

# NEURAL NETWORK & DEEP LEARNING

BY TAUTOLOGY

# Neural Network & Deep Learning

Neural Network

Deep Learning

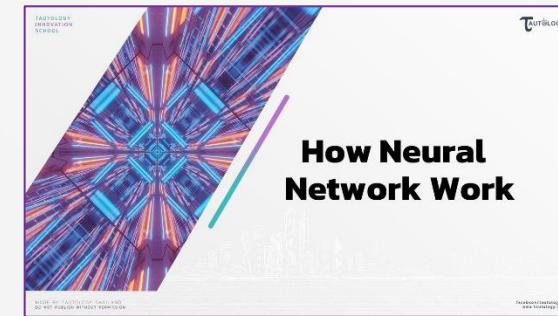
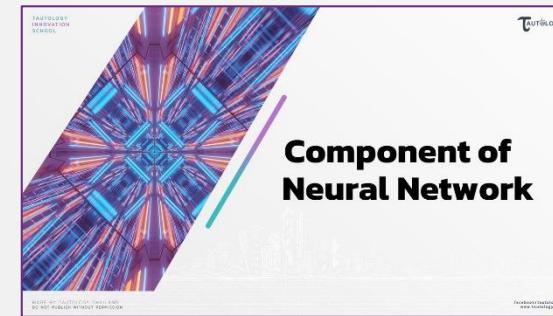
Architecture  
Interpretation

Deep Learning for  
Regression

Deep Learning for  
Classification

Workshop

# Neural Network



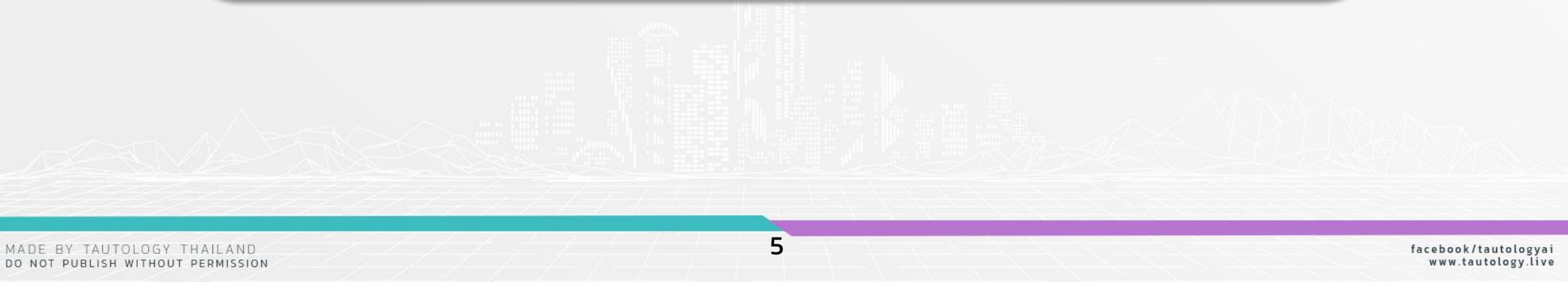


# What is Neural Network?

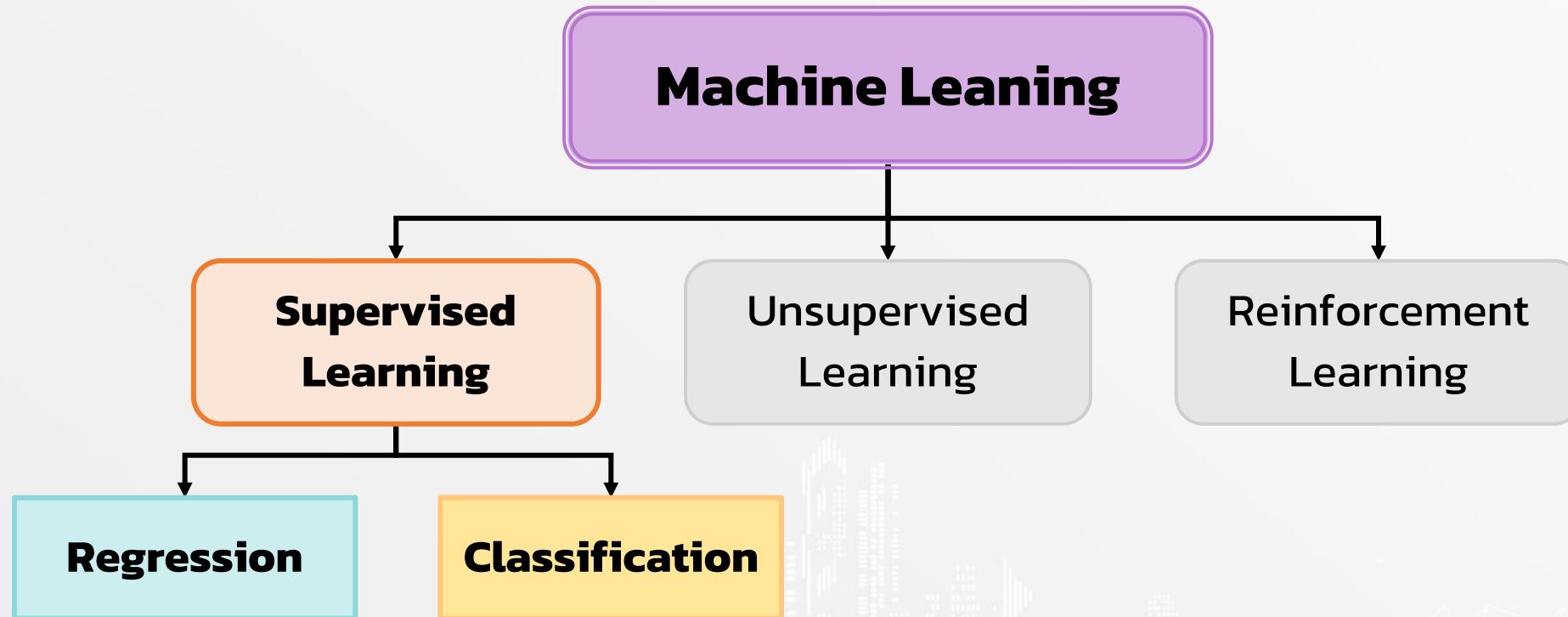
# What is Neural Network?

**Neural Network** เป็นหนึ่งใน algorithm ประเภท supervised learning ที่ใช้สำหรับแก้ปัญหา regression และ classification

โดยมีหลักการทำงาน คือ **การนำ nonlinear function มาประกอบกัน** เพื่อประมาณ nonlinear function ที่ซับซ้อนได้

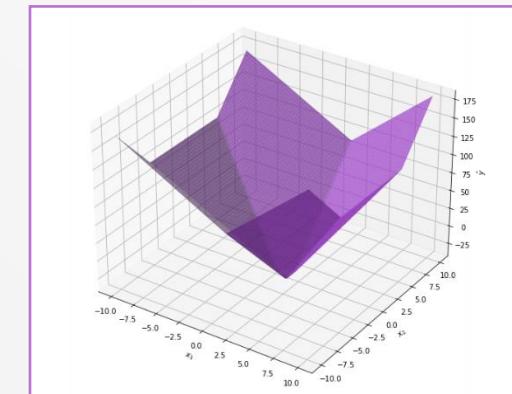
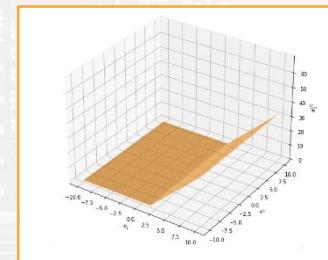
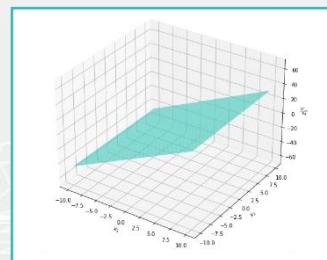
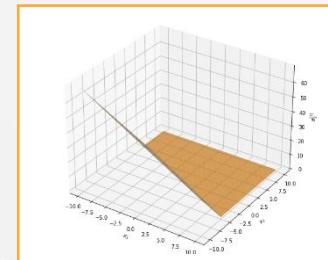
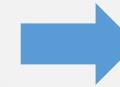
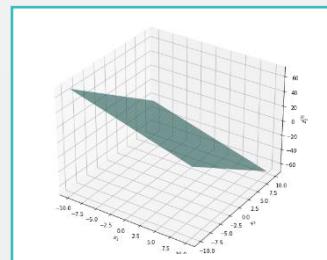
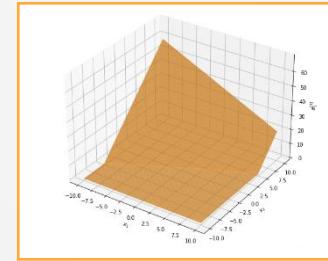
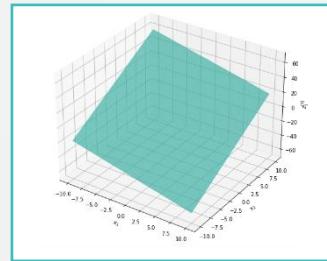


# What is Neural Network?



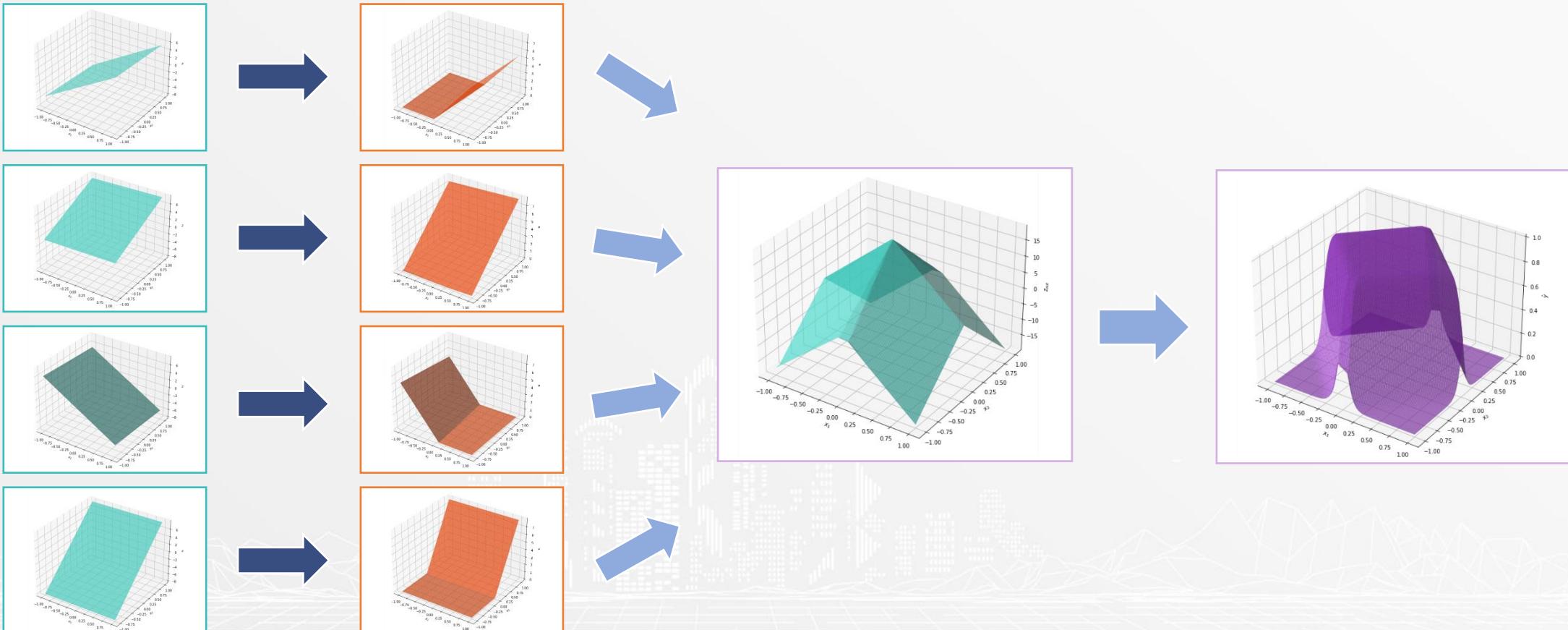
# What is Neural Network?

## Regression

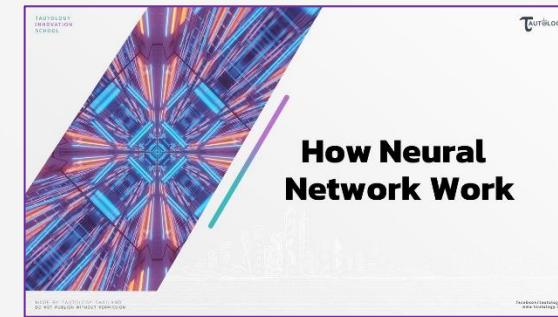
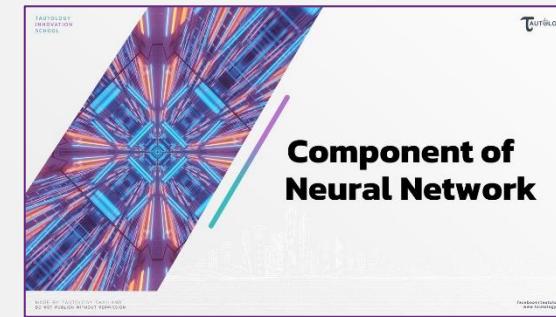
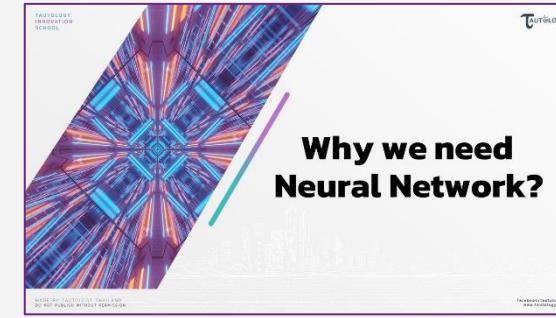
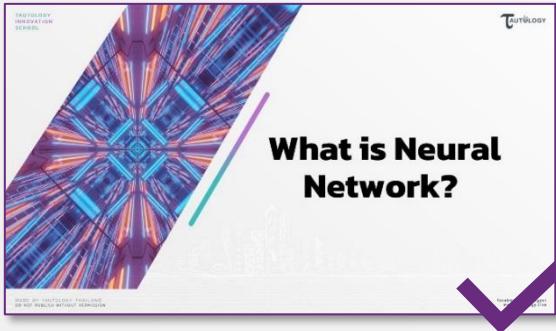


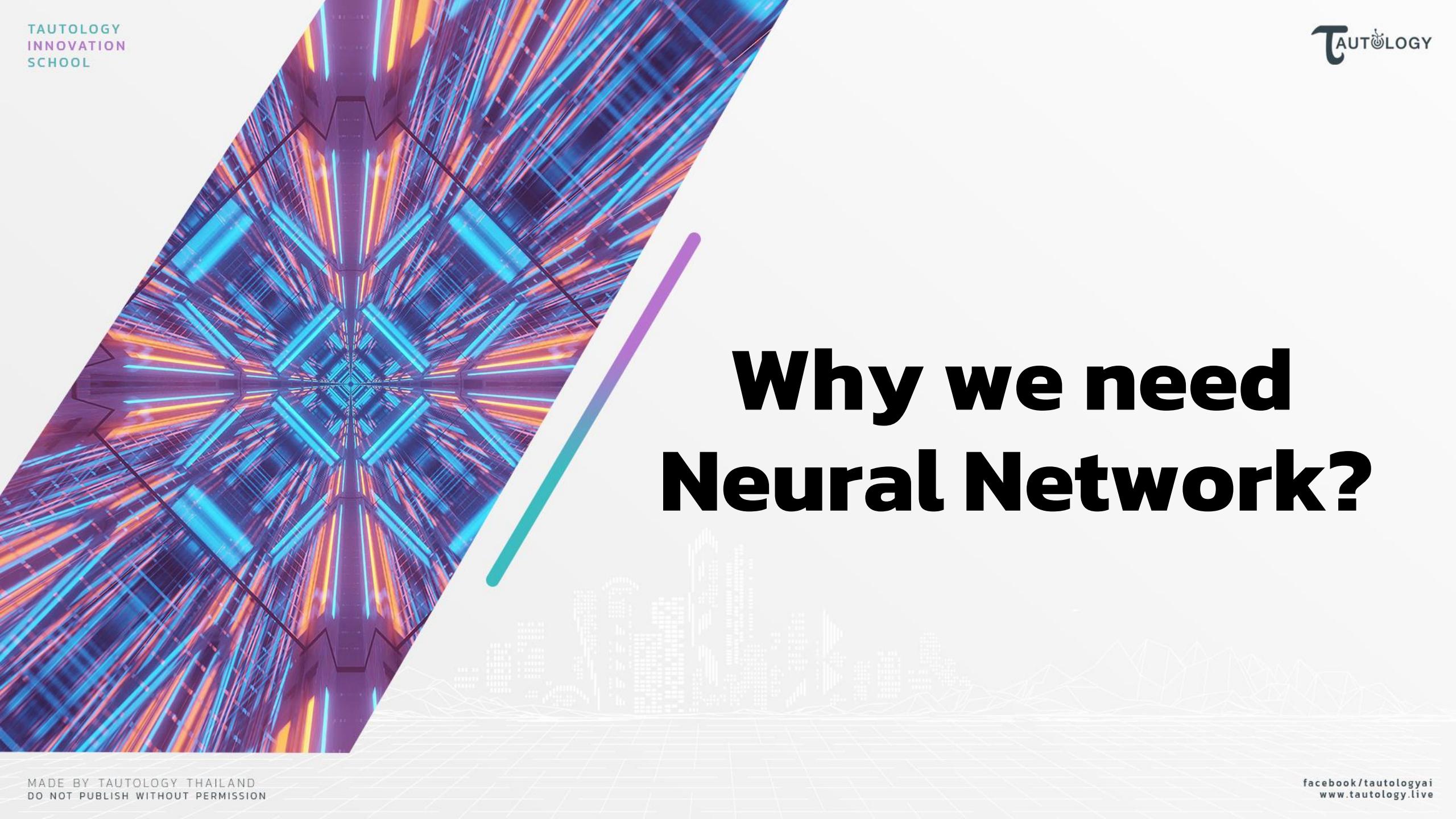
# What is Neural Network?

## Classification



# Neural Network



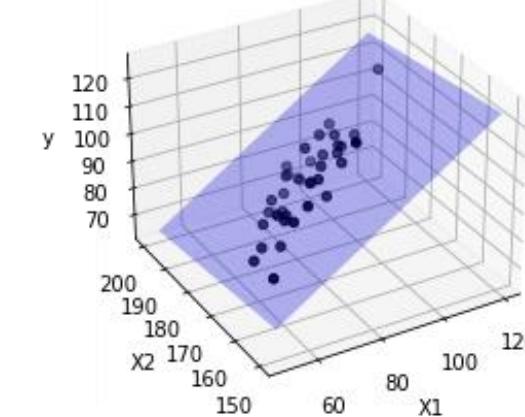


# Why we need Neural Network?

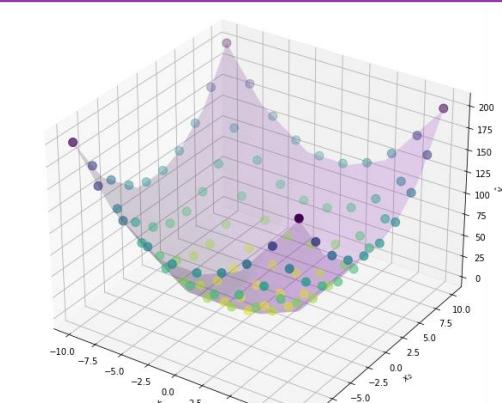
# Why we need Neural Network?

## Regression

Neural network ใช้เพื่อประมาณ nonlinear function



Linear Regression

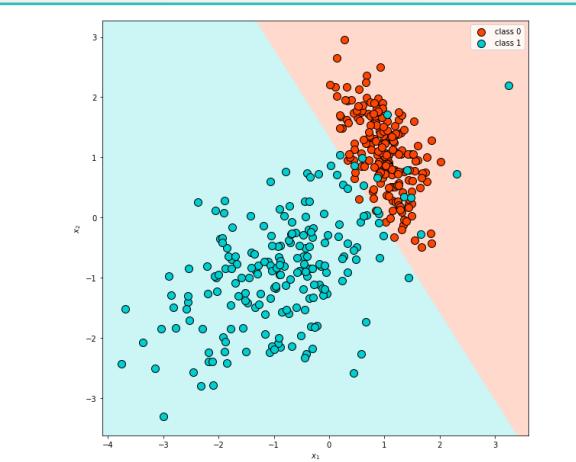


Neural Network

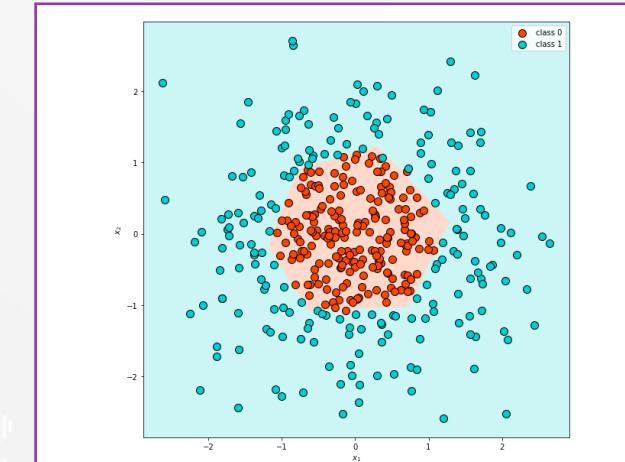
# Why we need Neural Network?

## Classification

Neural network สามารถสร้าง decision boundary ที่ไม่ใช่เส้นตรงได้



Logistic Regression

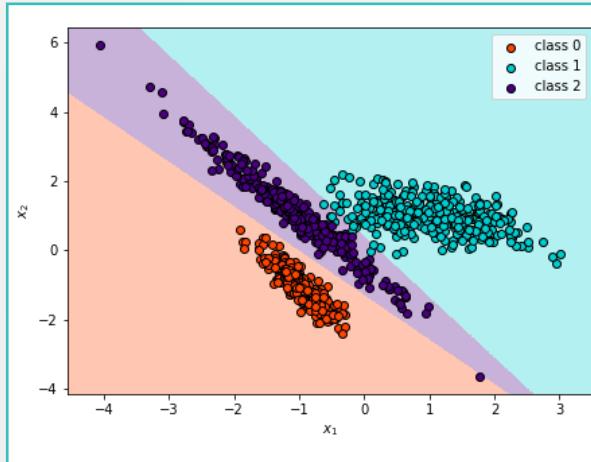


Neural Network

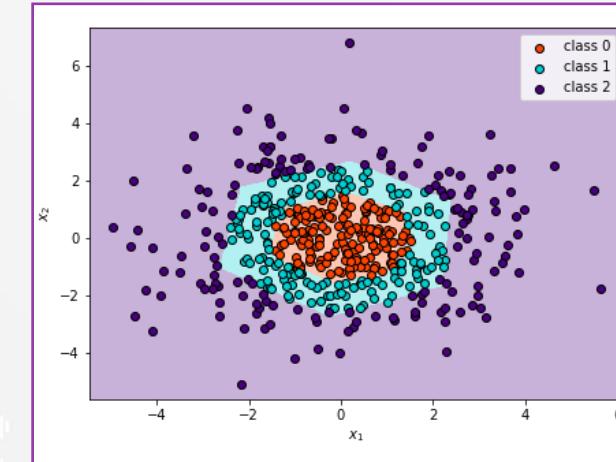
# Why we need Neural Network?

## Classification

Neural network สามารถสร้าง decision boundary ที่ไม่ใช่เส้นตรงได้

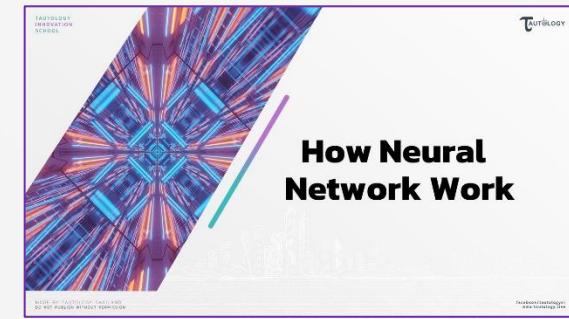
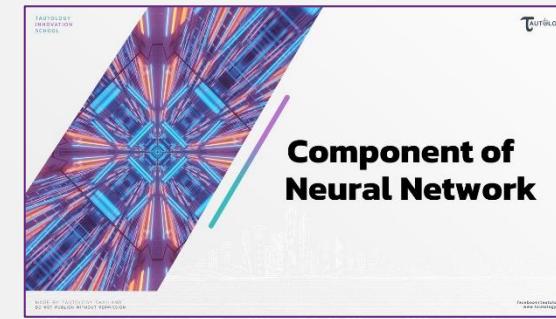
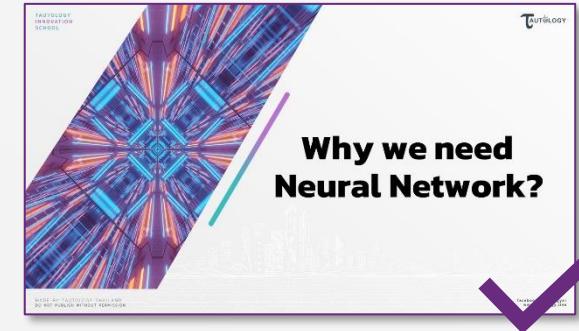
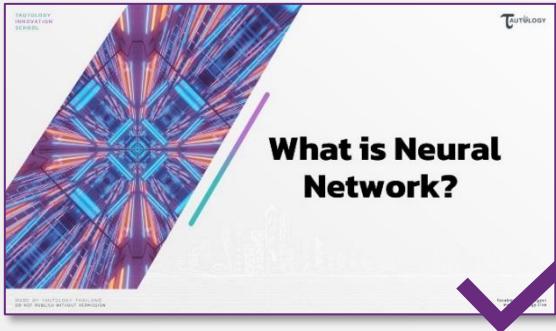


Logistic Regression



Neural Network

# Neural Network



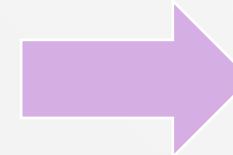


# Real World Application

# Real World Application



[2020, H. I. Lee et al] Urban Flood Prediction Using Deep Neural Network with Data Augmentation



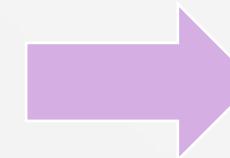
## การคาดการณ์ปริมาณน้ำสะสมในอ่างระบายน้ำแห่งหนึ่งในกรุงเทพฯ

โดยพิจารณาจากข้อมูลสกัดติดต่อ ณ ของวันที่ฝนตกหนักเป็นเวลา 6 ชั่วโมง เช่น ปริมาณน้ำฝนทึบหมัด ปริมาณน้ำฝนสูงสุดของแต่ละชั่วโมง ปริมาณน้ำฝนเฉลี่ยต่อชั่วโมง เป็นต้น

# Real World Application



[2016, J. Perricone et al.] Predicting Results for Professional Basketball Using NBA API Data



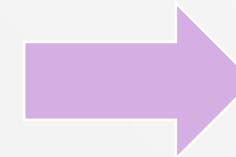
## การนำযผลバスเกตบอล NBA

โดยพิจารณาจากข้อมูลสถิติต่าง ๆ ของแต่ละทีม เช่น จำนวนครั้งในการยิงสามแต้ม คะแนน กี่ได้จากการยิงในพื้นที่ตัวเอง เปอร์เซ็นต์ความแม่นยำในการยิง เป็นต้น

# Real World Application



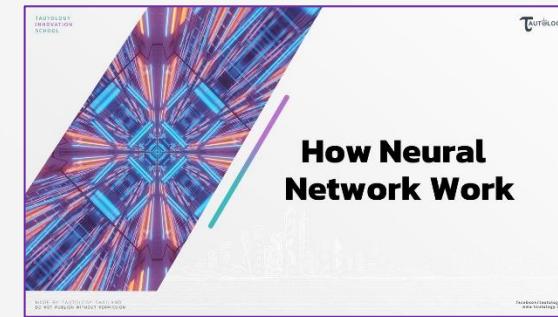
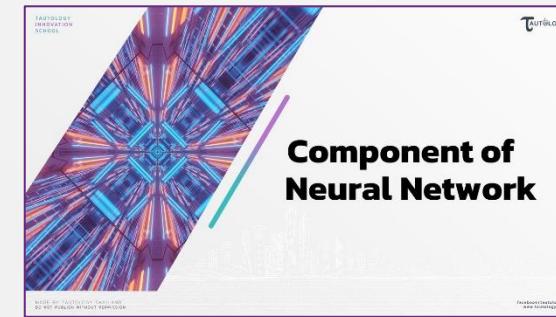
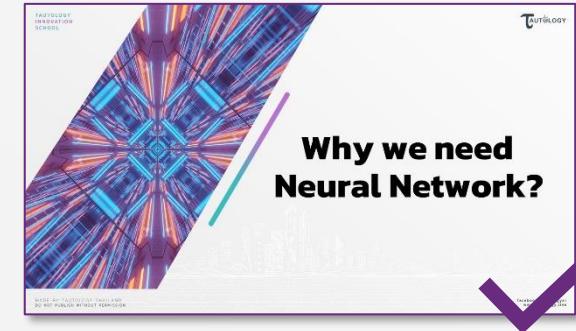
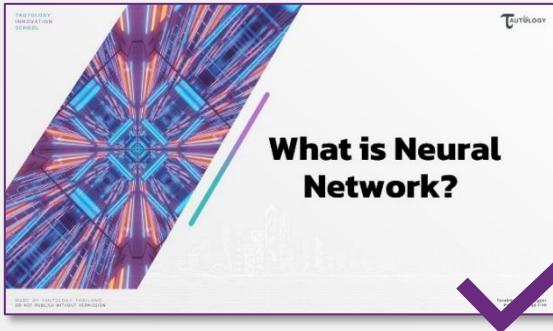
[2018, D. H. Mantzaris et al.] Medical Disease Prediction Using Artificial Neural Networks



