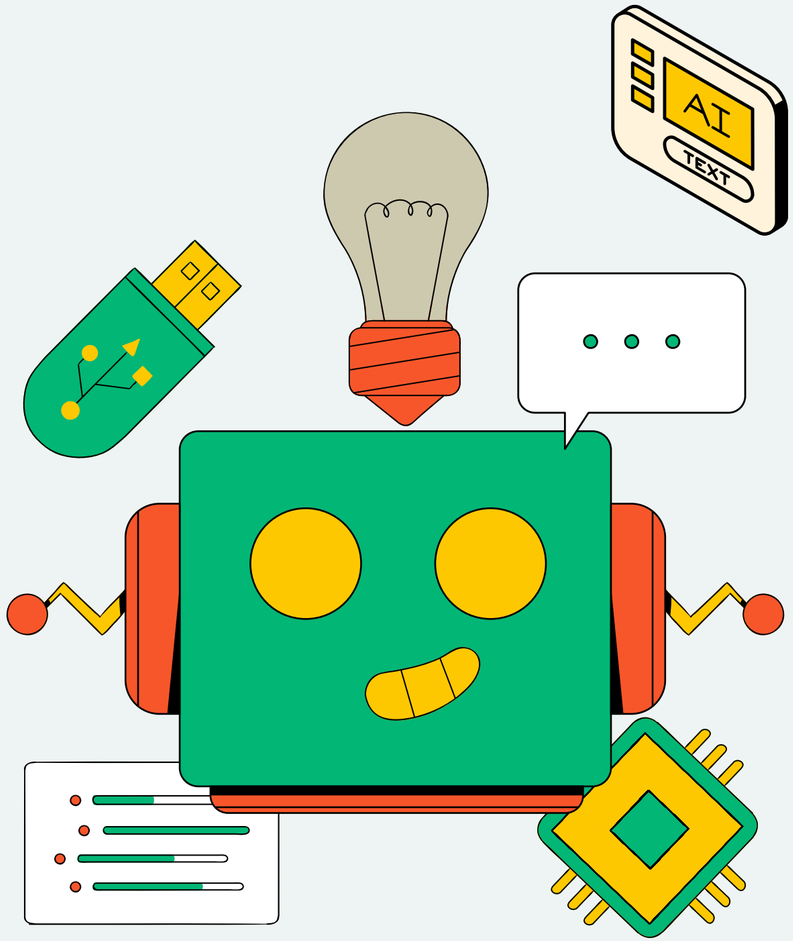
THYNK UNLIMITED

WE LEARN FOR THE FUTURE



INTRODUCTION TO DEEP LEARNING

PRESENTATION

PRESENTED BY:

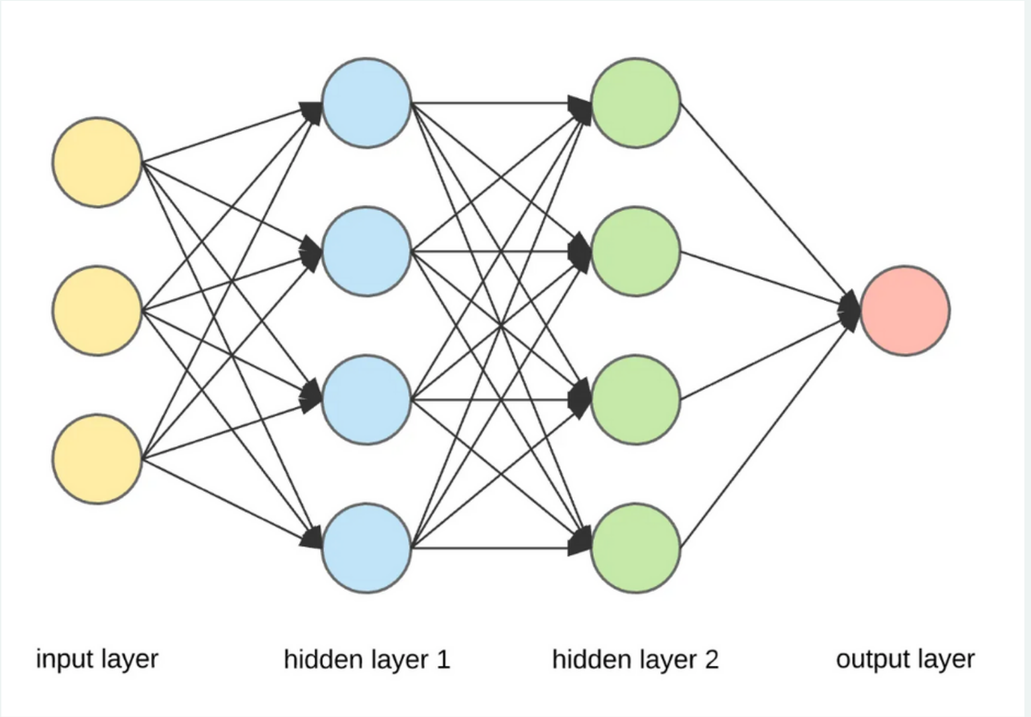
ZULFIKAR IRHAM

PRESENTATION OUTLINE

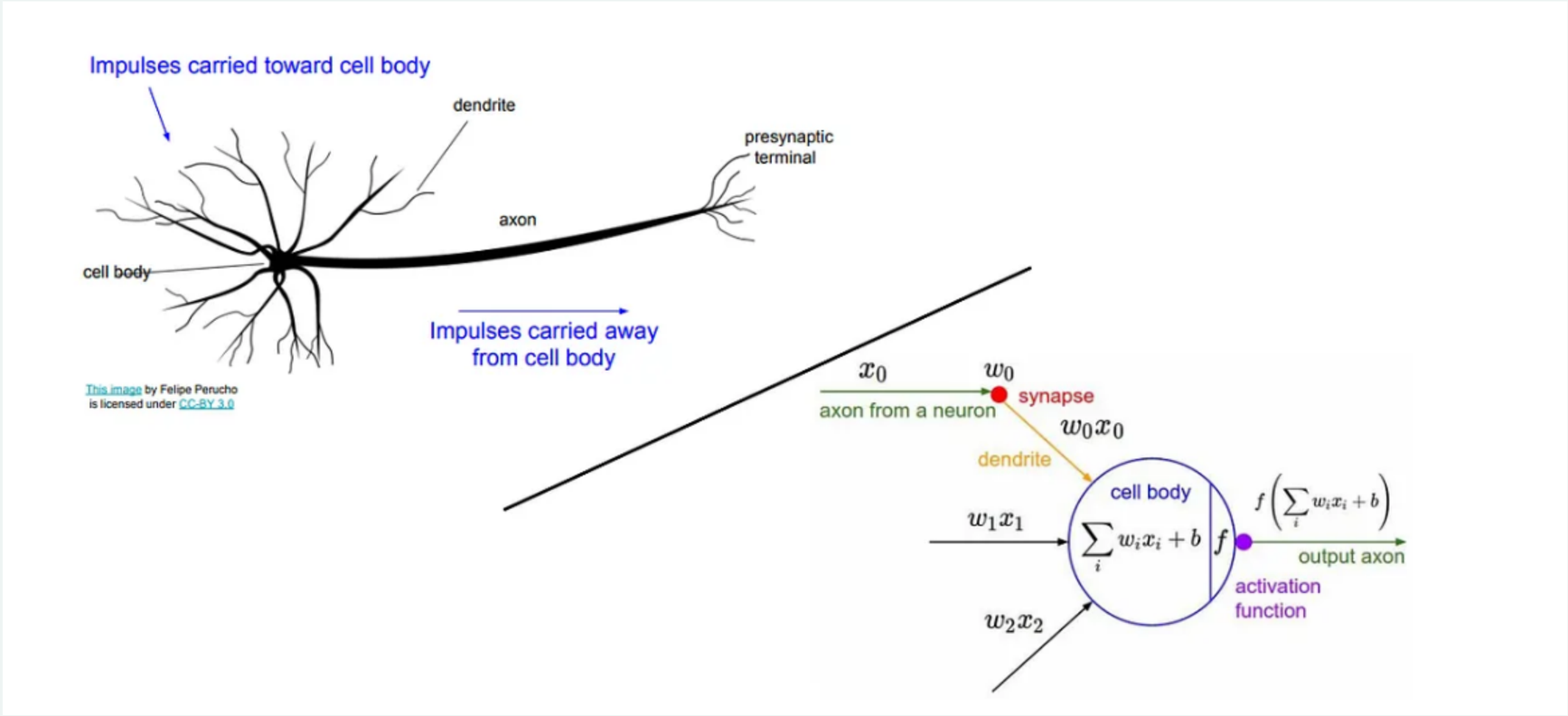
- Introduction
- Perbedaan Machine Learning dan Deep Learning
- Data Structure and Unstructure
- Aplikasi Deep Learning
- Library yang digunakan
- Tensor
- Teori Deep Learning
- Hands-On

INTRODUCTION
APA ITU DEEP LEARNING?

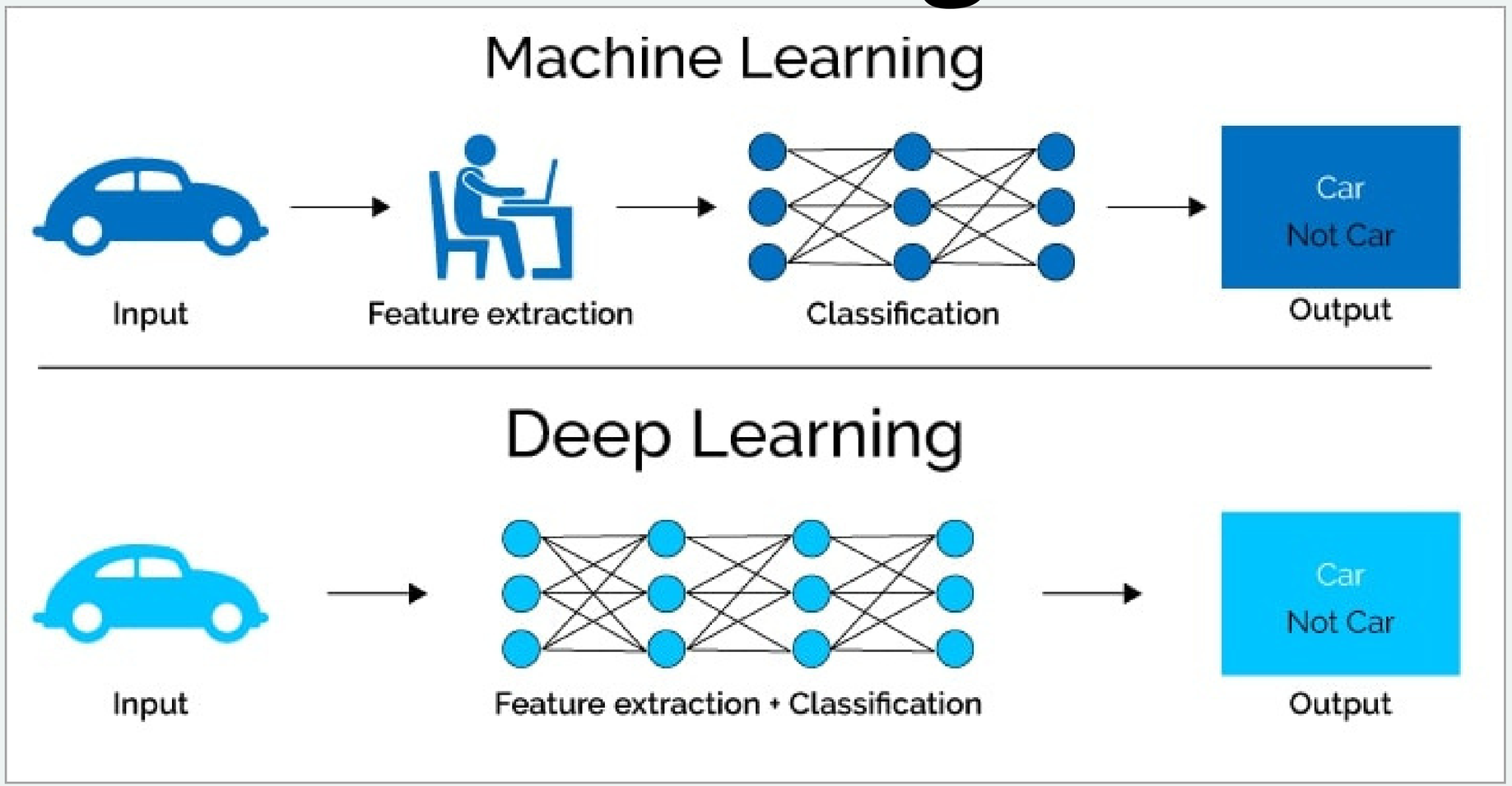
Deep learning adalah metode dalam kecerdasan buatan (AI)
yang mengajarkan komputer
untuk memproses data dengan
cara yang mirip dengan cara otak manusia berpikir dan belajar.



