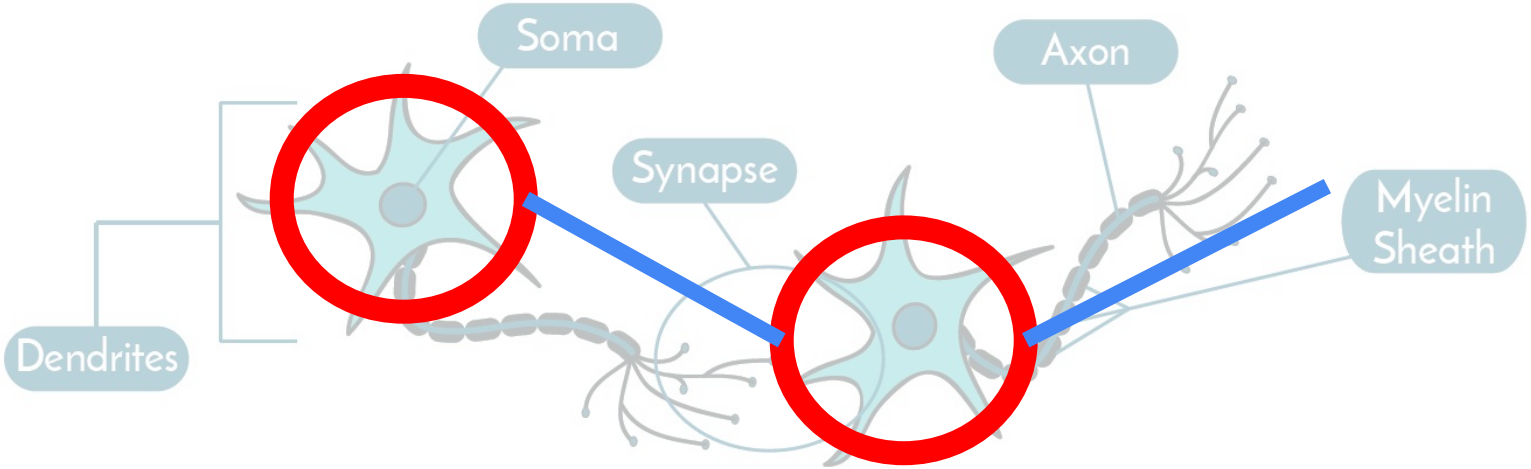
Content

1. Inspiration from **neurons**
2. Required **mathematics**
3. Basics of **Deep Learning**

Chapter #1

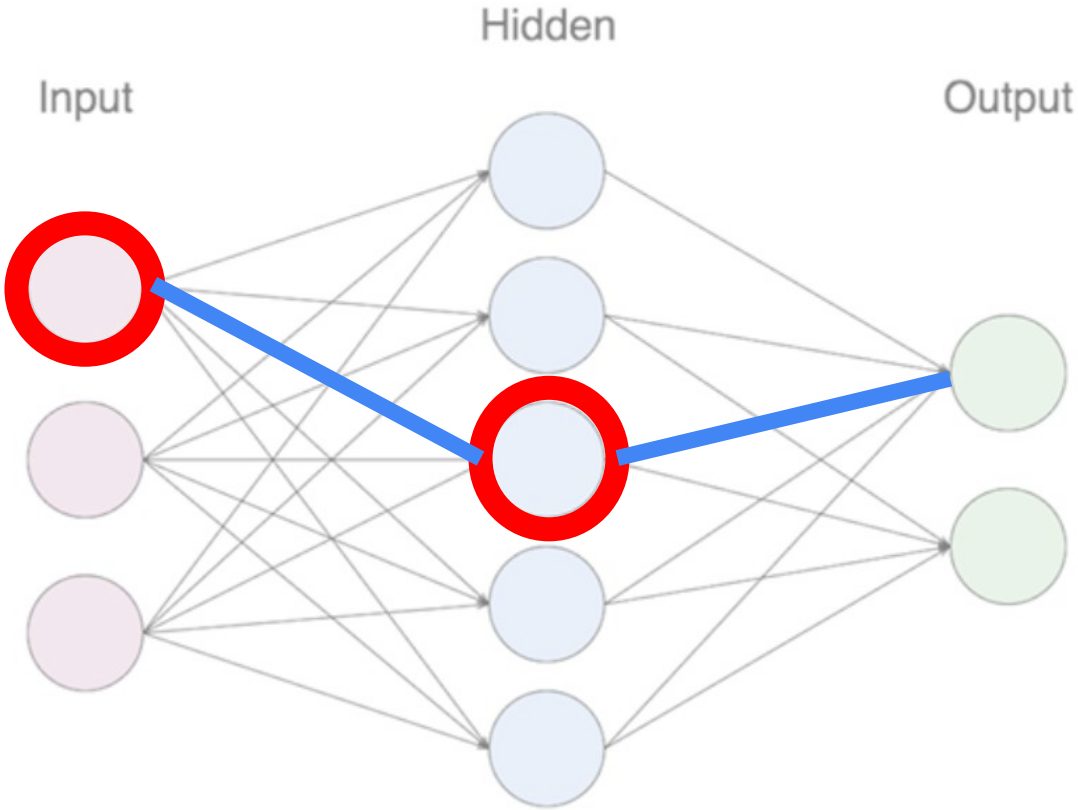
Inspiration from **neurons**

Inspiration from neurons



Inspiration for ANNs came from here

Inspiration from neurons



