Neural Network

Learning Outcome 2

Objective

After attending this session, you should be able

- Single neuron
- Importance of activation function
- Various kind of activation function
- Loss vs cost function
- Gradient Descent

Class Exercise: Neural Network History → Basics → Architectures → Applications → Future.

Group 1: History & Evolution of Neural Networks

Deliverable: A 5-minute group presentation showing the timeline and why each milestone mattered

Group 2: Neural Network Basics (Perceptron & Derivatives)

Deliverable: A mini-demo or sketch explaining perceptron → MLP → deeper nets.

Group 3: Architectures

Deliverable: A "Neural Net Architecture Cheat Sheet" to share with the class.

Group 4: Applications in Real Life

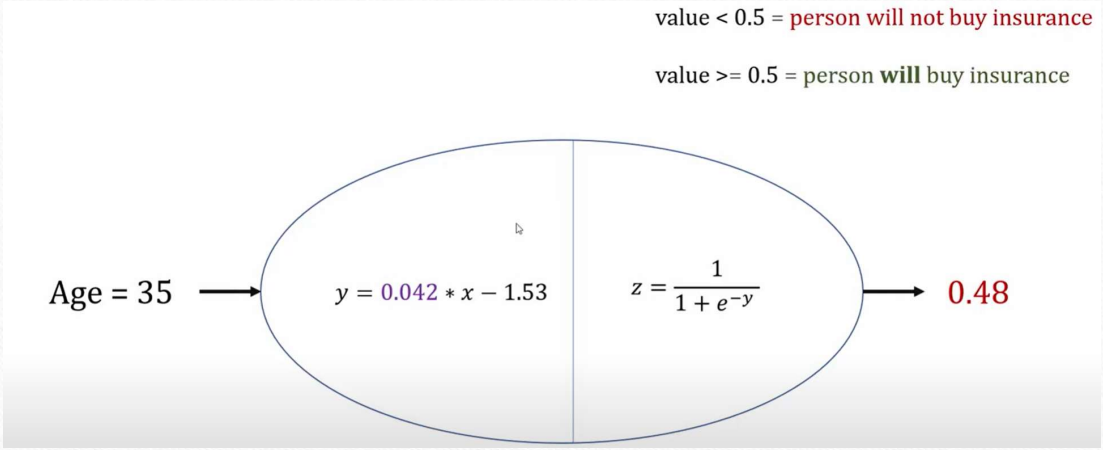
Deliverable: A quick mind map (whiteboard or digital tool like Miro) linking applications to architectures.

Group 5: Future Directions & Ethical Considerations

Deliverable: A 2–3 slide summary related to Discuss ethics, risks, and societal implications (bias, jobs, data privacy).

Single Neuron

- Find out the function that can predict if person will buy the insurance or not, using age of person as independent variable

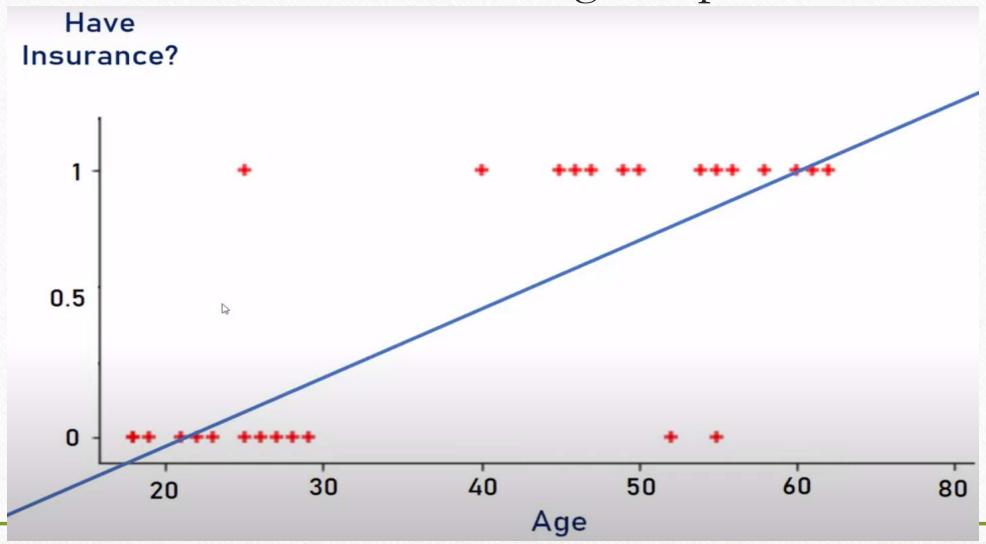


age	have_insurance
22	0
25	0
47	1
52	0
46	1
56	1
55	0
60	1
62	1
61	1
18	0
28	0
27	0
29	0
49	1

Binary Classification

- Problem formation:

Find the function that tell person required to buy insurance or not on basis of age of person

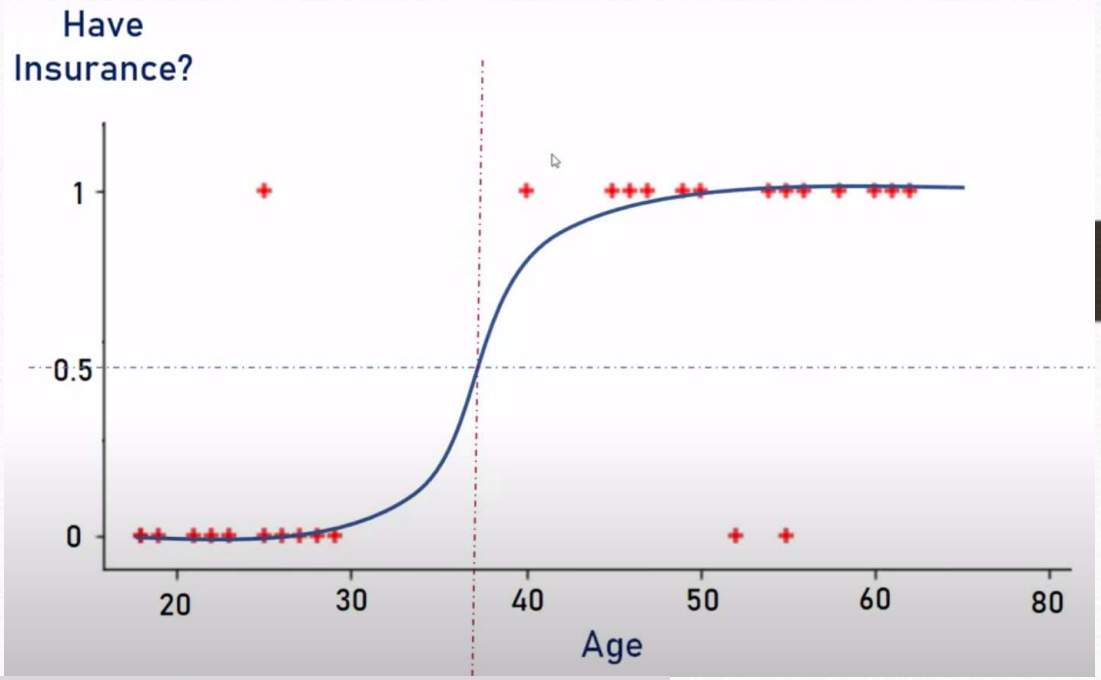


age	have_insurance
22	0
25	0
47	1
52	0
46	1
56	1
55	0
60	1
62	1
61	1
18	0
28	0
27	0
29	0