KESAMAAN DENGAN SARAF MANUSIA



Machine Learning vs Deep Learning



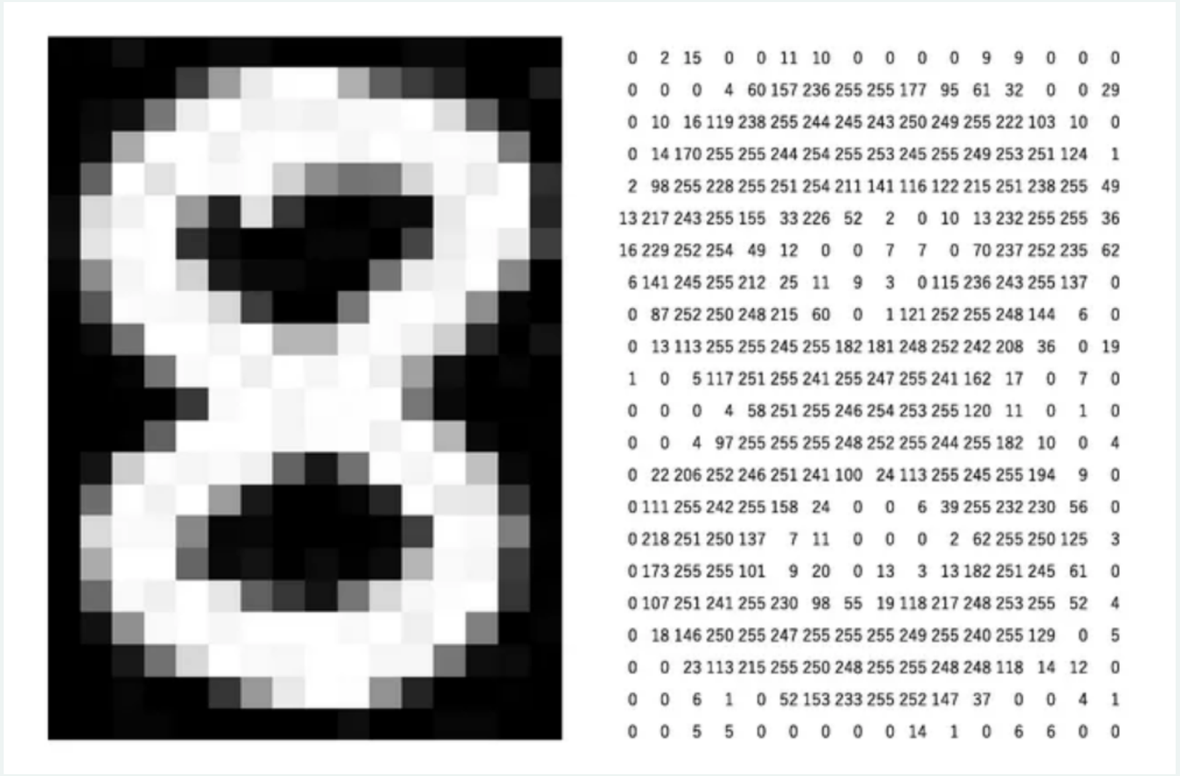
Data Structure

Data yang terorganisasi contohnya data tabular.
Machine Learning Task

	name	year	selling_price	km_driven	fuel	seller_type	transmission	owner
0	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner