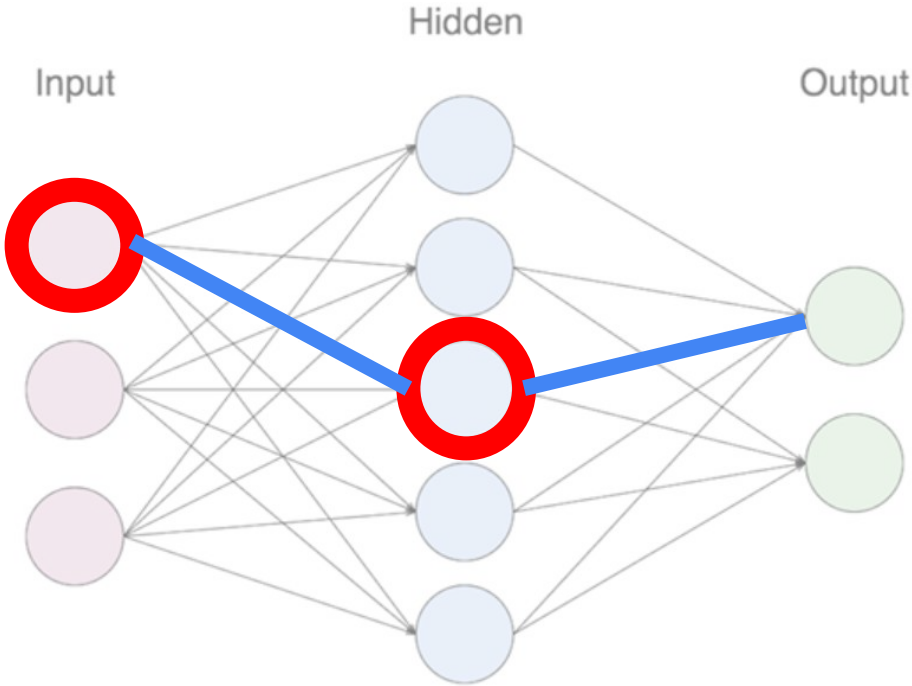
Chapter #2

Required **mathematics**

Required mathematics

- Why knowledge of math is needed in DL?
  - To get a deeper understanding of DL
- You don’t have to be math experts
- We’ll explore only the required math concepts

Required mathematics

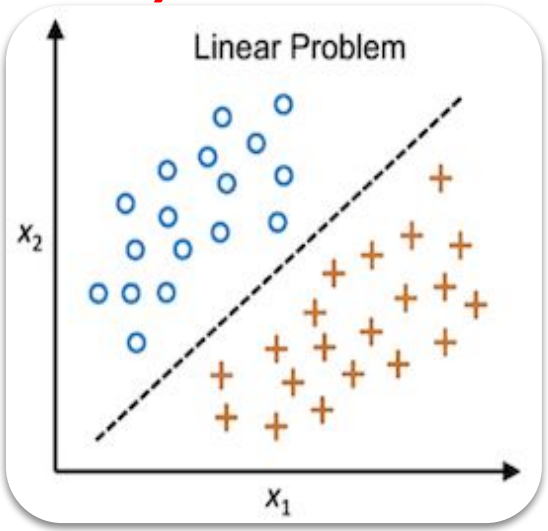


Idea is:

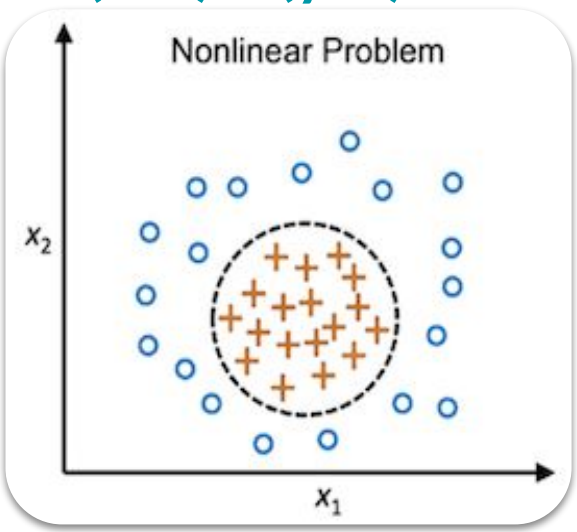
Mimic neurons on a machine using math

Required mathematics

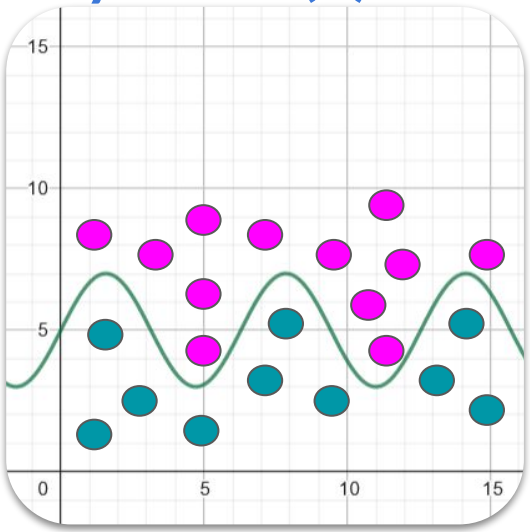
$y = wx+b$



$(x-a)^2+(y-b)^2=c$



$y = 2\sin(x)+5$



2D representation

Required mathematics

- In real time:
  - **Many features** have to be considered $(x_1, x_2, x_3...)$
  - It is an **N-dimensional problem**
  - **Non-linearity** is required!!

Chapter #3

Basics of **Deep Learning**

Basics of Deep Learning

- How will my machine automatically learn a math function based on the data I feed it?
- Ans: There should be
  - Automated **learning process**
  - Automated **math function creation**
  - Automated **feedback process** to fit correct function
  - **Non-linearity** involved

Basics of Deep Learning

Steps in ANNs learning/training:

1. Dataset preparation with predictors, truth labels
2. ANN initialization with randomness
3. Involve non-linearity to fit a good prediction function
4. Calculate the prediction with help of predictors
5. Check how close the ANN prediction is to the truth label
6. Use #5 for feedback and go back correct the params

Basics of Deep Learning

Steps in ANNs learning/training:

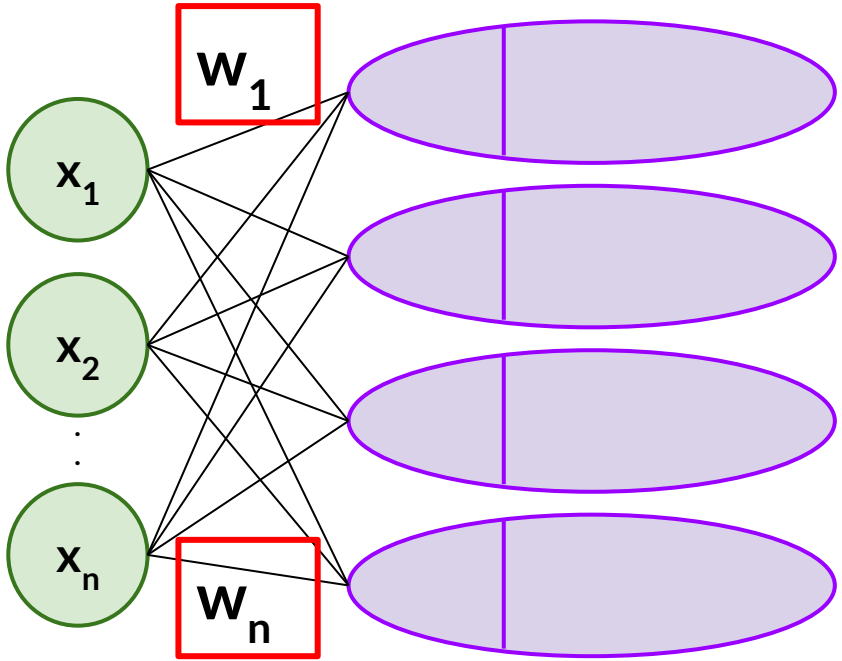
1. Dataset preparation with predictors, truth labels

x_1	x_2	x_3	$\ldots x_n$	y (truth label)

Basics of Deep Learning

Steps in ANNs learning/training:

2. ANN initialization with **randomness**

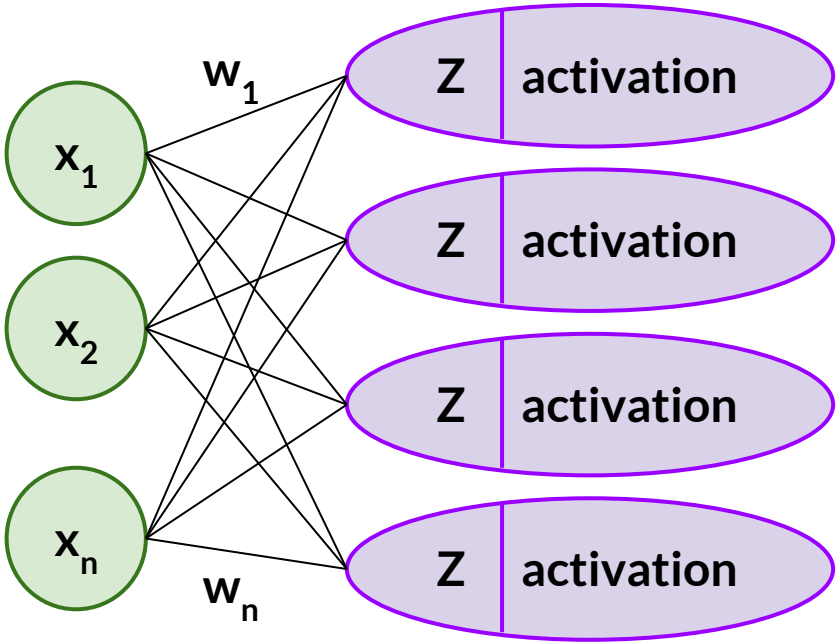


In this step the **weights and biases** are initialized randomly

Basics of Deep Learning

Steps in ANNs learning/training:

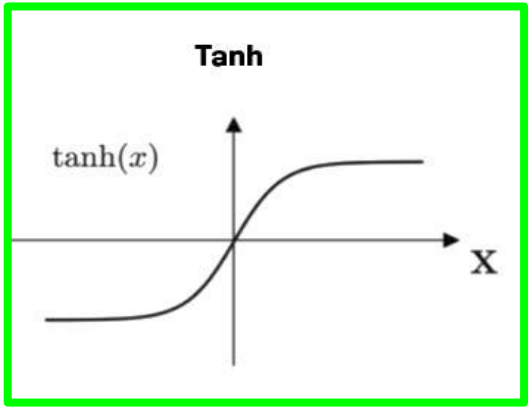
3. Involve **non-linearity** to fit a good prediction function