Sigmoid function

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

e = Euler's number ~ 2.71828



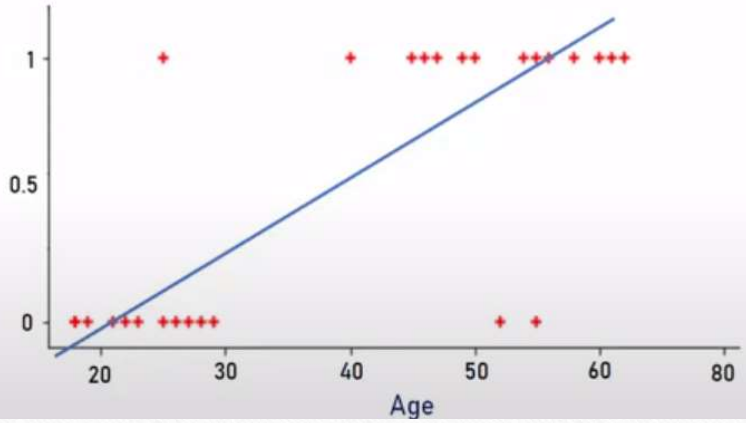
Sigmoid function converts input into range 0 to 1

Refer more on: https://machinelearningmastery.com/a-gentle-introduction-to-sigmoid-function/