**การวิเคราะห์ระดับความเสี่ยงที่จะเป็นโรคกระดูกพรุน**

โดยพิจารณาจากข้อมูลของผู้ป่วย ได้แก่ อายุ เพศ ส่วนสูง น้ำหนัก

# Neural Network



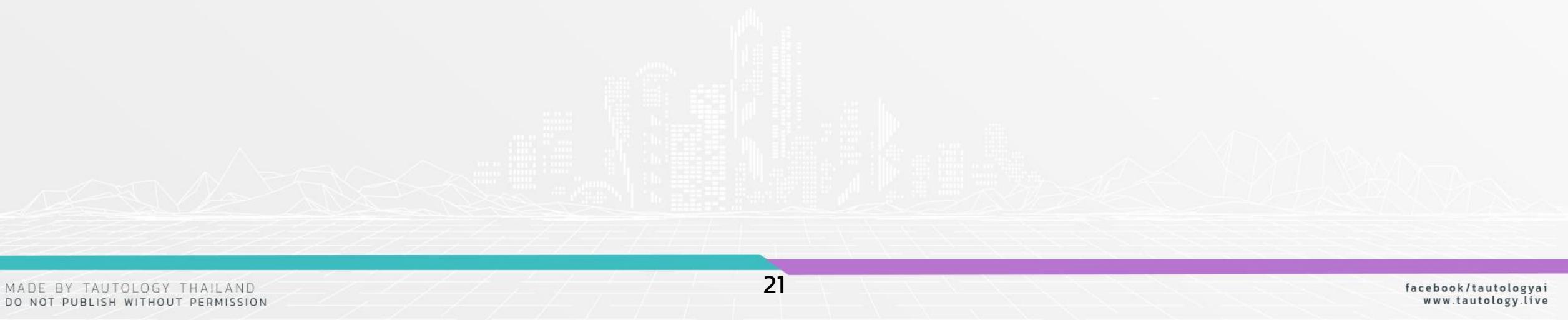
# Architecture of Neural Network

# Architecture of Neural Network

**Regression**

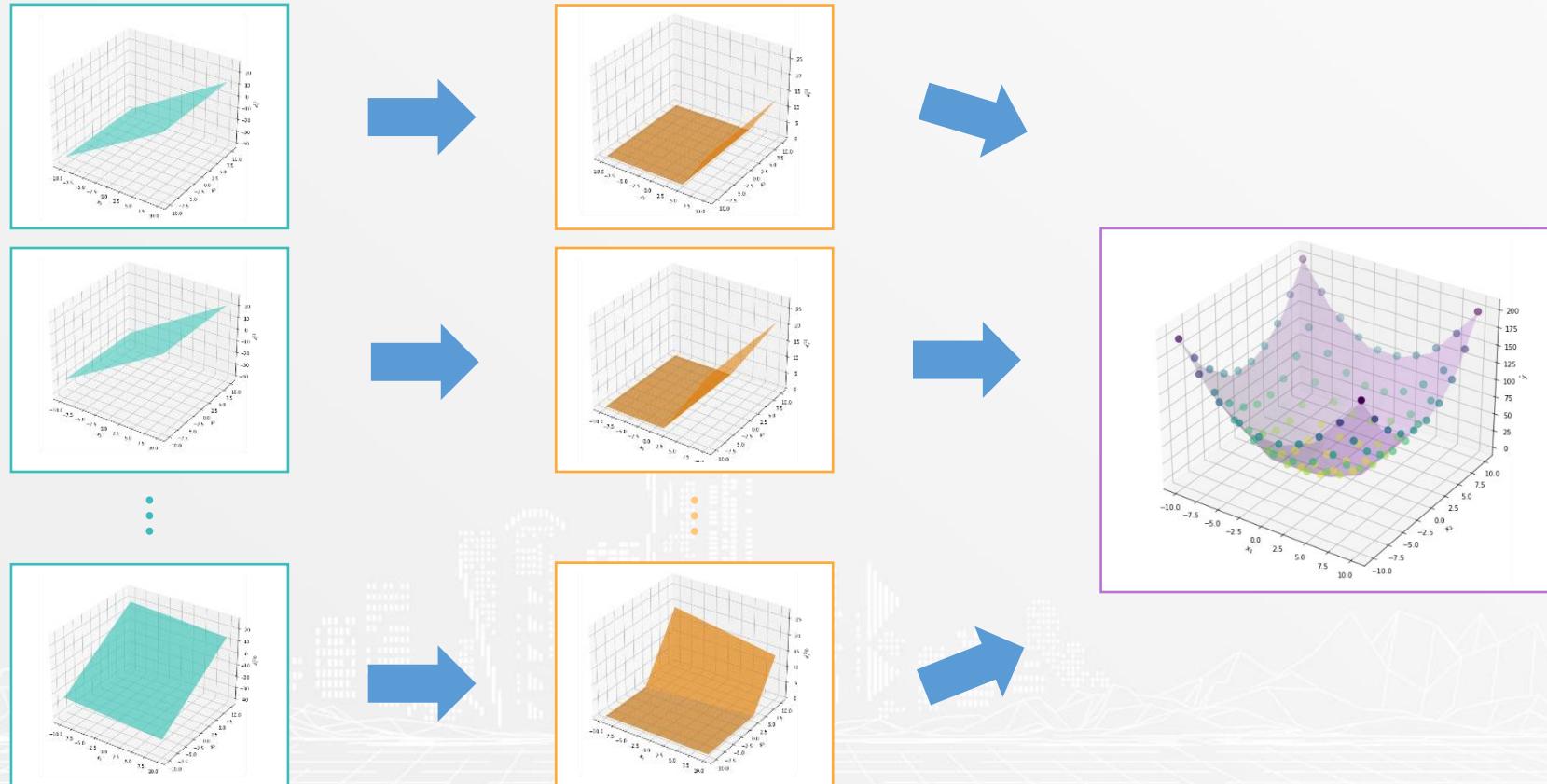
**Binary  
Classification**

**Multi-Class  
Classification**



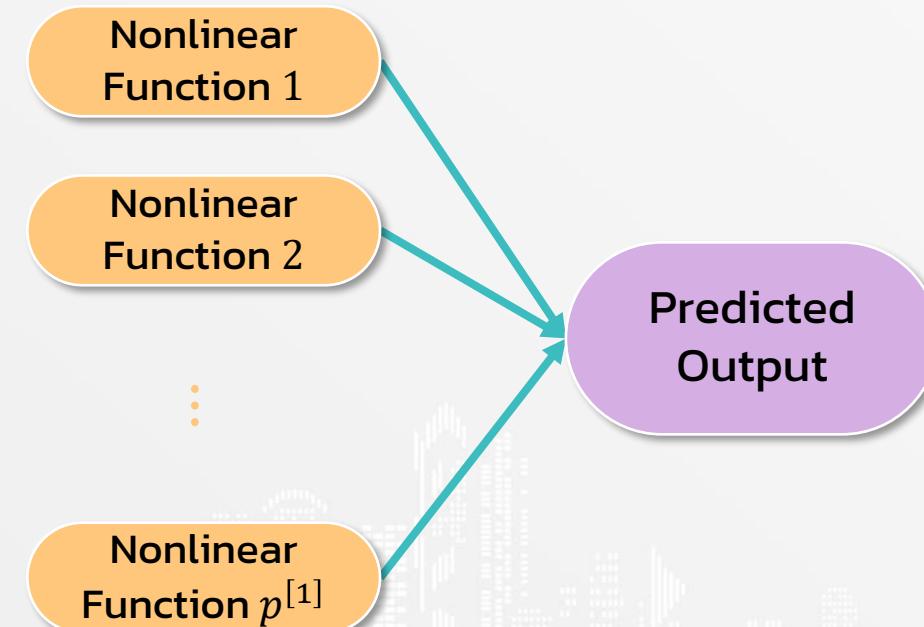
# Architecture of Neural Network

## Regression



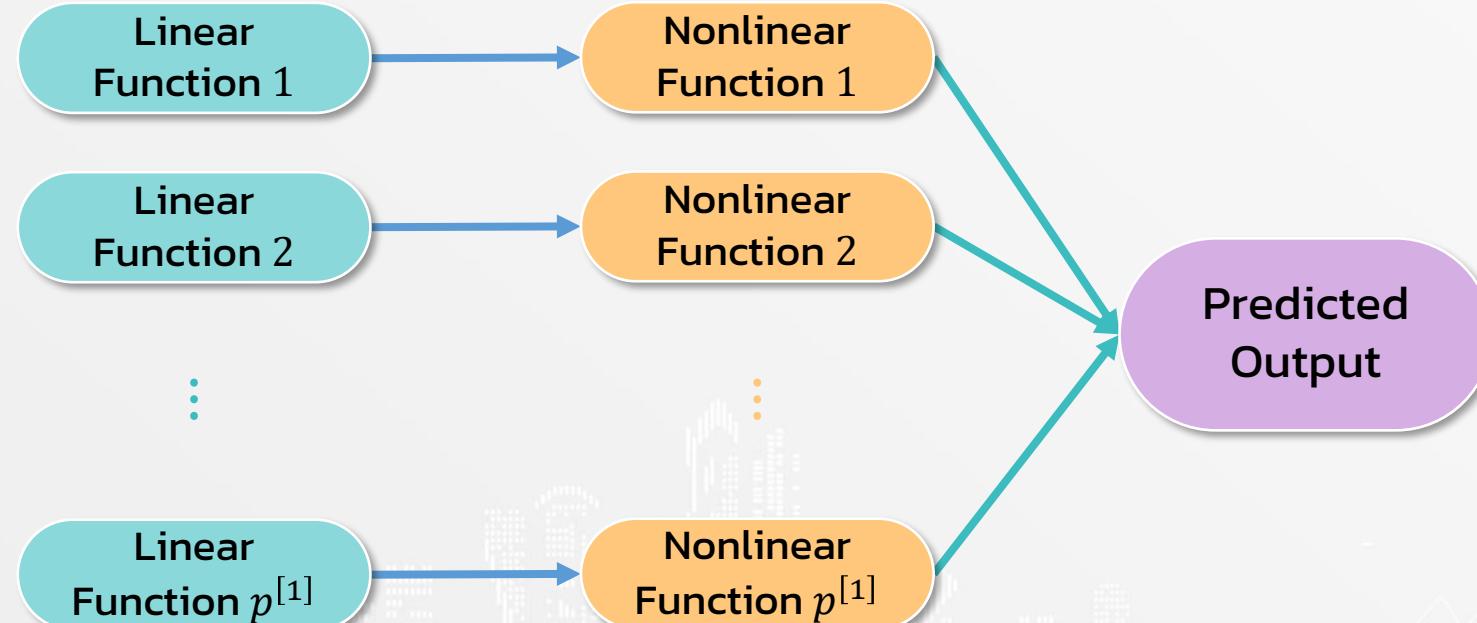
# Architecture of Neural Network

Regression

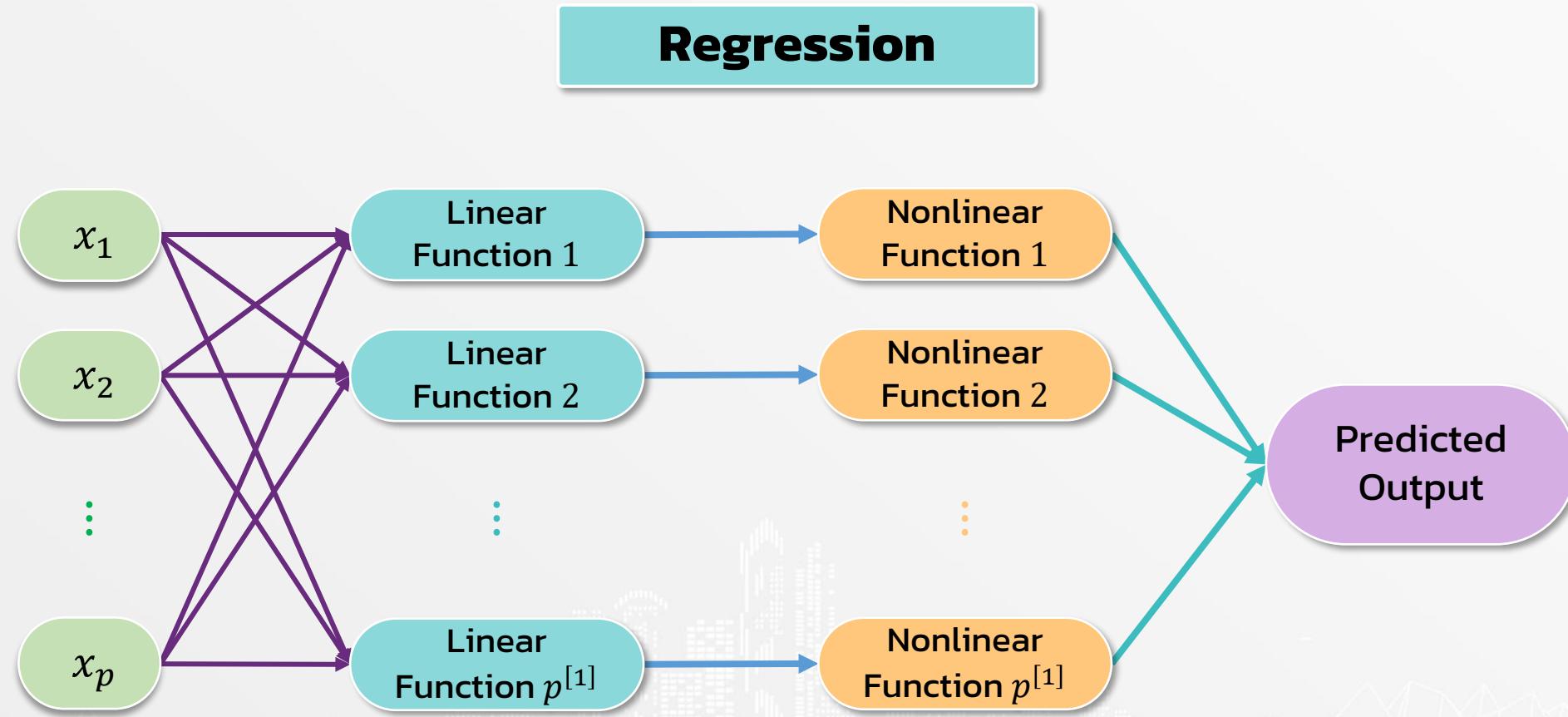


# Architecture of Neural Network

## Regression

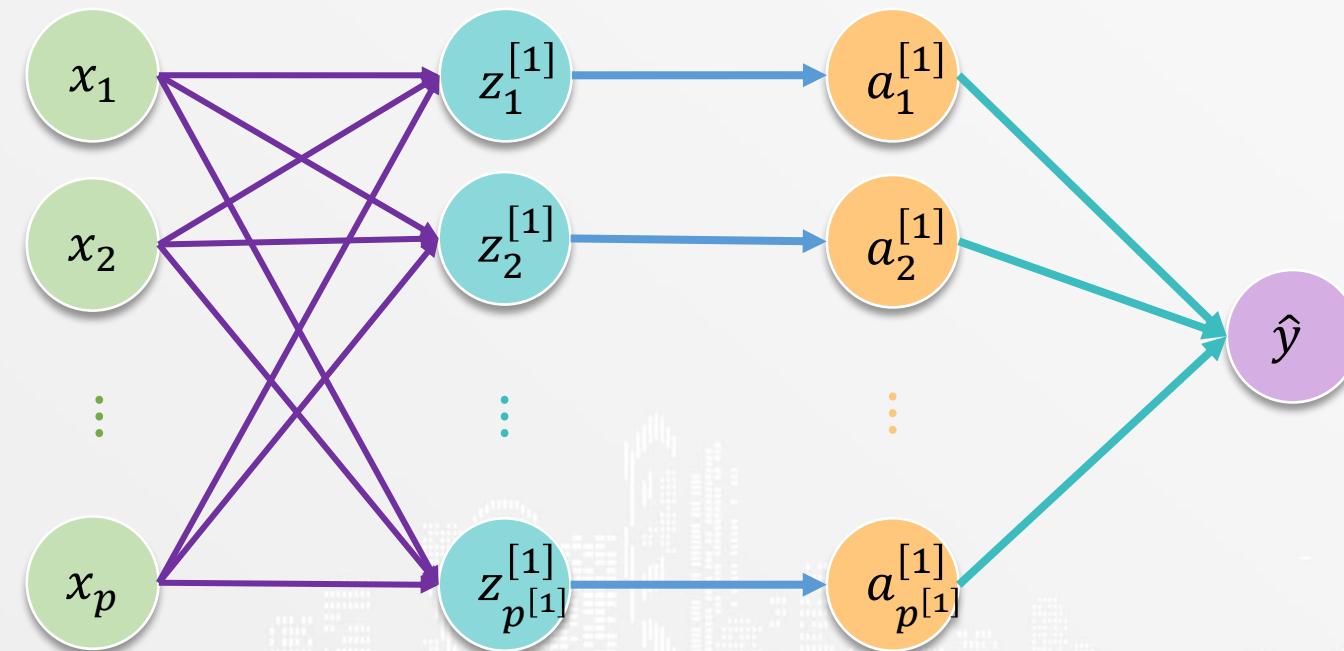


# Architecture of Neural Network



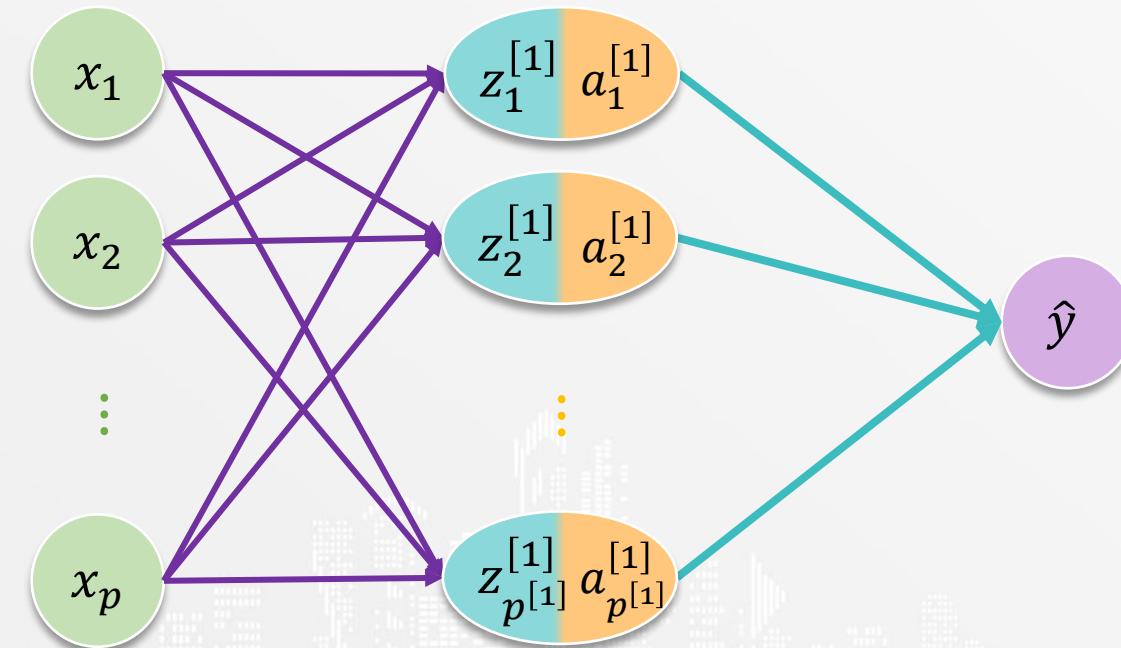
# Architecture of Neural Network

Regression



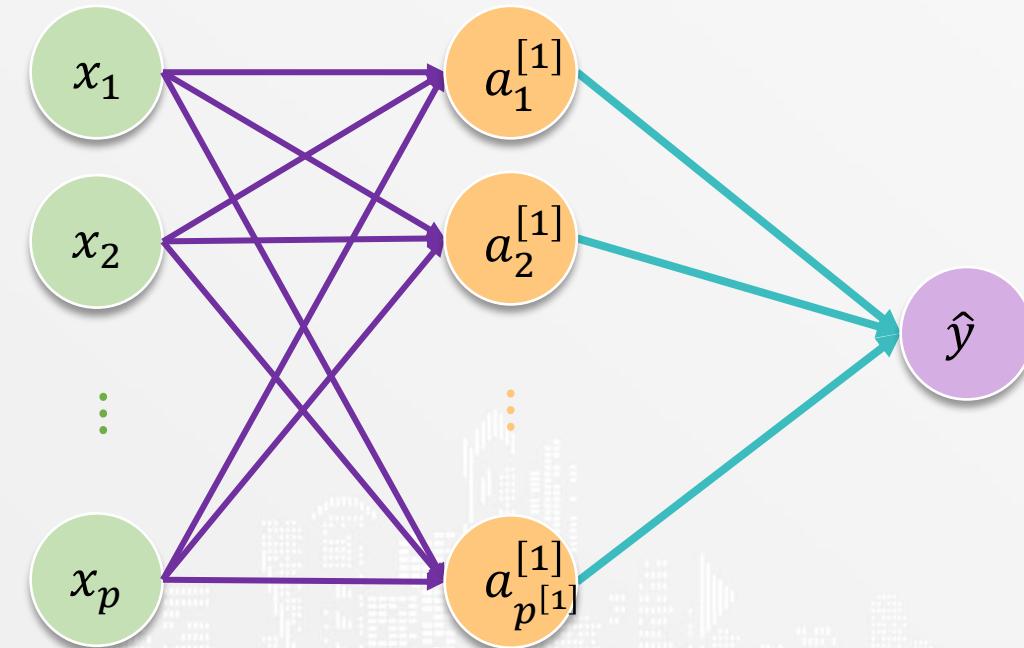
# Architecture of Neural Network

Regression



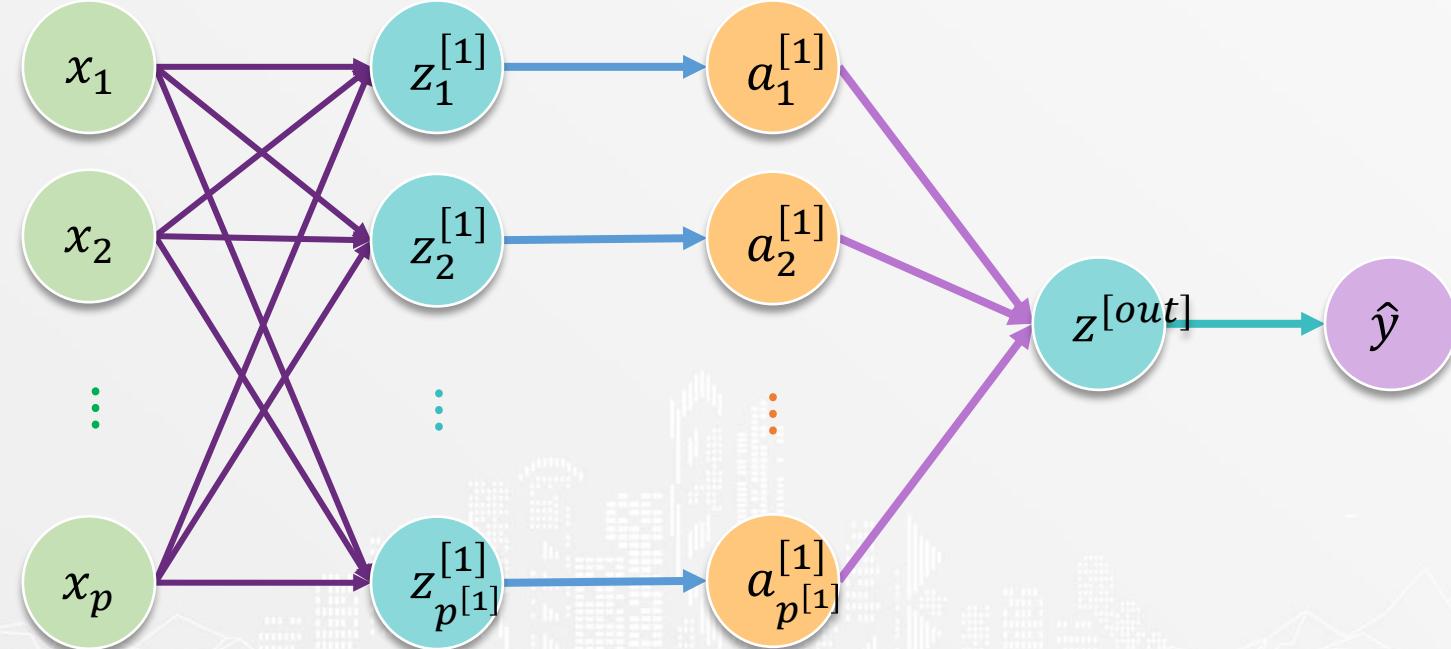
# Architecture of Neural Network

Regression



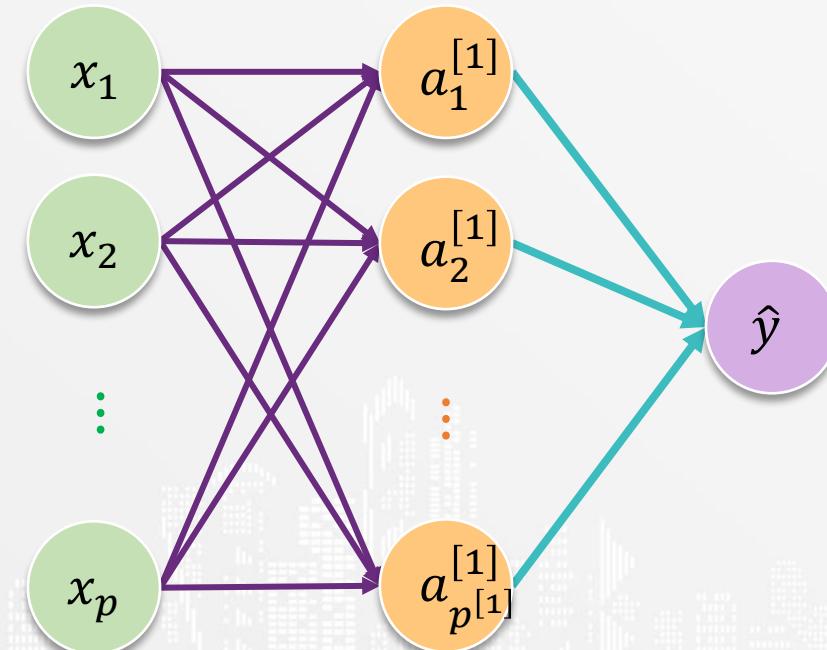
# Architecture of Neural Network

## Binary Classification



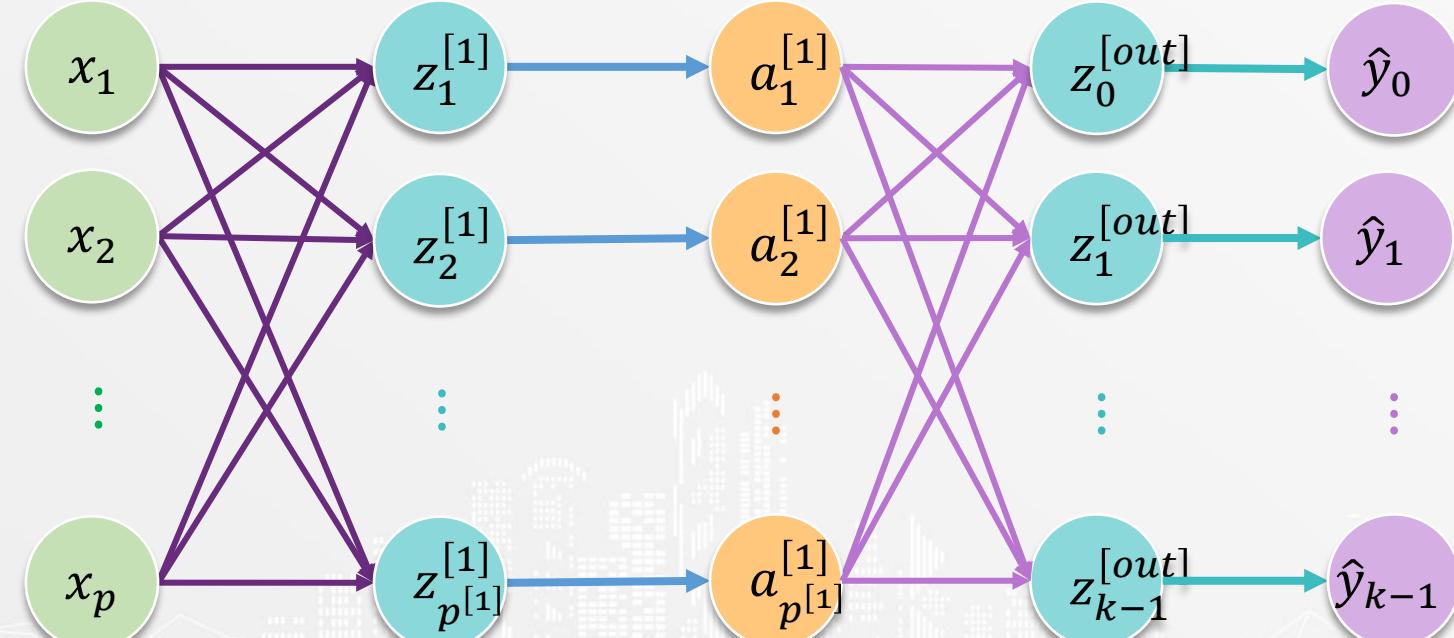
# Architecture of Neural Network

## Binary Classification



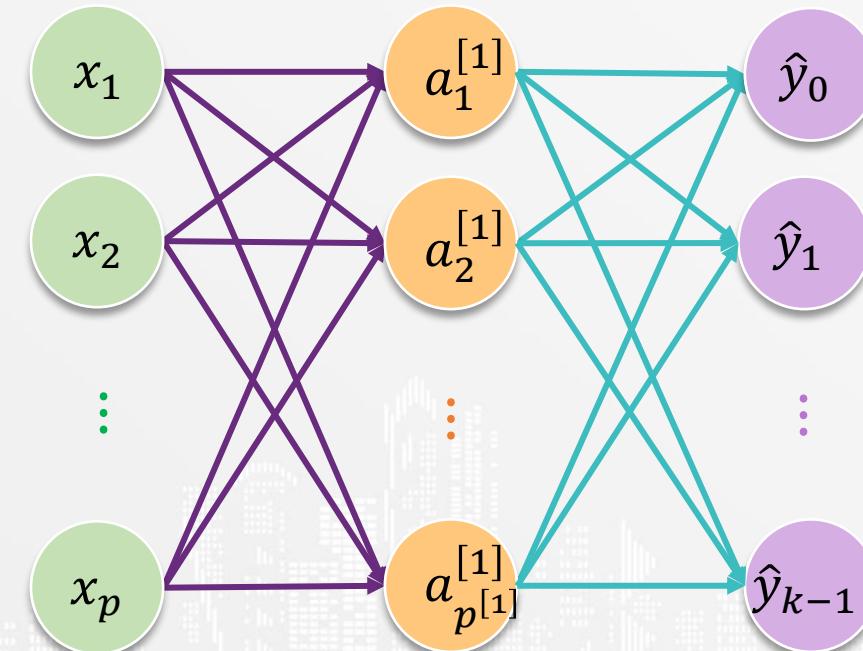
# Architecture of Neural Network

## Multi-Class Classification

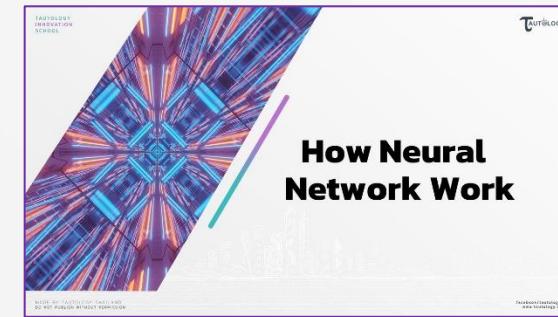
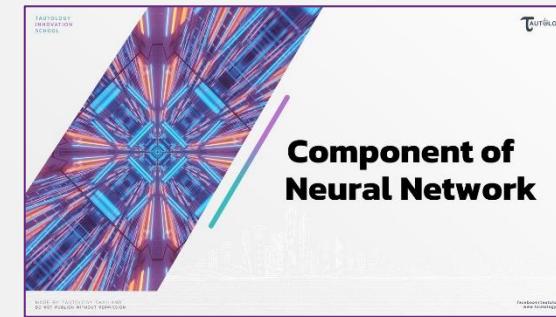
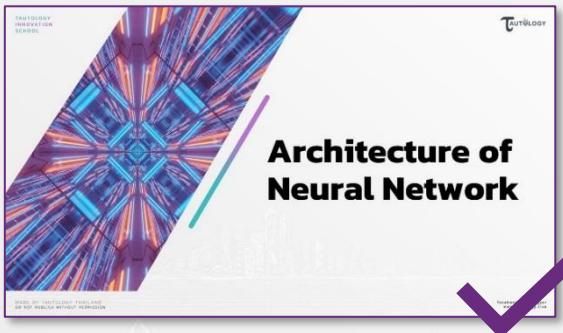
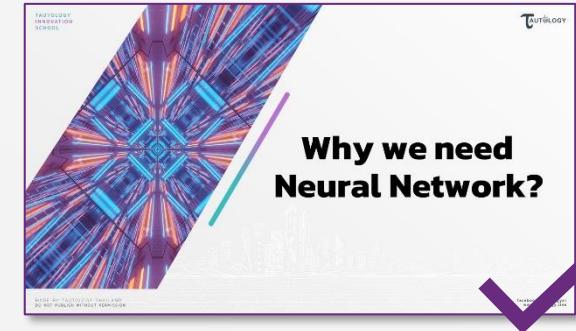
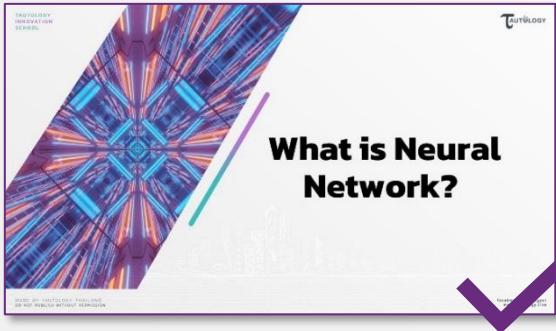


# Architecture of Neural Network

## Multi-Class Classification



# Neural Network





# Component of Neural Network



# Component of Neural Network

Hidden Node

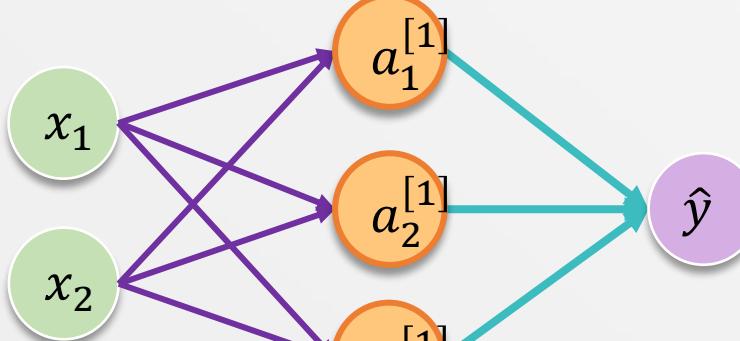
Hidden Layer

Weight & Bias

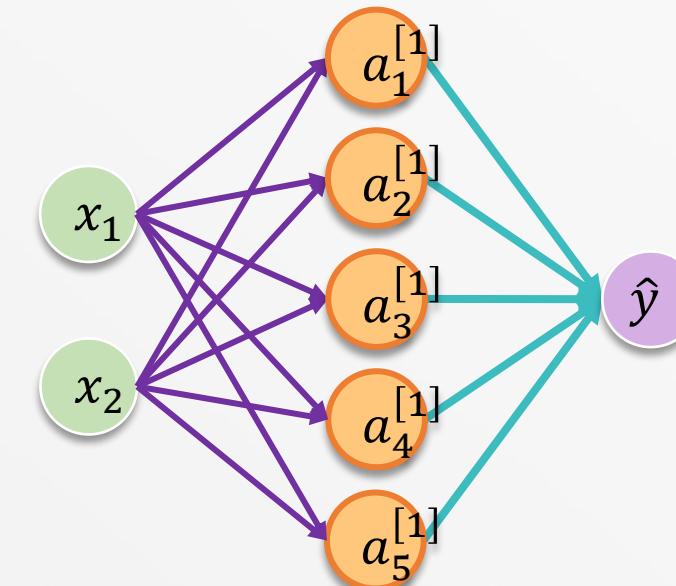
Activation  
Function

# Hidden Node

**Hidden Node** คือ จำนวนของ nonlinear function ที่ใช้



Hidden Node = 3



Hidden Node = 5

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



**Weight & Bias**

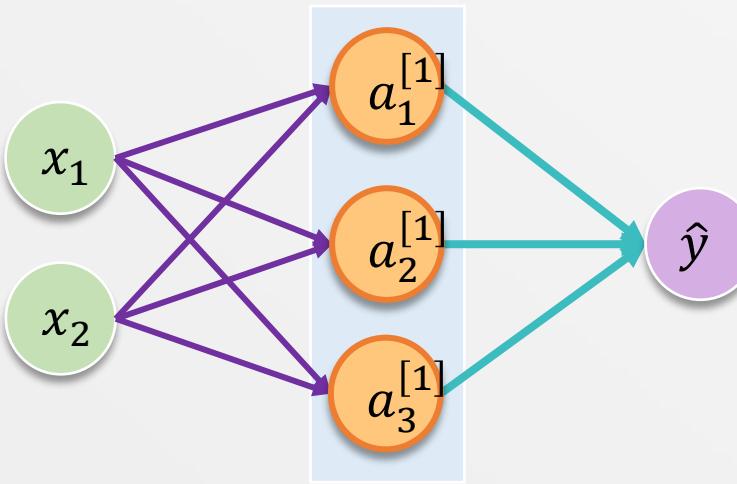


**Activation  
Function**

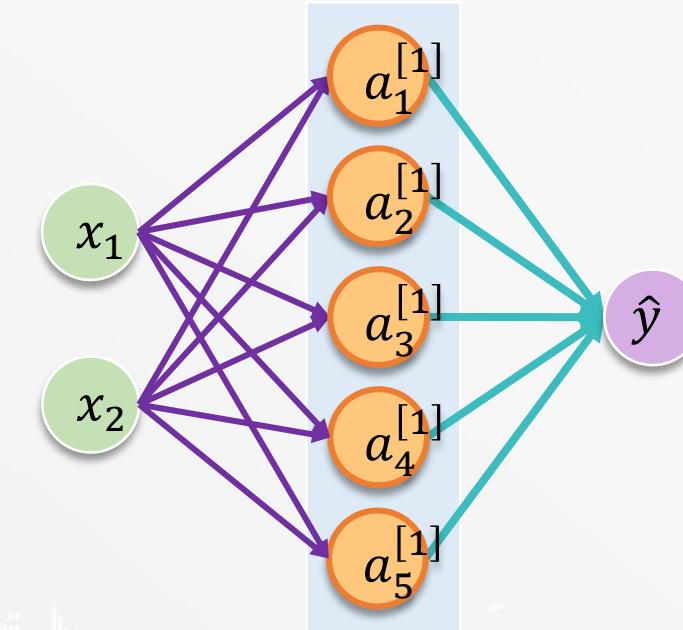


# Hidden Layer

**Hidden Layer** คือ ชั้นที่เก็บ hidden node



จำนวน Hidden Layer = 1  
(Hidden Node = 3)



จำนวน Hidden Layer = 1  
(Hidden Node = 5)

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



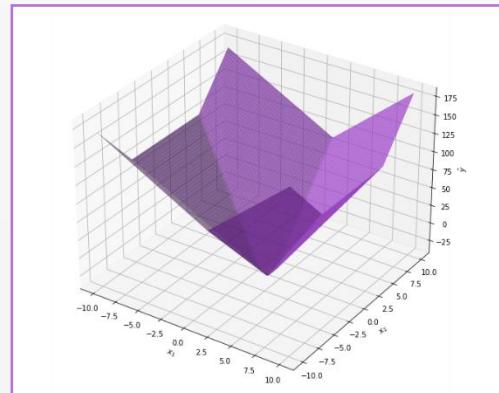
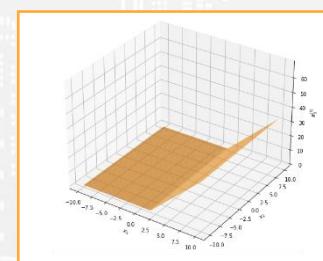
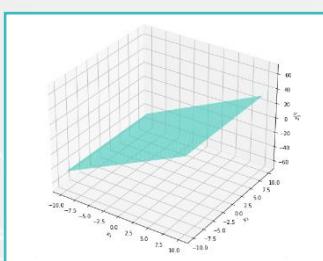
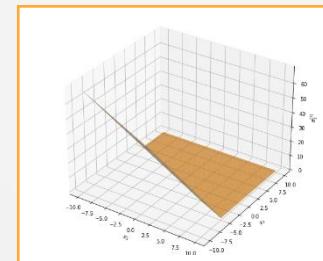
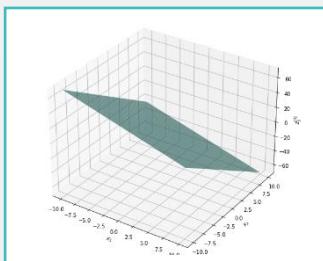
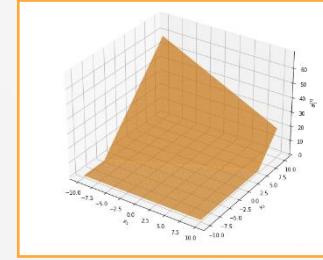
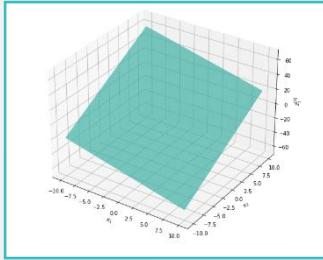
**Weight & Bias**