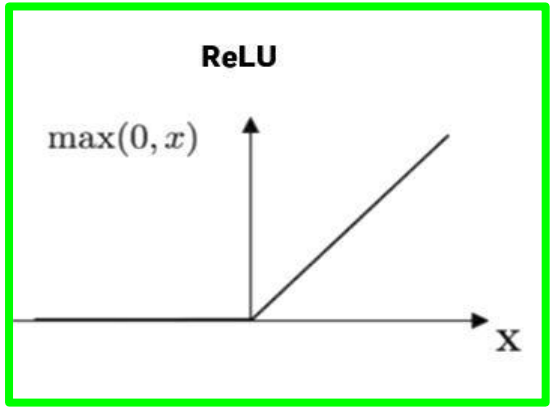


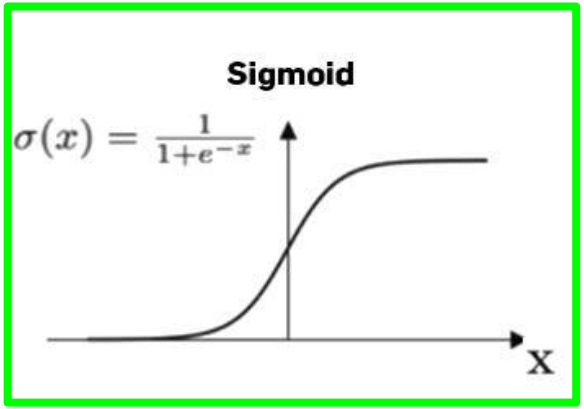
- $z = \Sigma (w_i x_i + b)$

Basics of Deep Learning

A few commonly used activation functions:





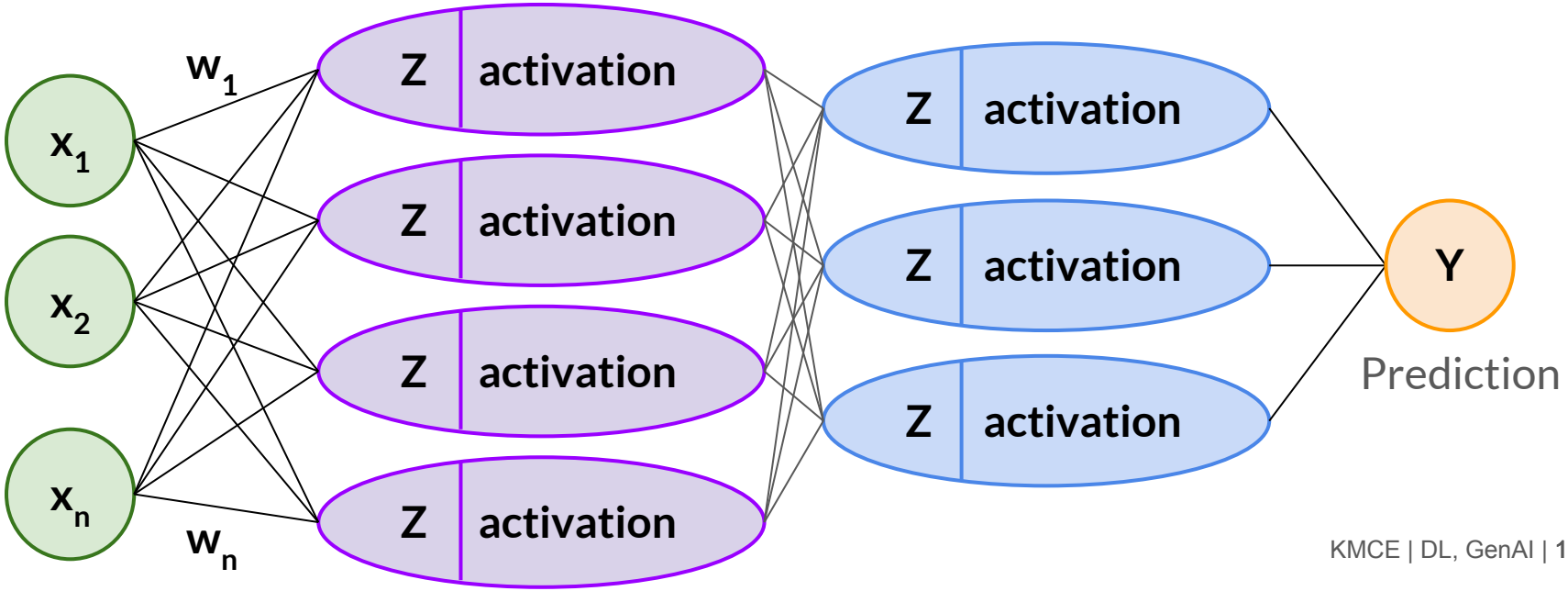


- $z = \Sigma (w_i x_i + b)$ is LINEAR
- Activation(z) i.e., tanh(z) or ReLU(z) or Sigmoid(z) is NON-LINEAR