Step 1

$$y = m * x + b$$

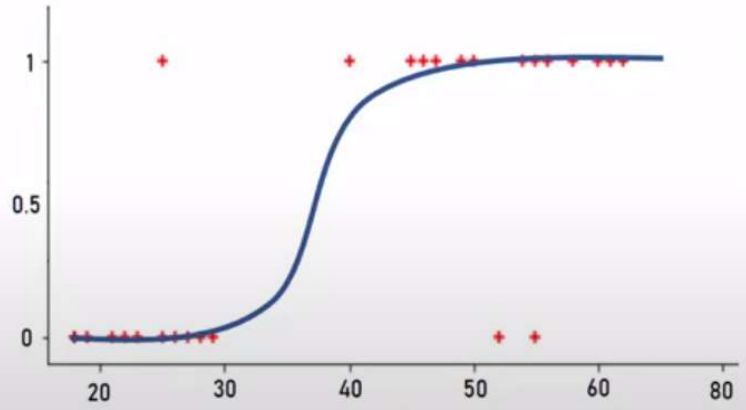
Age



Step 2

$$z = \frac{1}{1 + e^{-y}}$$

If person will buy insurance



Example with more features

$$y = 0.042 * x - 1.53$$

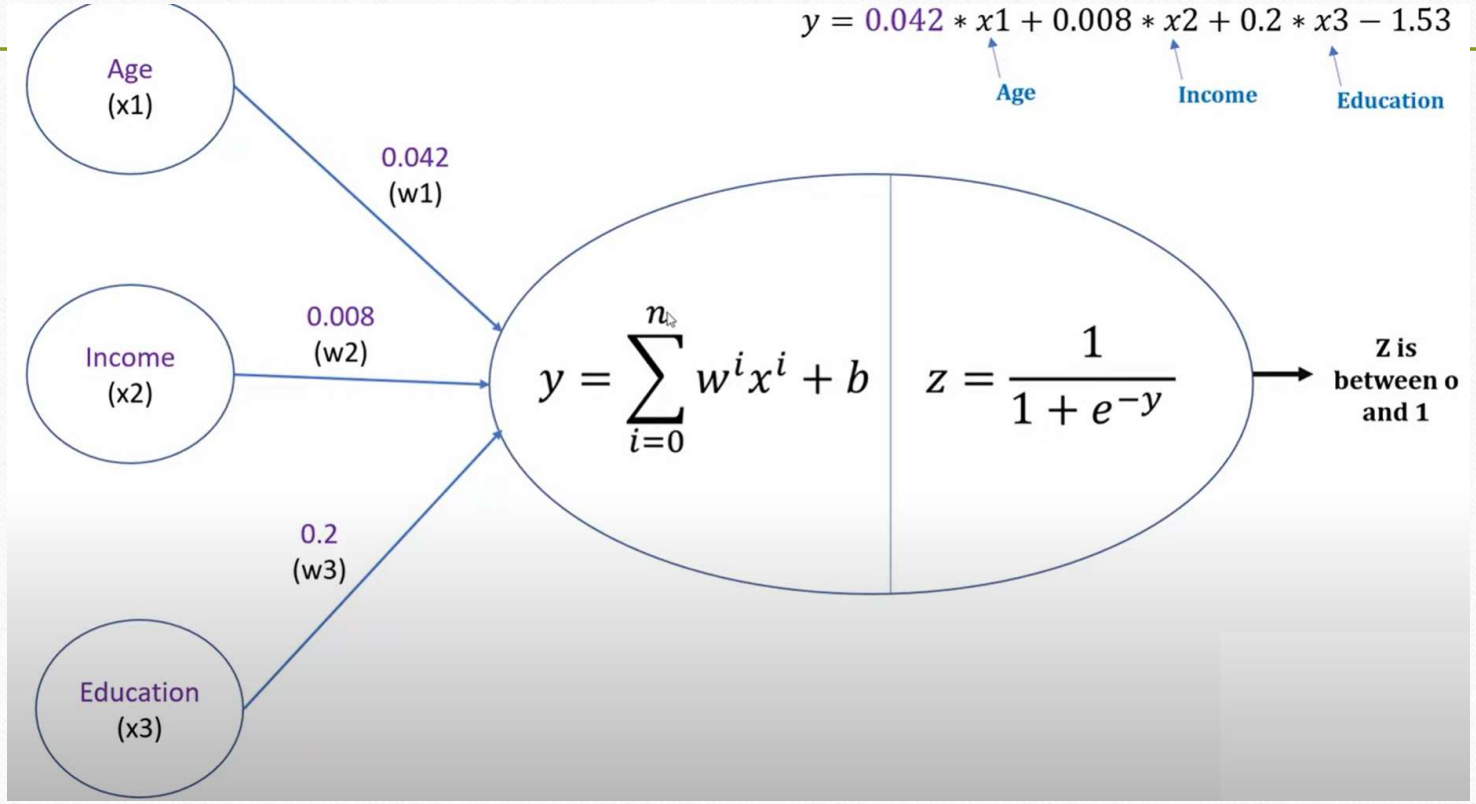
Age

$$y = 0.042 * x1 + 0.008 * x2 + 0.2 * x3 - 1.5$$

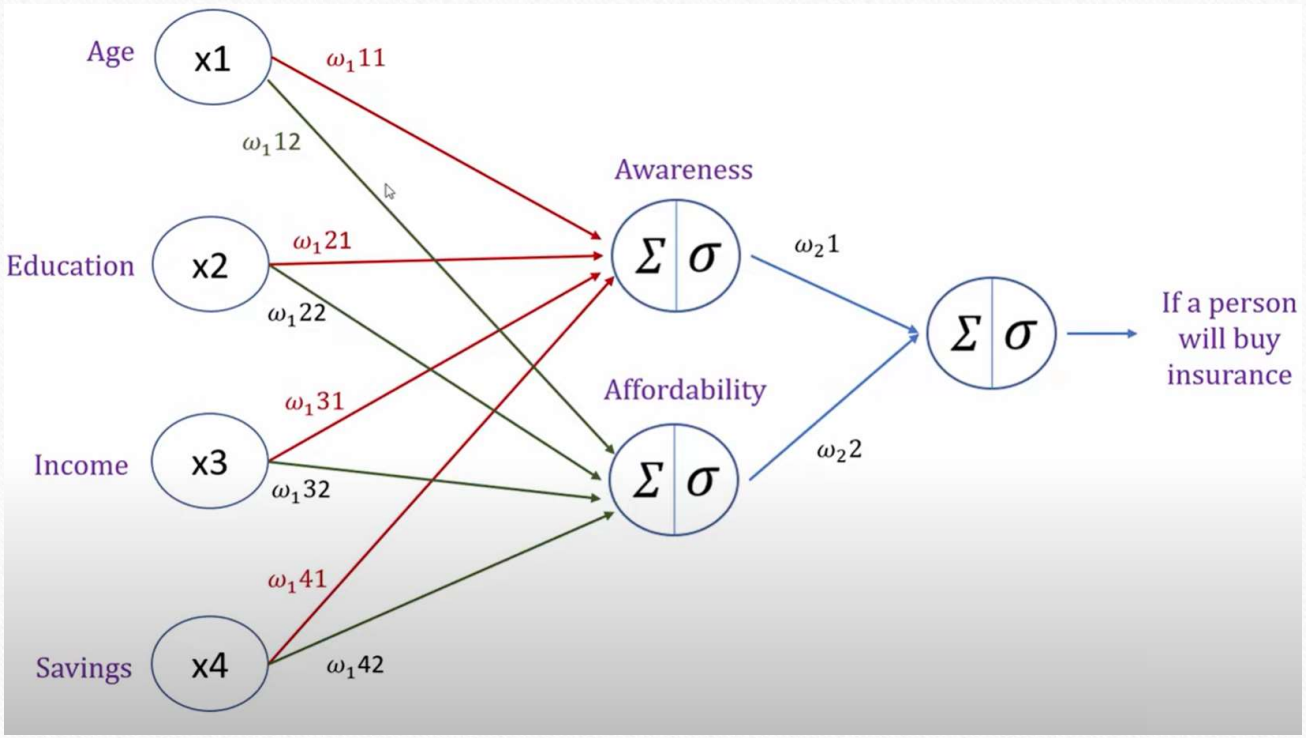
Age Income Education

$$y = w1 * x1 + w2 * x2 + w3 * x3 + \boldsymbol{b}$$

$$y = \sum_{i=0}^{n} w^i x^i + b$$

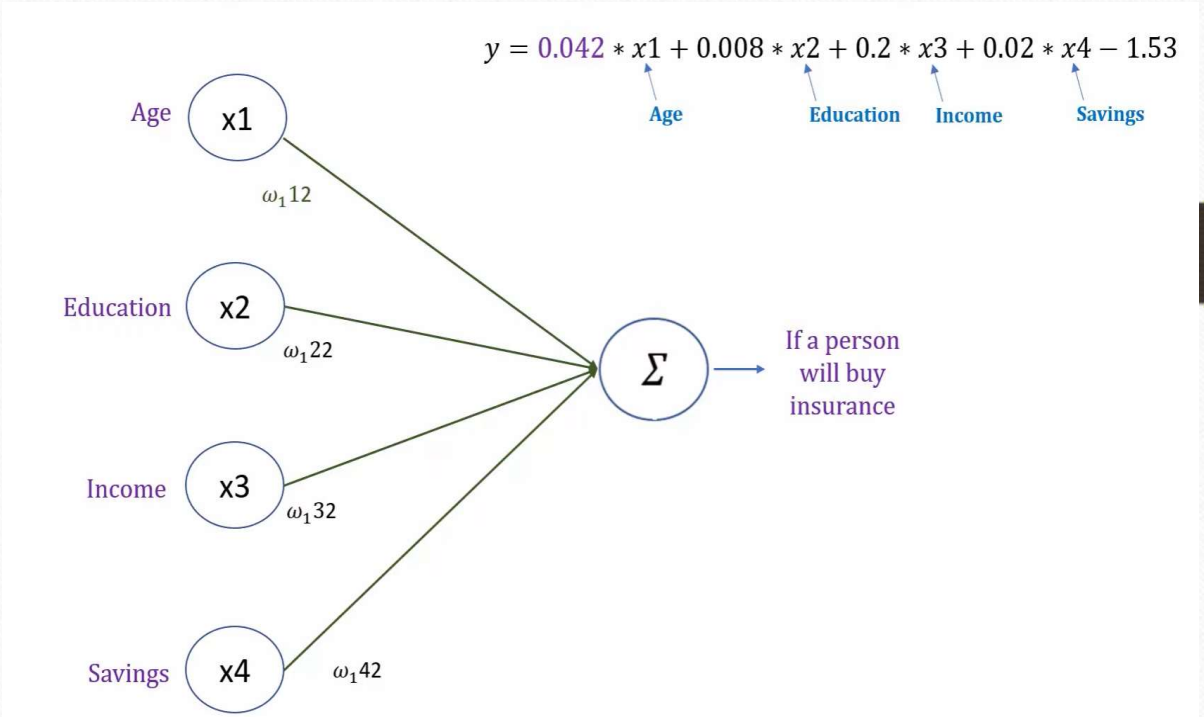

$y = 0.042 * x1 + 0.008 * x2 + 0.2 * x3 - 1.53$
Age
Income
Education
Age
(x1)
Income
(x2)
Education
(x3)
0.042
(w1)
0.008
(w2)
0.2
(w3)
$y = \sum_{i=0}^{n} w^i x^i + b$
$z = \frac{1}{1 + e^{-y}}$
Z is between o and 1

More real example including hidden layer



Not using activation function

- Our model become simple linear equation
- Many real-time problems are non-linear in nature. So, our model handle the non-linearity by introducing the activation function