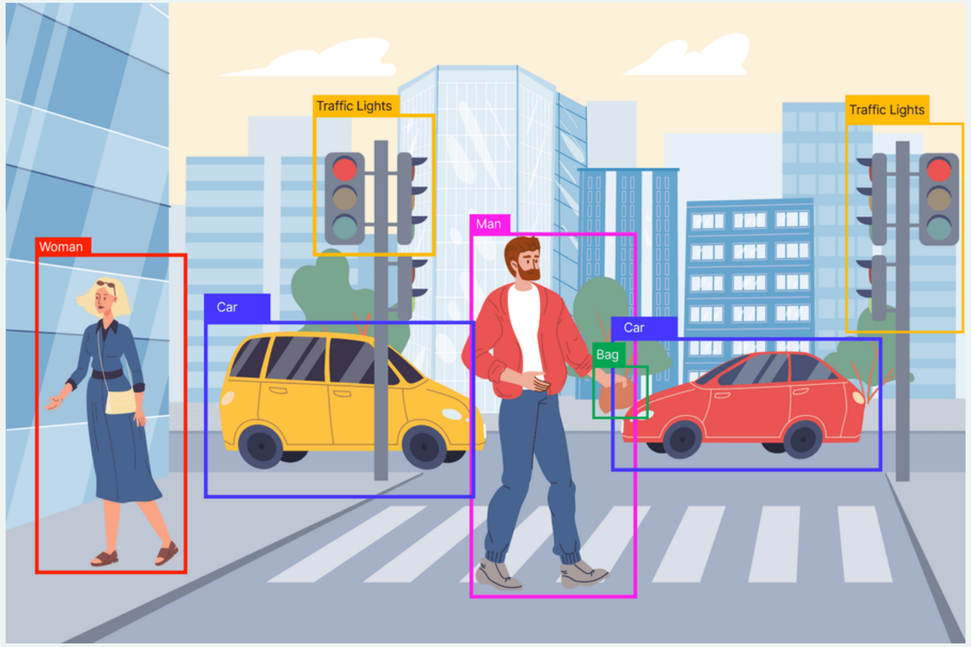
Data Unstructure

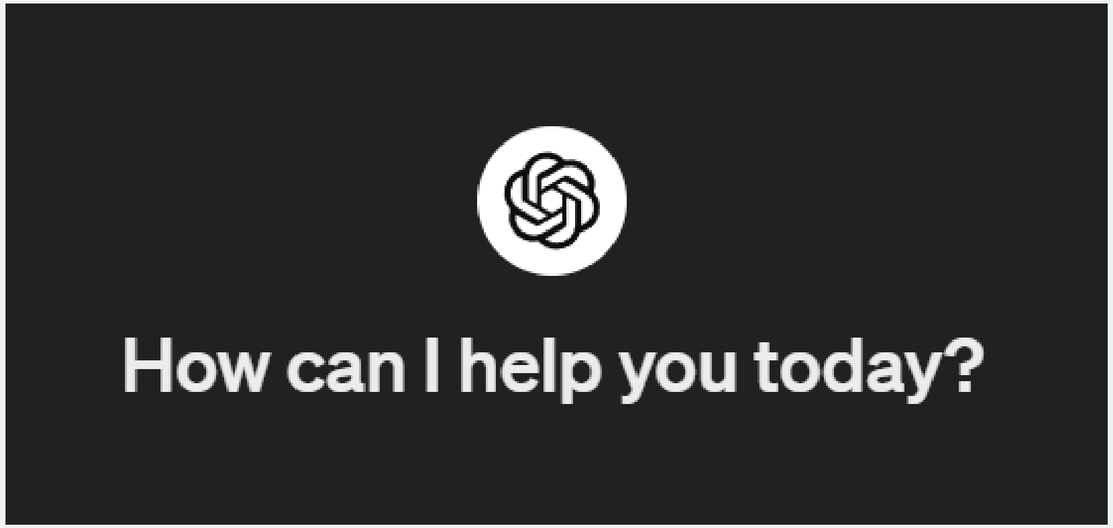
Unstructured data adalah data yang tidak memiliki format atau model data yang telah ditetapkan. Ini termasuk media, gambar, audio, data sensor, teks, dan banyak lagi.



0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	255	249	255	240	255	129	0	5
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	0