Basics of Deep Learning

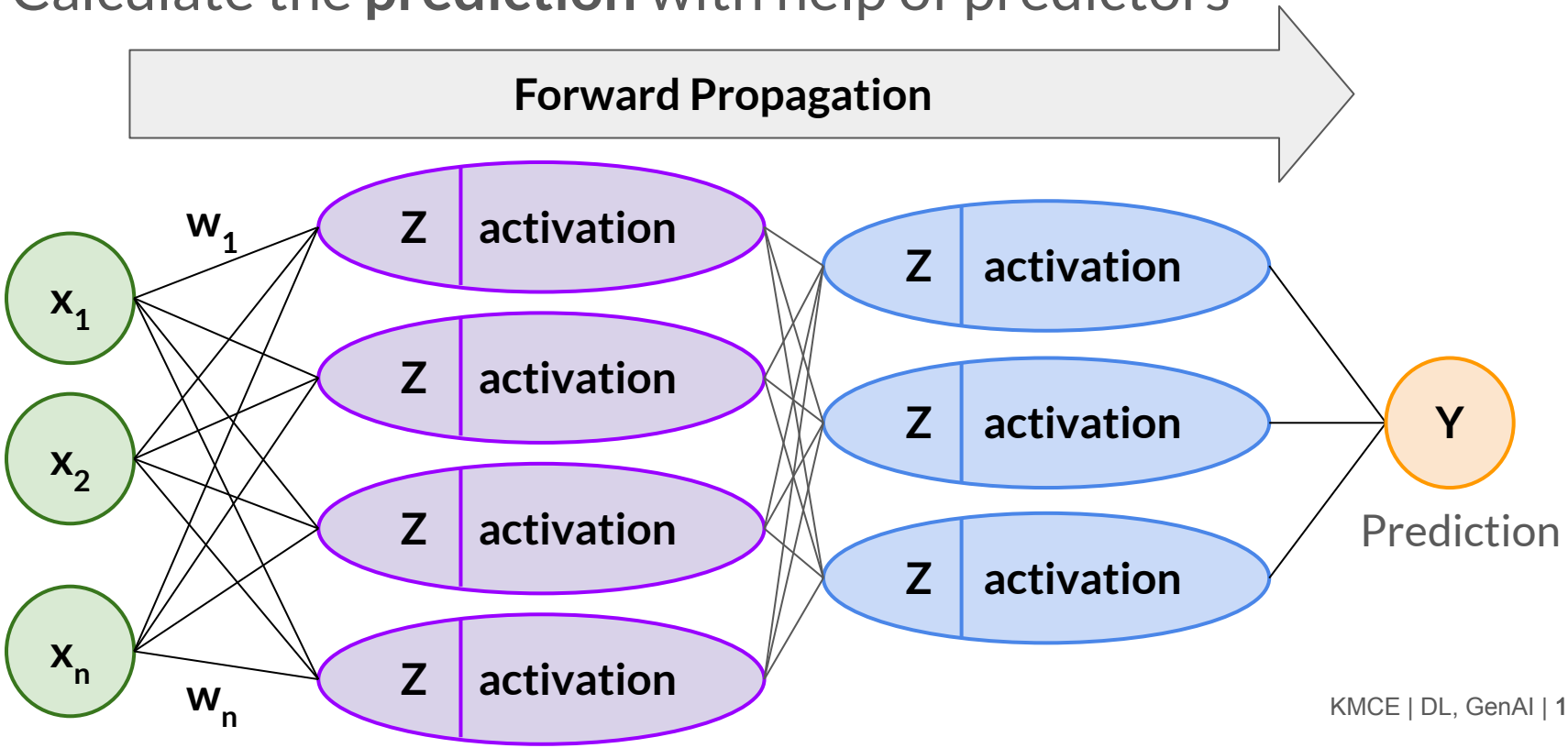
Steps in ANNs learning/training:

4. Calculate the **prediction** with help of predictors



Basics of Deep Learning

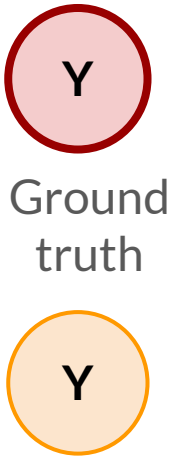
Calculate the **prediction** with help of predictors



Basics of Deep Learning

Steps in ANNs learning/training:

5. Check how close the ANN prediction is to the truth label



- Calculating difference between predicted values & ground truth labels is called a **LOSS FUNCTION**
- It gauges performance of an ANN

Basics of Deep Learning

- There are many loss functions
- A few commonly used loss functions are

REGRESSION

1. **MSE (Mean square error)**
2. **MAE (Mean absolute error)**

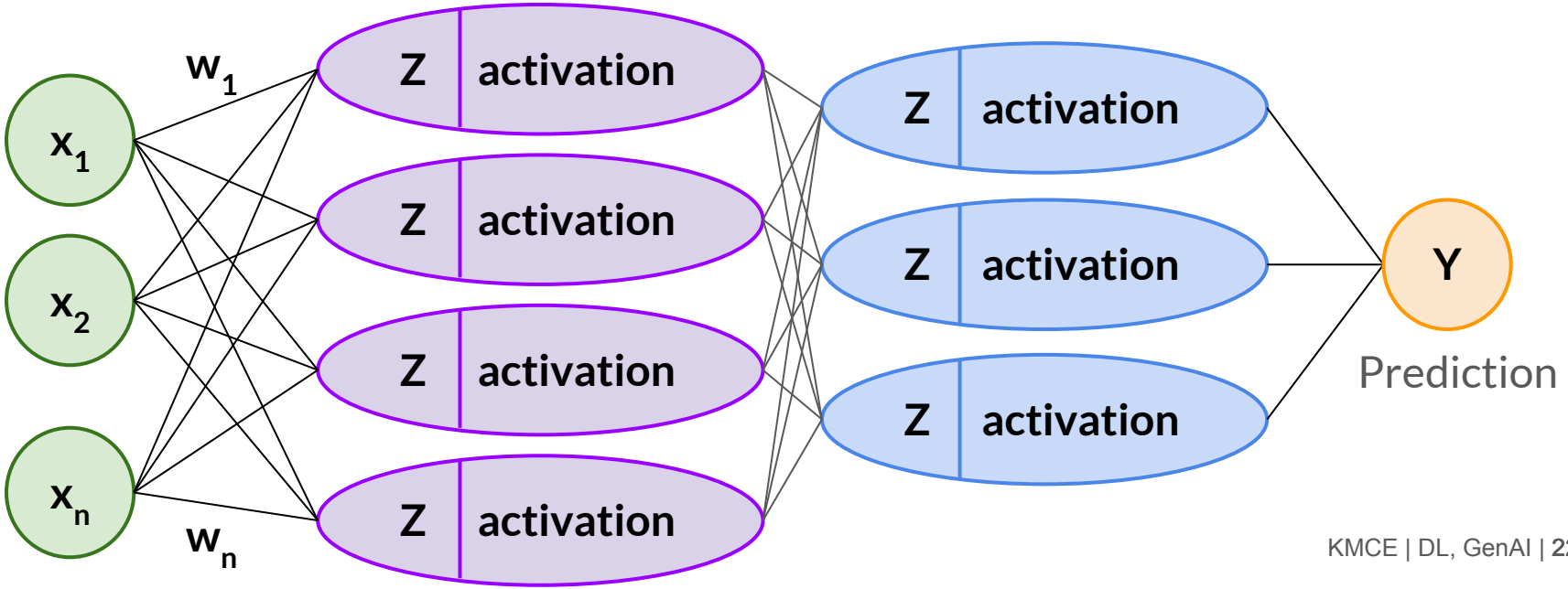
CLASSIFICATION

1. **Binary cross entropy/Log loss**
2. **Categorical Cross-Entropy loss**

Basics of Deep Learning

Steps in ANNs learning/training:

6. Use loss funcs for feedback, go back correct the params



Basics of Deep Learning

But how to correct the function?