Neural Network by example

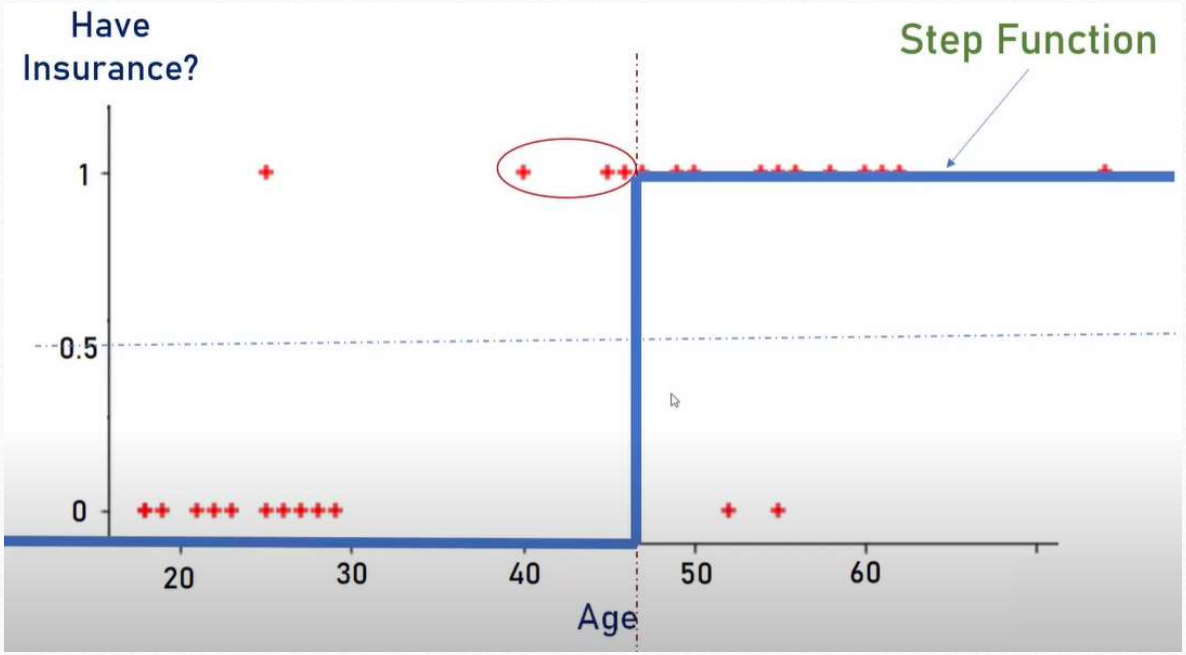
nose, eyes, ear detector

Leg, tail detector

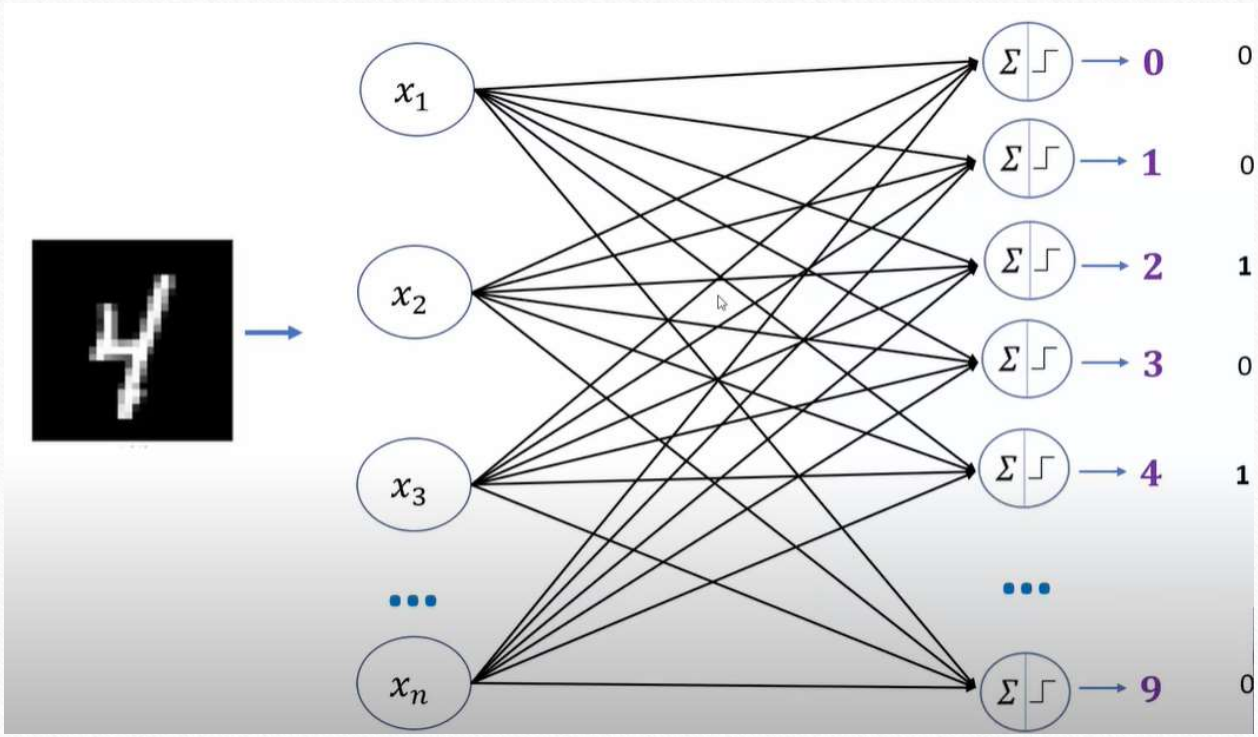
- Face detector = 0.2 eye +0.5 nose+0.3 ear
- Body detector = 0.4 leg+0.6 tail
- Lion detector = 0.6 face + 0.4 body
- Predict
- Backward propagation
- repeat the same process

Step function

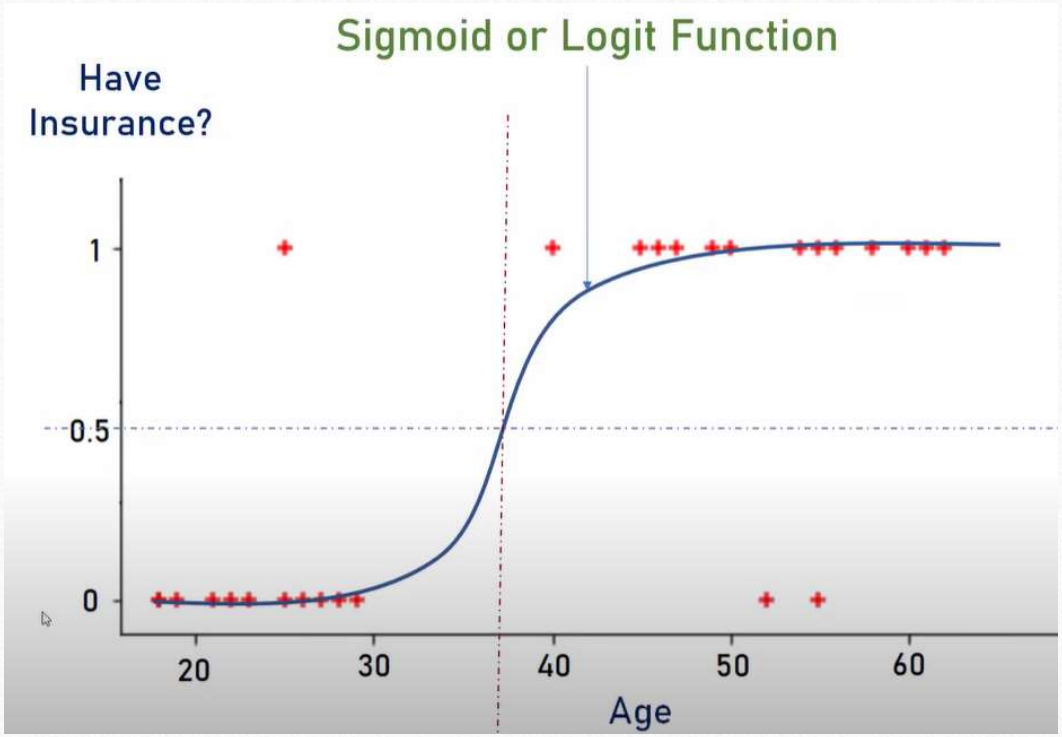
- Mis-classification issue with step function