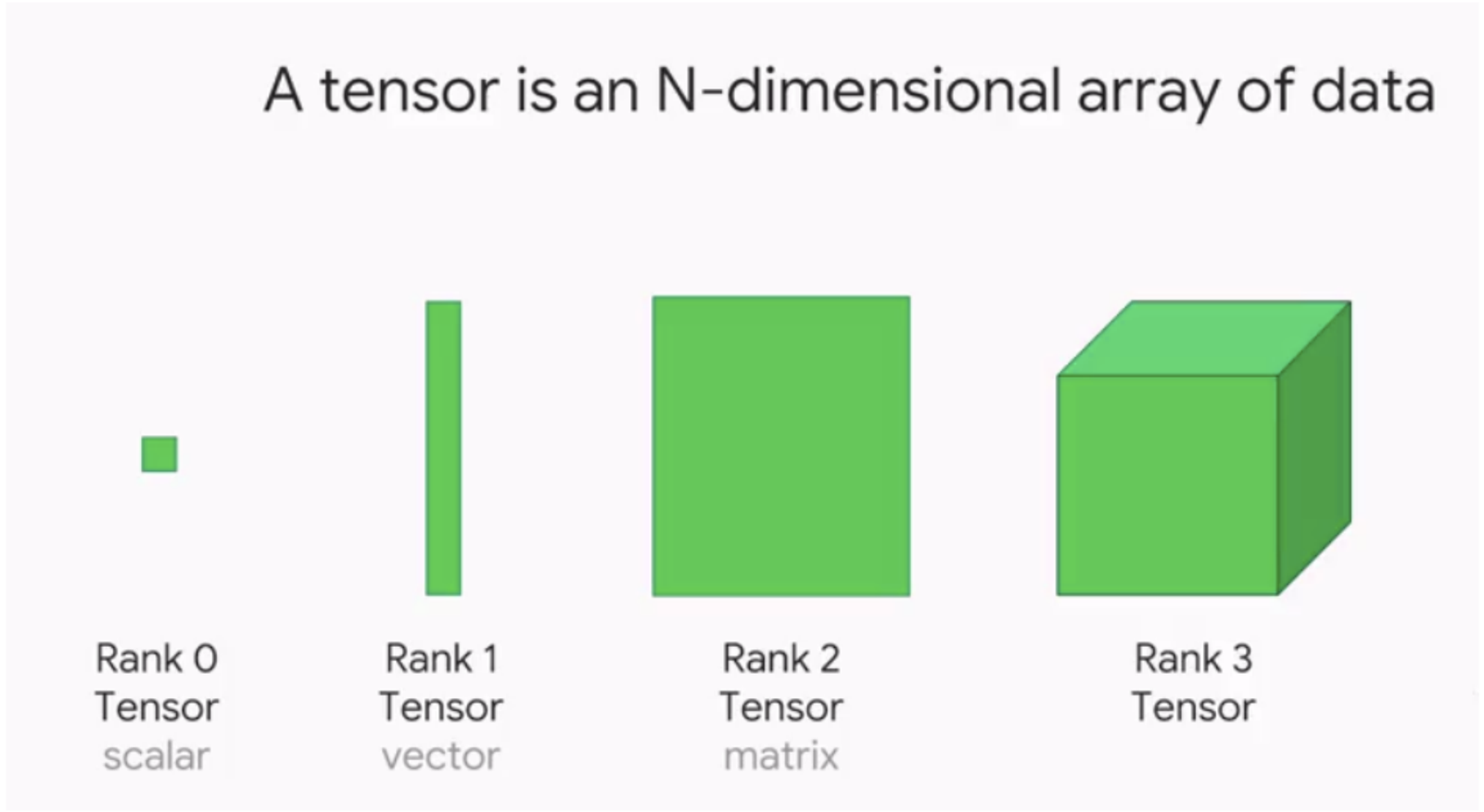
Aplikasi Deep Learning



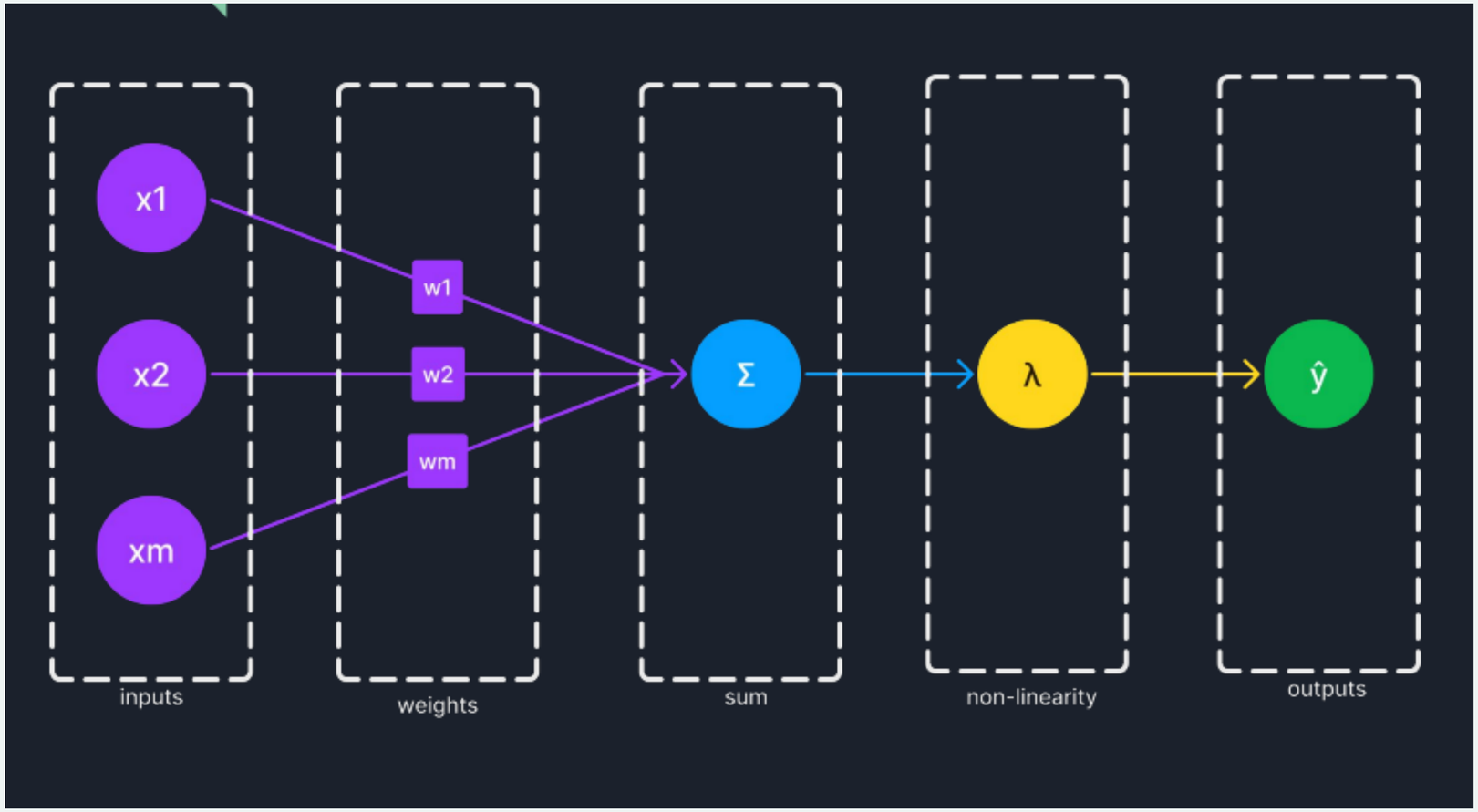


Library yang Digunakan

Tensor


A tensor is an N-dimensional array of data
Rank 0
Tensor
scalar
Rank 1
Tensor
vector
Rank 2
Tensor
matrix
Rank 3
Tensor

Neural Network



Output

Linear combination of inputs

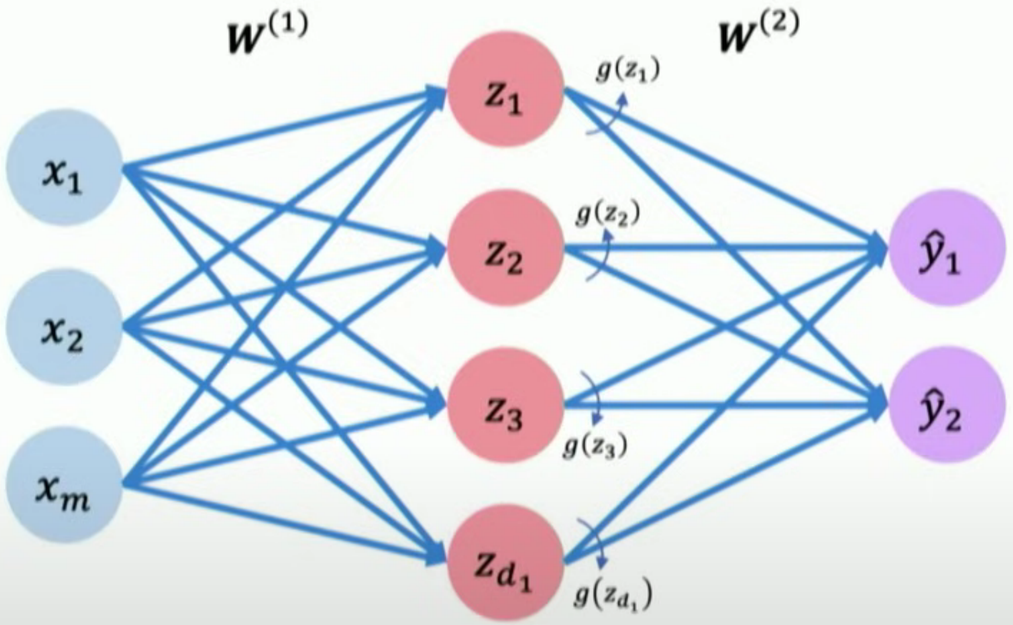
$$\hat{y} = g\left(\sum_{i=1}^{m} x_i \, w_i\right)$$

Non-linear activation function

$$\hat{y} = g(w_0 + \boldsymbol{X}^T \boldsymbol{W})$$

where: $\boldsymbol{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$ and $\boldsymbol{W} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$