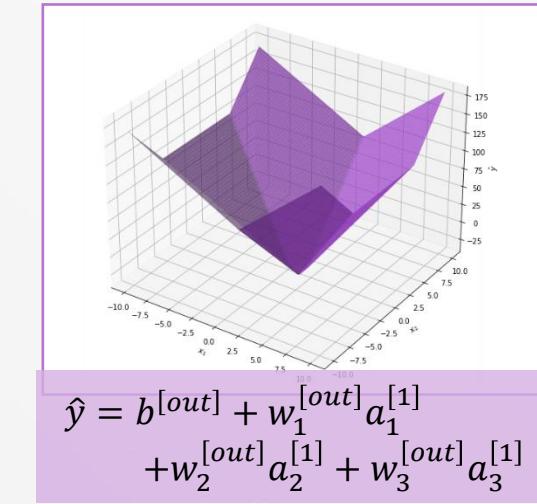
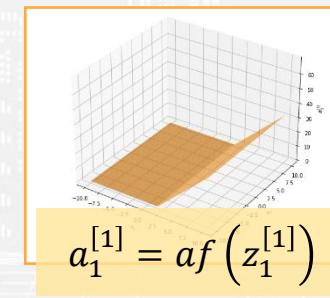
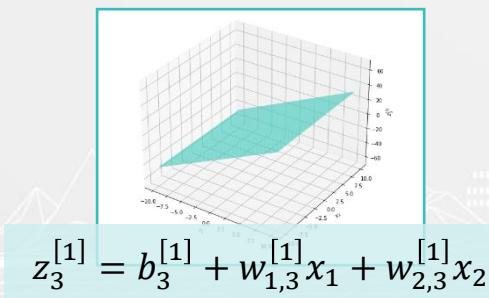
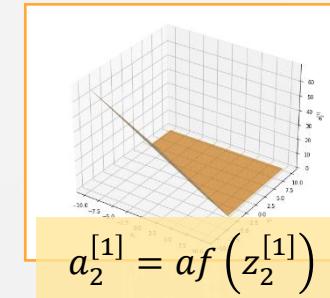
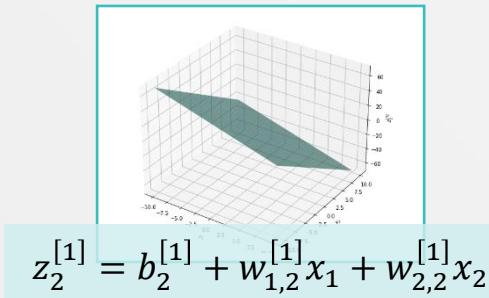
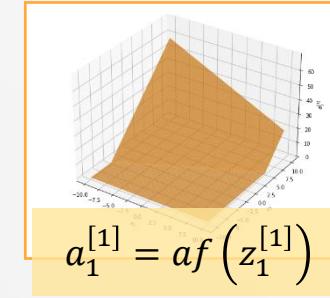
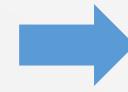
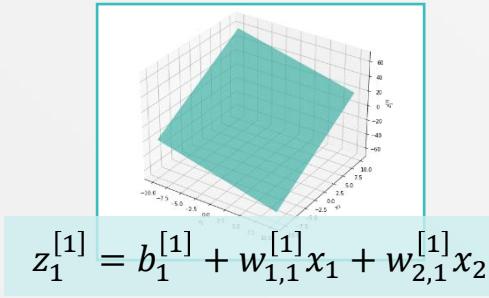
**Activation  
Function**



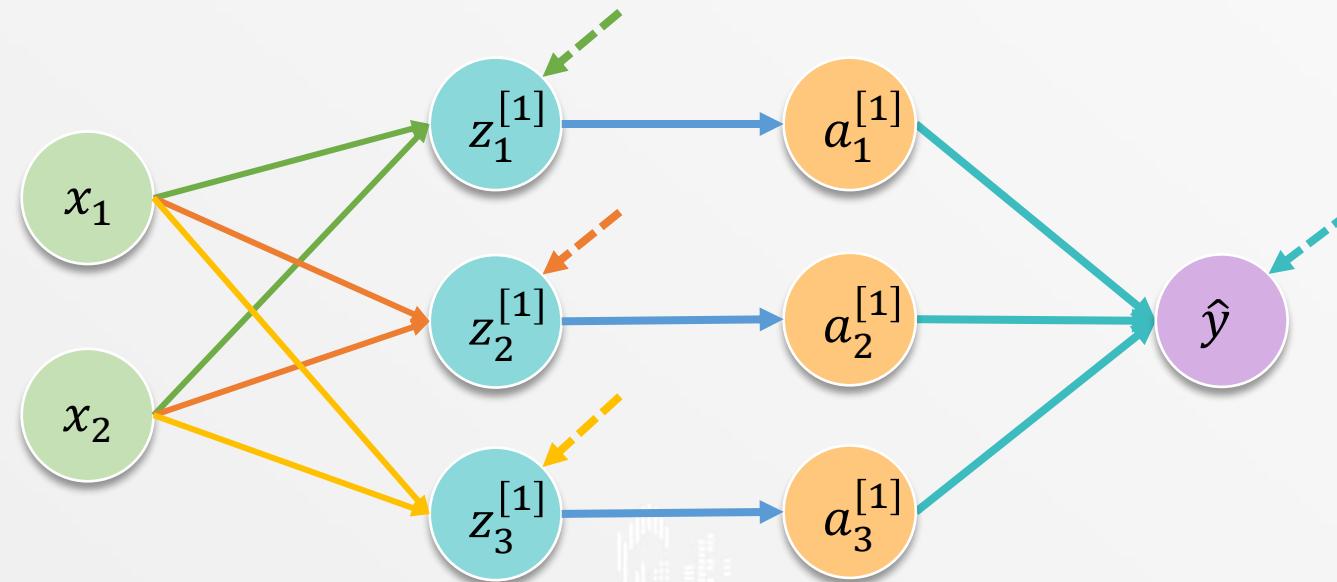
# Weight & Bias



# Weight & Bias



# Weight & Bias

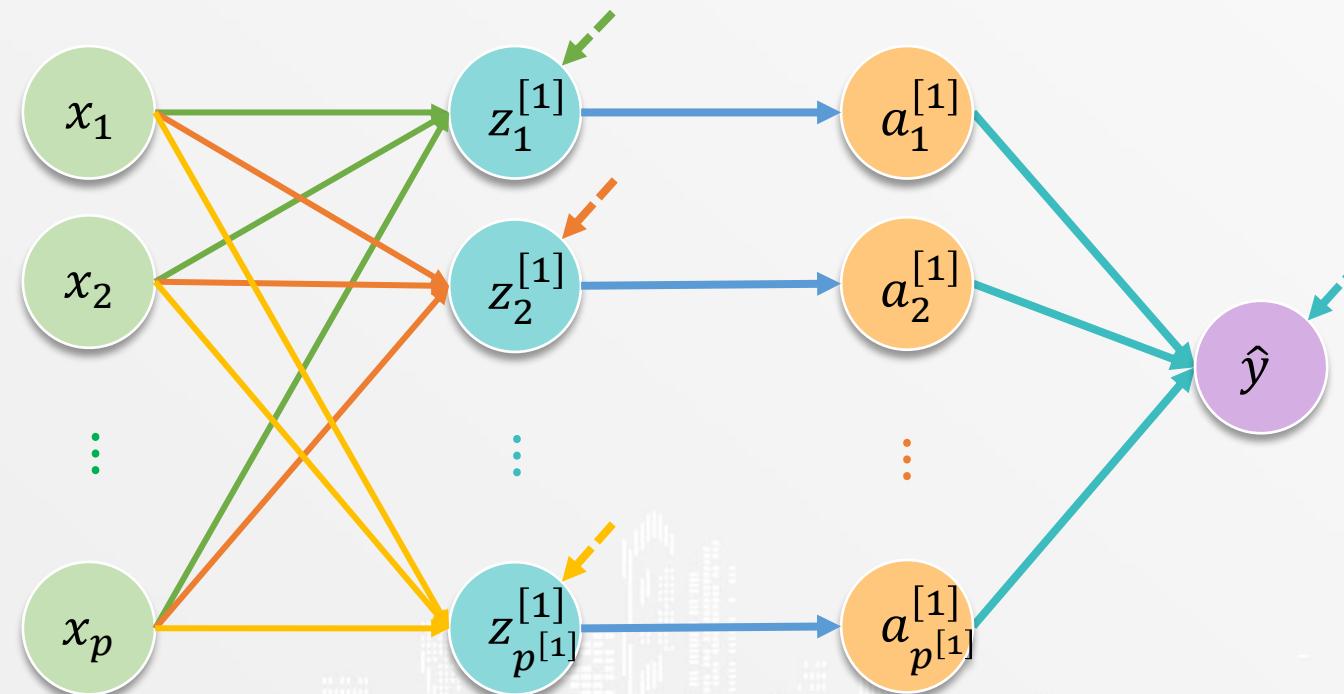


# Weight & Bias

$$\mathbf{b}^{[1]} = [b_1^{[1]} \quad b_2^{[1]} \quad b_3^{[1]}], \quad W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & w_{1,3}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & w_{2,3}^{[1]} \end{bmatrix}$$

$$\mathbf{b}^{[out]} = [b^{[out]}], \quad W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \\ w_3^{[out]} \end{bmatrix}$$

# Weight & Bias



# Weight & Bias

$$z_1^{[1]} = b_1^{[1]} + w_{1,1}^{[1]}x_1 + w_{2,1}^{[1]}x_2 + \cdots + w_{p,1}^{[1]}x_p$$

$$z_2^{[1]} = b_2^{[1]} + w_{1,2}^{[1]}x_1 + w_{2,2}^{[1]}x_2 + \cdots + w_{p,2}^{[1]}x_p$$

⋮

$$z_{p^{[1]}}^{[1]} = b_{p^{[1]}}^{[1]} + w_{1,p^{[1]}}^{[1]}x_1 + w_{2,p^{[1]}}^{[1]}x_2 + \cdots + w_{p,p^{[1]}}^{[1]}x_p$$

# Weight & Bias

$$a_1^{[1]} = af(z_1^{[1]})$$

$$a_2^{[1]} = af(z_2^{[1]})$$

⋮

$$a_{p^{[1]}}^{[1]} = af(z_{p^{[1]}}^{[1]})$$

$$\hat{y} = b^{[out]} + w_1^{[out]} a_1^{[1]} + w_2^{[out]} a_2^{[1]} + \dots + w_{p^{[1]}}^{[out]} a_{p^{[1]}}^{[1]}$$

# Weight & Bias

$$\mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & \dots & b_{p^{[1]}}^{[1]} \end{bmatrix}, \quad W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & \dots & w_{1,p^{[1]}}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & \dots & w_{2,p^{[1]}}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[1]} & w_{p,2}^{[1]} & \dots & w_{p,p^{[1]}}^{[1]} \end{bmatrix}$$

$$\mathbf{b}^{[out]} = [b^{[out]}], \quad W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \\ \vdots \\ w_{p^{[1]}}^{[out]} \end{bmatrix}$$

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



**Weight & Bias**

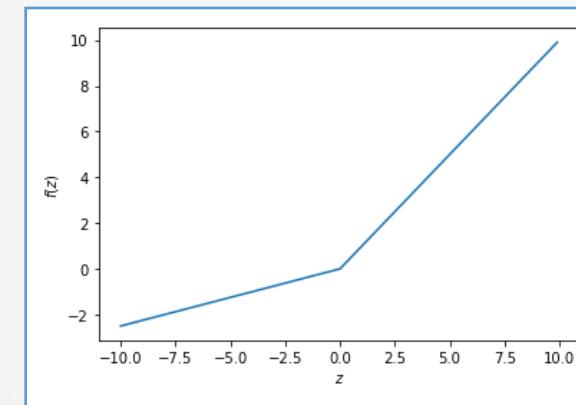
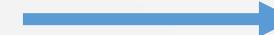
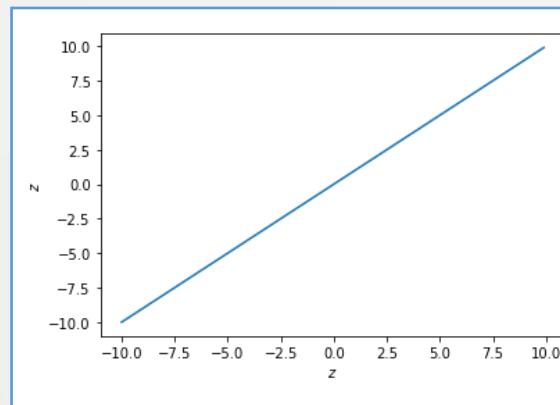


**Activation  
Function**



# Activation Function

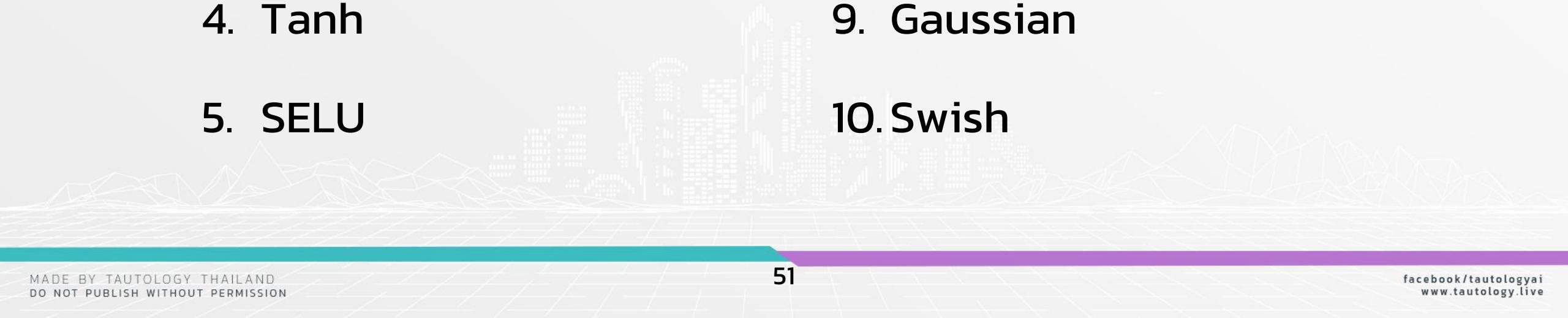
**Activation Function** คือ function ที่ส่งค่า linear function ไปเป็น nonlinear function



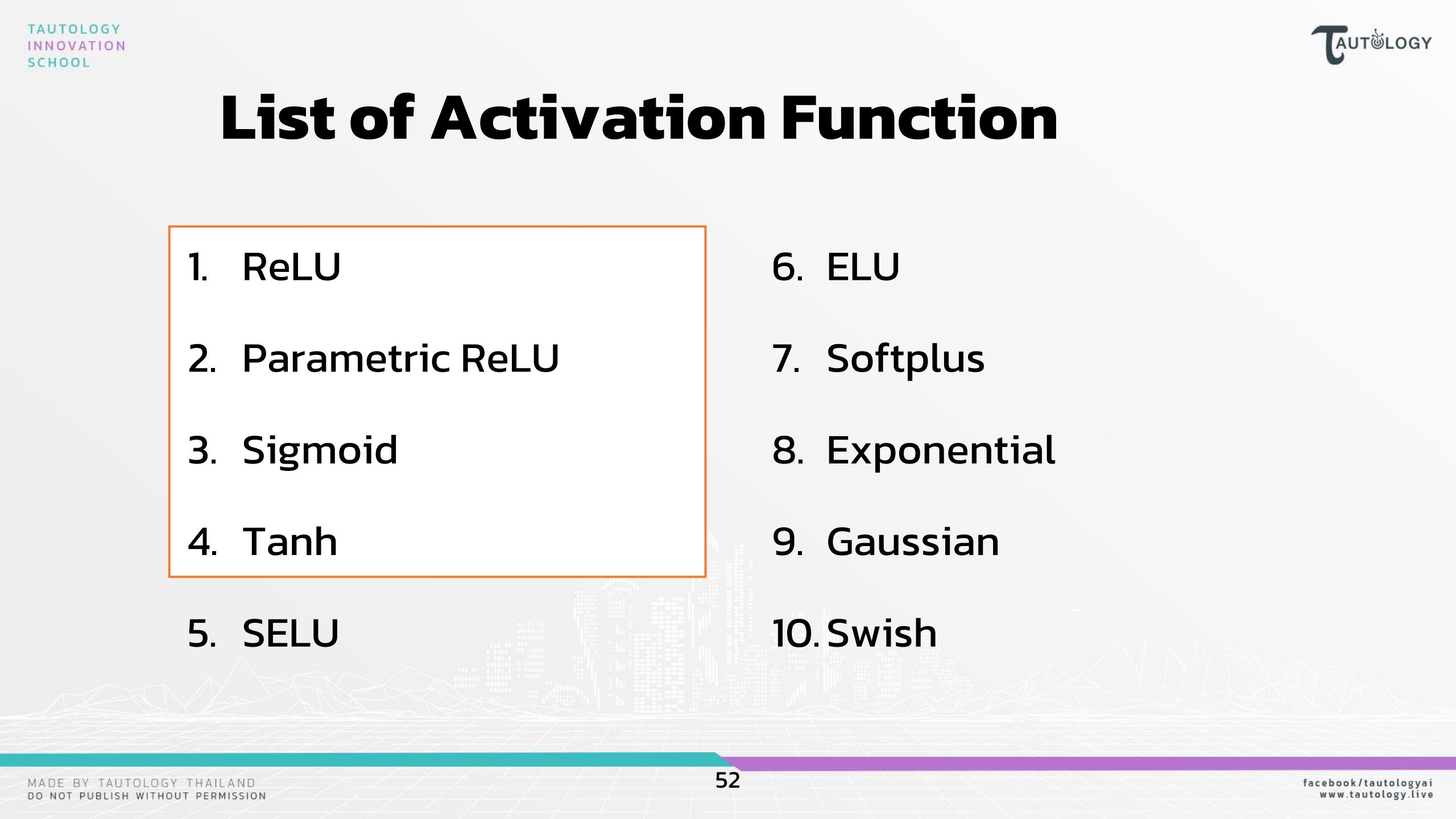
# Activation Function

- List of Activation Function
- Comparison of Activation Function
- What if no Activation Function

# List of Activation Function

- 
- 1. ReLU
  - 2. Parametric ReLU
  - 3. Sigmoid
  - 4. Tanh
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

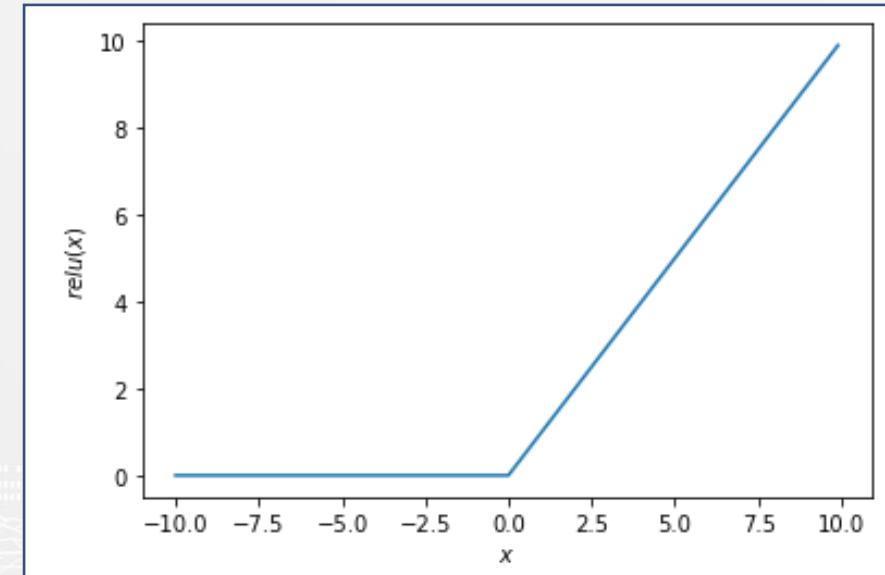
# List of Activation Function

- 
- 1. ReLU
  - 2. Parametric ReLU
  - 3. Sigmoid
  - 4. Tanh
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

# List of Activation Function

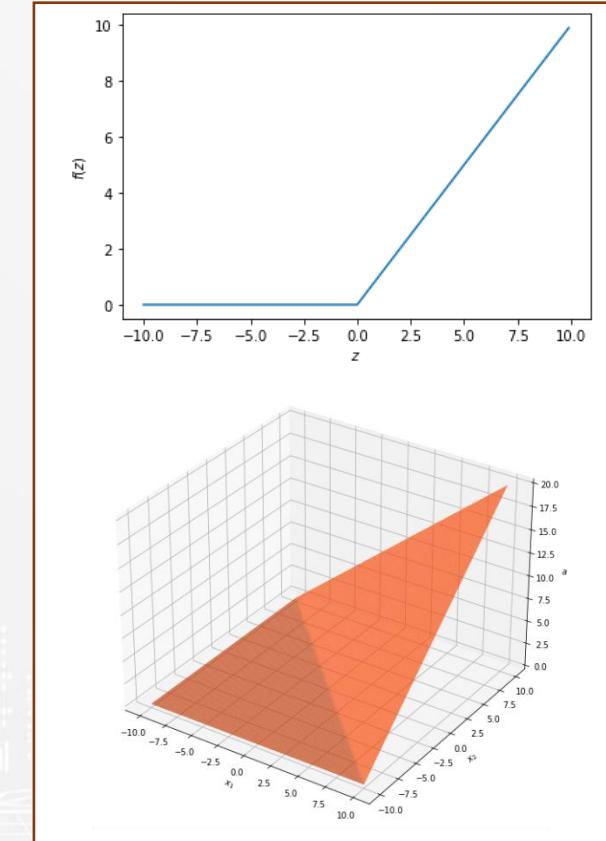
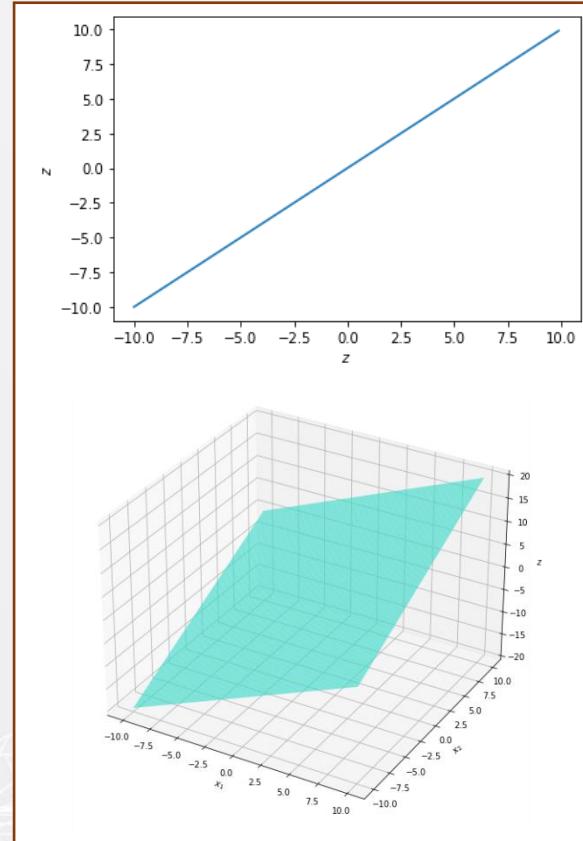
1. **ReLU (Rectified Linear Unit)** – สร้างจุดหักโดยใช้จุด 0 เป็นตัวกำหนด

$$\text{ReLU}(z) = \begin{cases} z & ; \quad z > 0 \\ 0 & ; \quad z \leq 0 \end{cases}$$



# List of Activation Function

## 1. ReLU (Rectified Linear Unit)



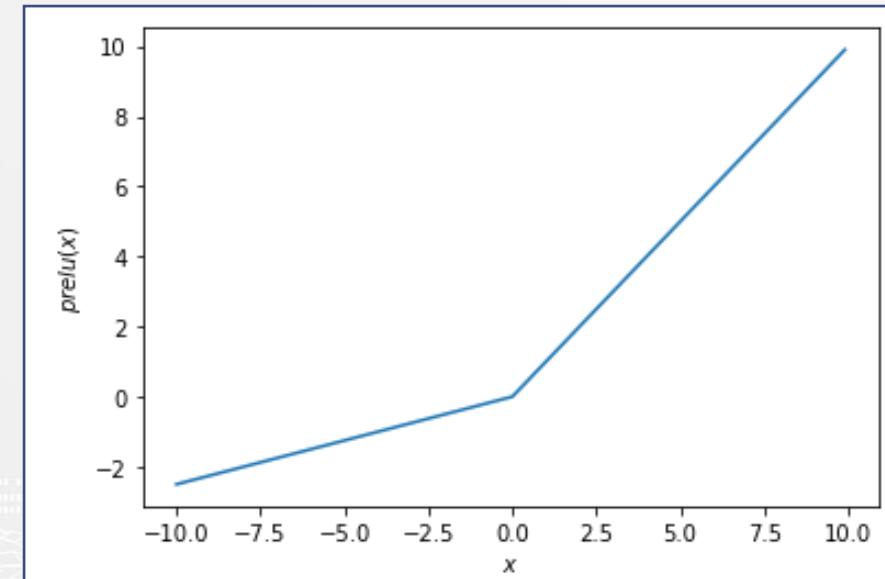
# List of Activation Function

1. ReLU ✓
2. Parametric ReLU
3. Sigmoid
4. Tanh
5. SELU
6. ELU
7. Softplus
8. Exponential
9. Gaussian
10. Swish

# List of Activation Function

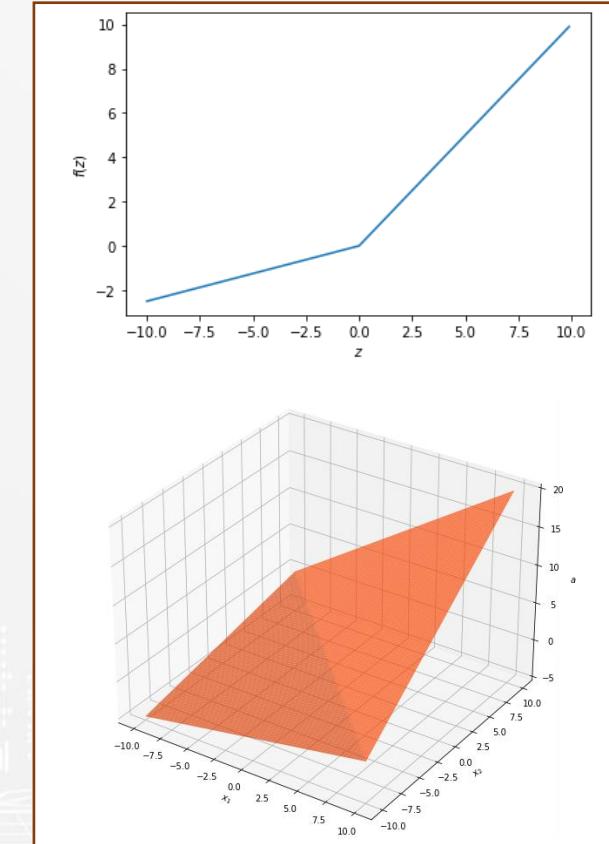
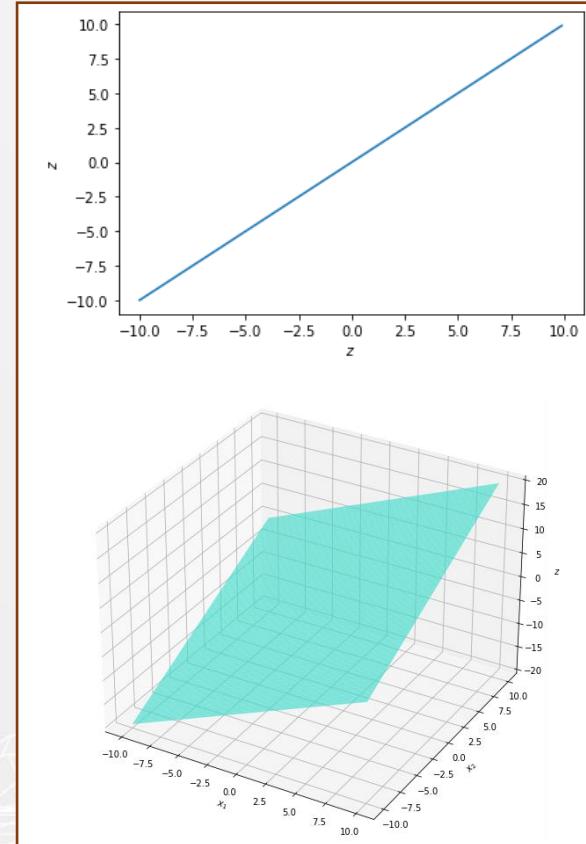
2. **Parametric ReLU** – สร้างจุดหักโดยใช้จุด 0 เป็นตัวกำหนด

$$PReLU(z) = \begin{cases} z & ; \quad z > 0 \\ \alpha z & ; \quad z < 0, \alpha > 0 \end{cases}$$

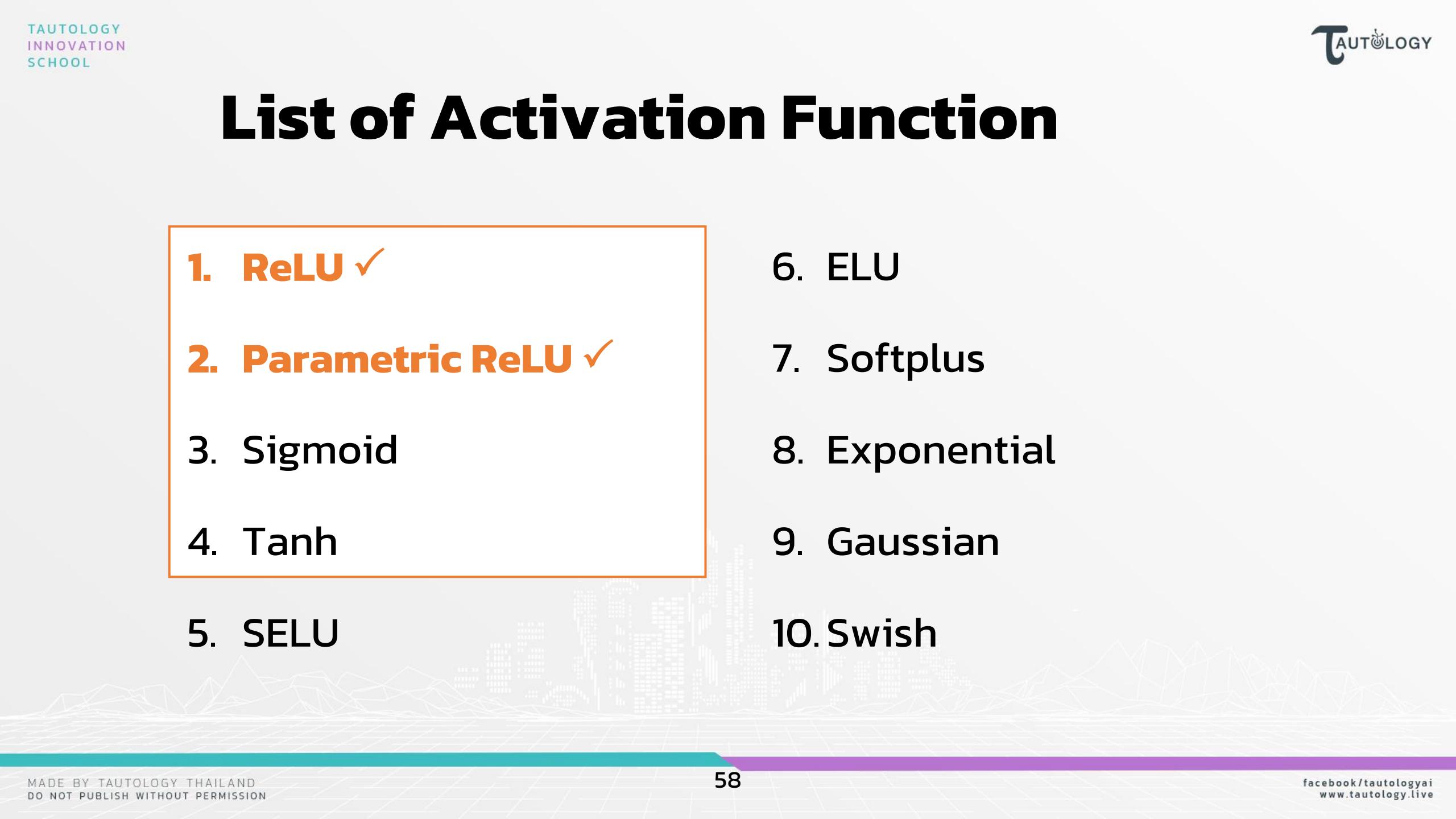


# List of Activation Function

## 2. Parametric ReLU



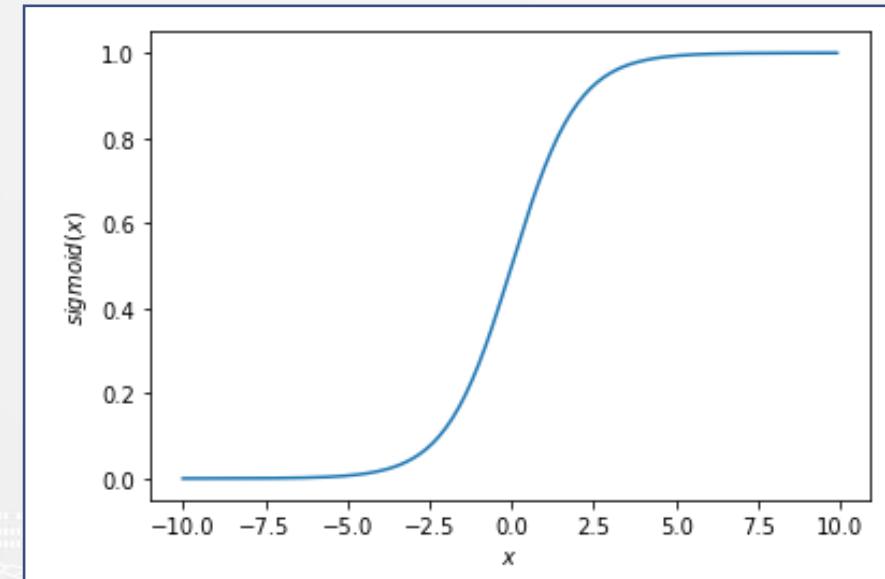
# List of Activation Function

- 
1. ReLU ✓
  2. Parametric ReLU ✓
  3. Sigmoid
  4. Tanh
  5. SELU
  6. ELU
  7. Softplus
  8. Exponential
  9. Gaussian
  10. Swish

# List of Activation Function

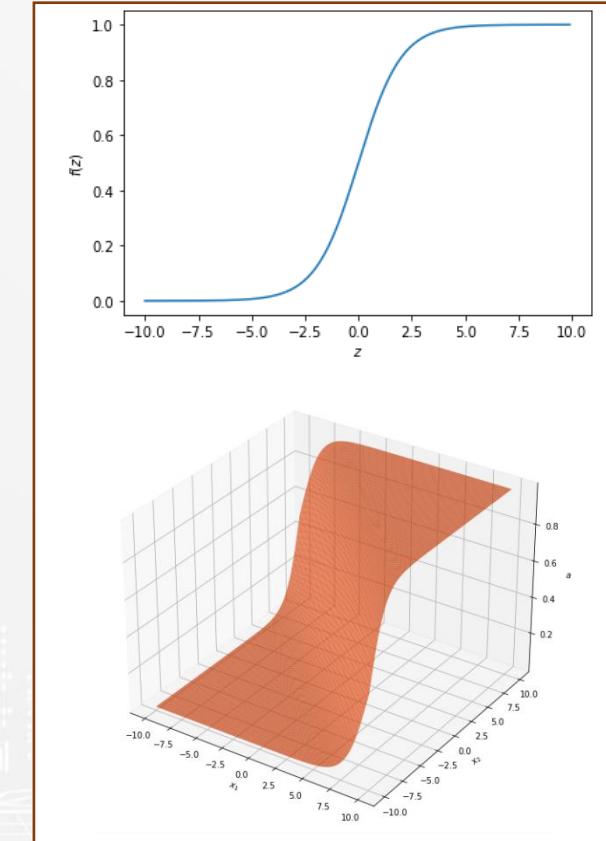
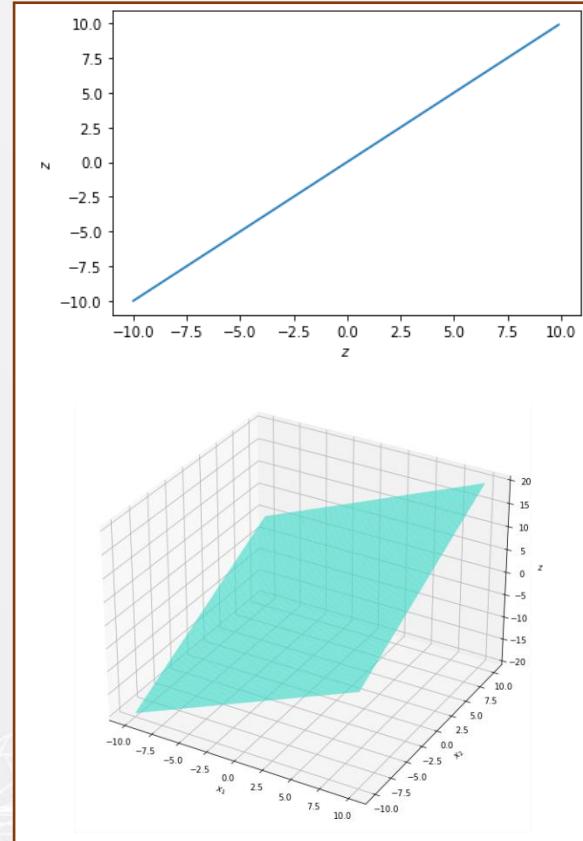
## 3. Sigmoid