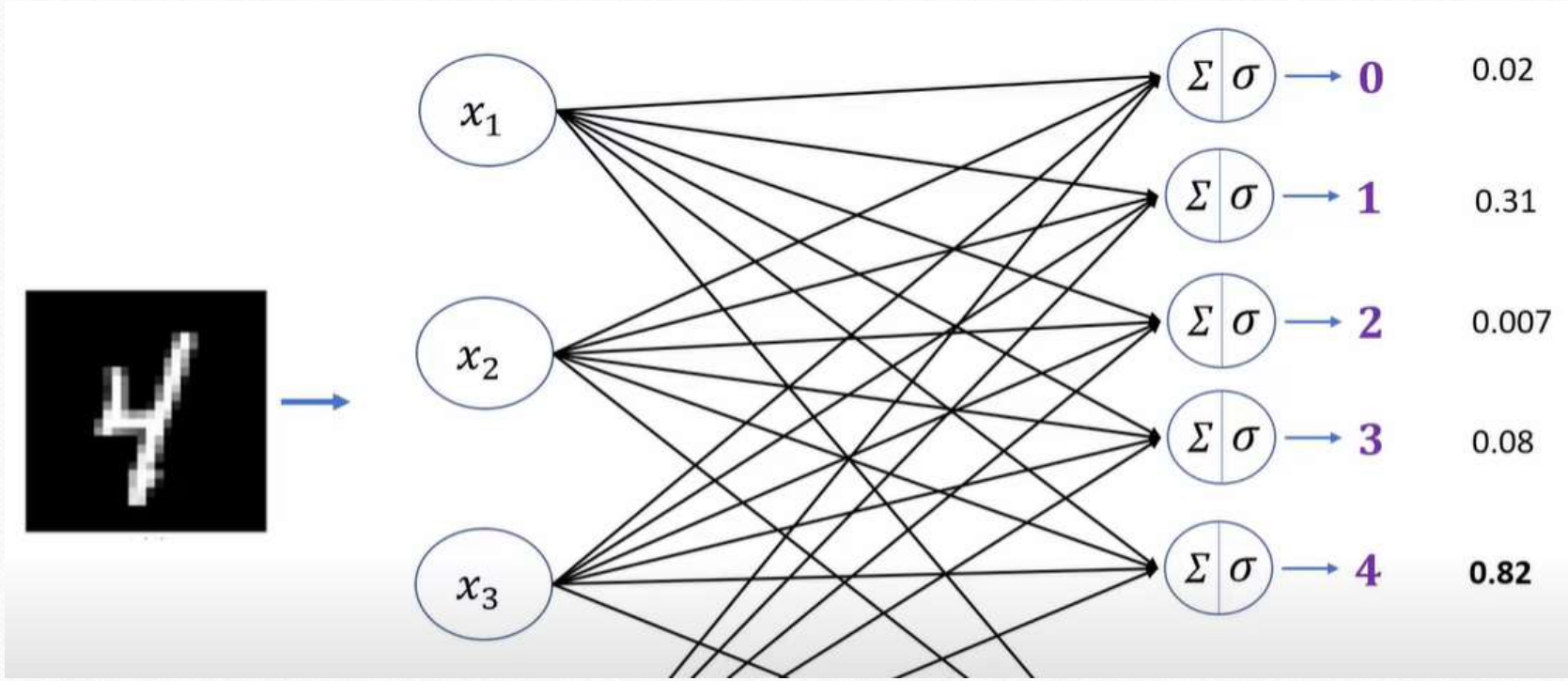


Mis-classification issue with step function



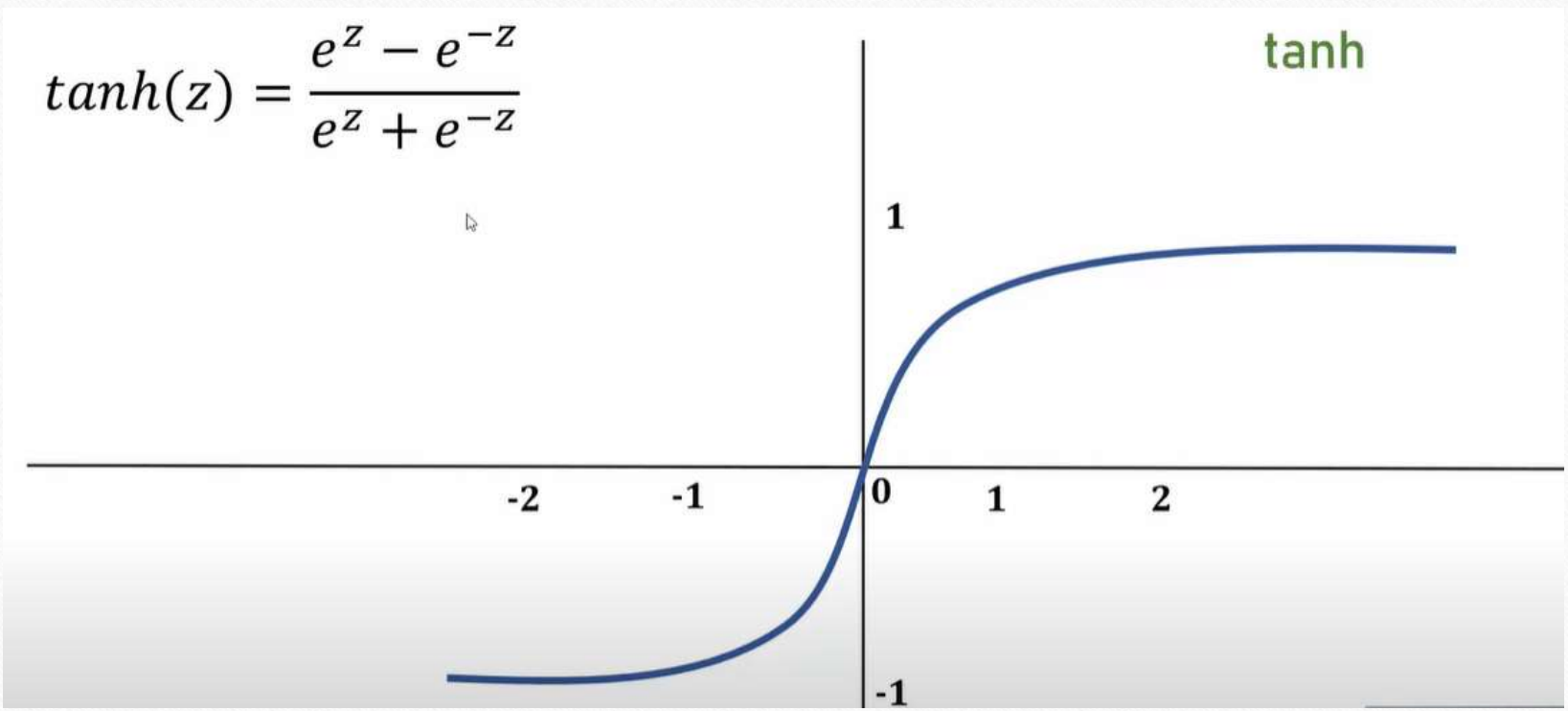
Sigmoid function