Single Layer Neural Network

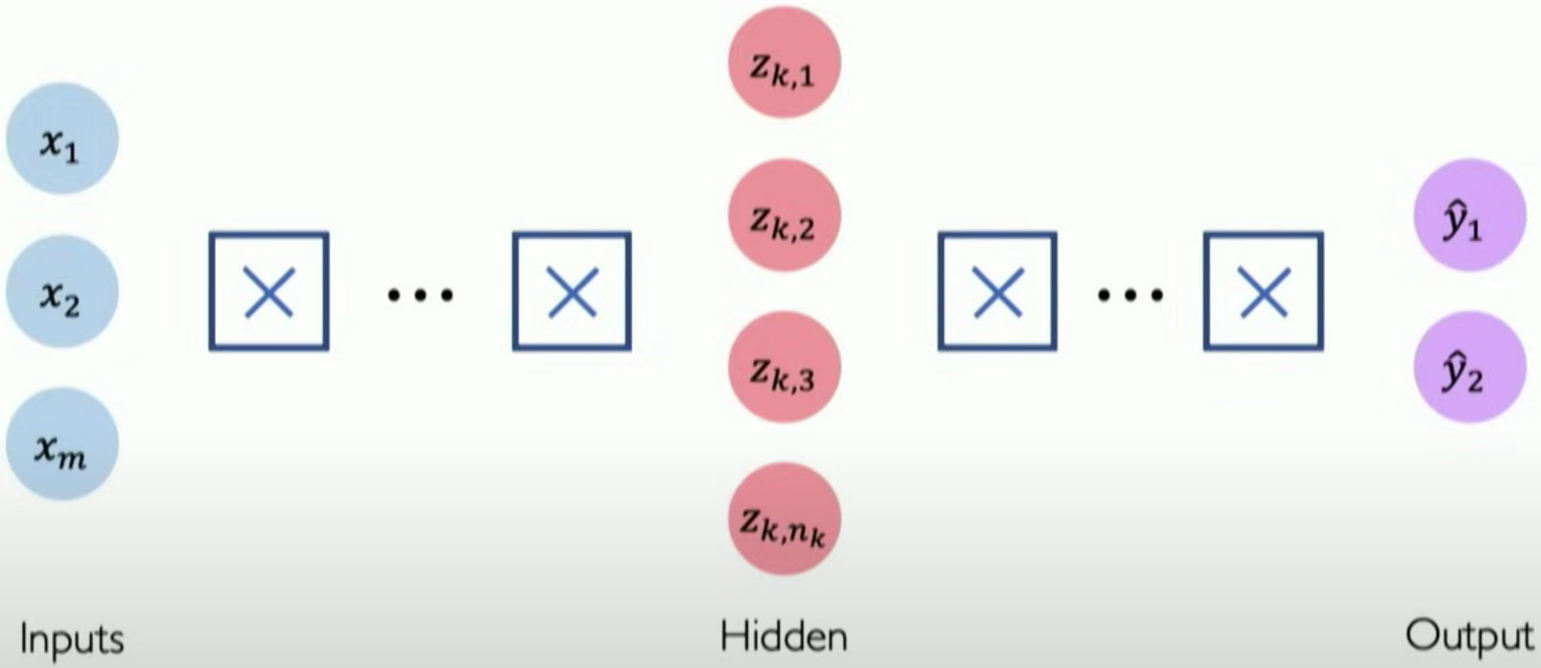


Inputs Hidden Final Output

$$z_i = w_{0,i}^{(1)} + \sum_{j=1}^{m} x_j \, w_{j,i}^{(1)}$$

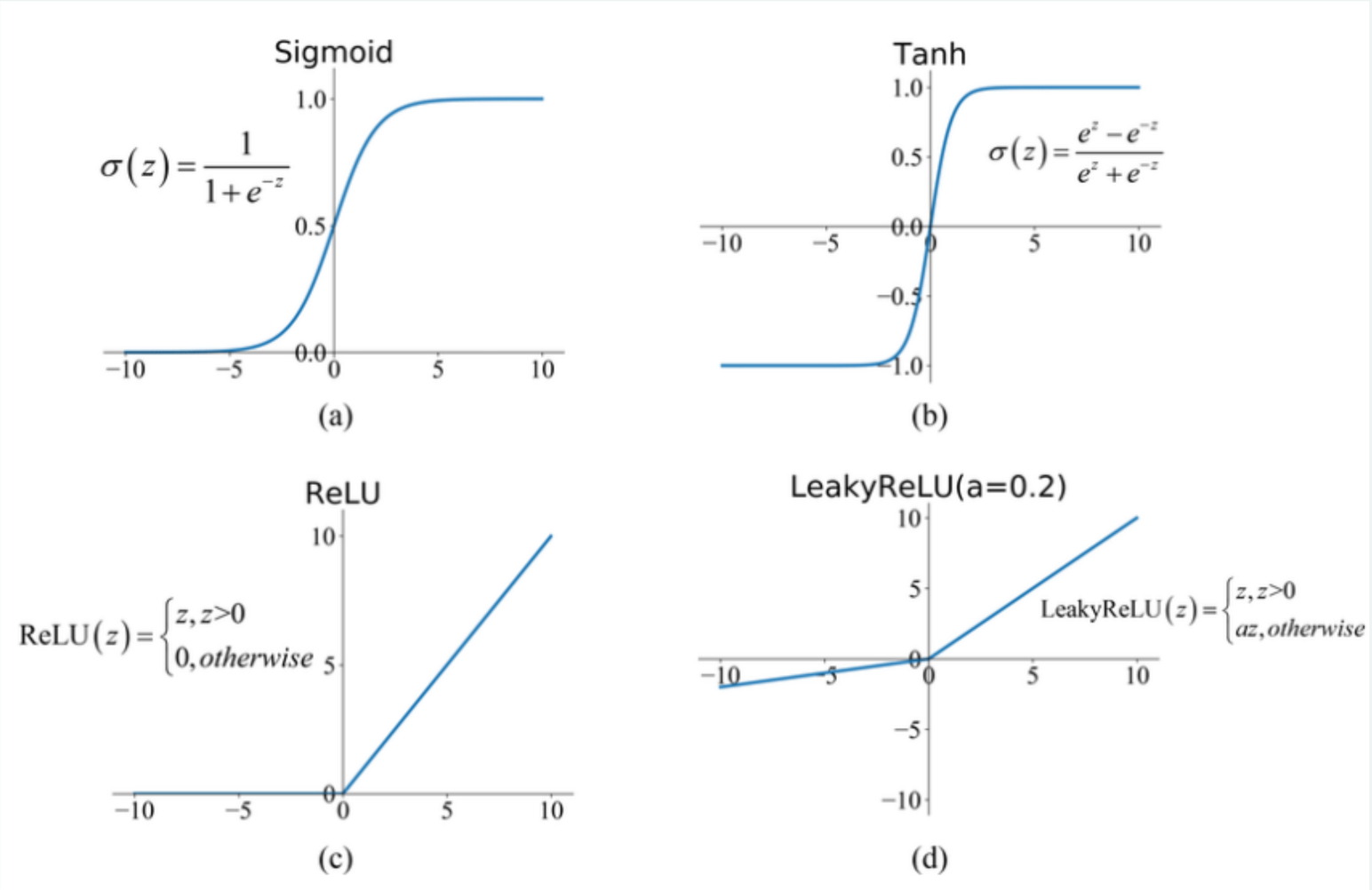
$$\hat{y}_i = g\left(w_{0,i}^{(2)} + \sum_{j=1}^{d_1} g(z_j) \, w_{j,i}^{(2)} \right)$$