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

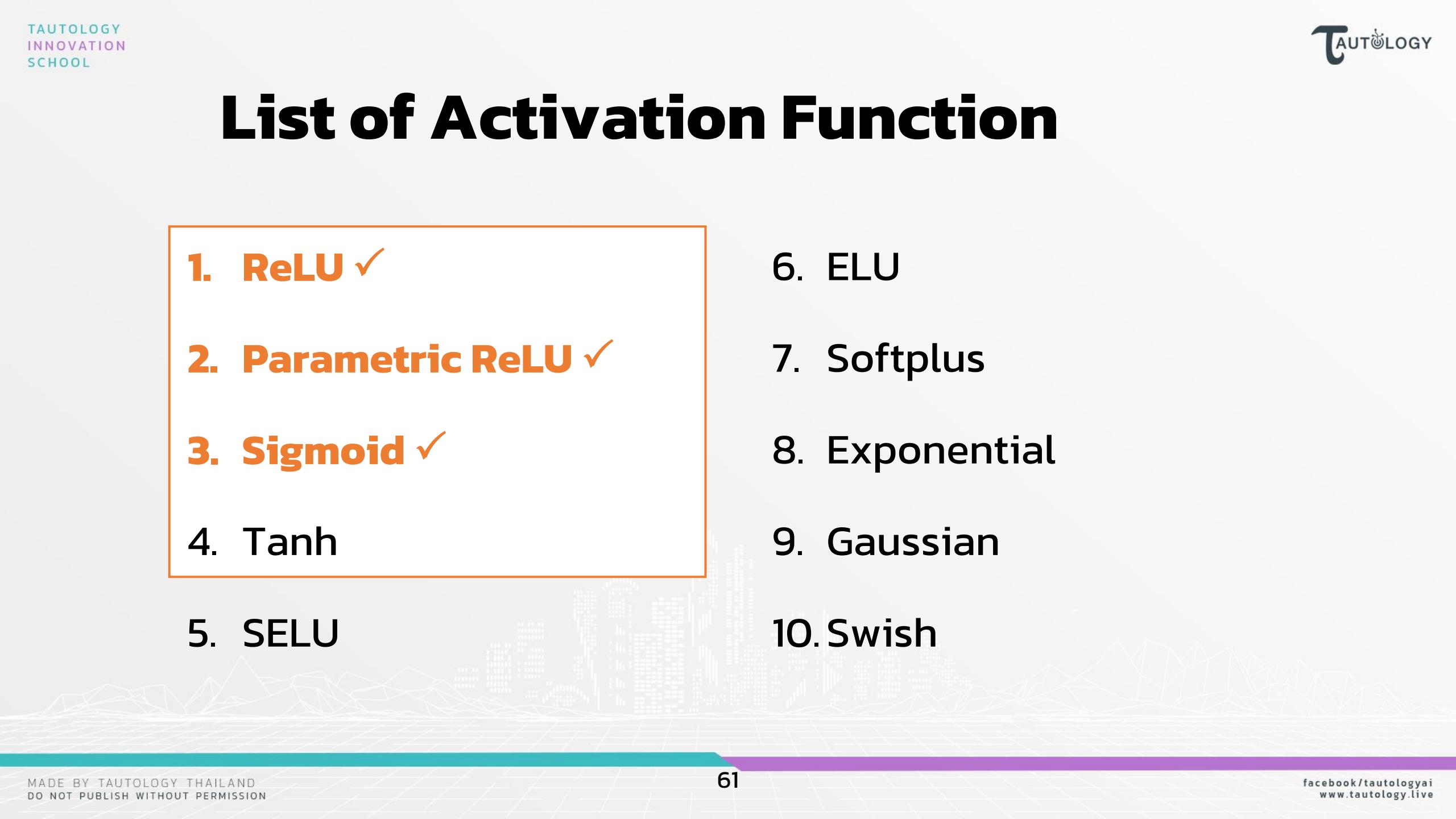


# List of Activation Function

## 3. Sigmoid



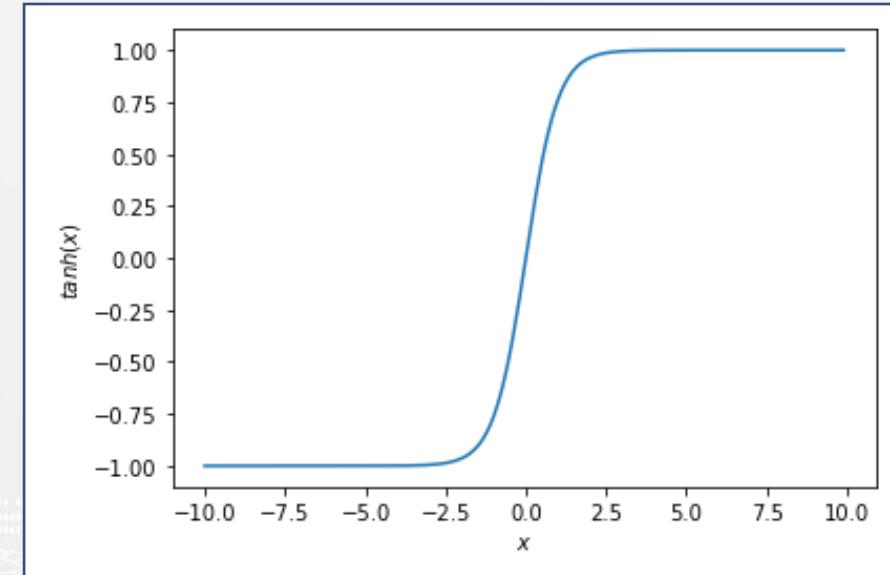
# List of Activation Function

- 
- 1. ReLU ✓
  - 2. Parametric ReLU ✓
  - 3. Sigmoid ✓
  - 4. Tanh
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

# List of Activation Function

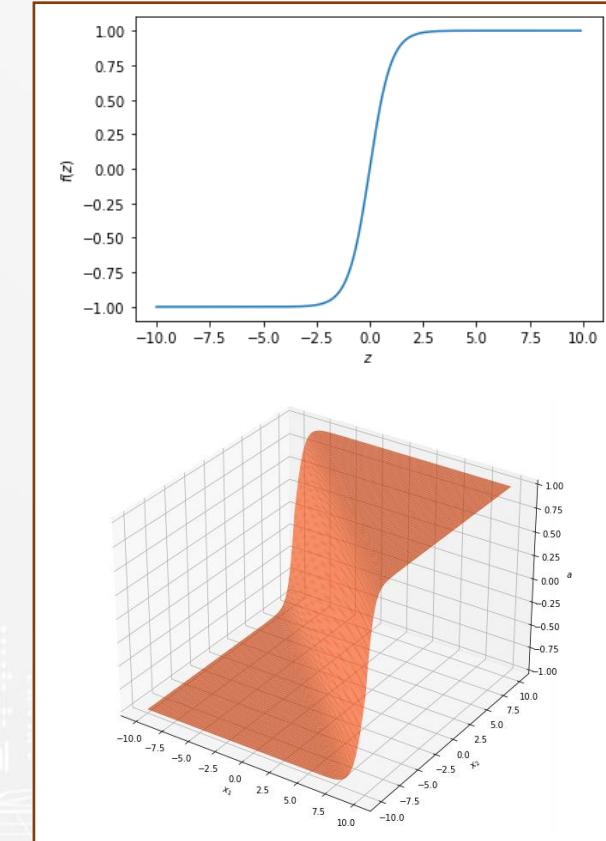
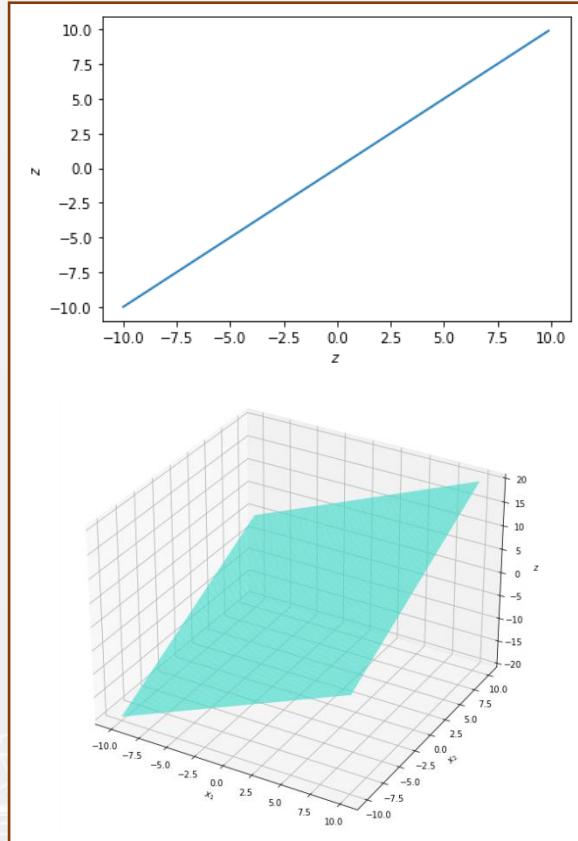
## 4. Tanh

$$\text{Tanh}(x) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

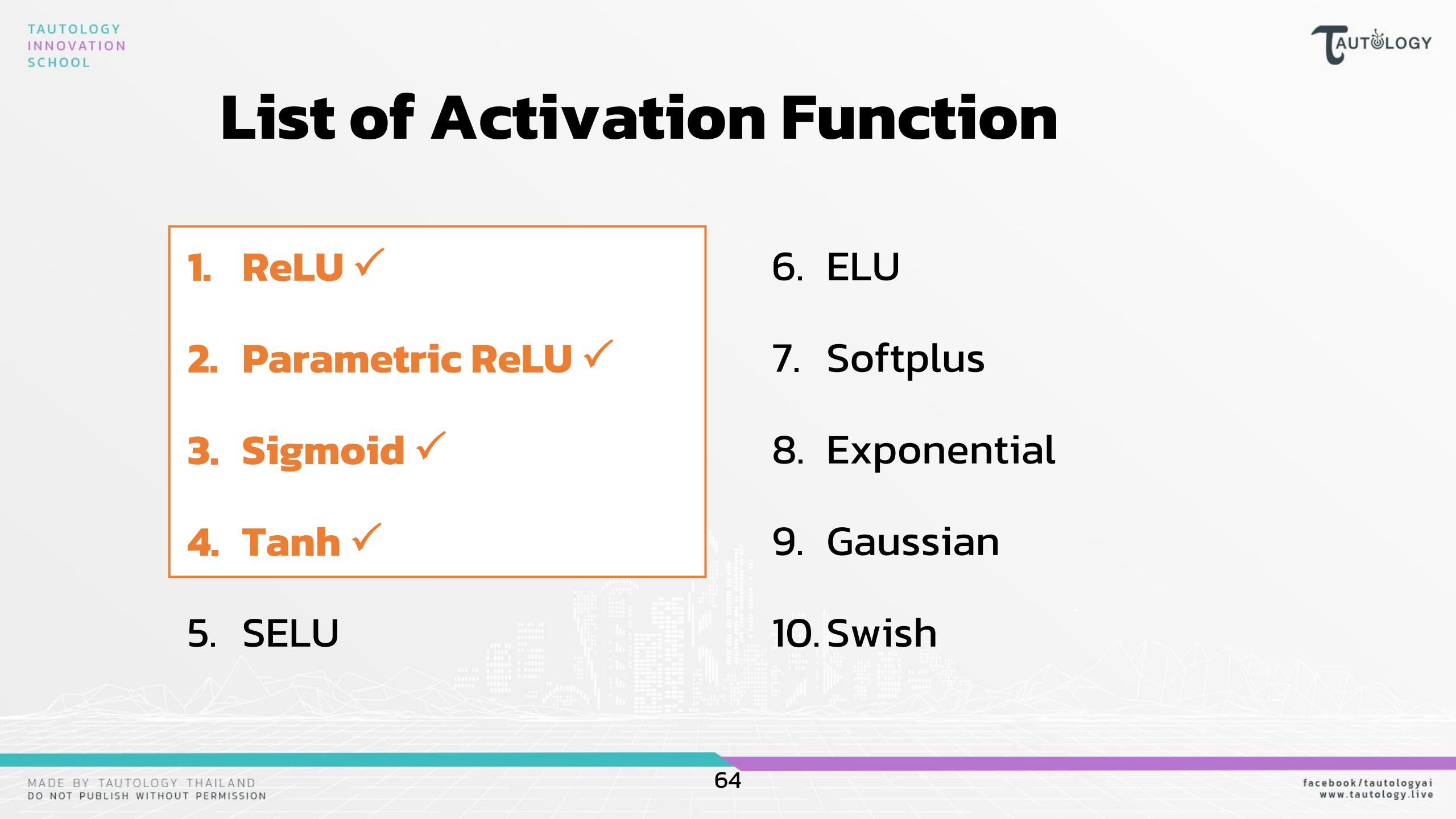


# List of Activation Function

## 4. Tanh

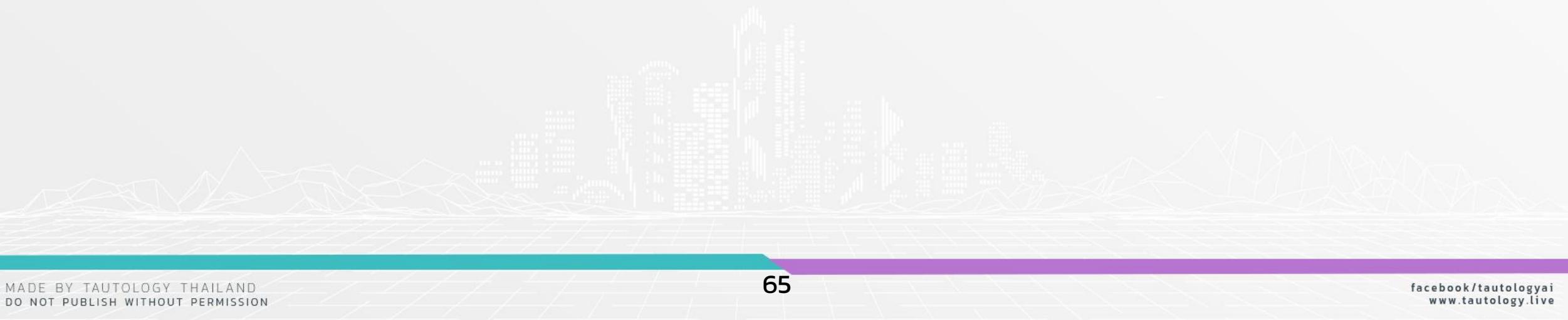


# List of Activation Function

- 
- 1. ReLU ✓
  - 2. Parametric ReLU ✓
  - 3. Sigmoid ✓
  - 4. Tanh ✓
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

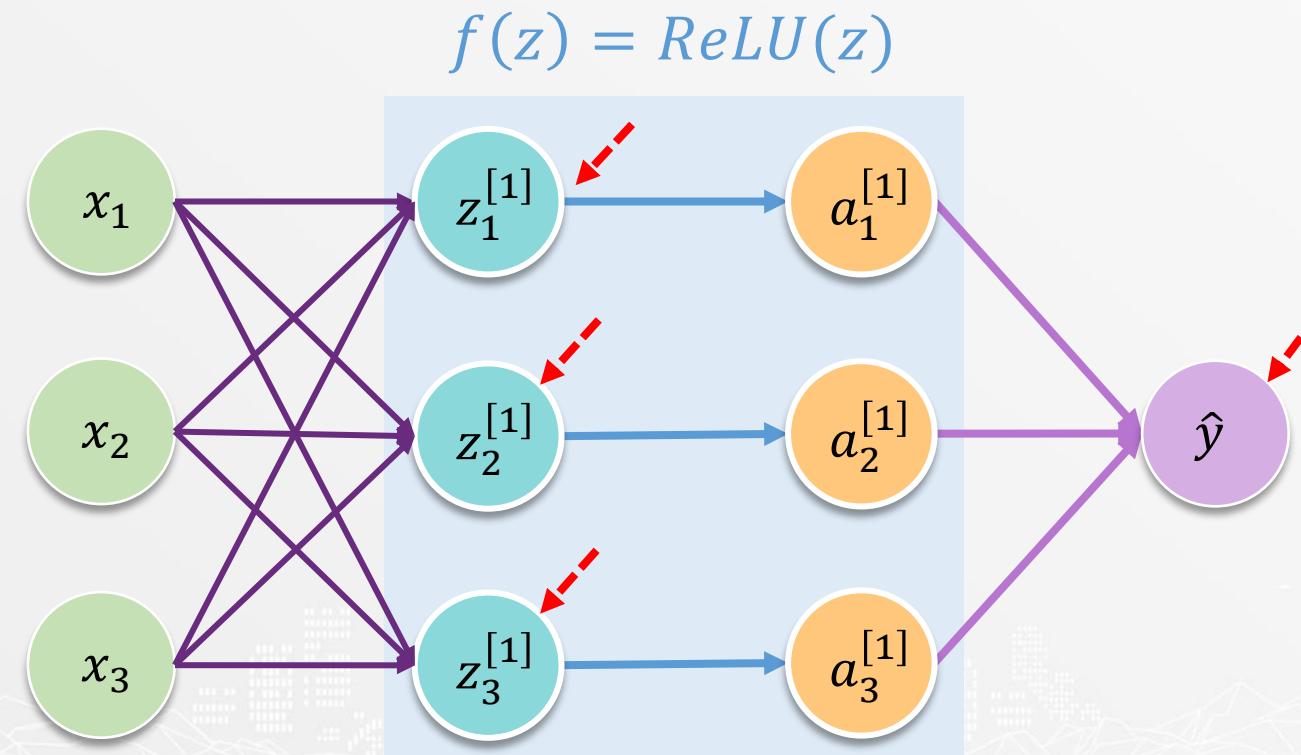
# Activation Function

- List of Activation Function**
- Comparison of Activation Function
- What if no Activation Function



# Comparison of Activation Function

## 1. ReLU (Rectified Linear Unit)



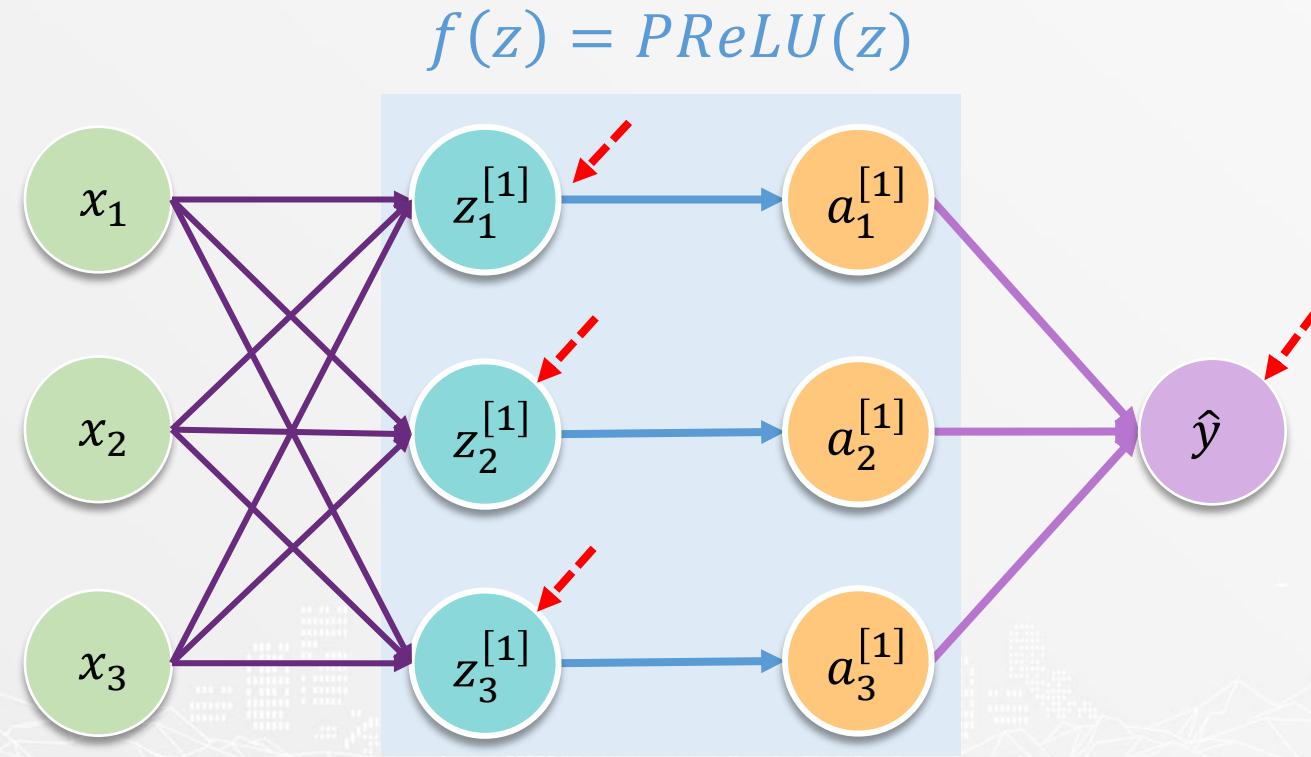
# Comparison of Activation Function

## 1. ReLU (Rectified Linear Unit)



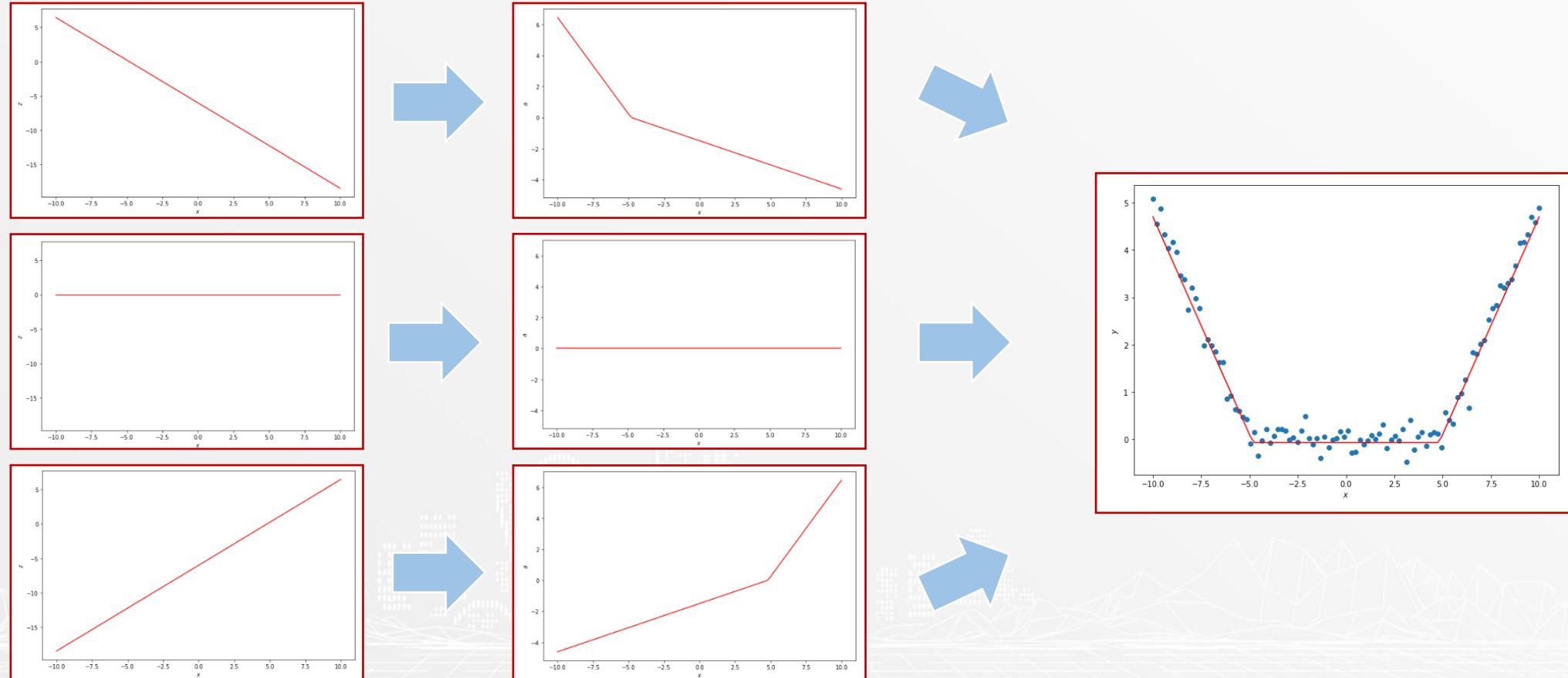
# Comparison of Activation Function

## 2. Parametric ReLU



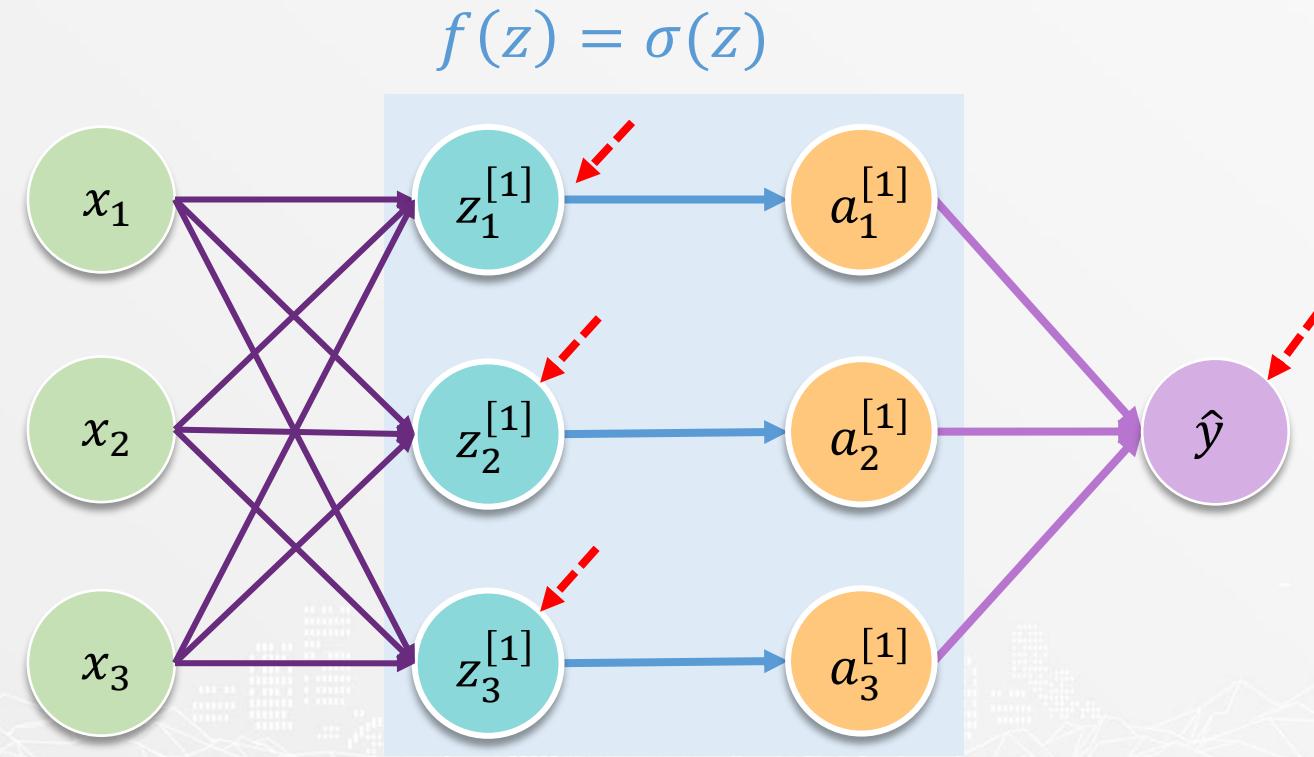
# Comparison of Activation Function

## 2. Parametric ReLU



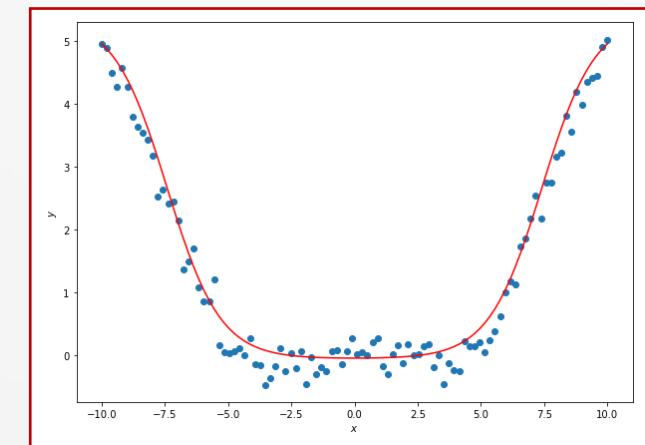
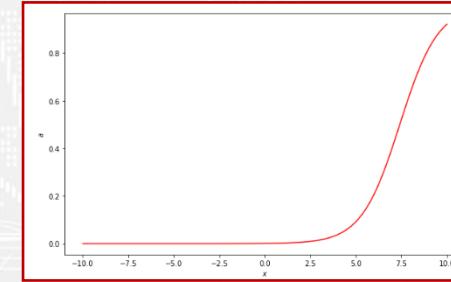
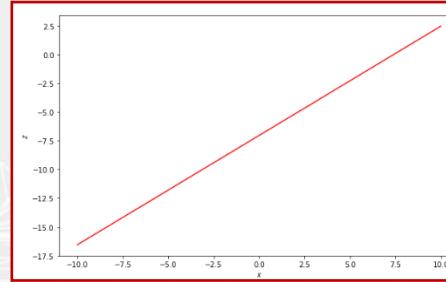
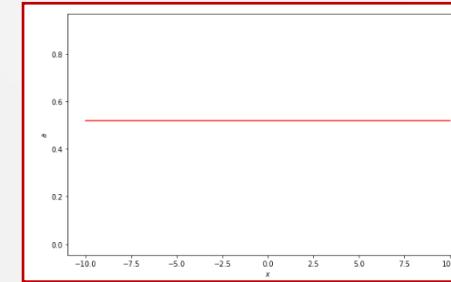
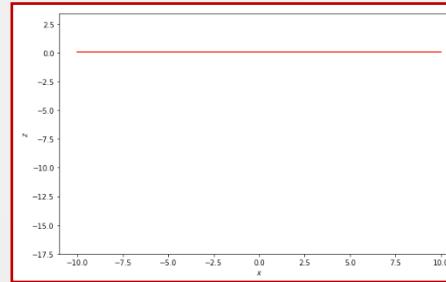
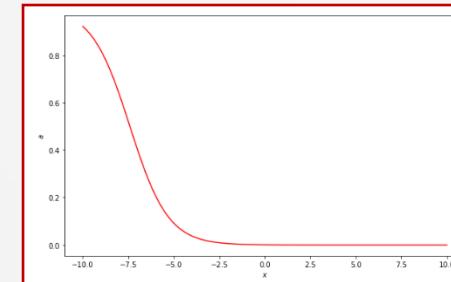
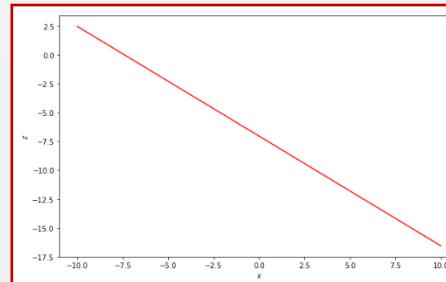
# Comparison of Activation Function

## 3. Sigmoid



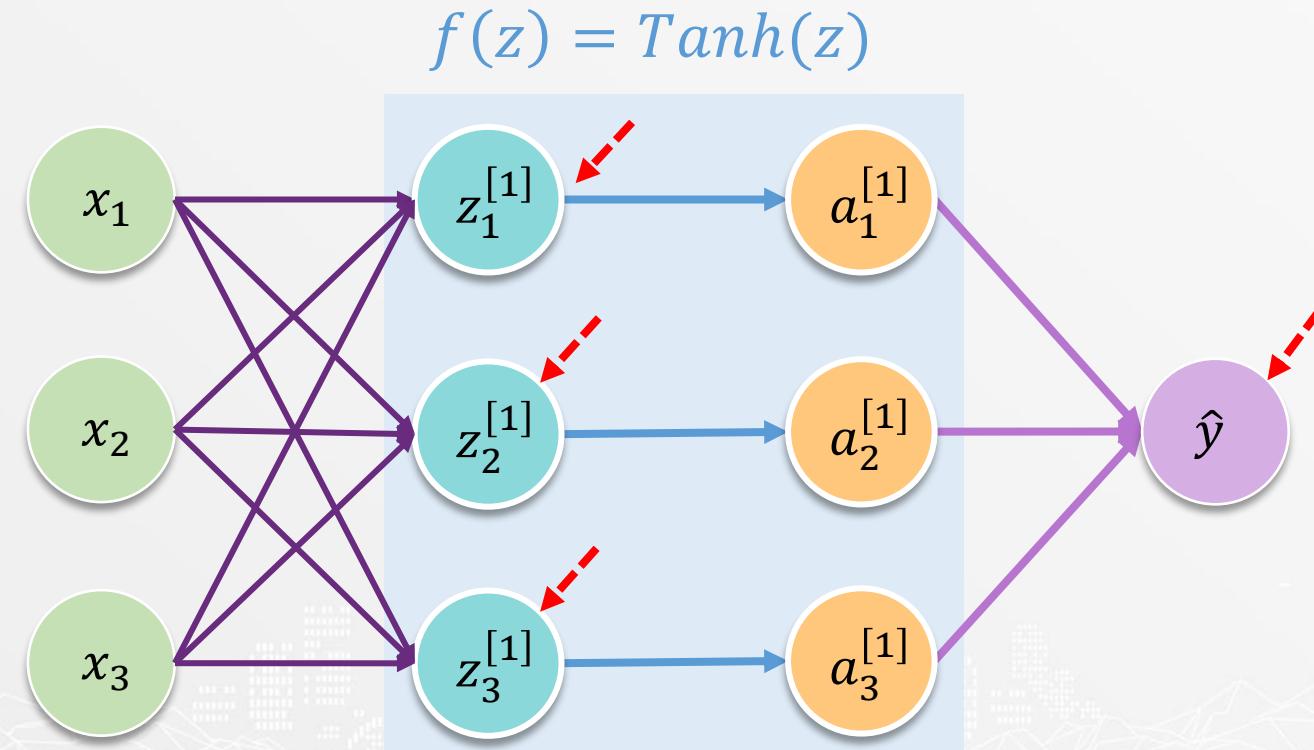
# Comparison of Activation Function

## 3. Sigmoid



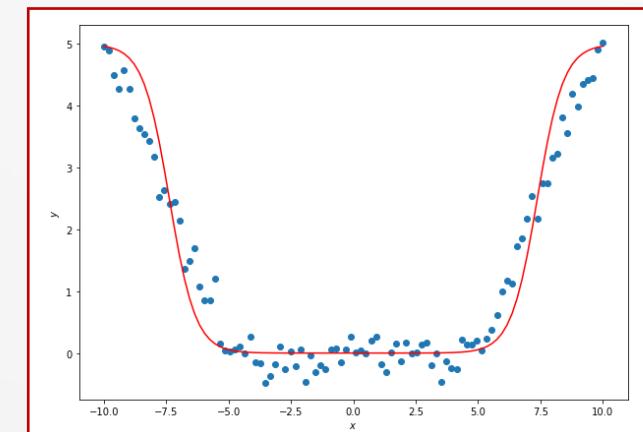
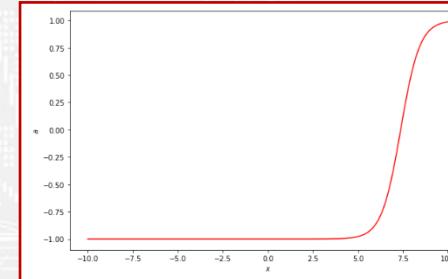
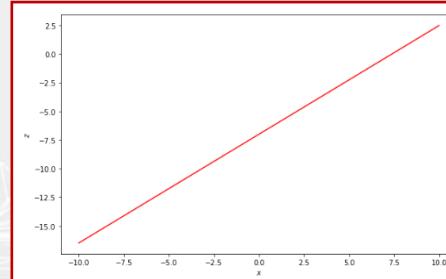
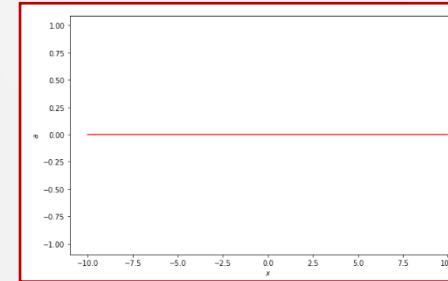
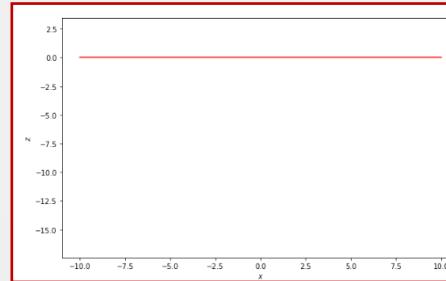
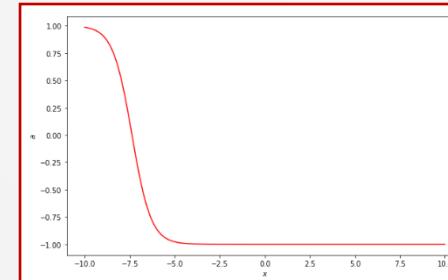
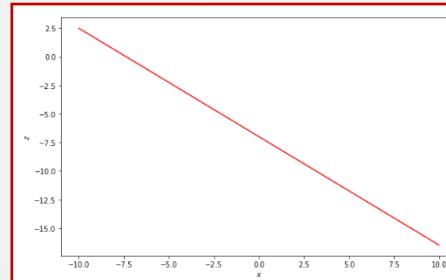
# Comparison of Activation Function