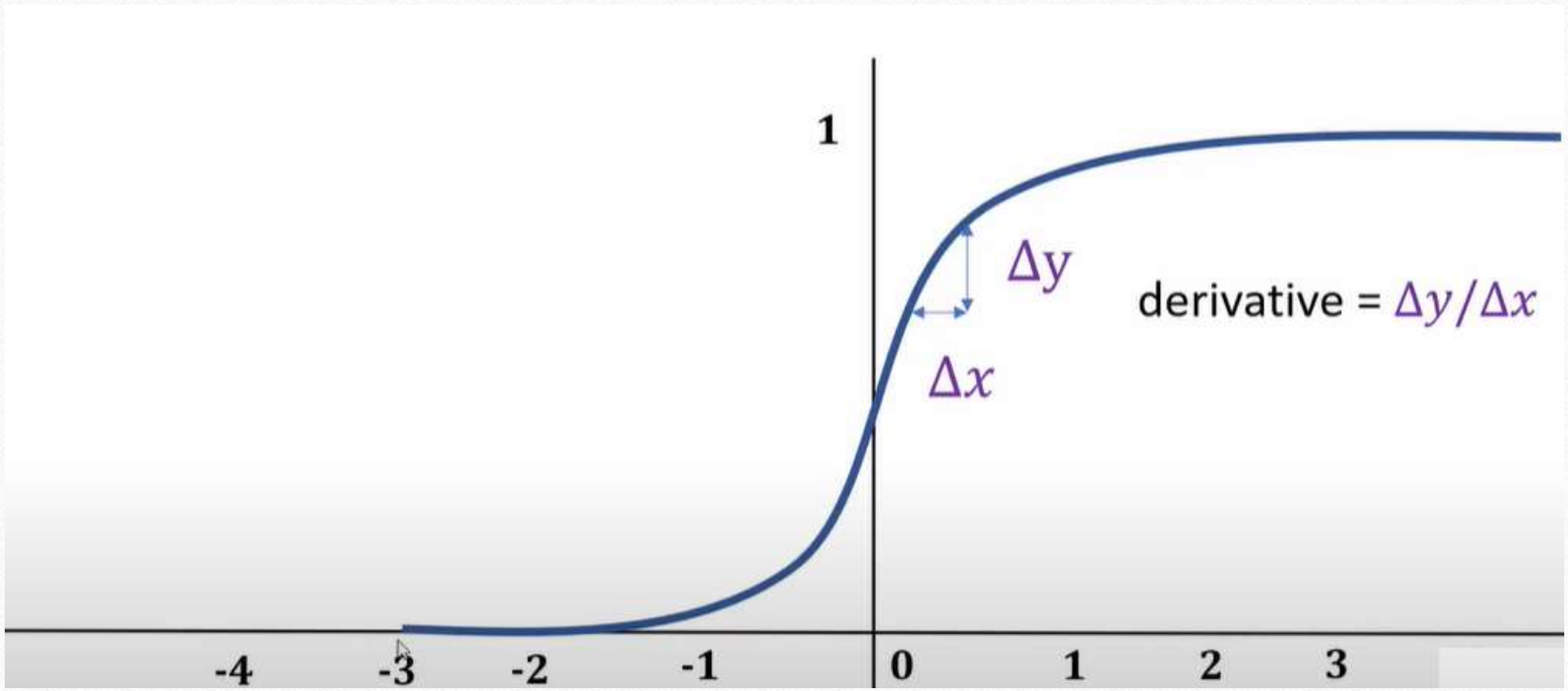


Tanh function

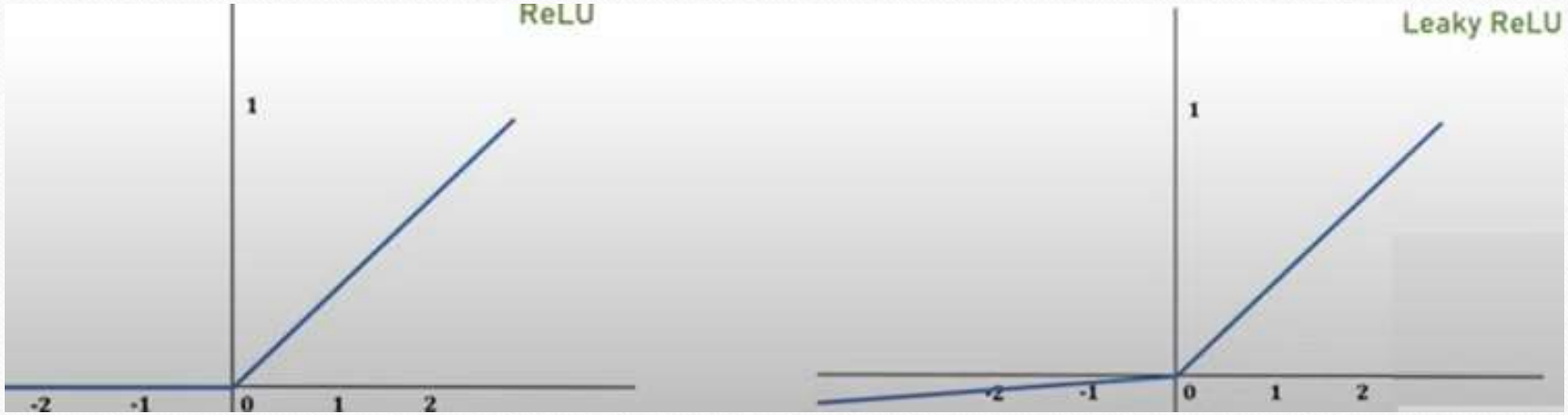
$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$