Deep Neural Network



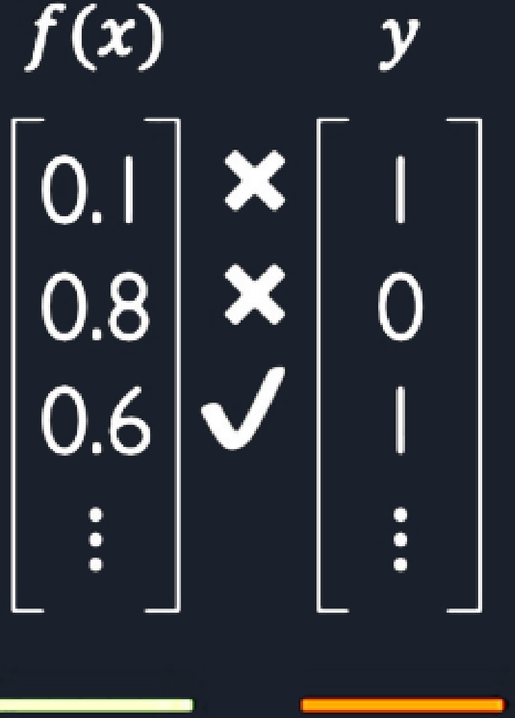
$$z_{k,i} = w_{0,i}^{(k)} + \sum_{j=1}^{n_{k-1}} g(z_{k-1,j})\, w_{j,i}^{(k)}$$

Activation Function



Loss Function

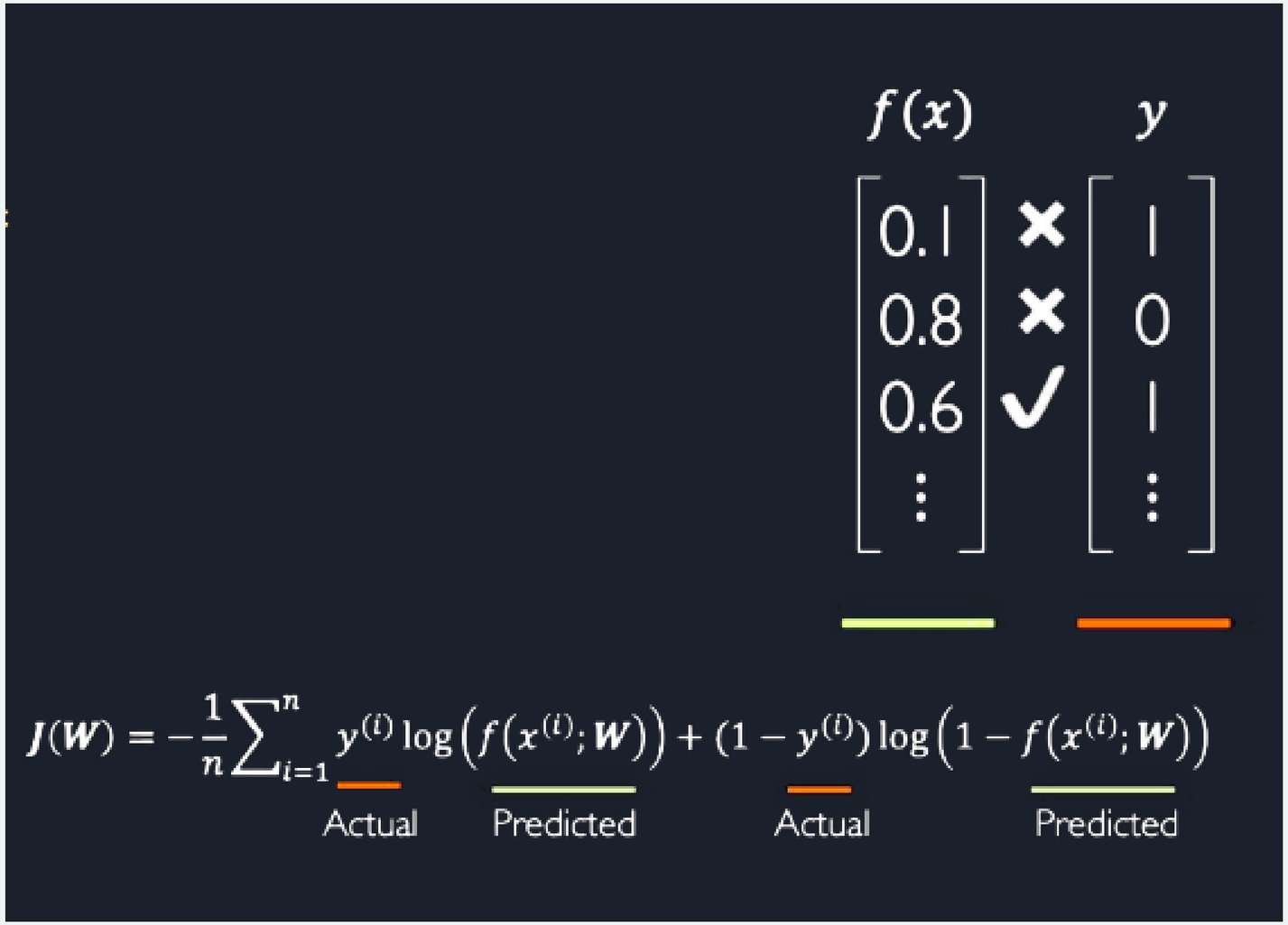
Loss measure the cost of incurred from incorrect predictions