## 4. Tanh



# Comparison of Activation Function

## 4. Tanh



# Comparison of Activation Function

Function	Range	บริเวณที่เกิดการเปลี่ยนแปลงความชัน	Cost per epoch	จำนวน epoch ที่ใช้ในการลู่เข้า
ReLU	$[0, \infty)$	เกิดรอยหัก	น้อยที่สุด	มากกว่า PReLU
PReLU	$(-\infty, \infty)$	เกิดรอยหัก	มากกว่า ReLU	น้อยกว่า ReLU
Sigmoid	$(0, 1)$	smooth	มากกว่า ReLU, PReLU	มากกว่า Tanh
Tanh	$(-1, 1)$	smooth	มากที่สุด	น้อยกว่า Sigmoid

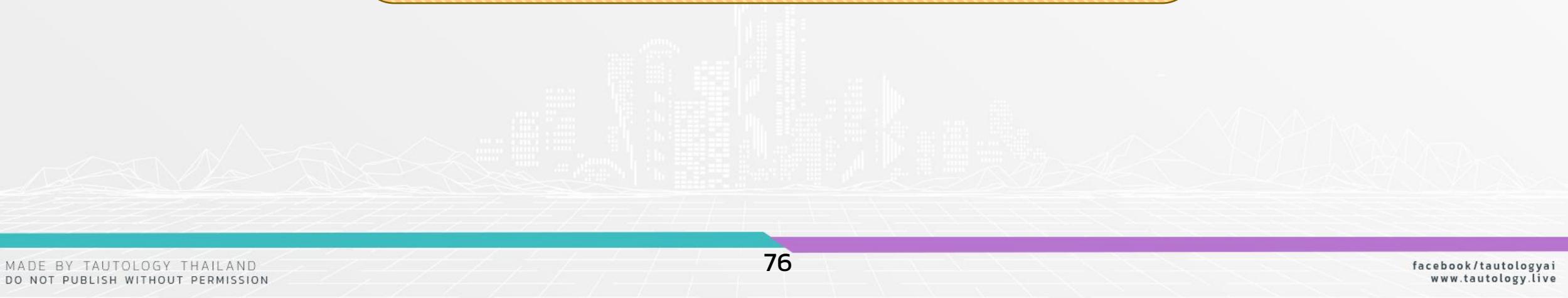
# Activation Function

- List of Activation Function**
- Comparison of Activation Function**
- What if no Activation Function**



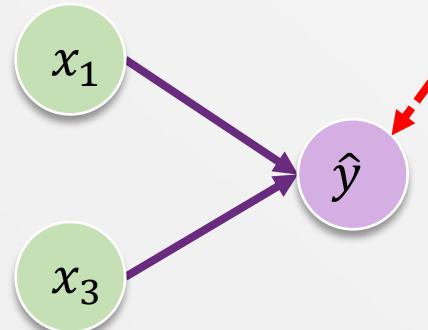
# What if no Activation Function

แล้วจะเกิดอะไรขึ้น ถ้าหากเราเพิ่ม hidden layer  
เพียงอย่างเดียว โดยที่ไม่ใส่ activation function



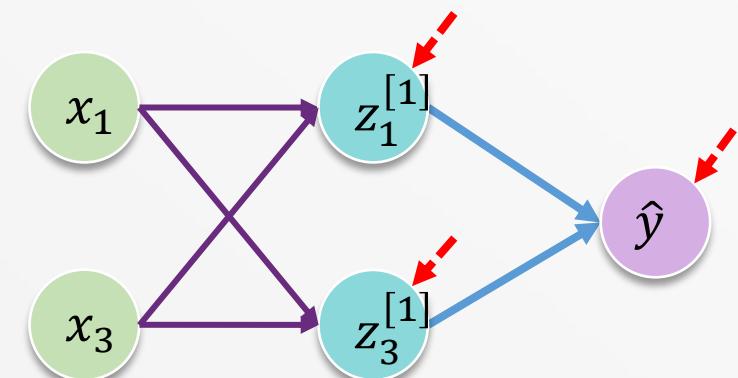
# What if no Activation Function

Linear Regression



$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2$$

Neural Network  
Regression



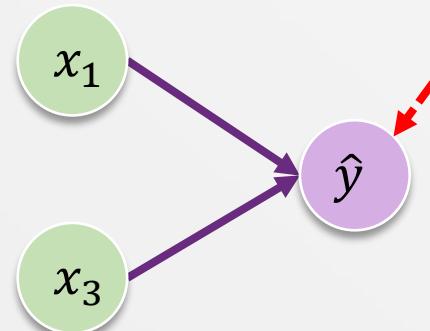
$$\hat{y} = b^{[out]} + w_1^{[out]} z_1 + w_2^{[out]} z_2 \quad (1)$$

$$z_1 = b_1^{[1]} + w_{1,1}^{[1]} x_1 + w_{2,1}^{[1]} x_2 \quad (2)$$

$$z_2 = b_2^{[1]} + w_{1,2}^{[1]} x_1 + w_{2,2}^{[1]} x_2 \quad (3)$$

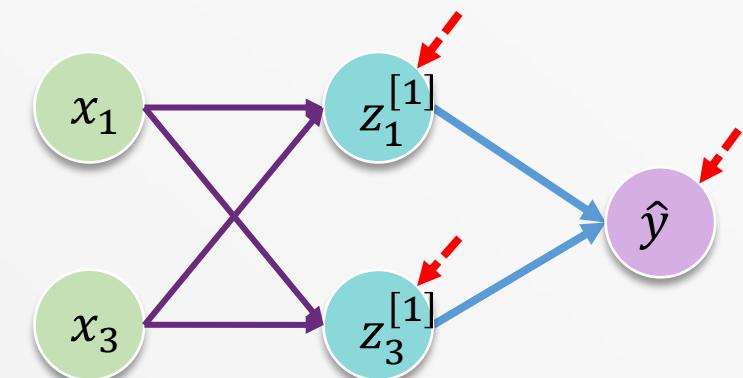
# What if no Activation Function

Linear Regression



$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2$$

Neural Network  
Regression



แทนค่า (2), (3) ลง (1)

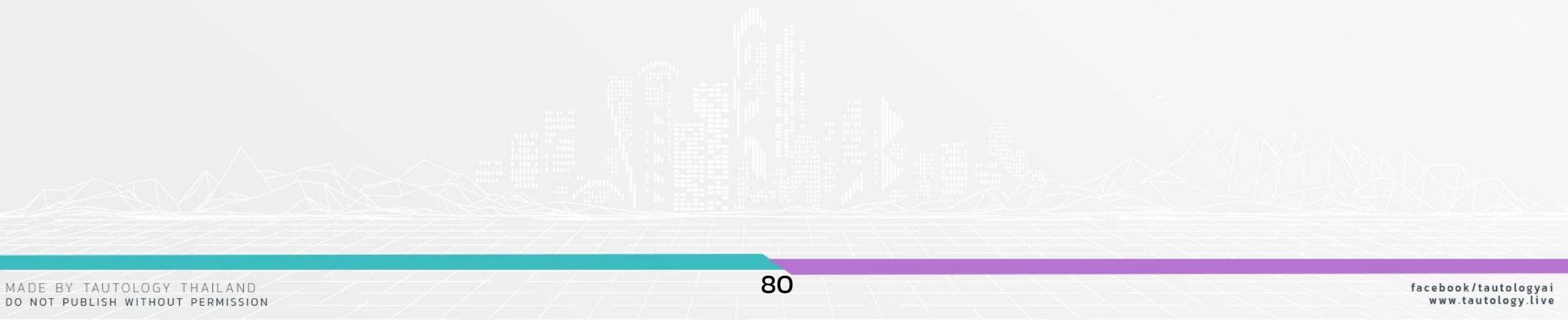
$$\begin{aligned}\hat{y} &= b^{[out]} + w_1^{[out]}(2) + w_2^{[out]}(3) \\ &\downarrow \\ \hat{y} &= c_0 + c_1 x_1 + c_2 x_2\end{aligned}$$

# What if no Activation Function

**“ Linear + Linear → Linear ”**

# Activation Function

- List of Activation Function**
- Comparison of Activation Function**
- What if no Activation Function**



# Component of Neural Network

**Hidden Node**



**Hidden Layer**



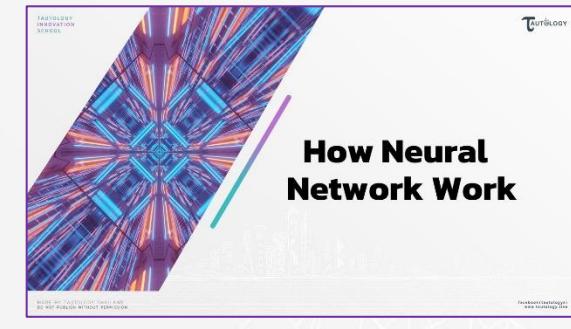
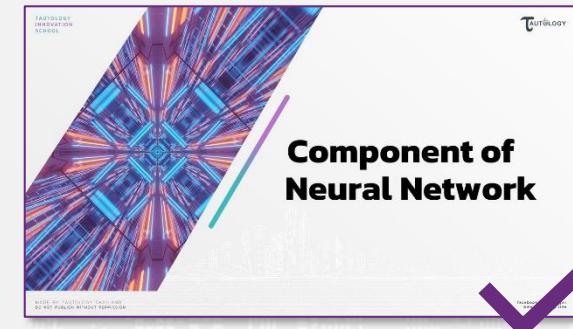
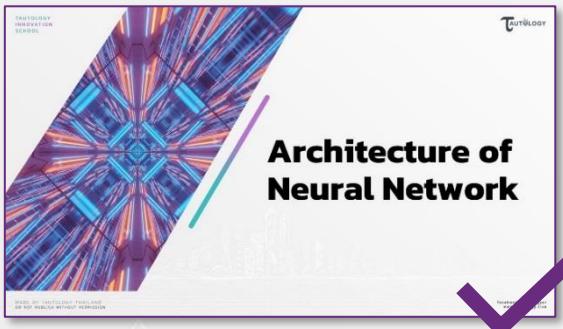
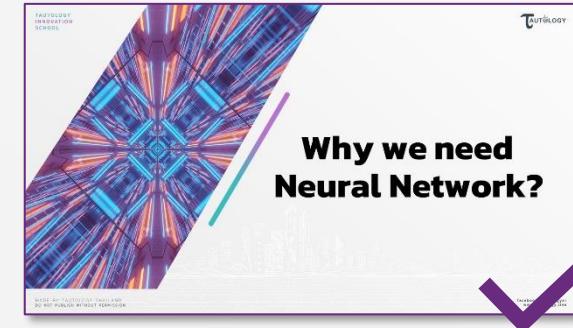
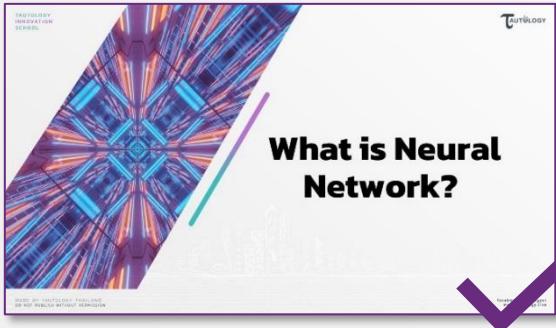
**Weight & Bias**



**Activation  
Function**



# Neural Network





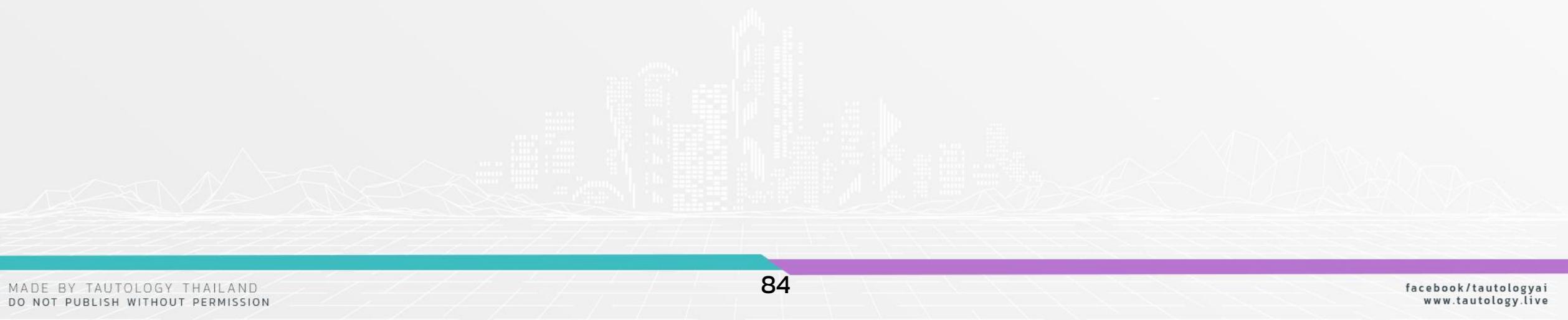
# How Neural Network Work



# How Neural Network Work

**Regression**

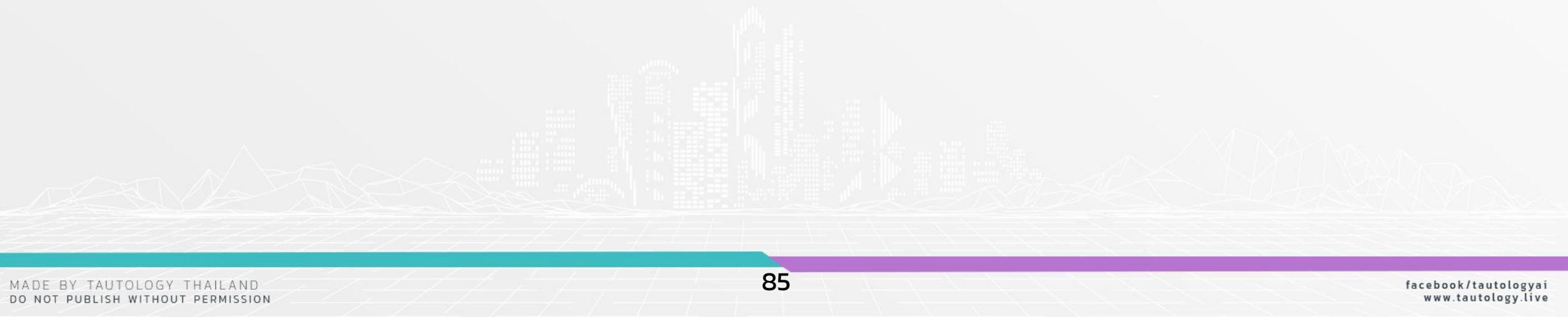
**Classification**



# How Neural Network Work

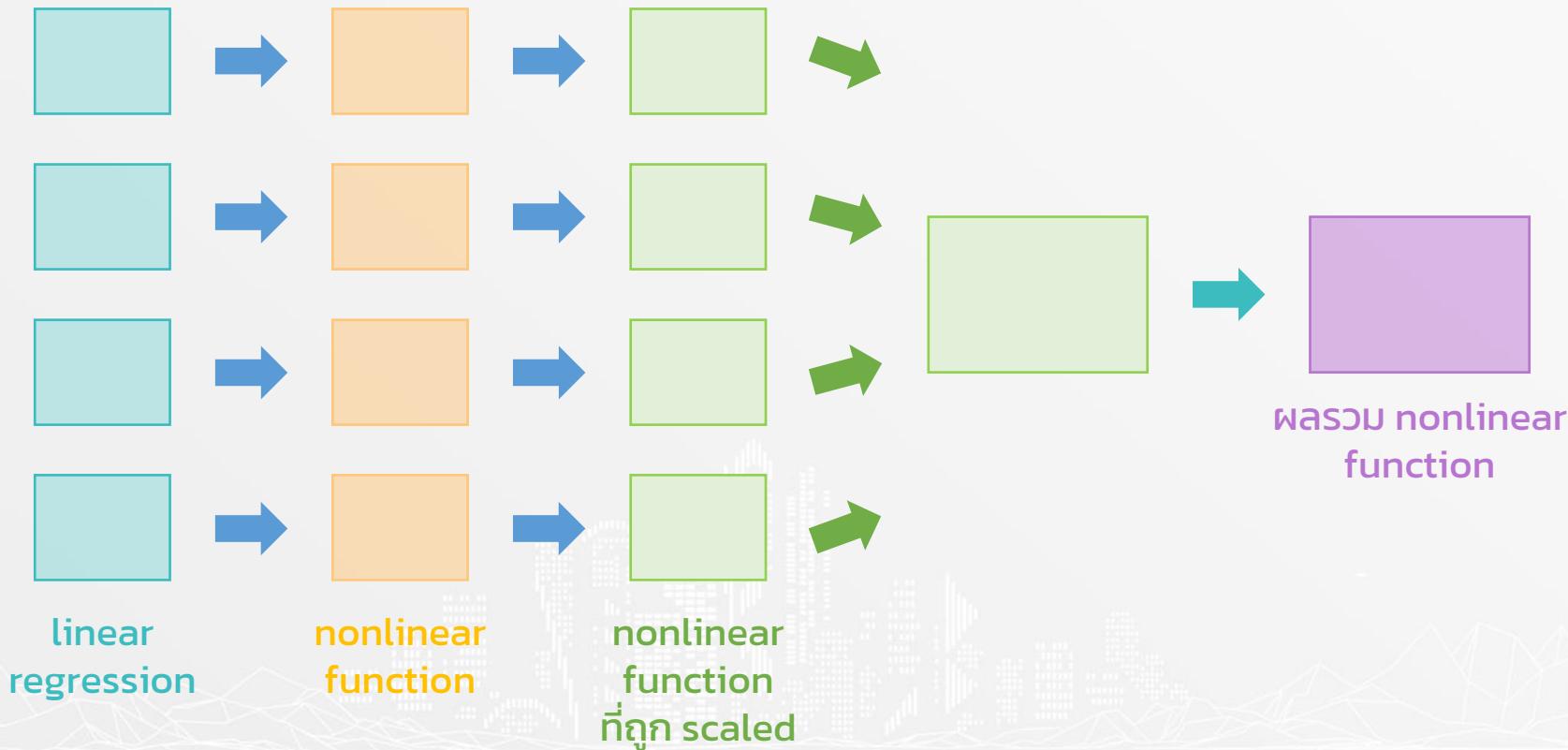
**Regression**

**Classification**



# How Neural Network Work

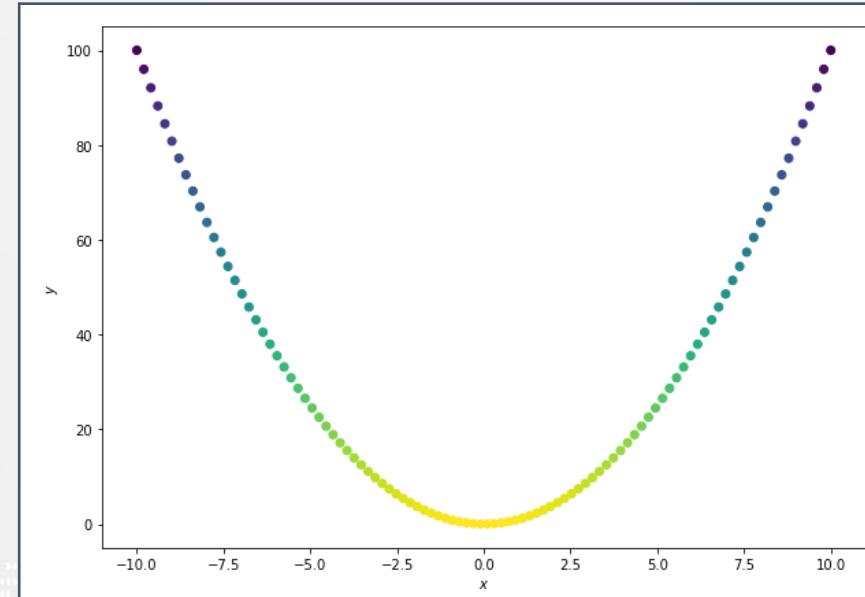
## Regression



# How Neural Network Work

Regression

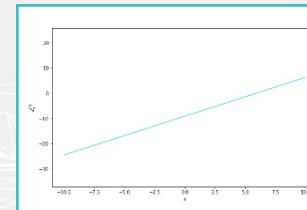
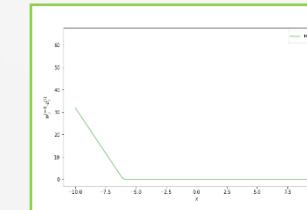
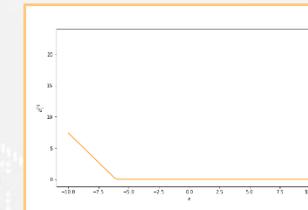
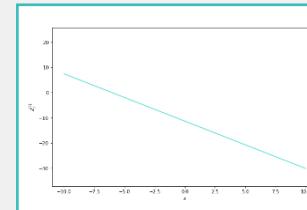
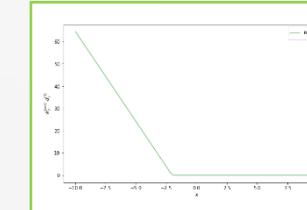
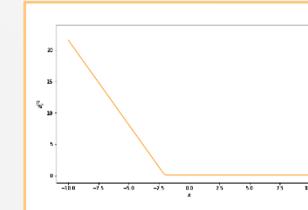
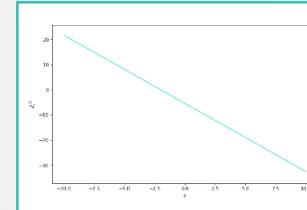
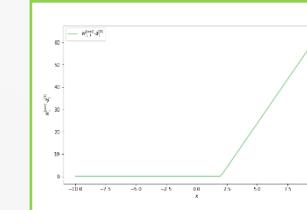
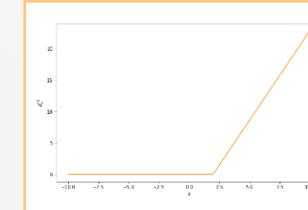
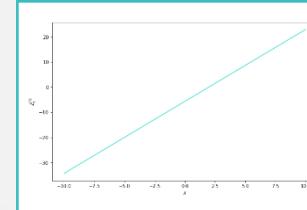
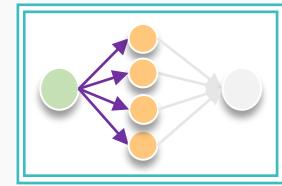
Example 1



# How Neural Network Work

## Regression

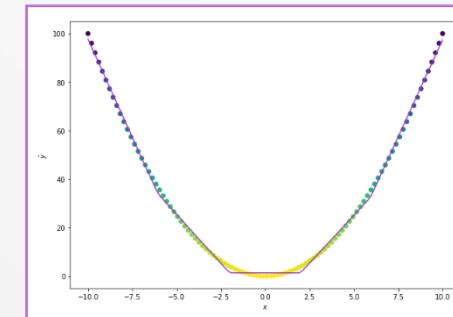
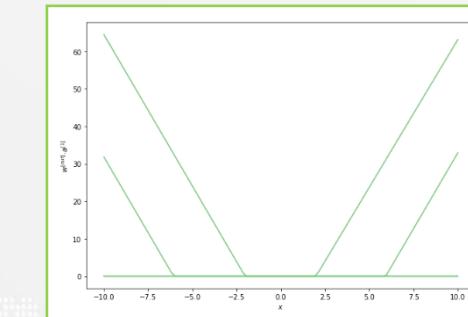
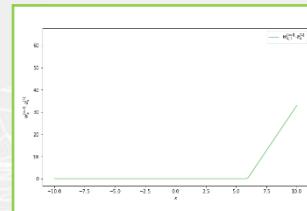
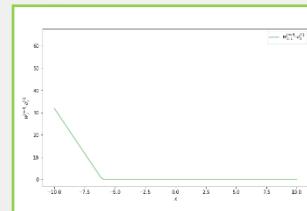
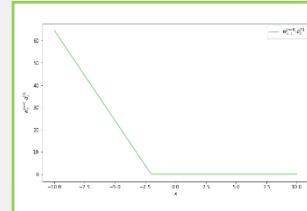
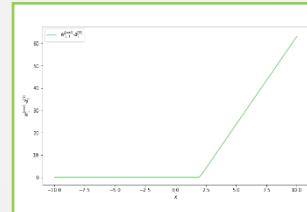
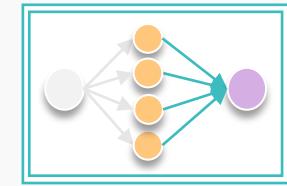
## Example 1



# How Neural Network Work

Regression

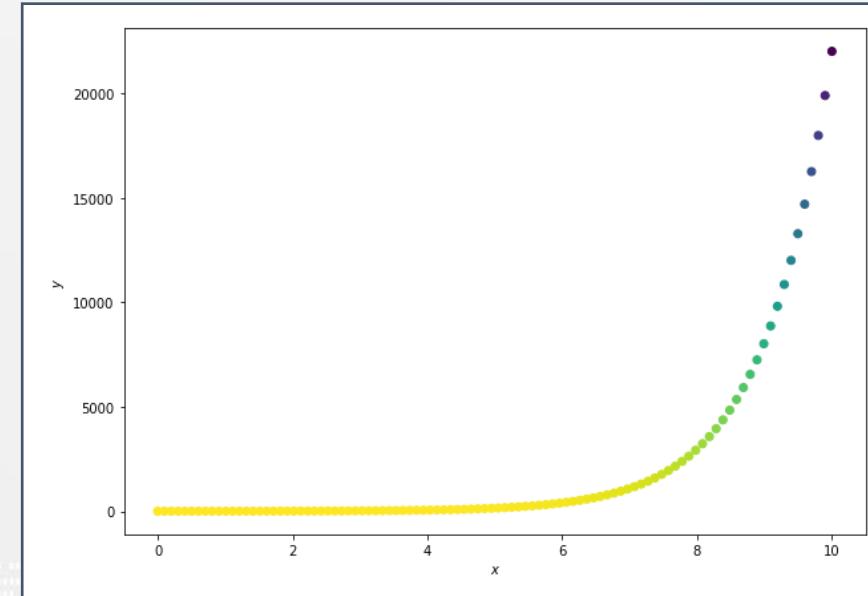
Example 1



# How Neural Network Work

Regression

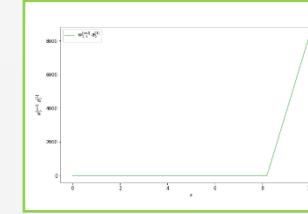
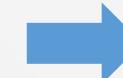
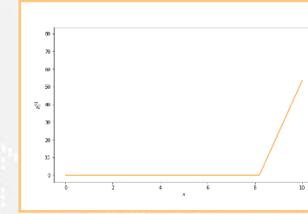
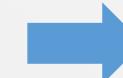
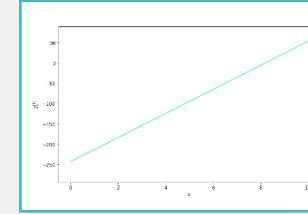
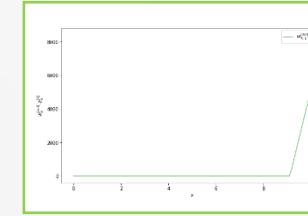
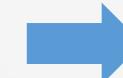
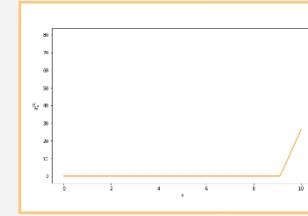
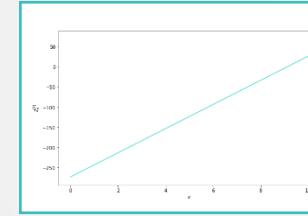
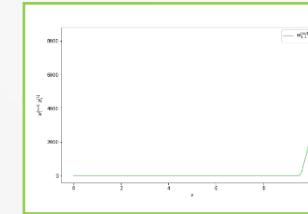
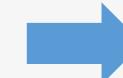
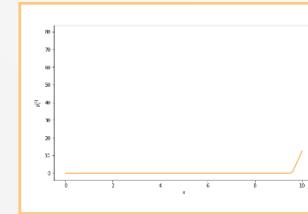
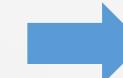
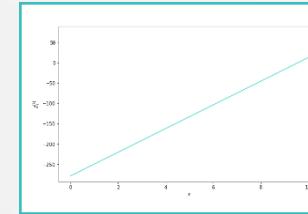
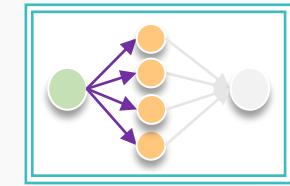
Example 2



# How Neural Network Work

Regression

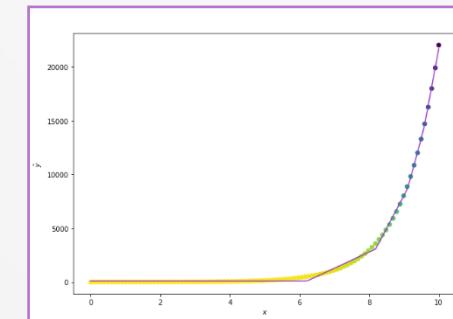
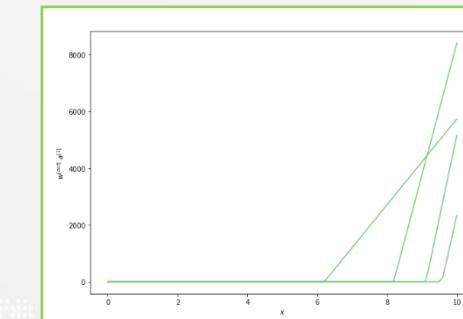
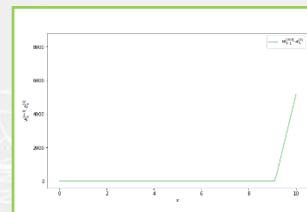
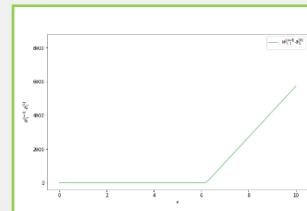
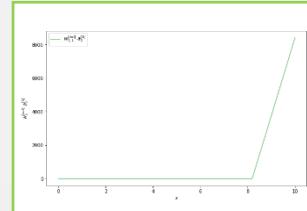
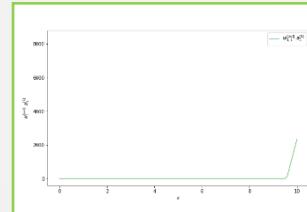
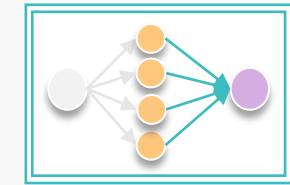
Example 2



# How Neural Network Work

Regression

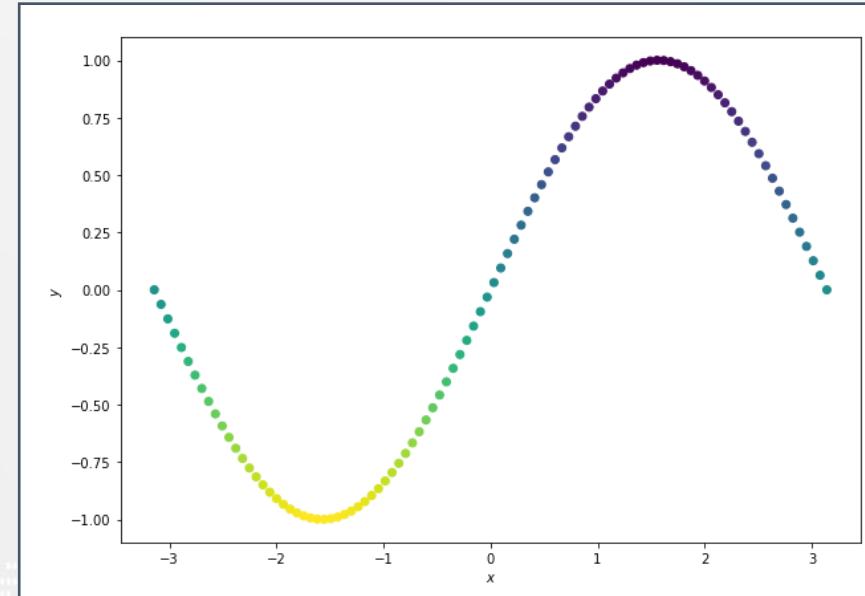
Example 2



# How Neural Network Work

Regression

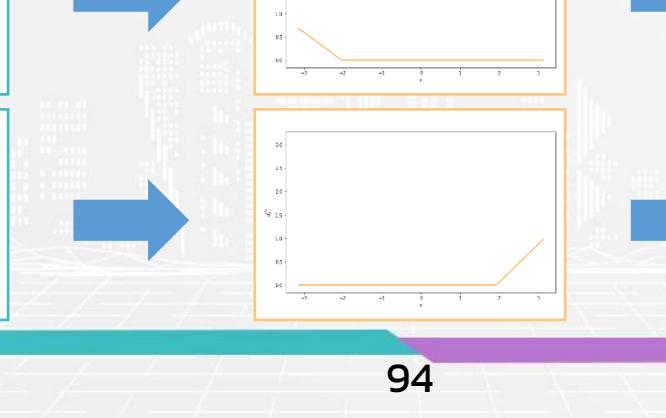
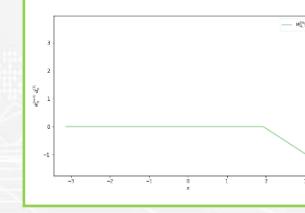
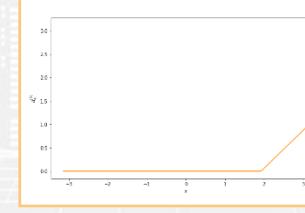
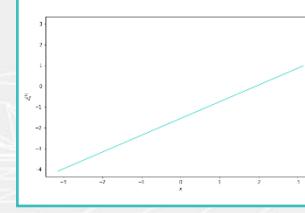
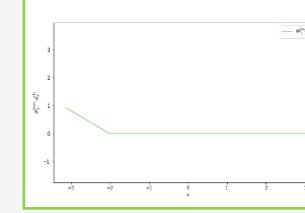
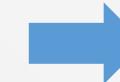
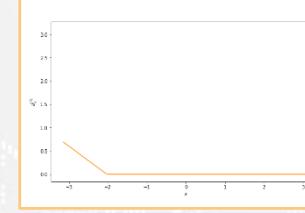
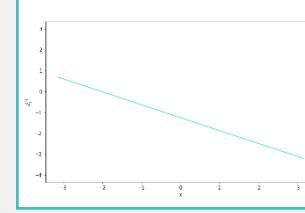
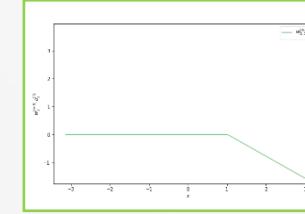
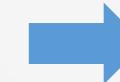
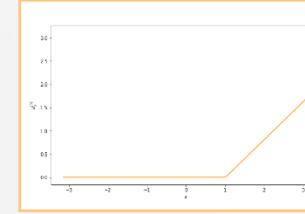
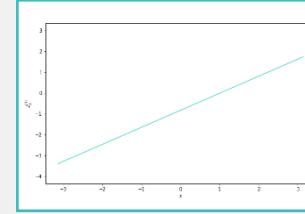
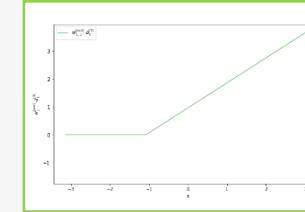
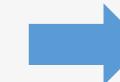
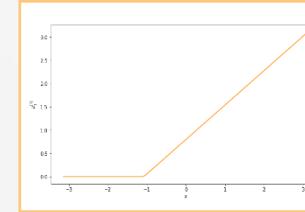
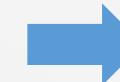
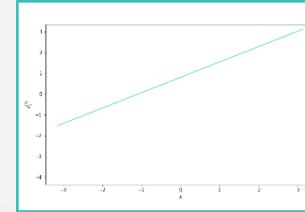
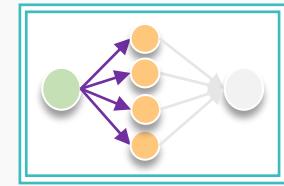
Example 3