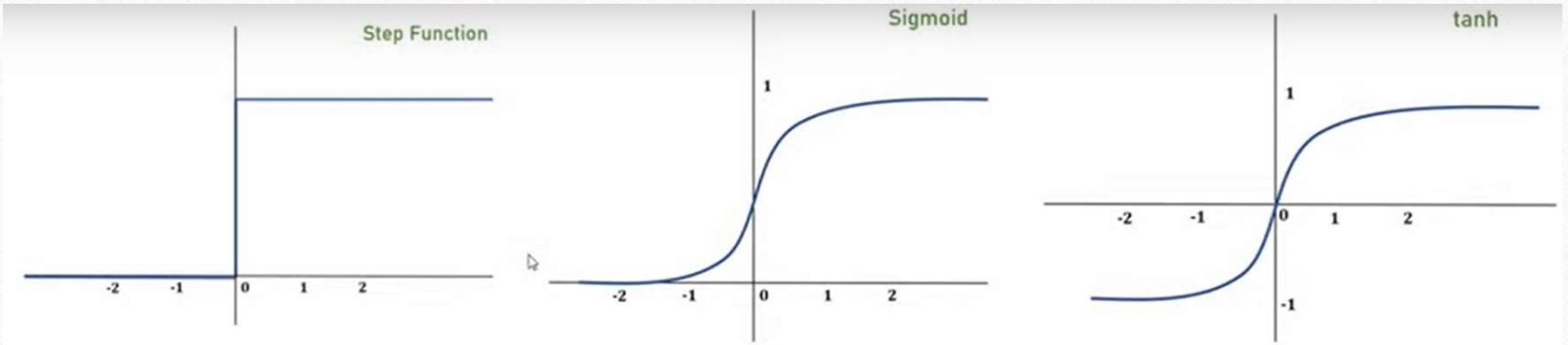
Issue with sigmoid and tanh



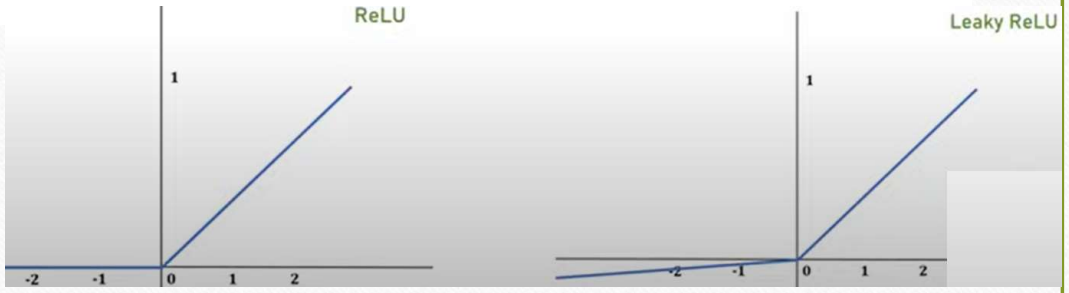
ReLU and Leaky ReLU



Activation functions

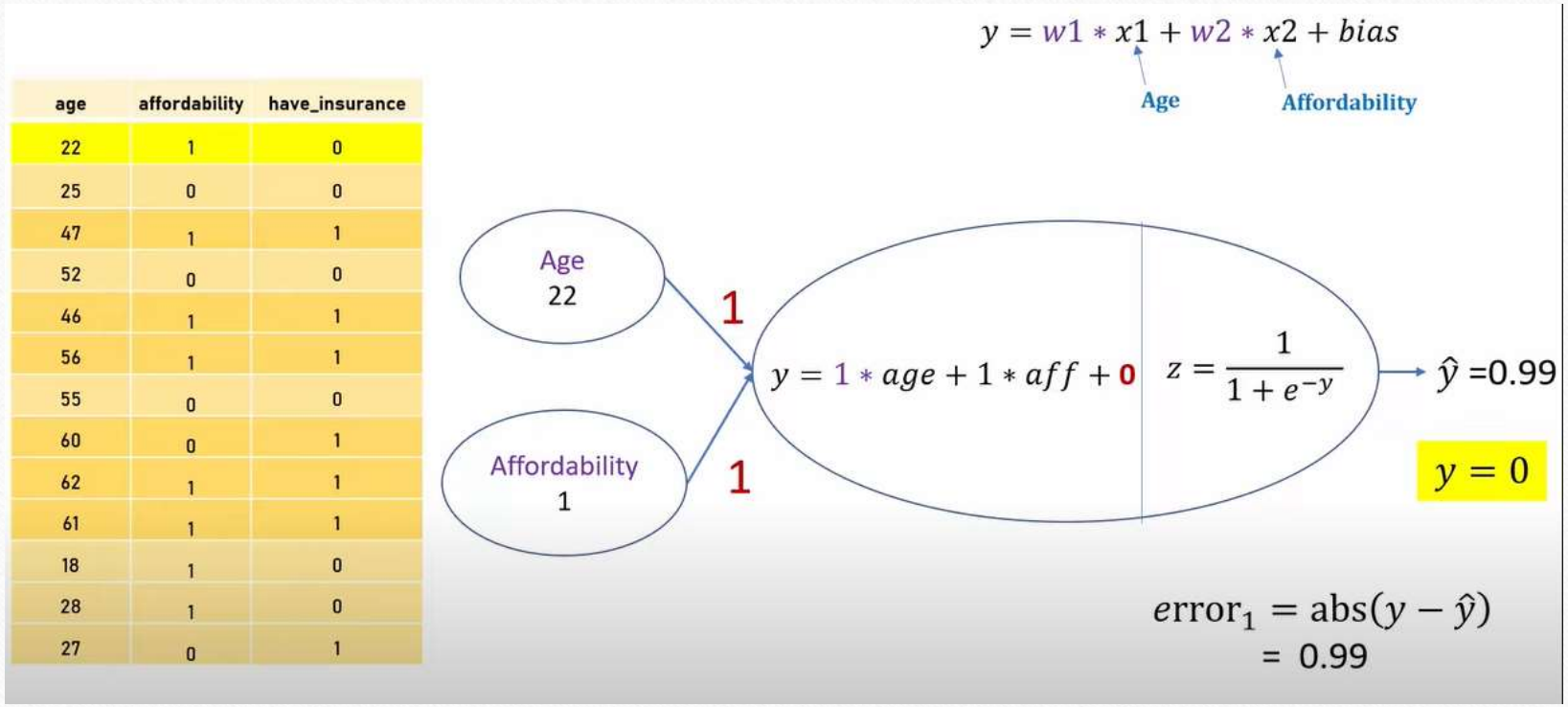


An Activation Function decides whether a neuron should be activated or not. This means that it will decide whether the neuron's input to the network is important or not in the process of prediction using simpler mathematical operations.



Introduction: Error

age	affordability	have_insurance
22	1	0
25	0	0
47	1	1
52	0	0
46	1	1
56	1	1
55	0	0
60	0	1
62	1	1
61	1	1
18	1	0
28	1	0
27	0	1



Loss vs Cost function

$$\text{Total error} = error_1 + error_2 + \ldots + error_{13}$$

$$= \sum_{i=1}^{n} abs(y_i - \hat{y}_i)$$

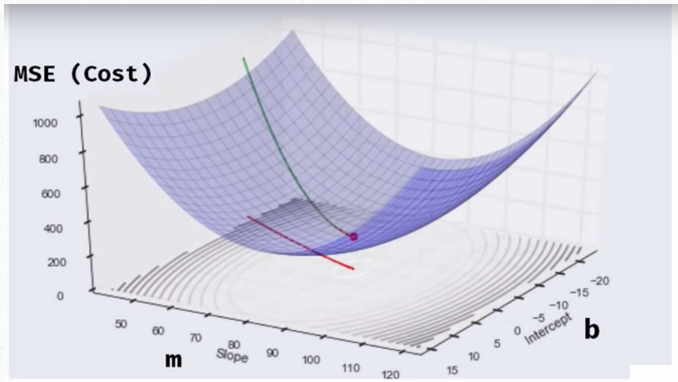
Loss

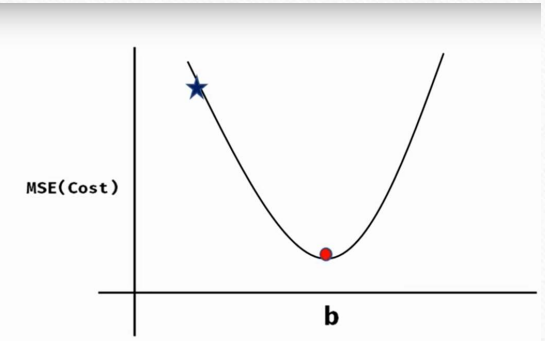
$$Mean\ Absolute\ Error\ (MAE) = \frac{1}{n}\sum_{i=1}^{n} abs(y_i - \hat{y}_i)$$