- **x values** are input data, they cannot be changed
- $\mathbf{w_i}$, **b** the weights are biases can be tweaked

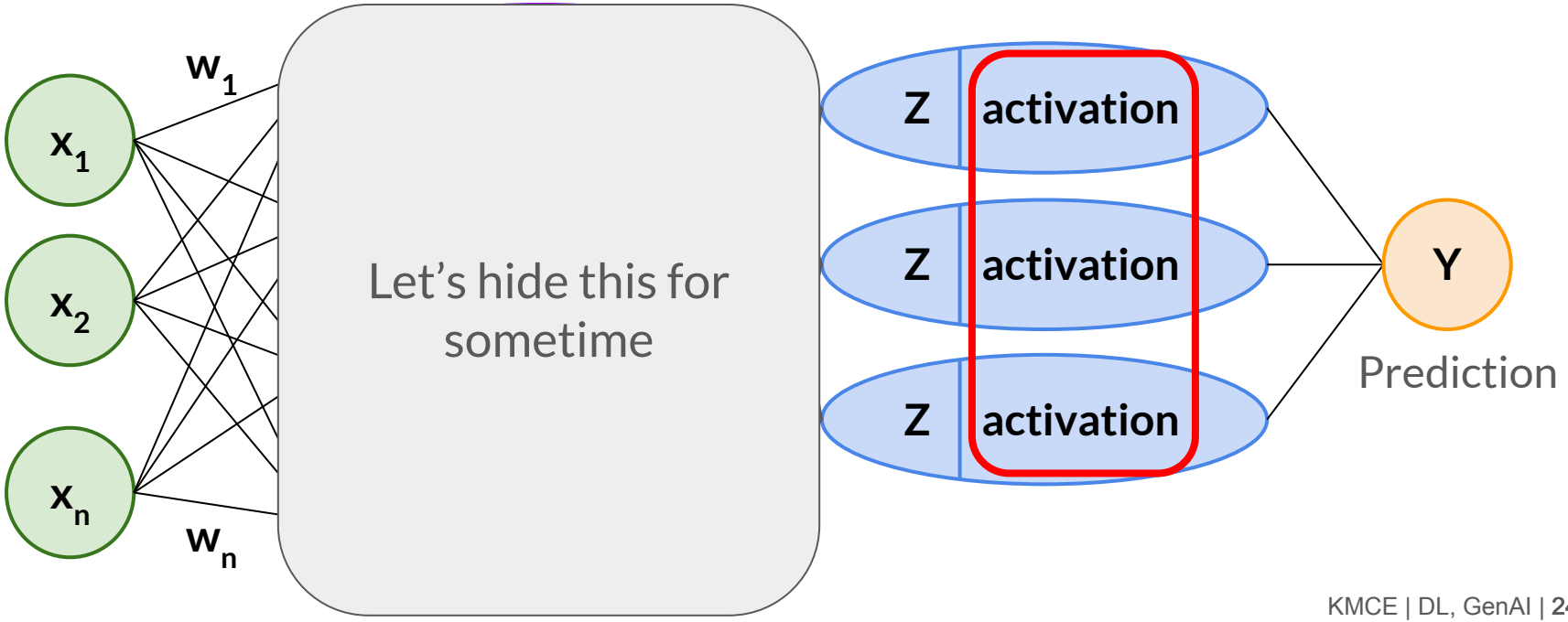
By how much should the weights, biases change?

- That where we use **derivatives**

Basics of Deep Learning

Derivative of activation function

- **Measure each neuron's (& its activation) contribution to error**



Basics of Deep Learning

- Derivative tells how much the the output is impacted when the internal variables are slightly changed
- Then we can decide and tweak the weights accordingly

Basics of Deep Learning

Backpropagation: Gradient estimation method commonly used for training neural networks to compute the network parameter updates (wiki def)