# How Neural Network Work

Regression

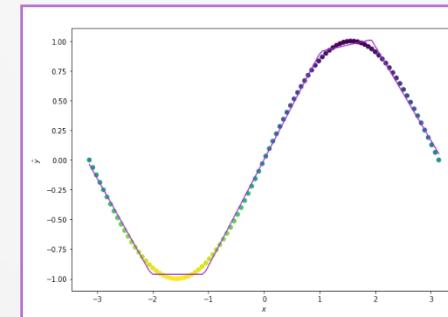
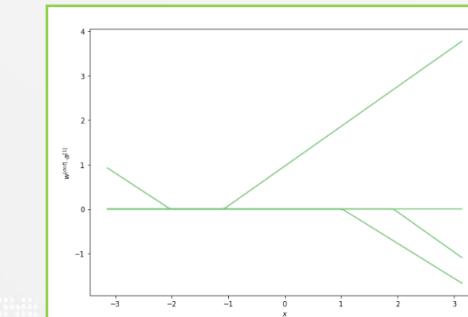
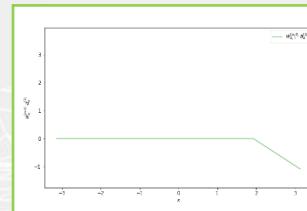
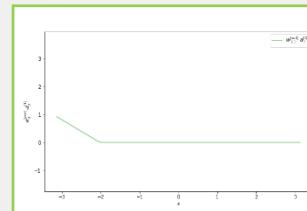
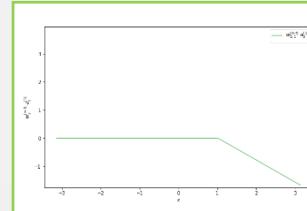
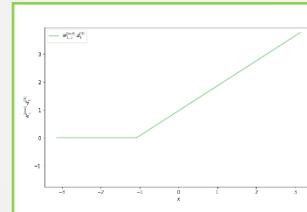
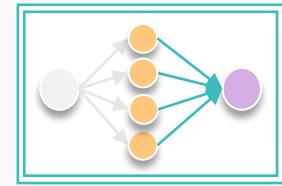
Example 3



# How Neural Network Work

Regression

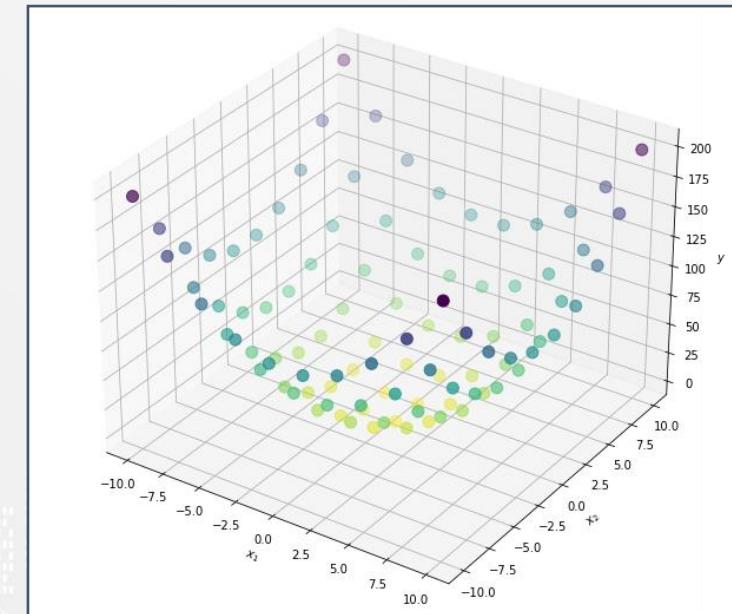
Example 3



# How Neural Network Work

Regression

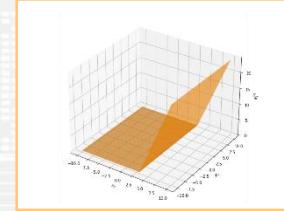
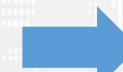
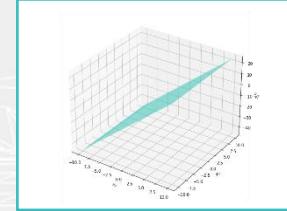
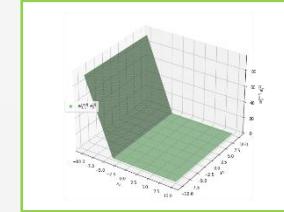
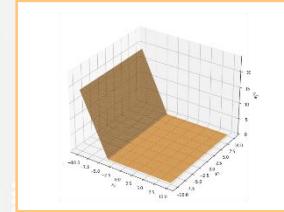
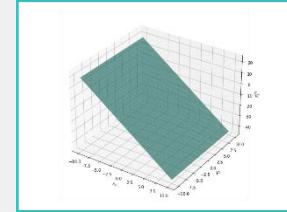
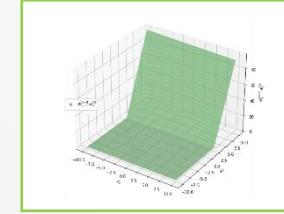
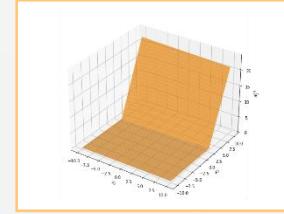
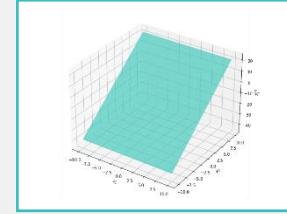
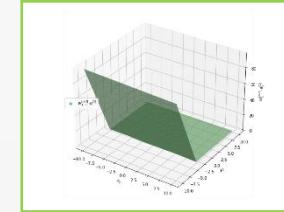
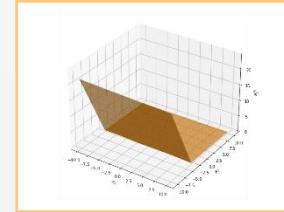
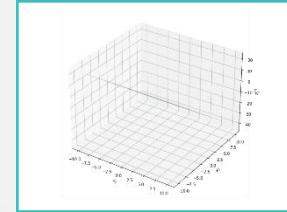
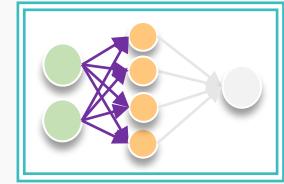
Example 4



# How Neural Network Work

Regression

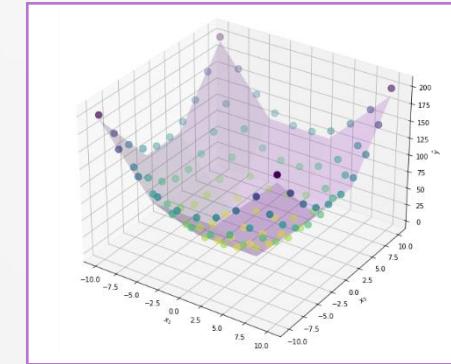
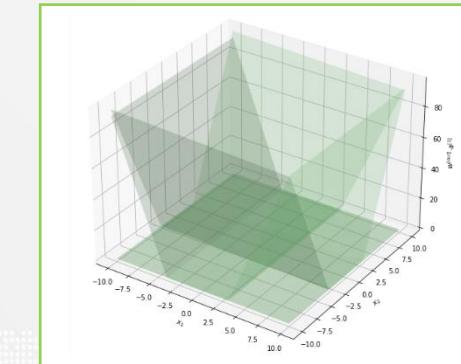
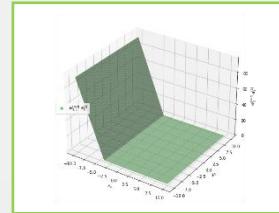
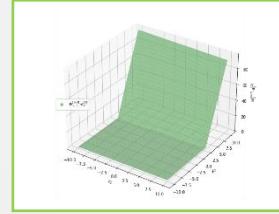
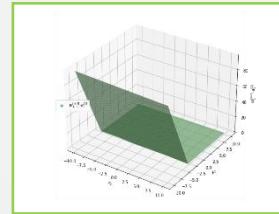
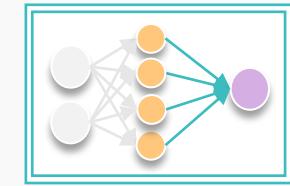
Example 4



# How Neural Network Work

Regression

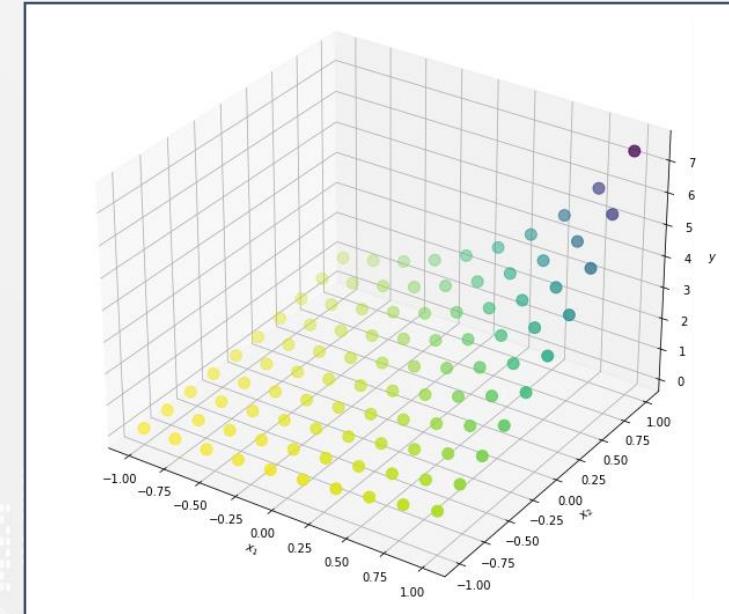
Example 4



# How Neural Network Work

Regression

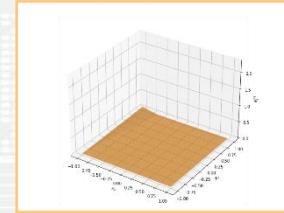
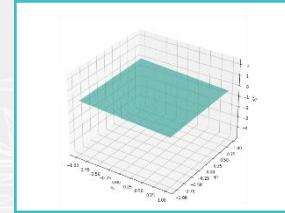
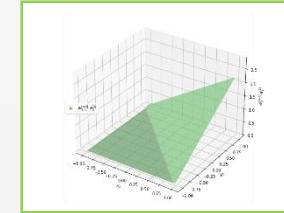
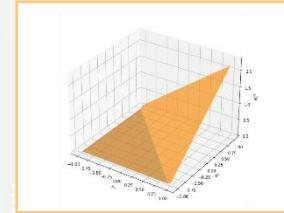
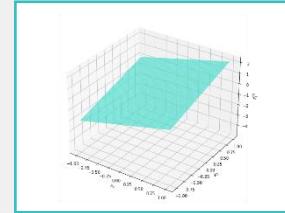
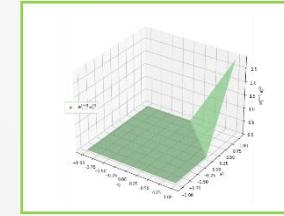
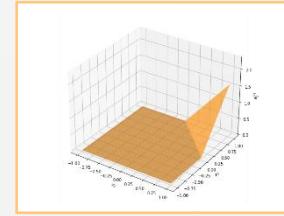
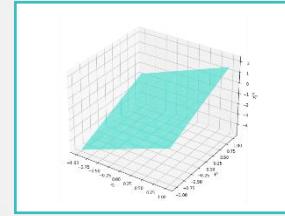
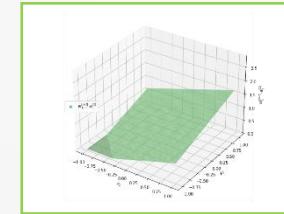
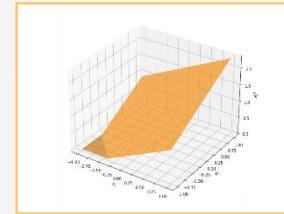
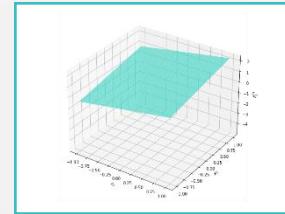
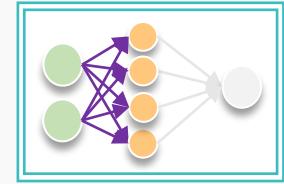
Example 5



# How Neural Network Work

Regression

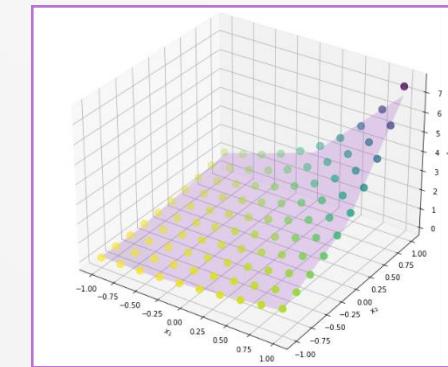
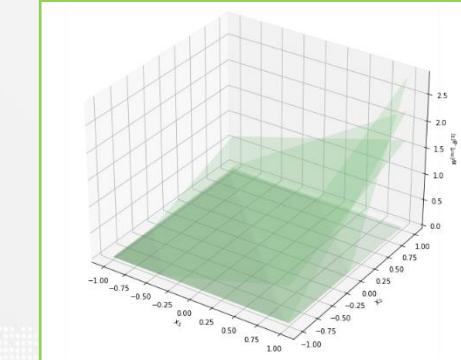
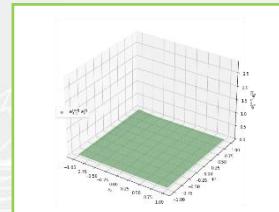
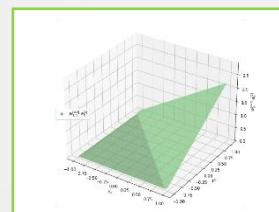
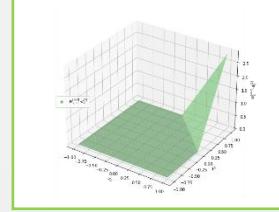
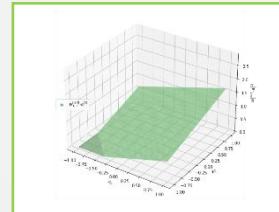
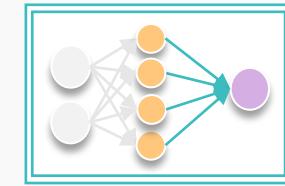
Example 5



# How Neural Network Work

Regression

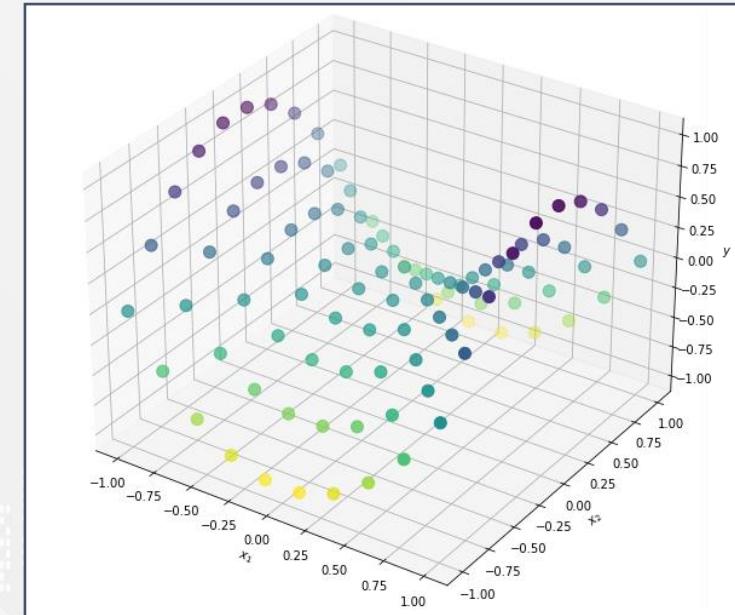
Example 5



# How Neural Network Work

Regression

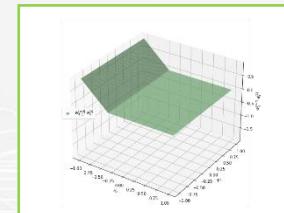
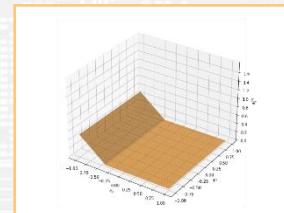
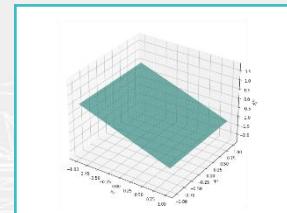
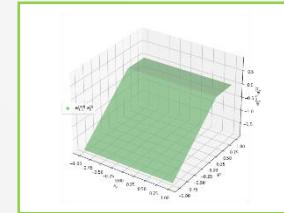
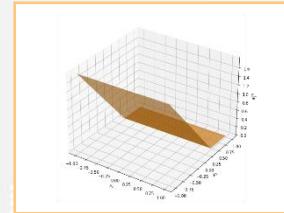
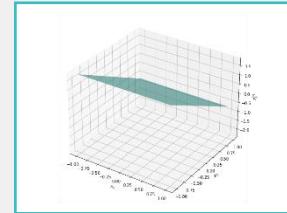
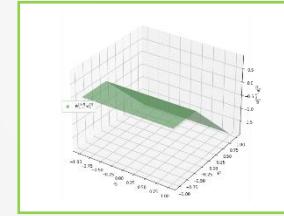
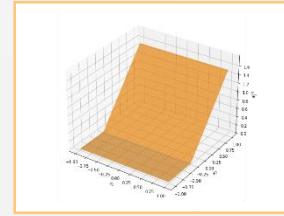
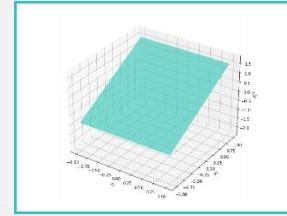
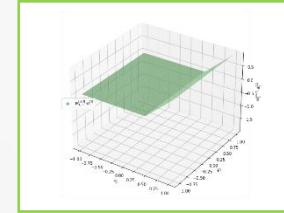
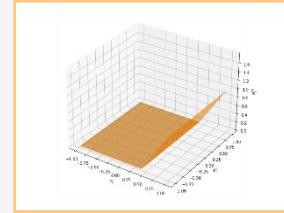
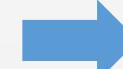
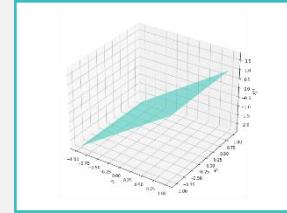
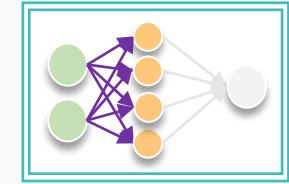
Example 6



# How Neural Network Work

Regression

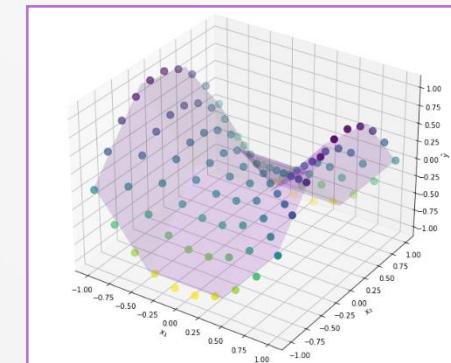
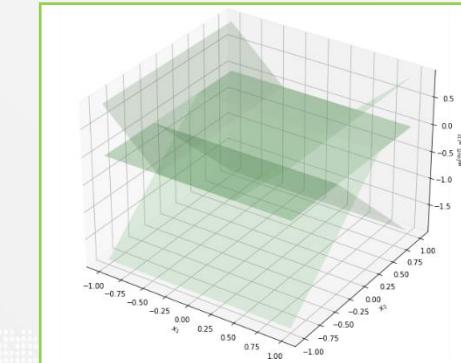
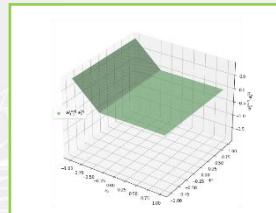
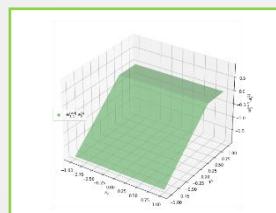
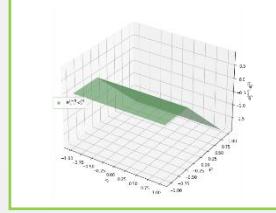
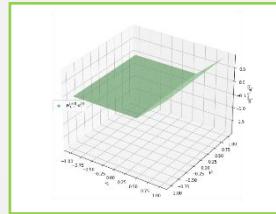
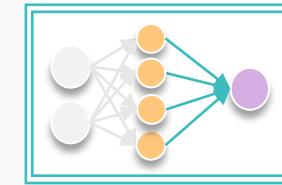
Example 6



# How Neural Network Work

Regression

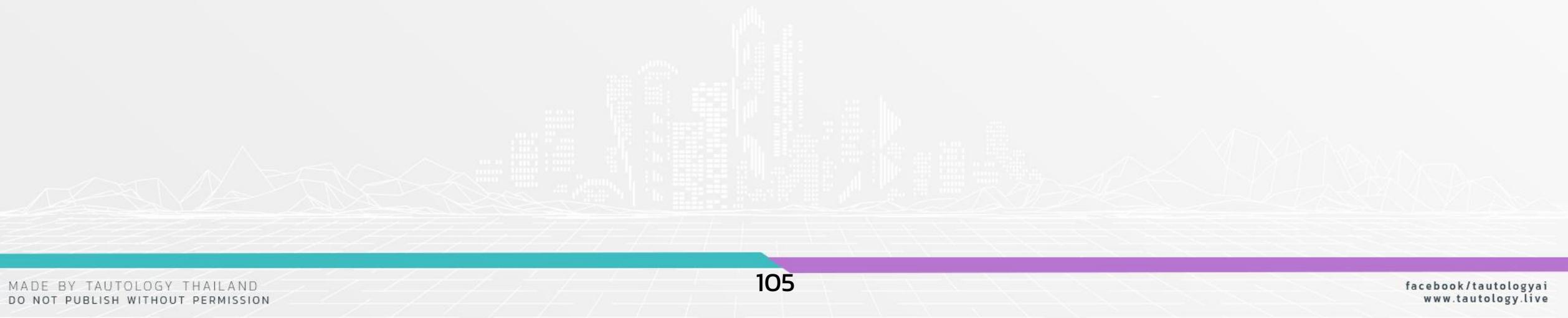
Example 6



# How Neural Network Work

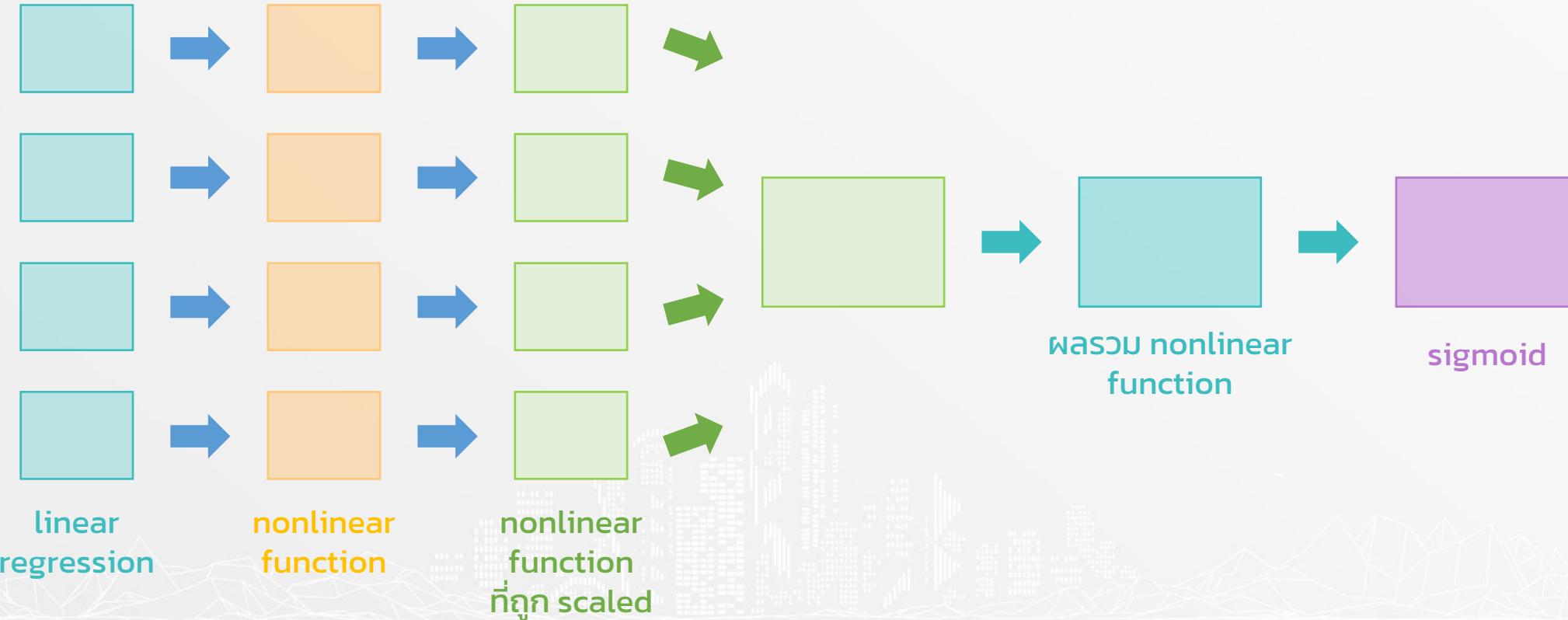
Regression

Classification



# How Neural Network Work

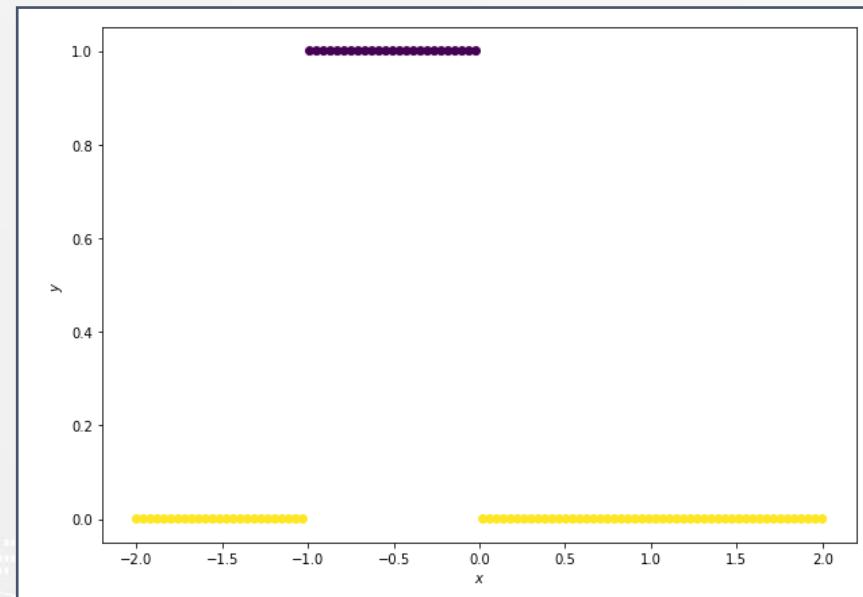
## Binary Classification



# How Neural Network Work

## Classification

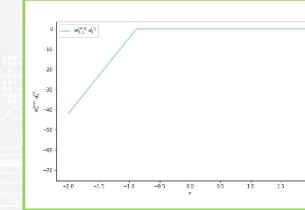
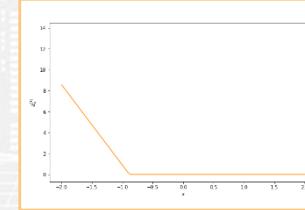
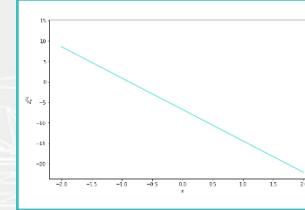
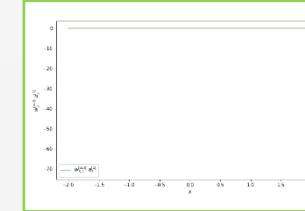
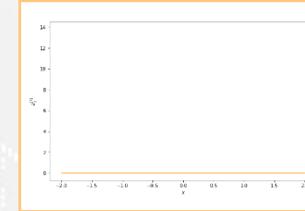
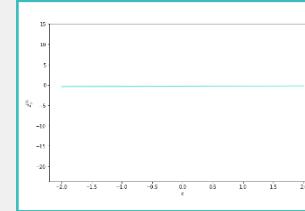
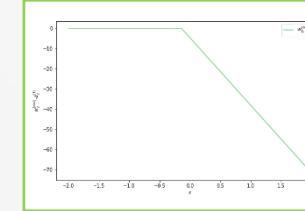
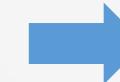
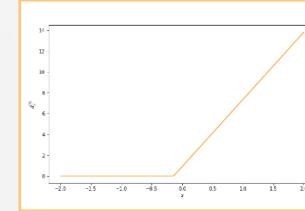
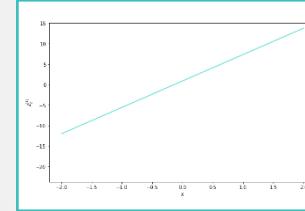
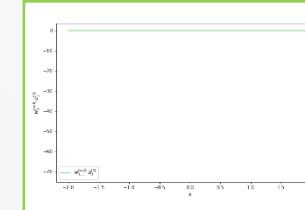
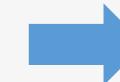
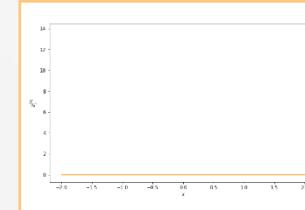
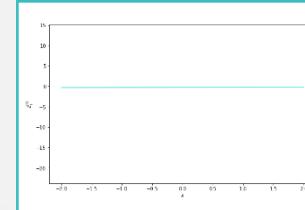
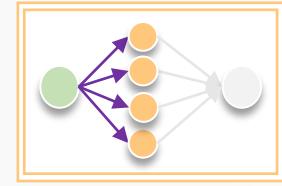
### Example 1: Binary Classification



# How Neural Network Work

## Classification

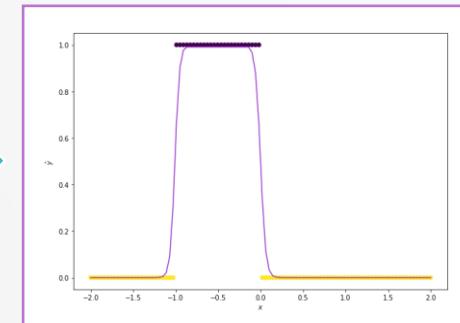
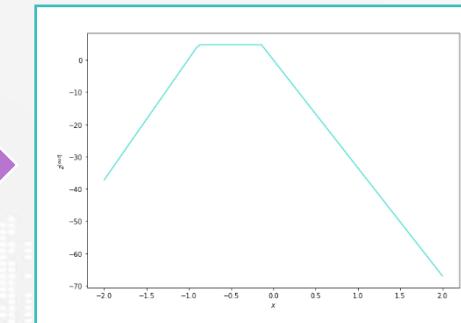
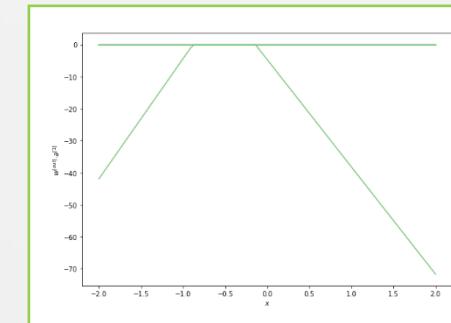
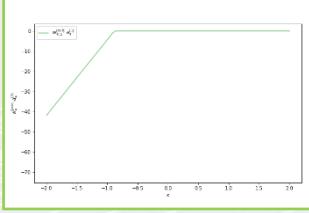
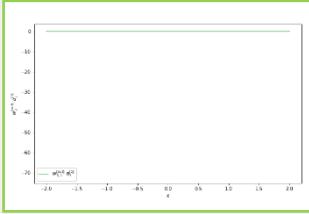
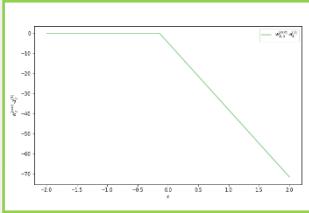
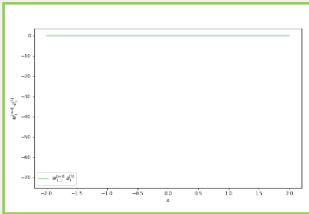
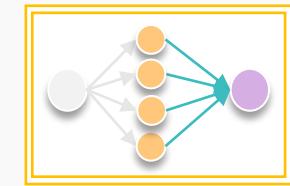
## Example 1



# How Neural Network Work

Classification

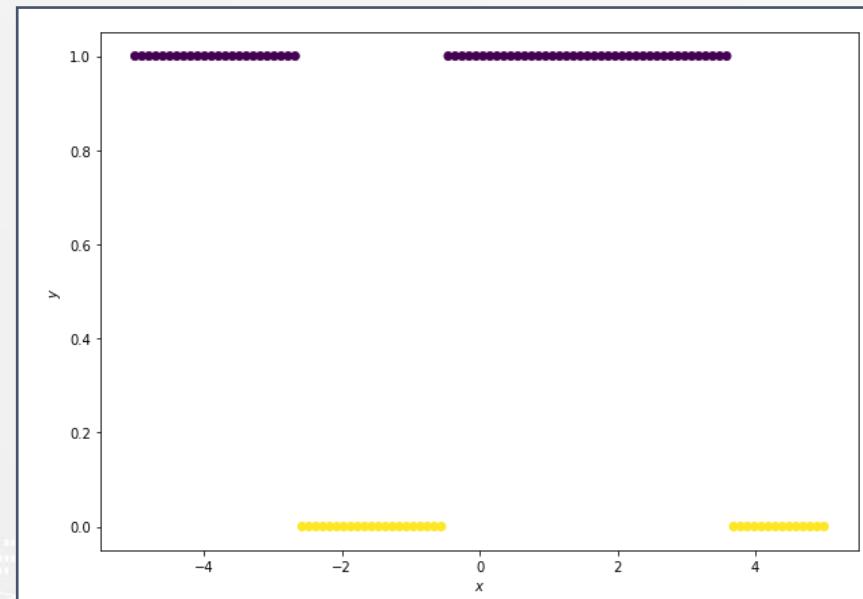
Example 1



# How Neural Network Work

## Classification

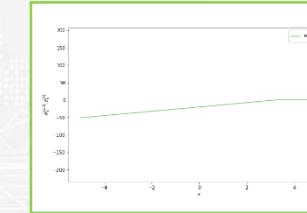
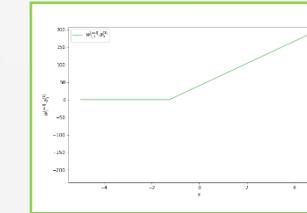
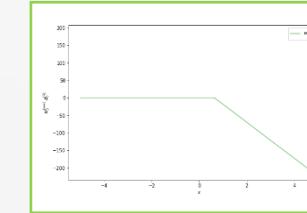
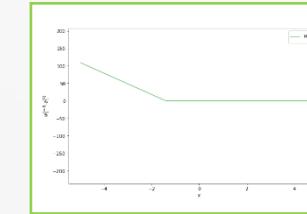
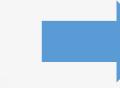
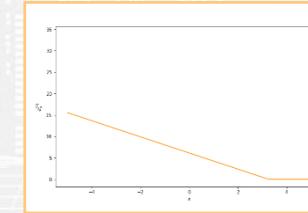
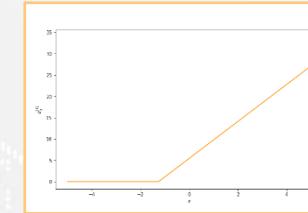
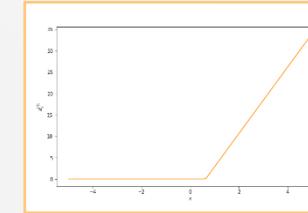
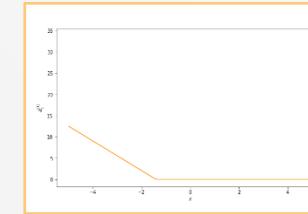
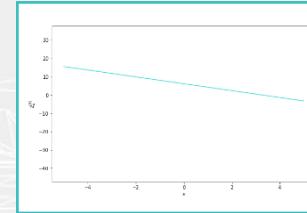
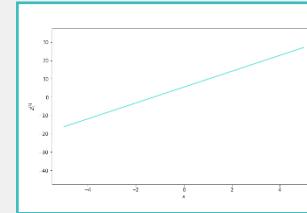
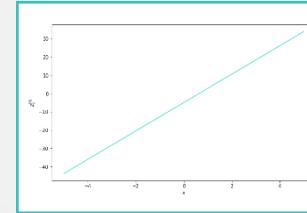
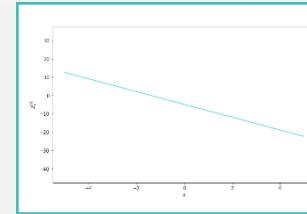
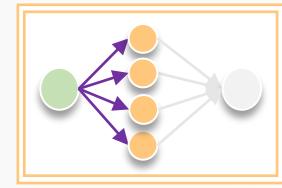
### Example 2 : Binary Classification