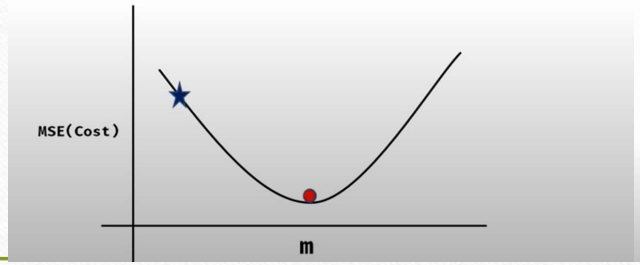
Cost Function

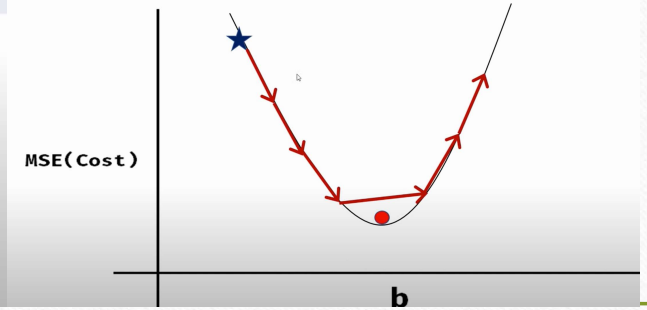
Gradient Descent

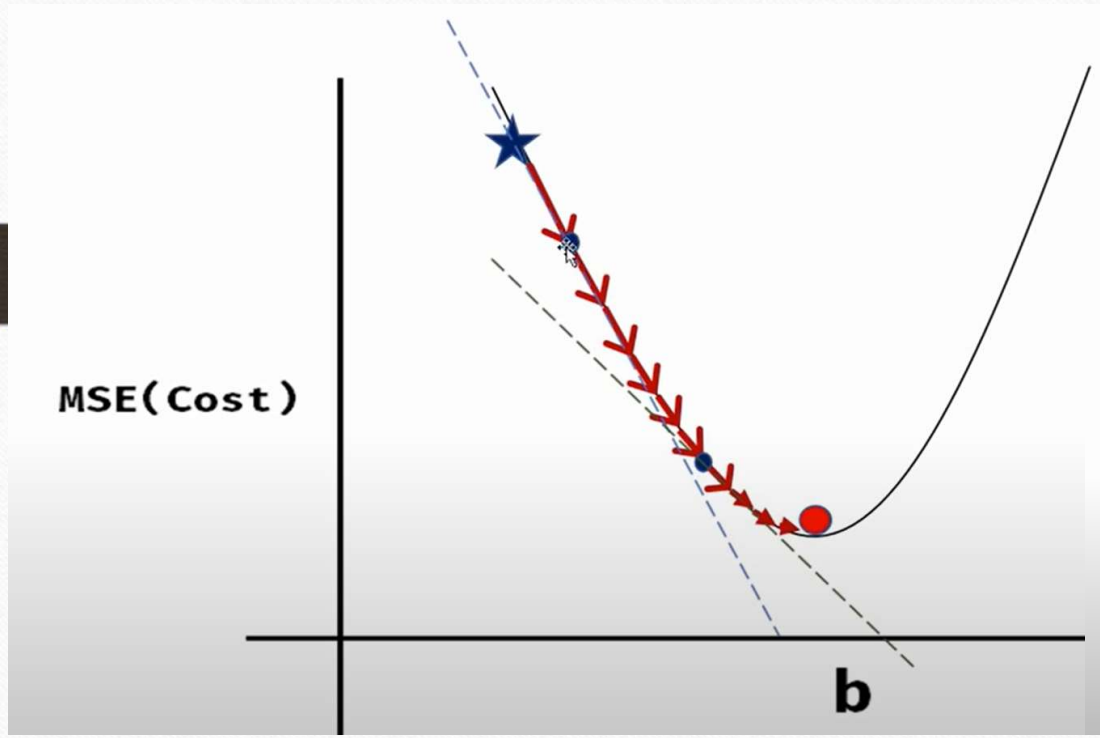
- Its an algorithm that finds best fit line for the given training data set











$$f(x, y) = x^3 + y^2$$

$$\partial f / \partial x = 3x^2 + 0 = 3x^2$$

$$\partial f / \partial y = 0 + 2y = 2y$$

Cost function: Mean Squared Error

$$mse = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_{predicted}\right)^2$$

$$mse = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - (mx_i + b)\right)^2$$

Cost Function

$$mse = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - (mx_i + b)\right)^2$$

$$\partial/\partial m = \frac{2}{n}\sum_{i=1}^{n} -x_i\left(y_i - (mx_i + b)\right)$$

$$\partial/\partial b = \frac{2}{n}\sum_{i=1}^{n} -\left(y_i - (mx_i + b)\right)$$

Learning rate

$$m = m - \text{learning rate} * {}^{\partial}/_{\partial m}$$

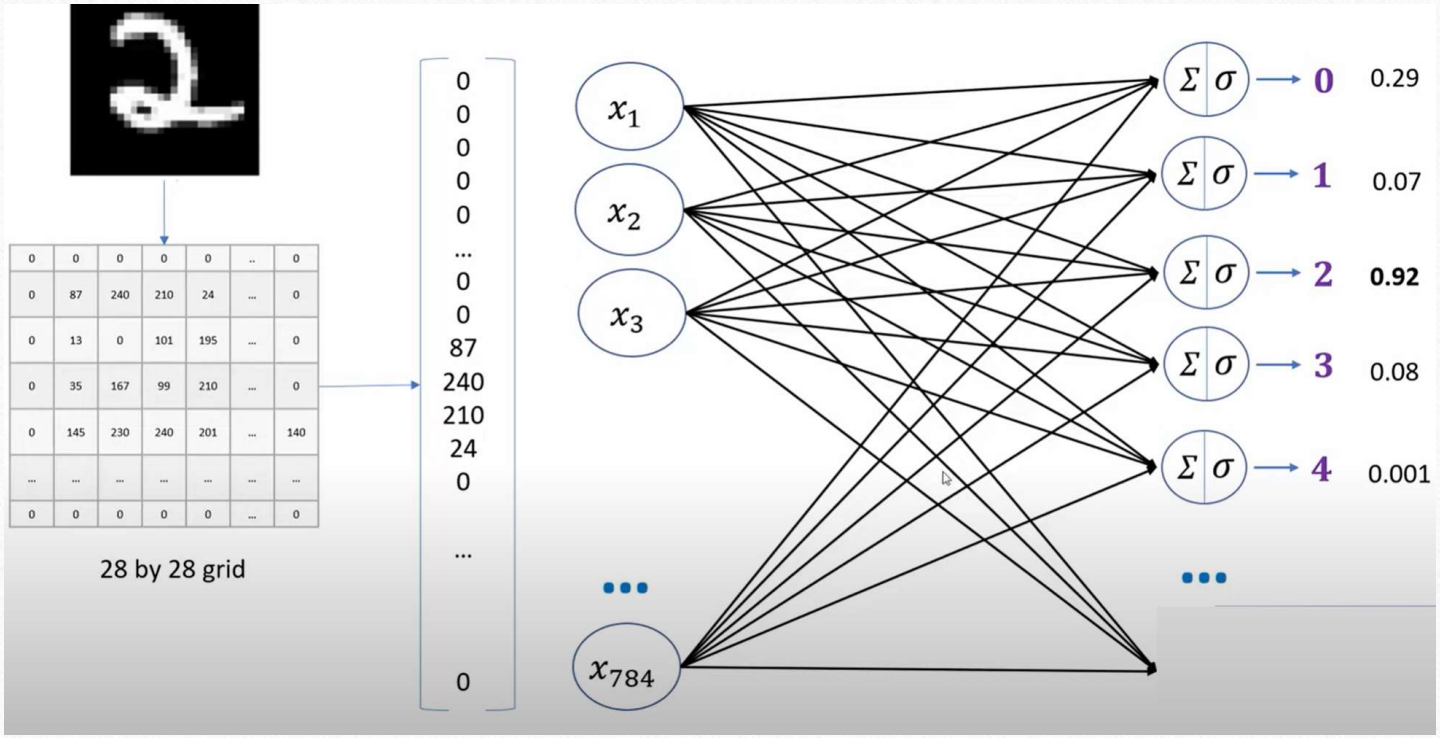
$$b = b - \text{learning rate} * {}^{\partial}/_{\partial b}$$

Learning rate start with 0.001 and then you update as per requirement

Introduction and Installation of TensorFlow

- TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.
- TensorFlow was developed by the Google Brain team for internal Google use in research and production.
- Pytorch is from facebook.
- Keras is not full fledged deep learning framework like tensorflow and pytorch
- Now tensorflow 2.13.0
- Pip install tensorflow

Example: Handwritten Digits Classification



- An optimizer is a function or an algorithm that modifies the attributes of the neural network, such as weights and learning rate. Thus, it helps in reducing the overall loss and improve the accuracy
- One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only ONCE.
- Since one epoch is too big to feed to the computer at once we divide it in several smaller batches
- sparse_categorical_crossentropy: Used as a loss function for multi-class classification model where the output label is assigned integer value (0, 1, 2, 3…). This loss function is mathematically same as the categorical_crossentropy

  (more information: https://www.tensorflow.org/api_docs/python/tf/keras/losses)

Handwritten Digits Classification using neural network

Summary

- Single neuron
- Importance of activation function
- Various kind of activation function
- Loss vs cost function
- Gradient Descent

Group Presentation

- Feature selections techniques
- Feature Transformation techniques
- Handle Imbalanced data for a classification problem
- Evaluation metrics of classifier
- Evaluation metrics of regressor

Time: 1 hour for group discussion and preparing small presentation

Presentation time per group: 10 min

Himanshu Patel, Instructor
Saskatchewan Polytechnic
email: patelh@saskpolytech.ca
Mining building, saskatoon