$$\mathcal{L}\left(f\left(x^{(i)}; \boldsymbol{W}\right), y^{(i)}\right)$$

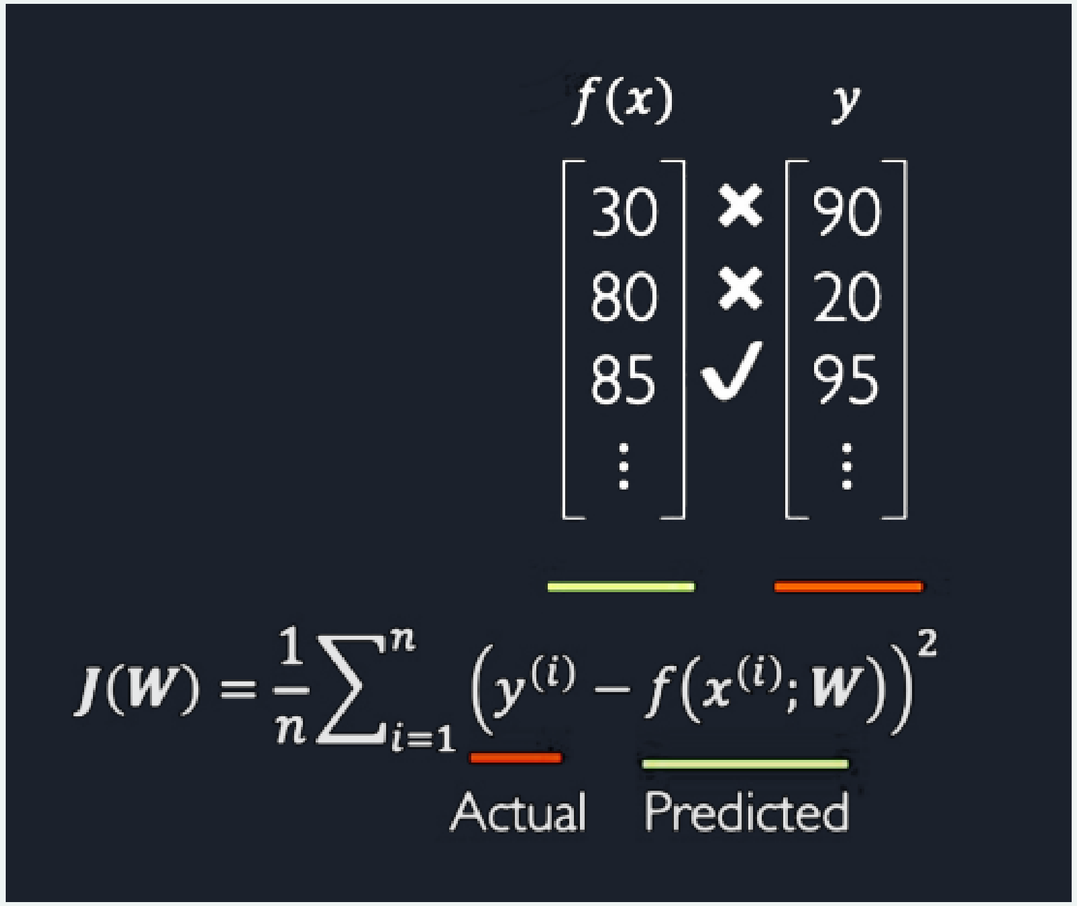
Predicted Actual

Loss Function Cross Entropy Loss



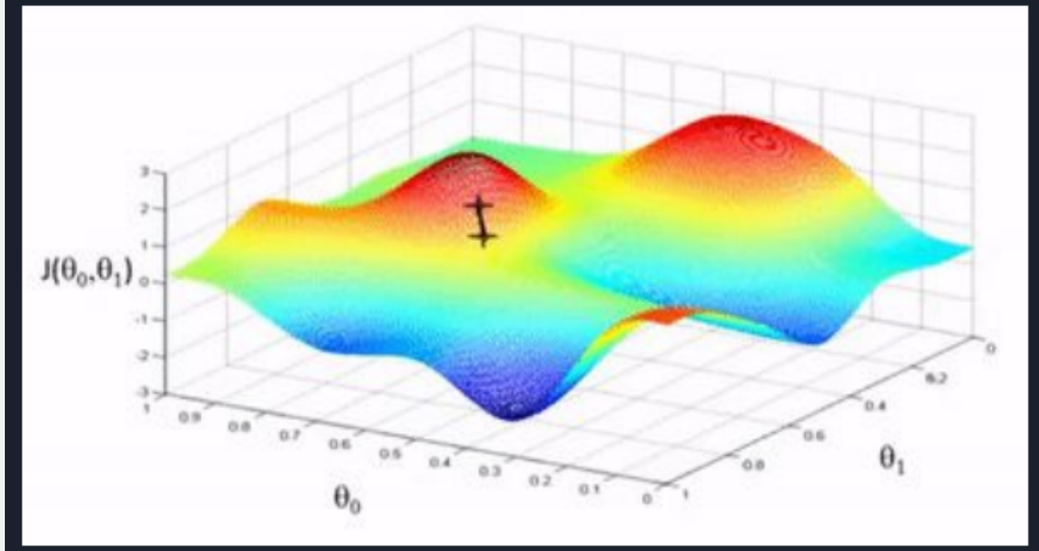
- Menghitung probabiliti dari 0 ke 1
- Binary Classification.
- Multiclass Classification

Loss Function Mean Squared Error Loss



- Menghitung rata2 kuadrat jarak dari hasil asli dan prediksi
- Buat regresi model.

Gradient Descent



Algorithm

1. Initialize weights randomly $\sim \mathcal{N}(0, \sigma^2)$
2. Loop until convergence:
3. Compute gradient, $\frac{\partial J(\boldsymbol{W})}{\partial \boldsymbol{W}}$
4. Update weights, $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \frac{\partial J(\boldsymbol{W})}{\partial \boldsymbol{W}}$
5. Return weights

Optimizer

- Adam
- SGD
- RMSProp
- Adagrad

Backpropagation