# How Neural Network Work

## Classification

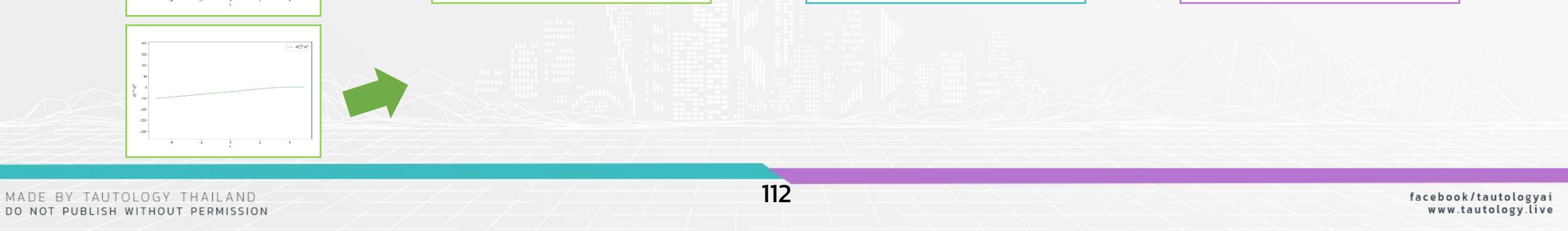
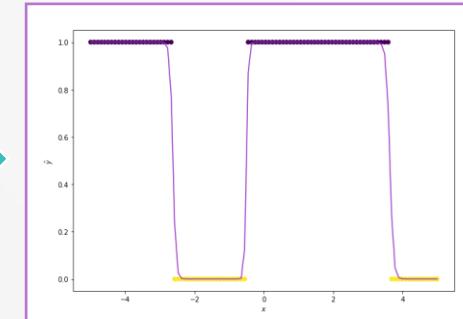
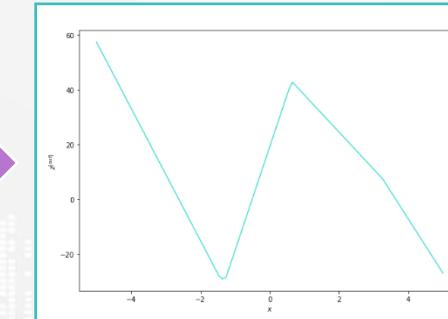
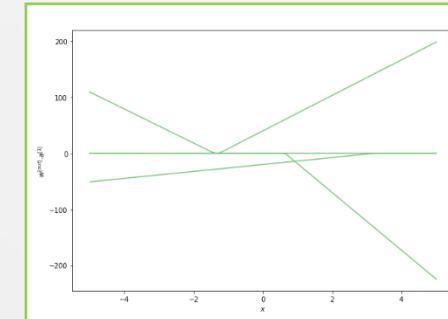
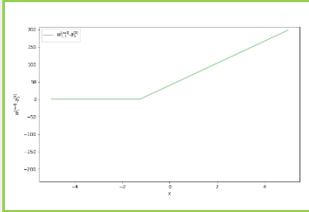
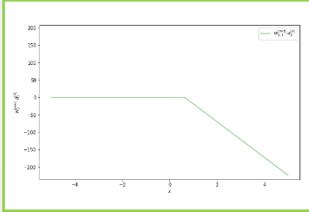
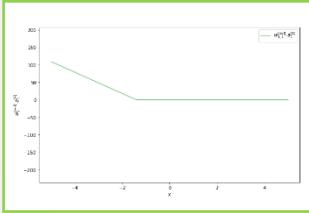
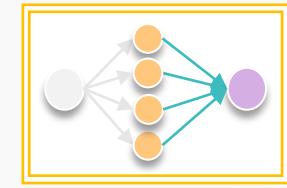
## Example 2



# How Neural Network Work

Classification

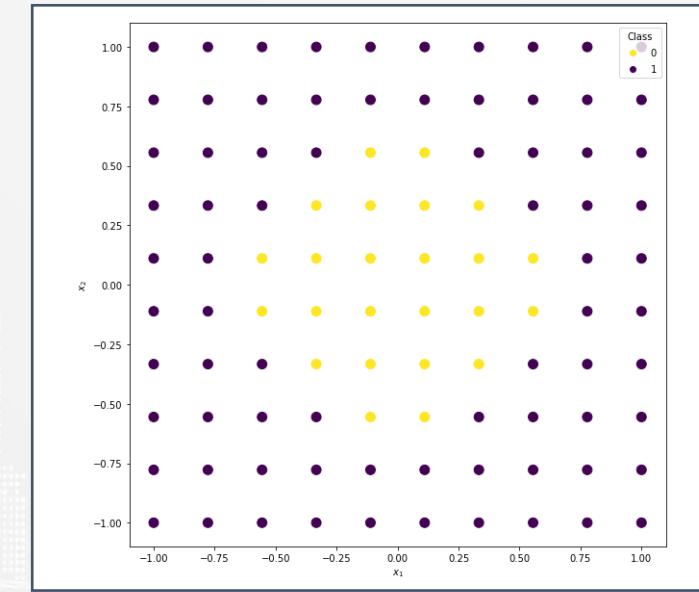
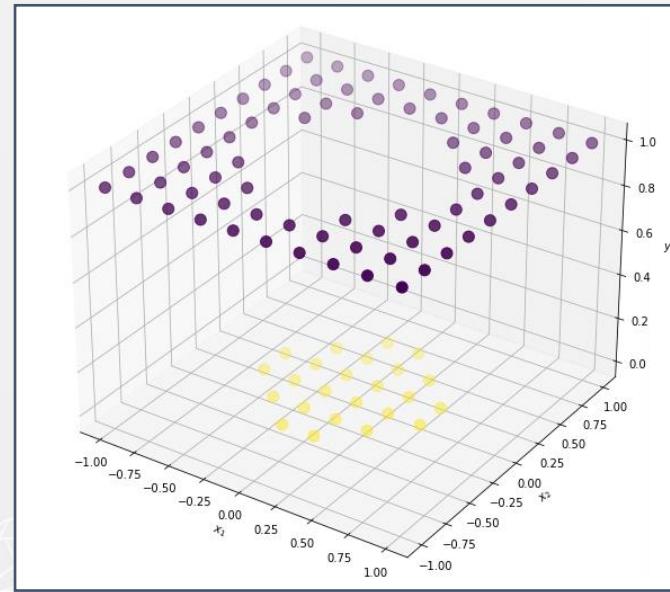
Example 2



# How Neural Network Work

## Classification

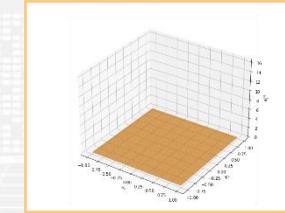
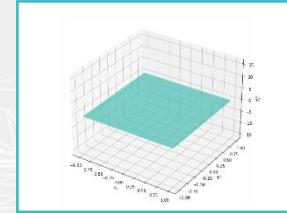
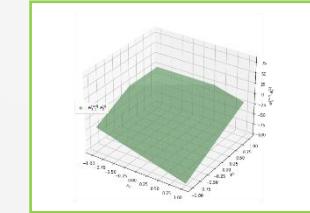
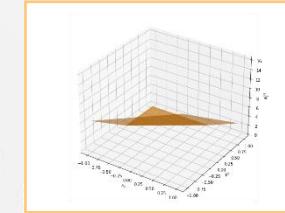
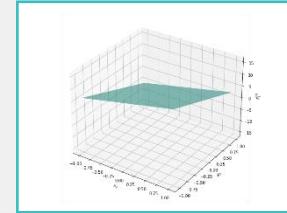
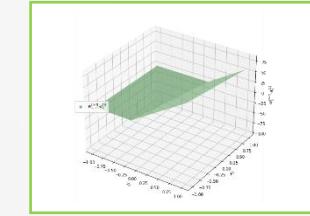
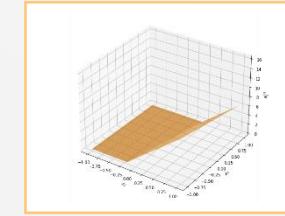
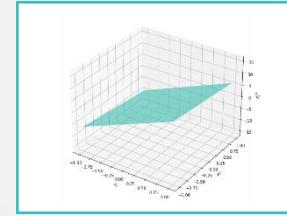
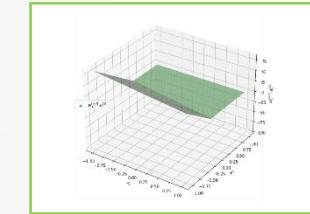
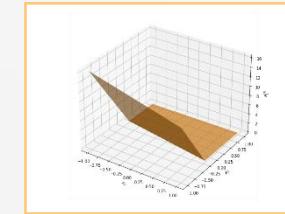
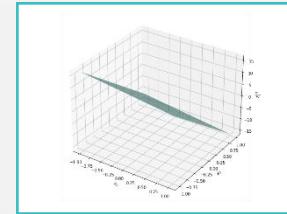
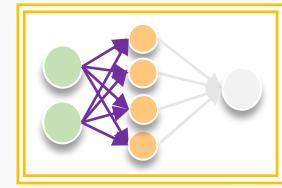
### Example 3 : Binary Classification



# How Neural Network Work

Classification

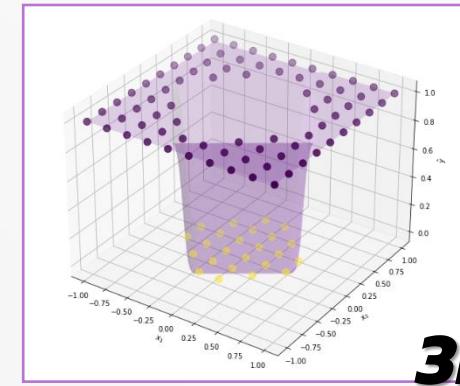
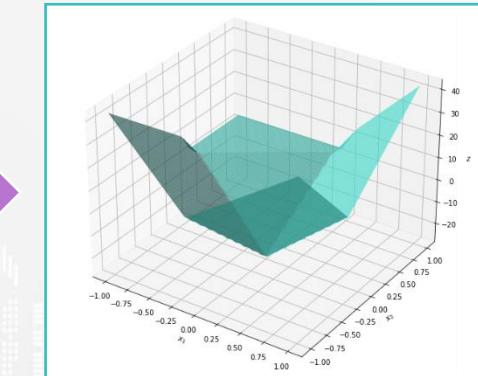
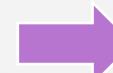
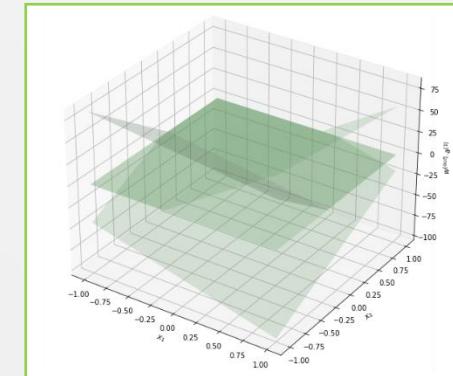
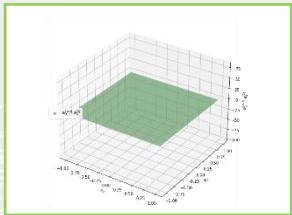
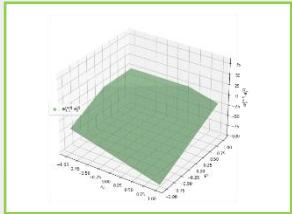
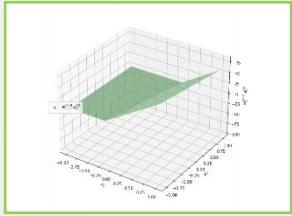
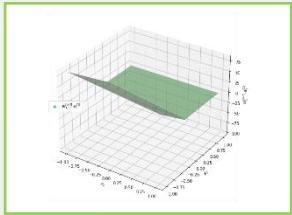
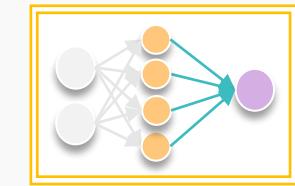
Example 3



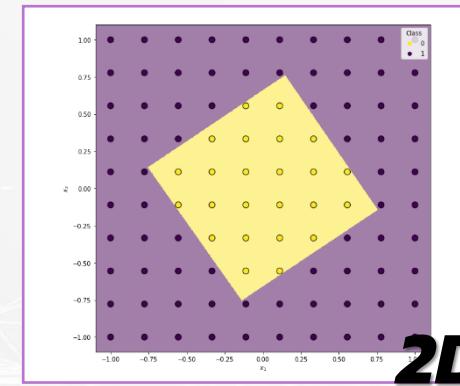
# How Neural Network Work

**Classification**

**Example 3**



**3D**

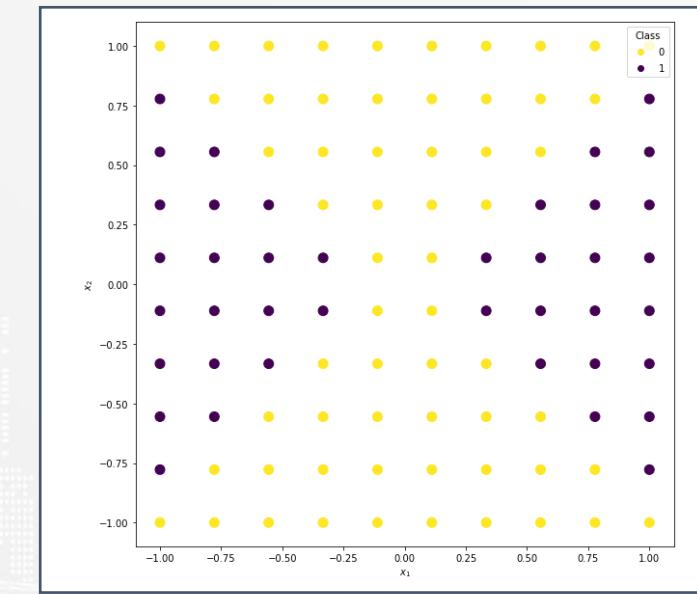
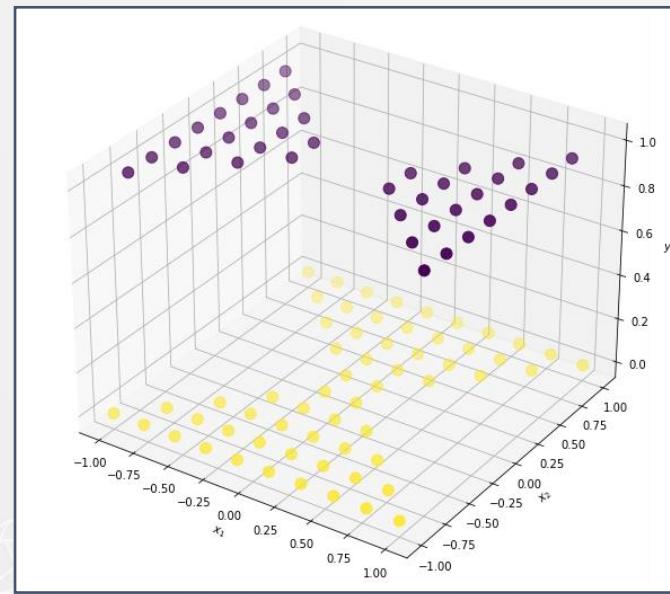


**2D**

# How Neural Network Work

## Classification

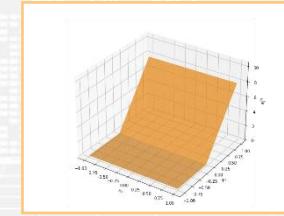
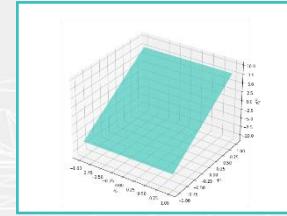
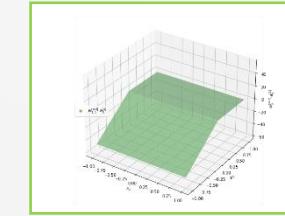
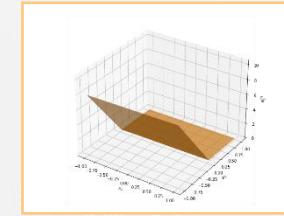
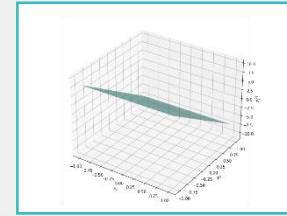
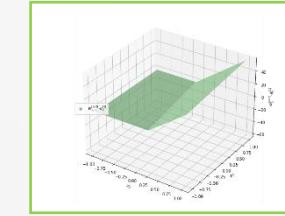
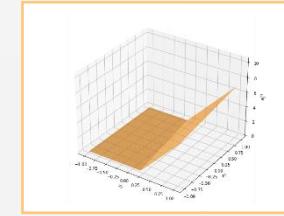
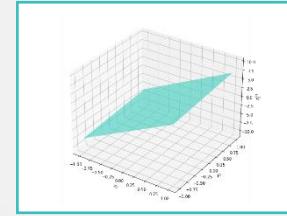
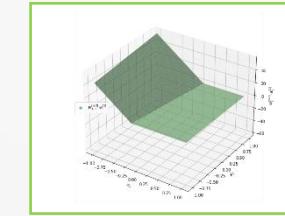
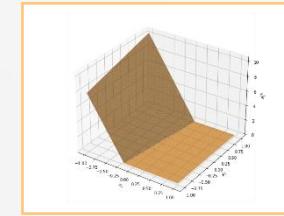
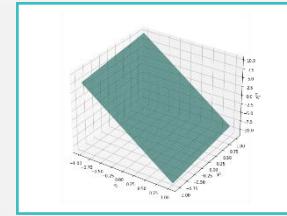
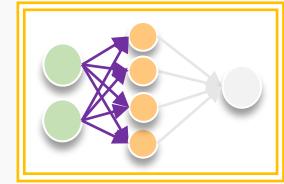
### Example 4 : Binary Classification



# How Neural Network Work

Classification

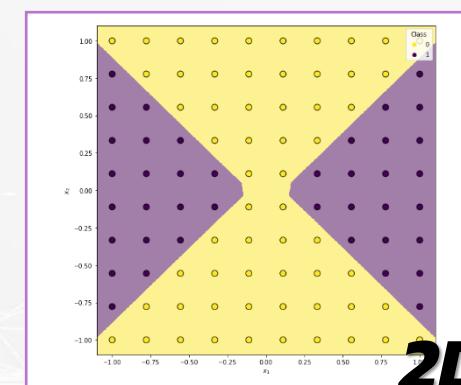
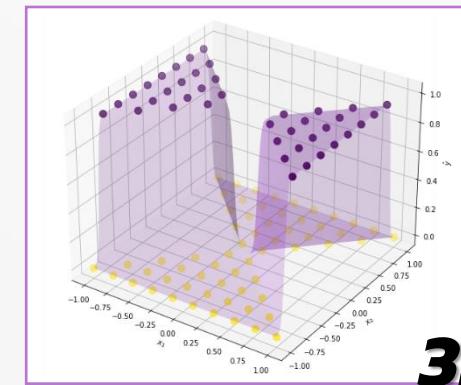
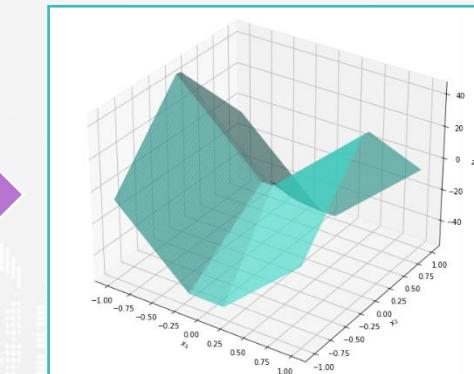
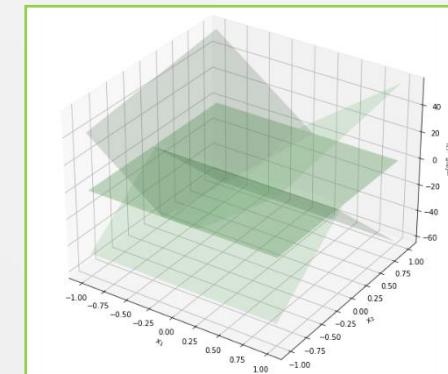
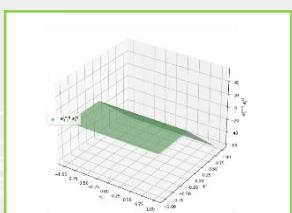
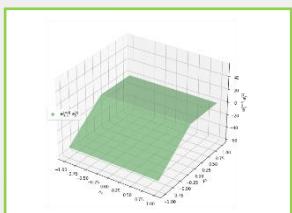
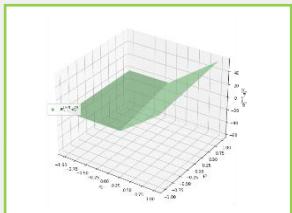
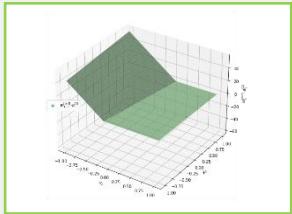
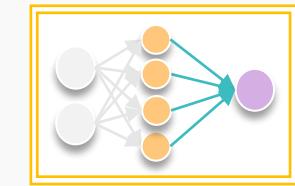
Example 4



# How Neural Network Work

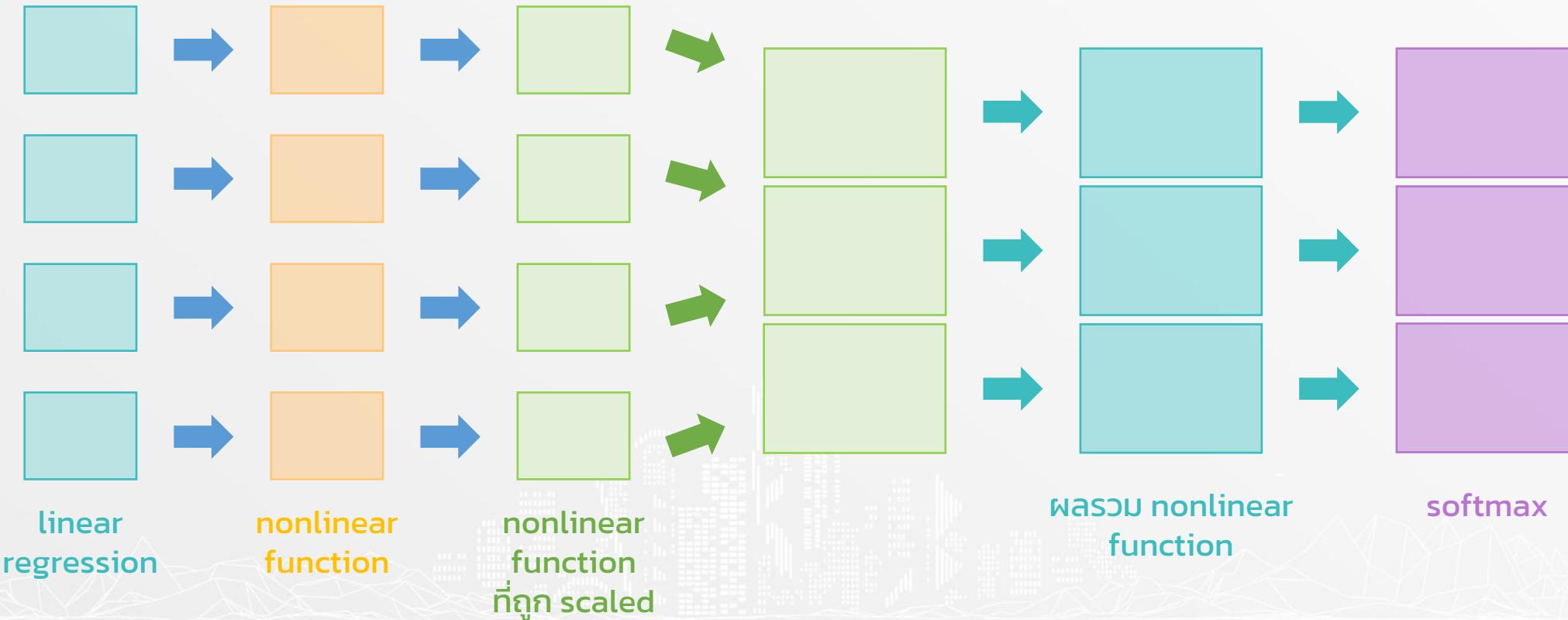
**Classification**

**Example 4**



# How Neural Network Work

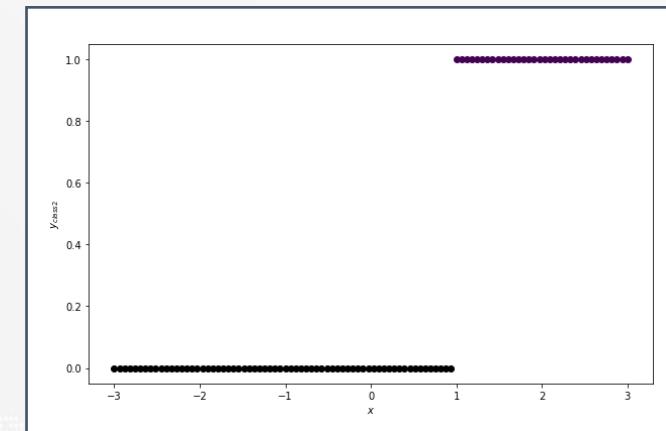
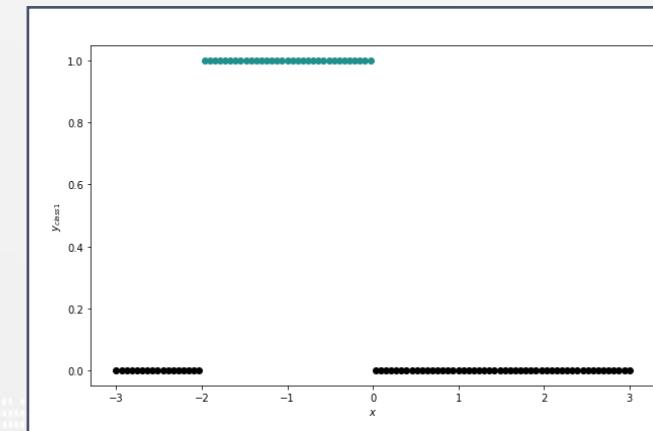
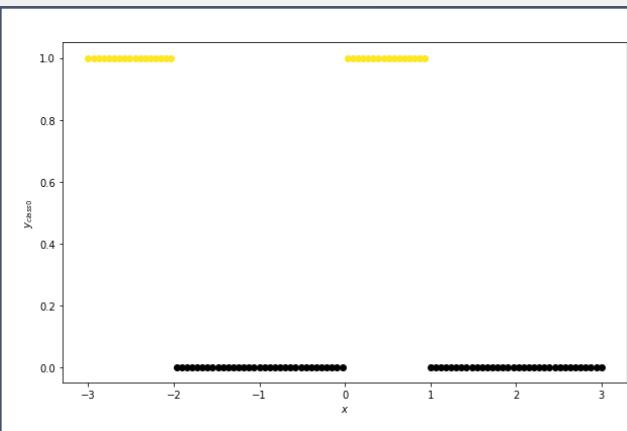
## Multi-Class Classification



# How Neural Network Work

## Classification

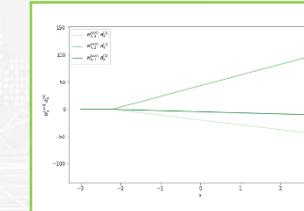
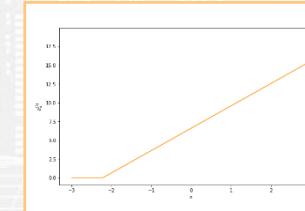
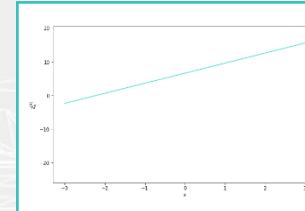
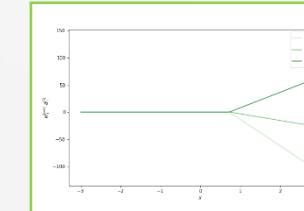
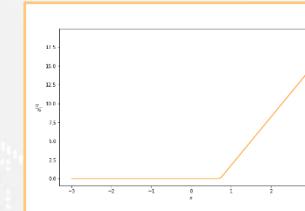
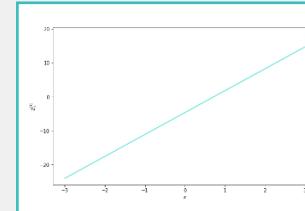
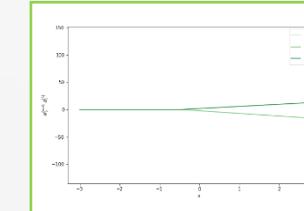
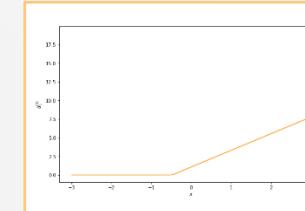
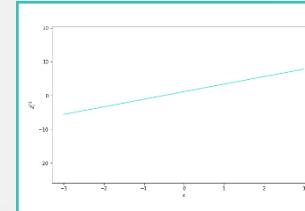
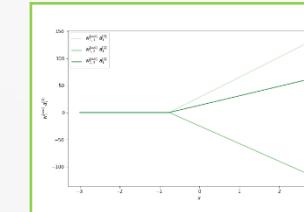
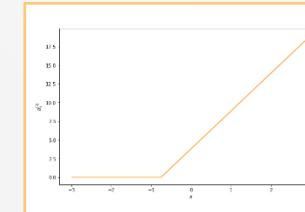
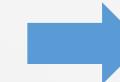
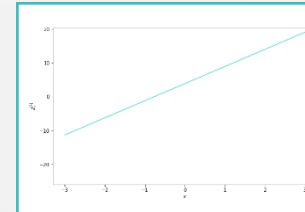
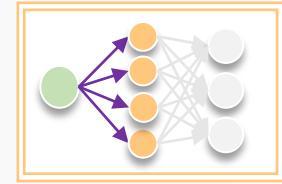
### Example 5 : Multi-Class Classification



# How Neural Network Work

## Classification

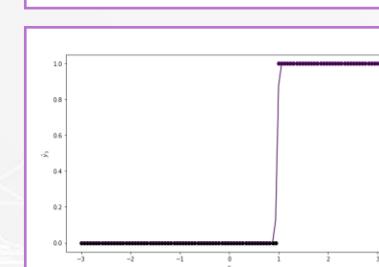
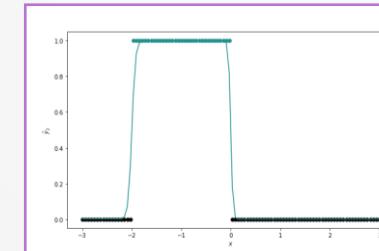
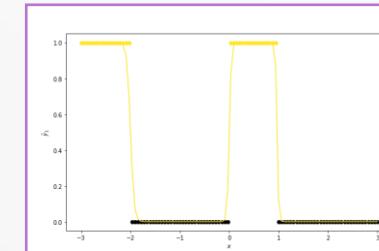
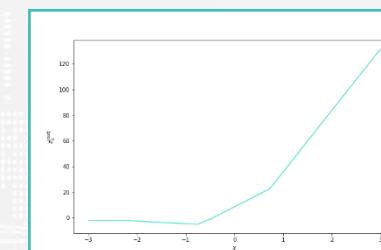
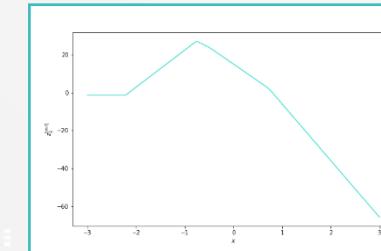
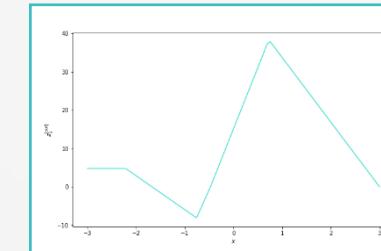
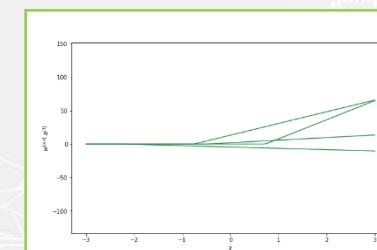
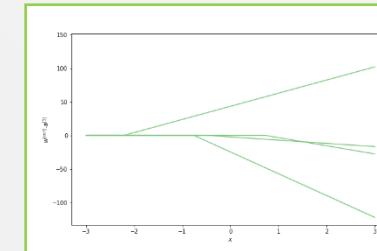
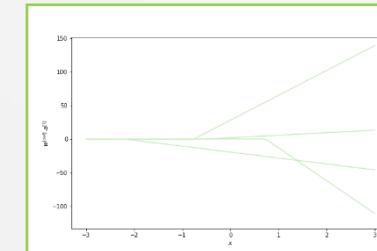
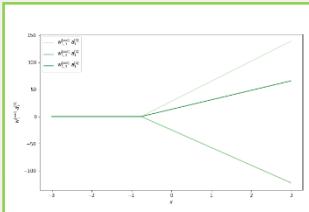
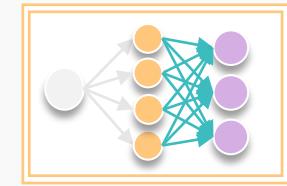
## Example 5



# How Neural Network Work

## Classification

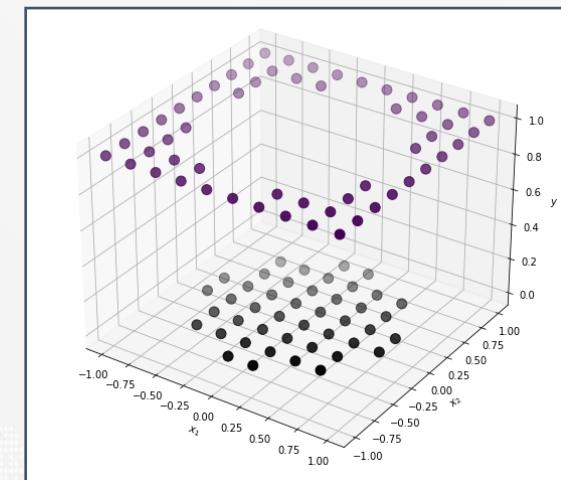
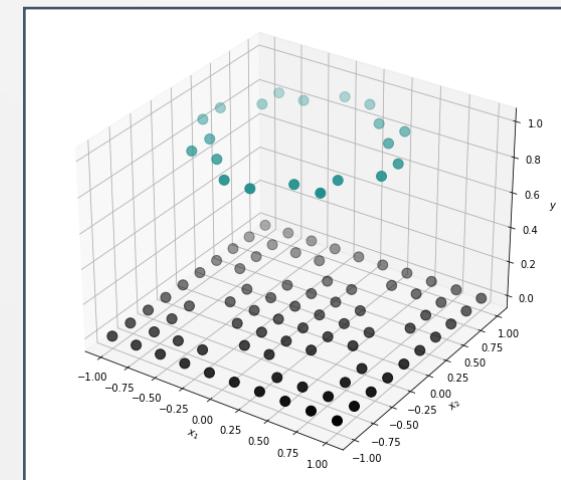
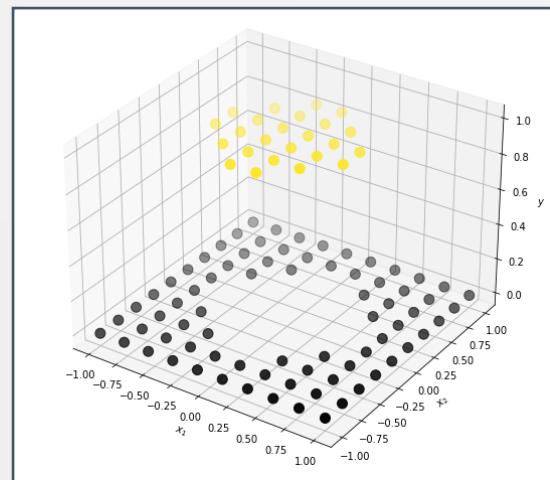
## Example 5



# How Neural Network Work

## Classification

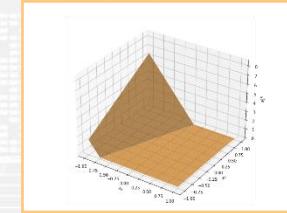
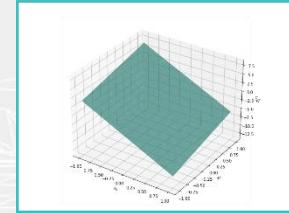
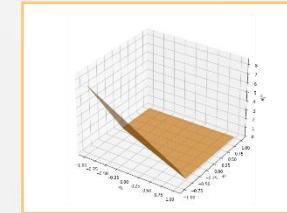
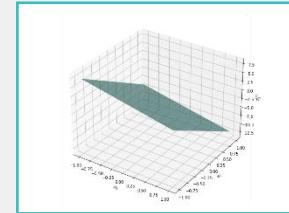
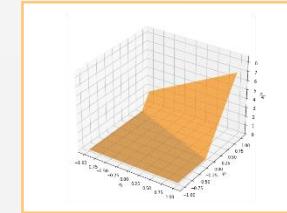
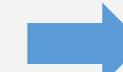
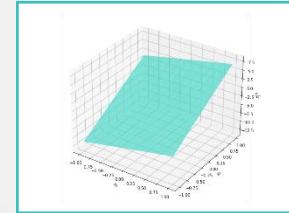
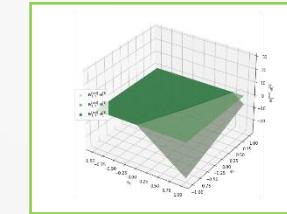
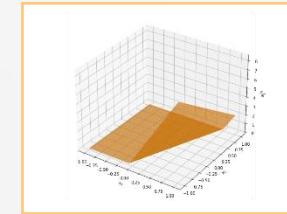
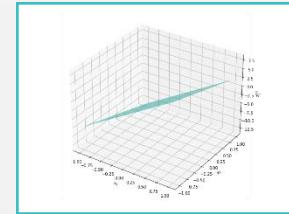
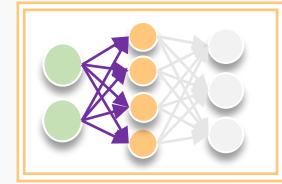
### Example 6 : Multi-Class Classification



# How Neural Network Work

Classification

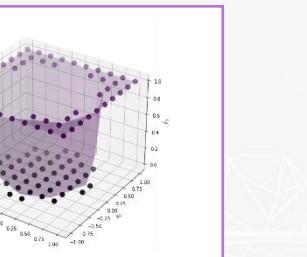
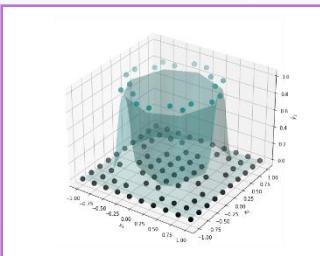
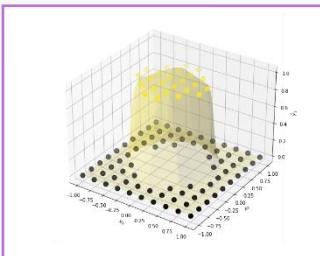
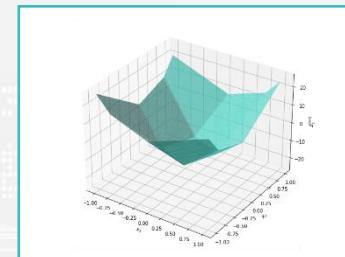
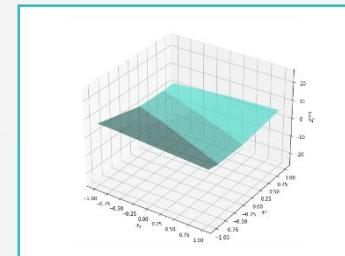
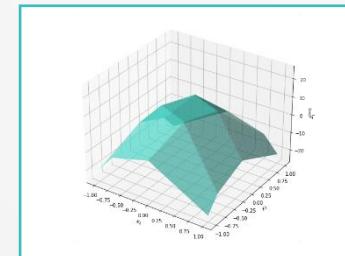
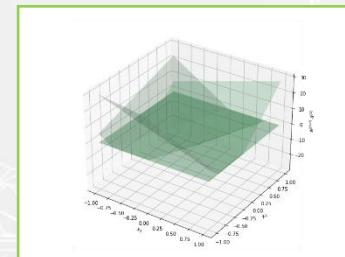
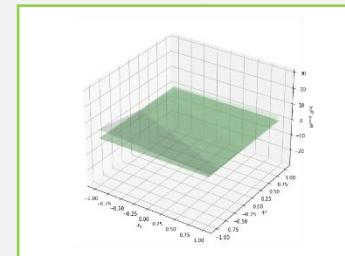
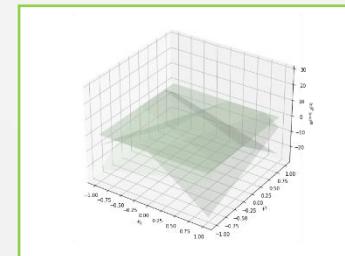
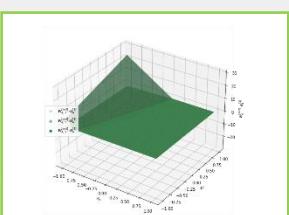
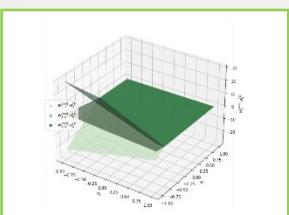
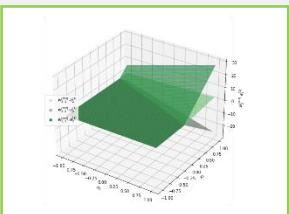
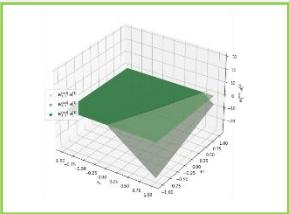
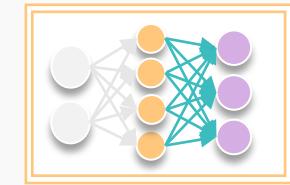
Example 6



# How Neural Network Work

## Classification

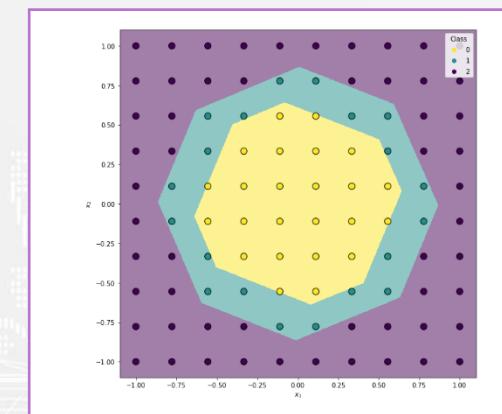
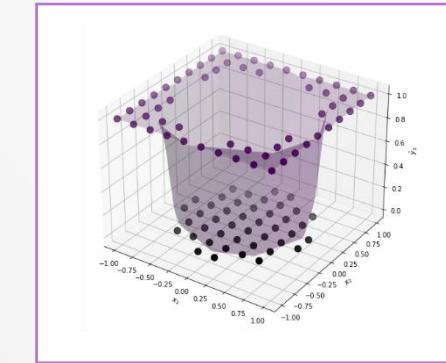
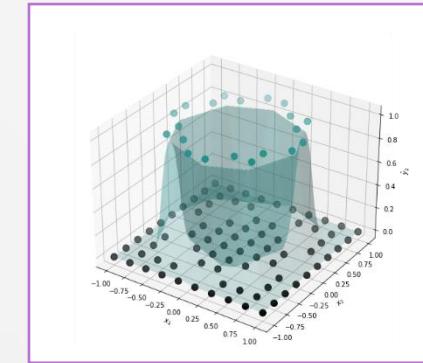
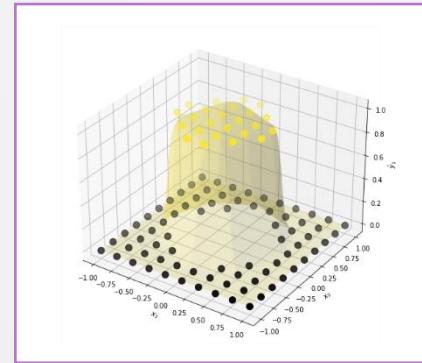
## Example 6



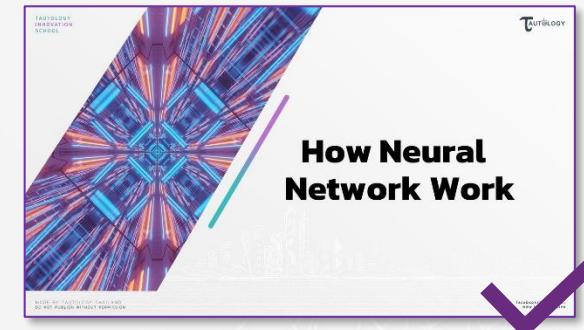
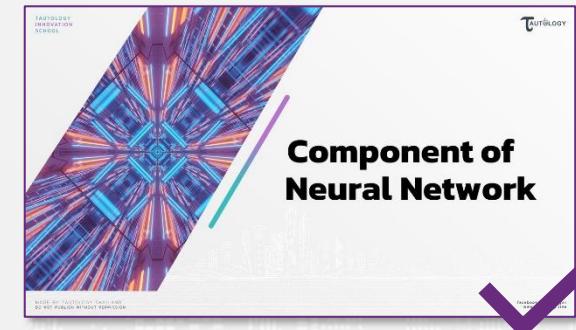
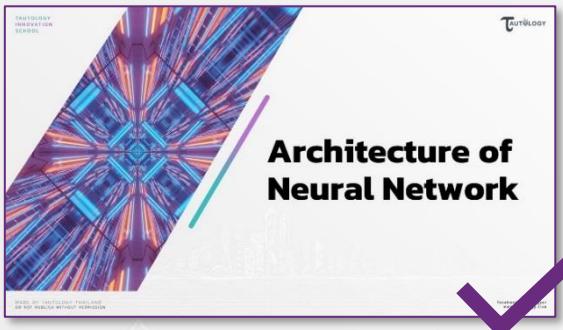
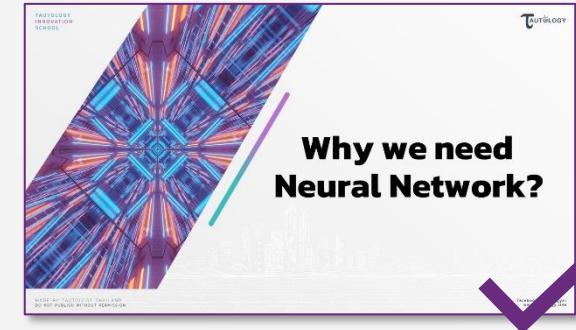
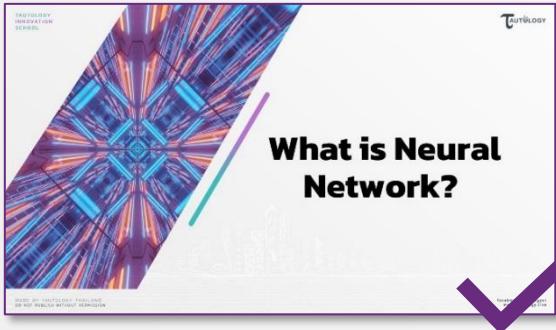
# How Neural Network Work

Classification

Example 6



# Neural Network



# Neural Network & Deep Learning

**Neural Network**



**Deep Learning**



**Architecture  
Interpretation**



**Deep Learning for  
Regression**



**Deep Learning for  
Classification**



**Workshop**



# Deep Learning



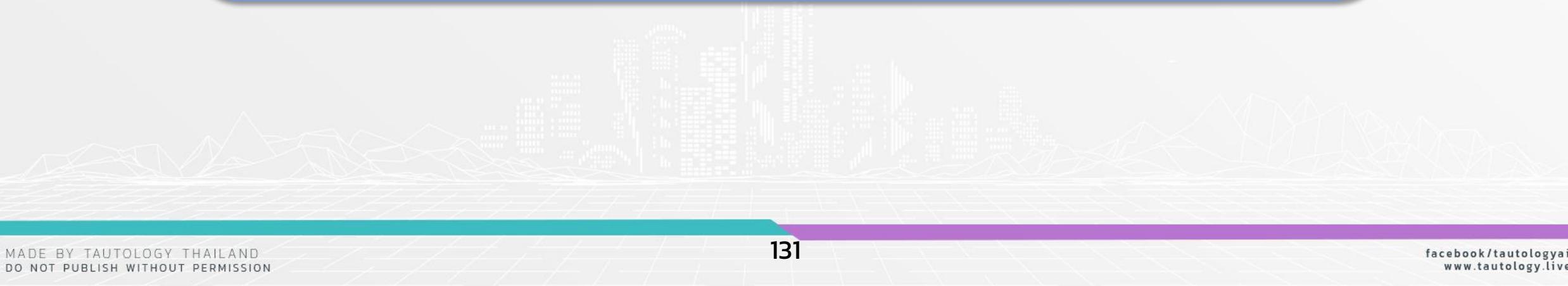


# What is Deep Learning?

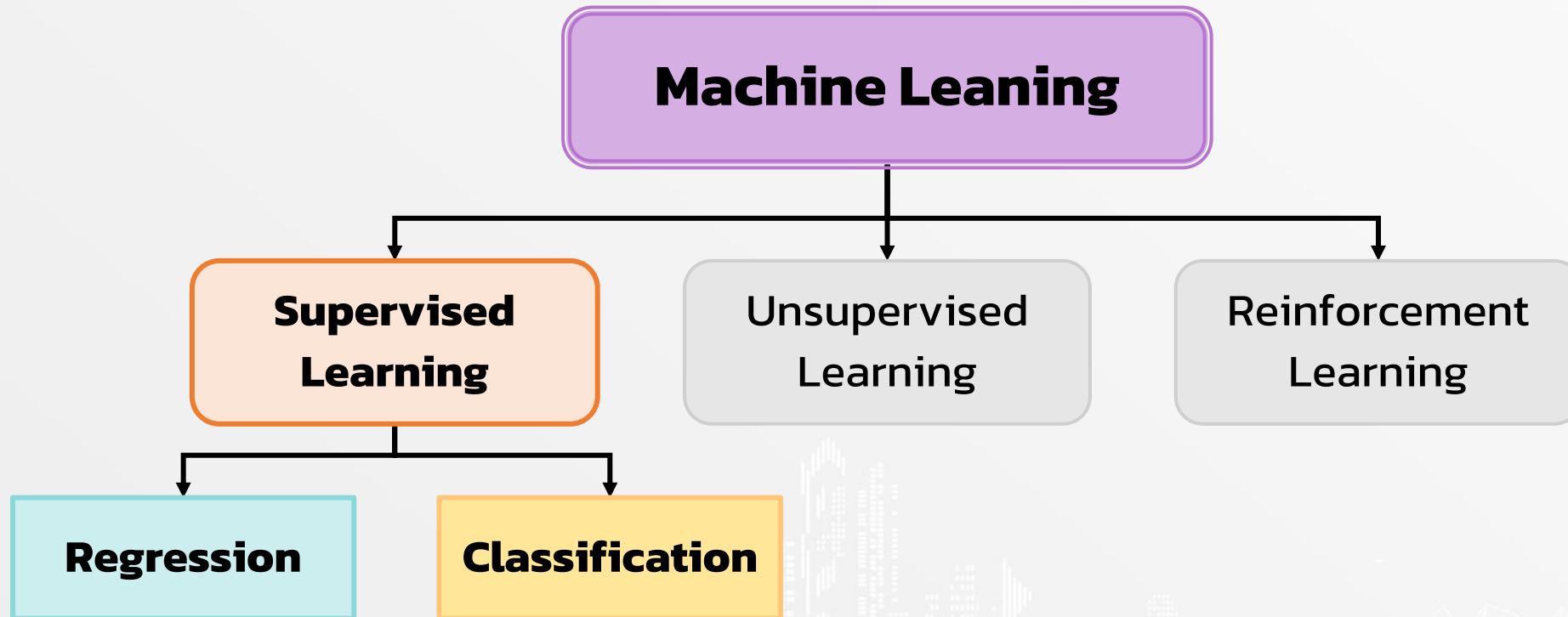


# What is Deep Learning?

**Deep Learning** คือ Neural Network ที่มีการเพิ่มจำนวน **hidden layer** เข้าไป เพื่อเพิ่มประสิทธิภาพการทำงานในด้าน **computational cost** และยังคงรักษาความสามารถในการประมาณ **nonlinear function** ที่ซับซ้อนไว้ด้วย



# What is Deep Learning?



# Deep Learning

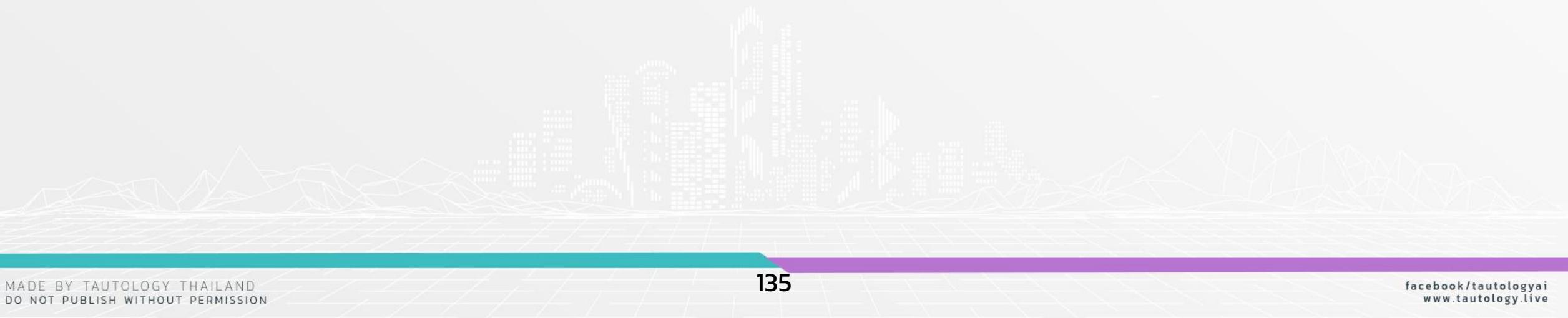


# Architecture of Deep Learning

# Architecture of Deep Learning

**Regression**

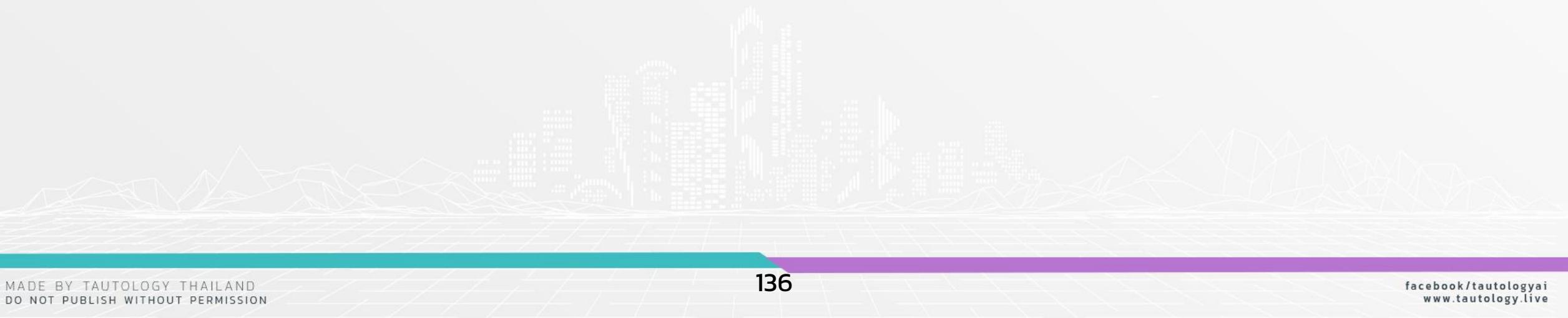
**Classification**



# Architecture of Deep Learning

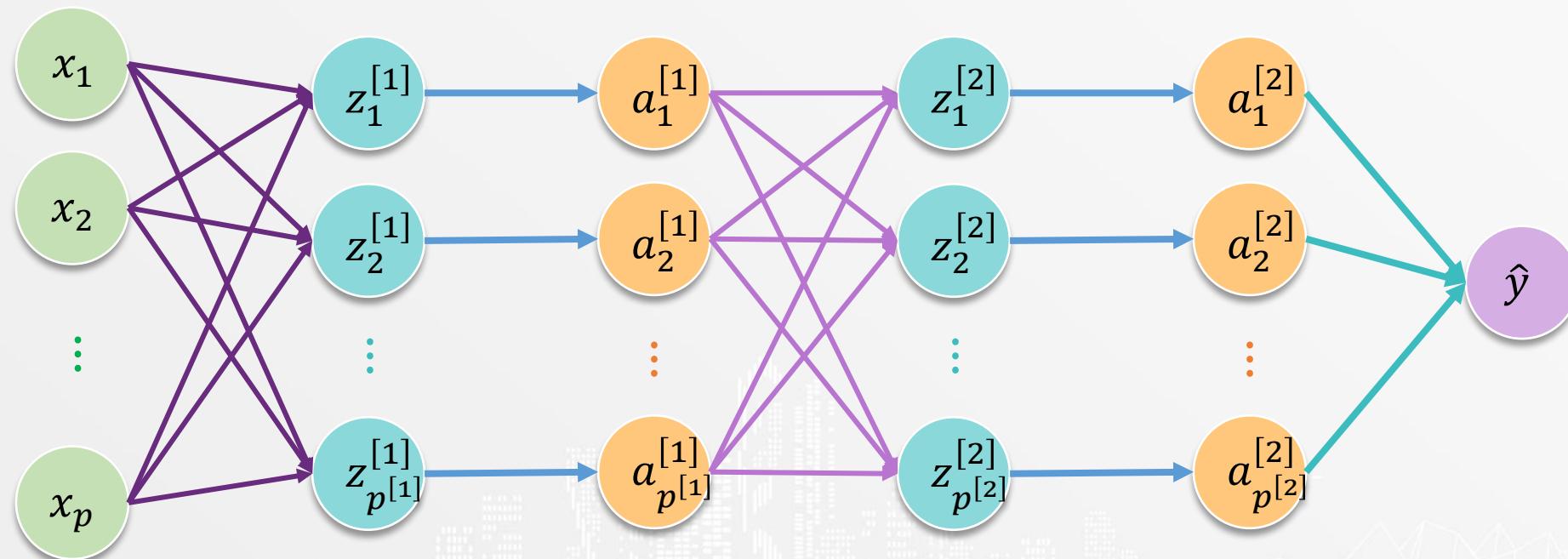
**Regression**

**Classification**



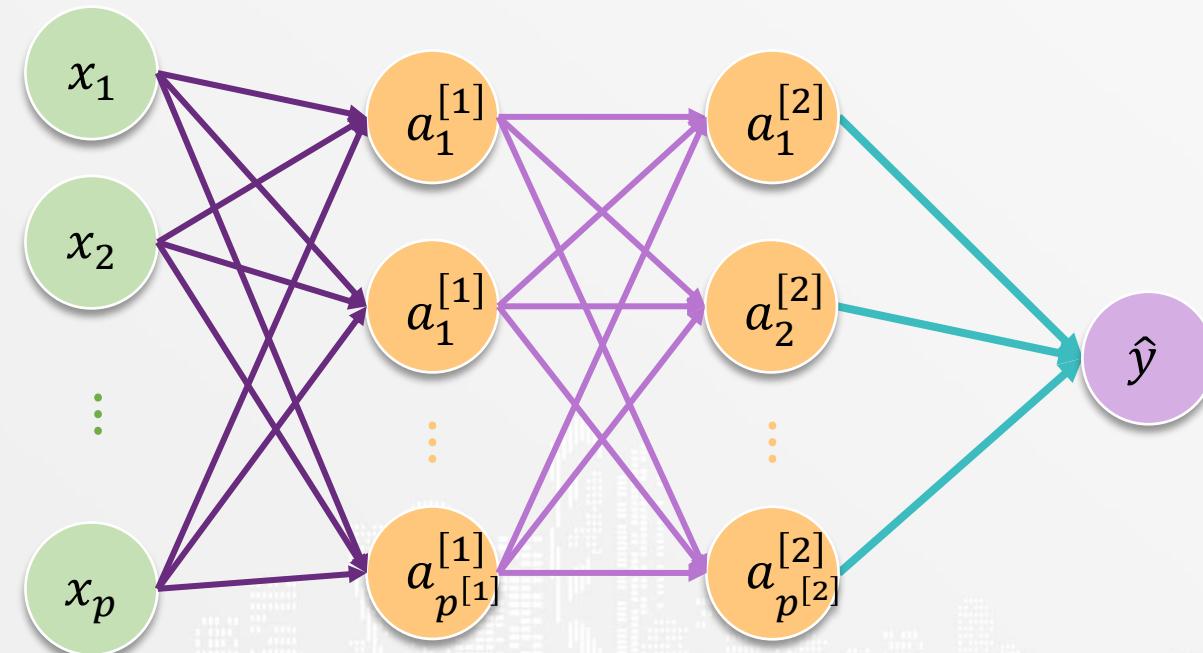
# Architecture of Deep Learning

Regression



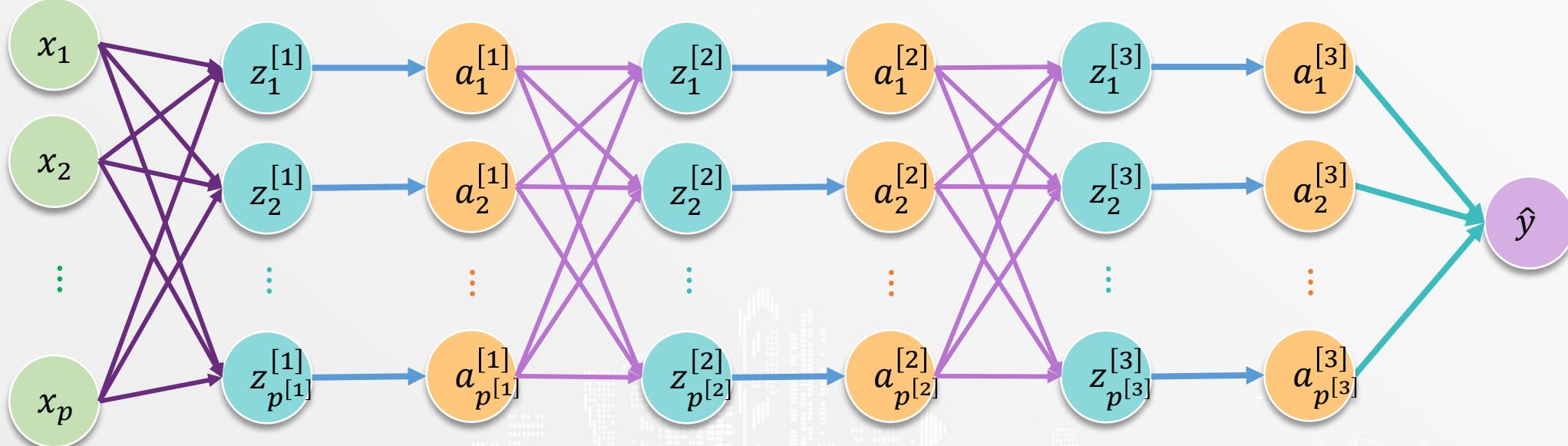
# Architecture of Deep Learning

Regression



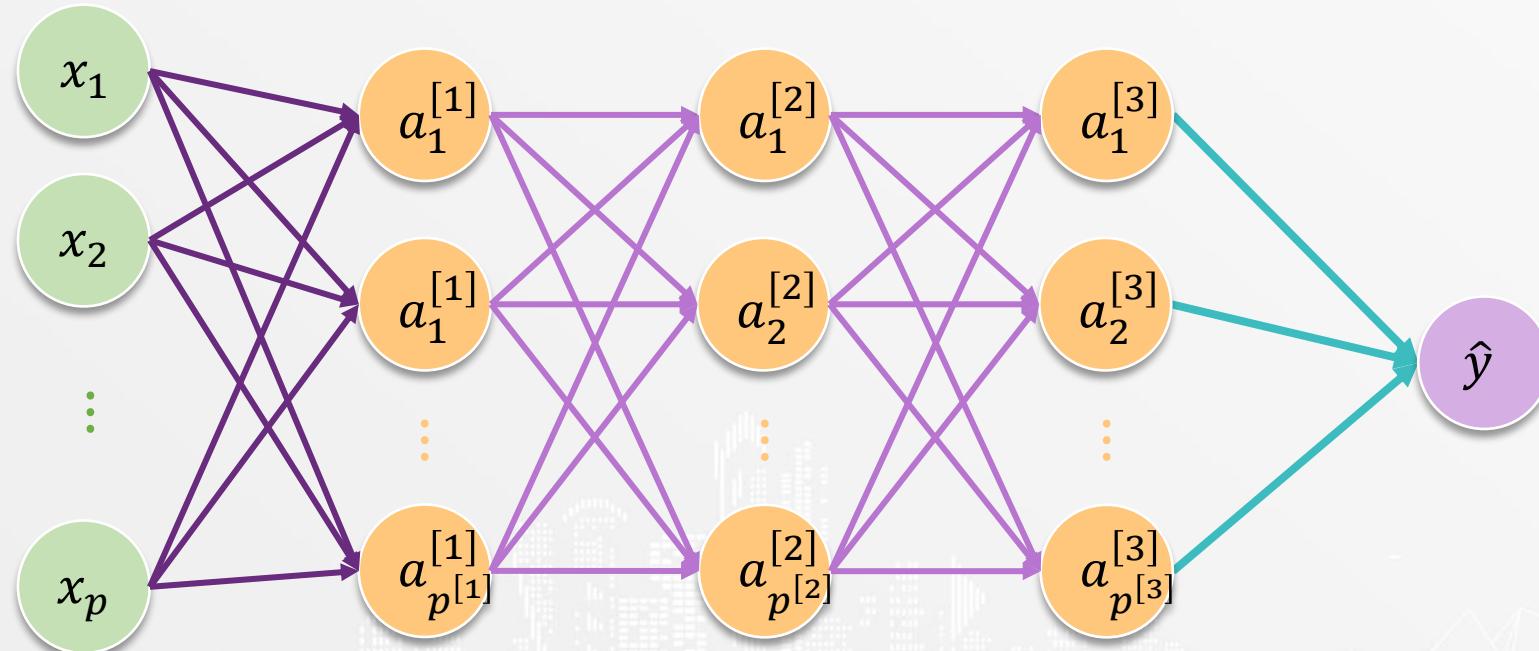
# Architecture of Deep Learning

Regression



# Architecture of Deep Learning

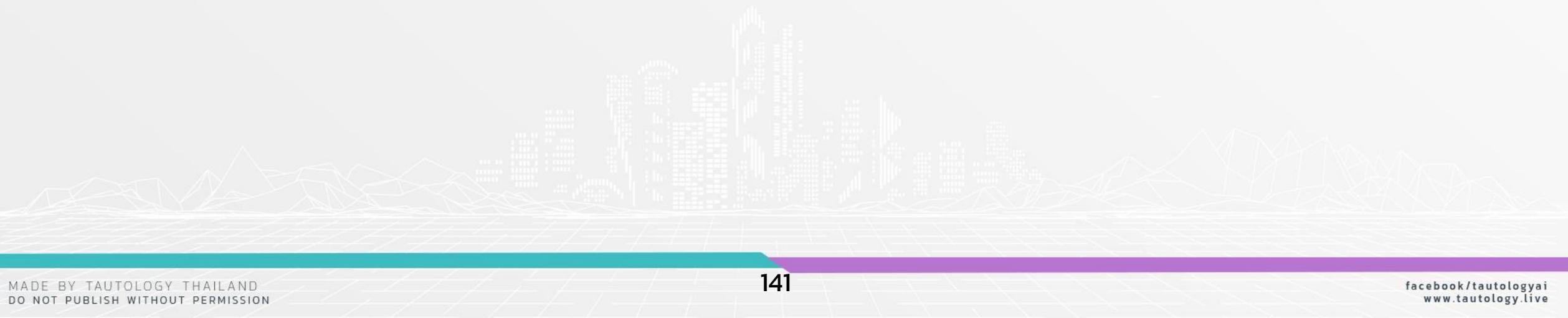
Regression



# Architecture of Deep Learning

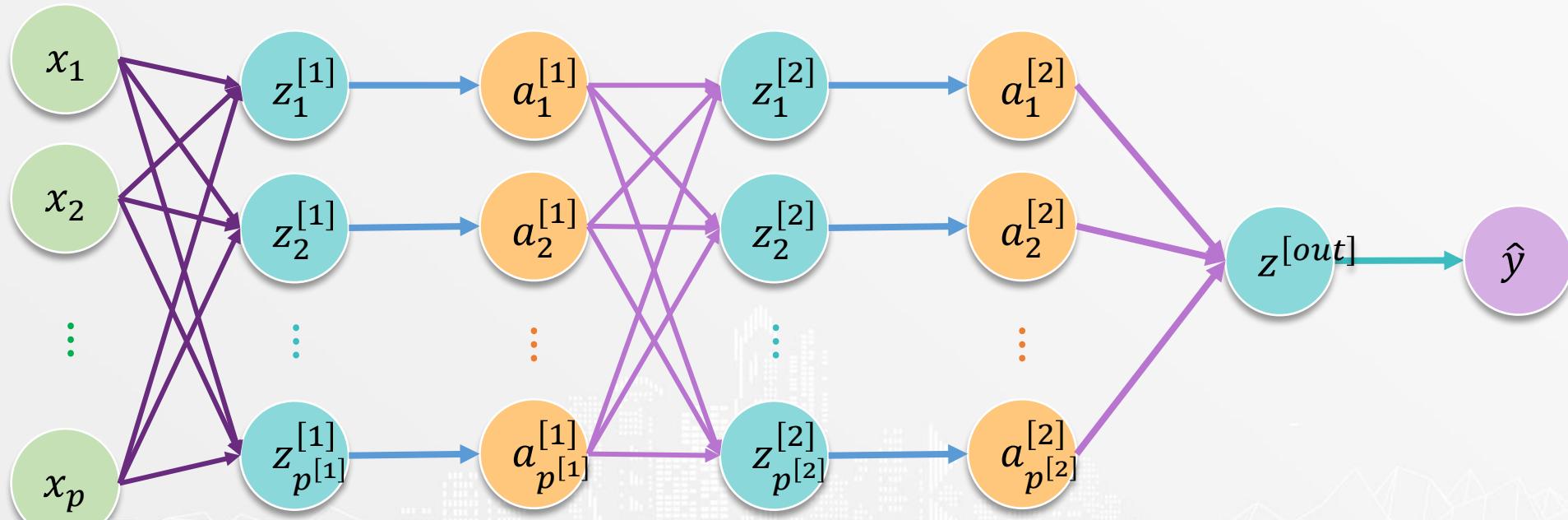
Regression

Classification



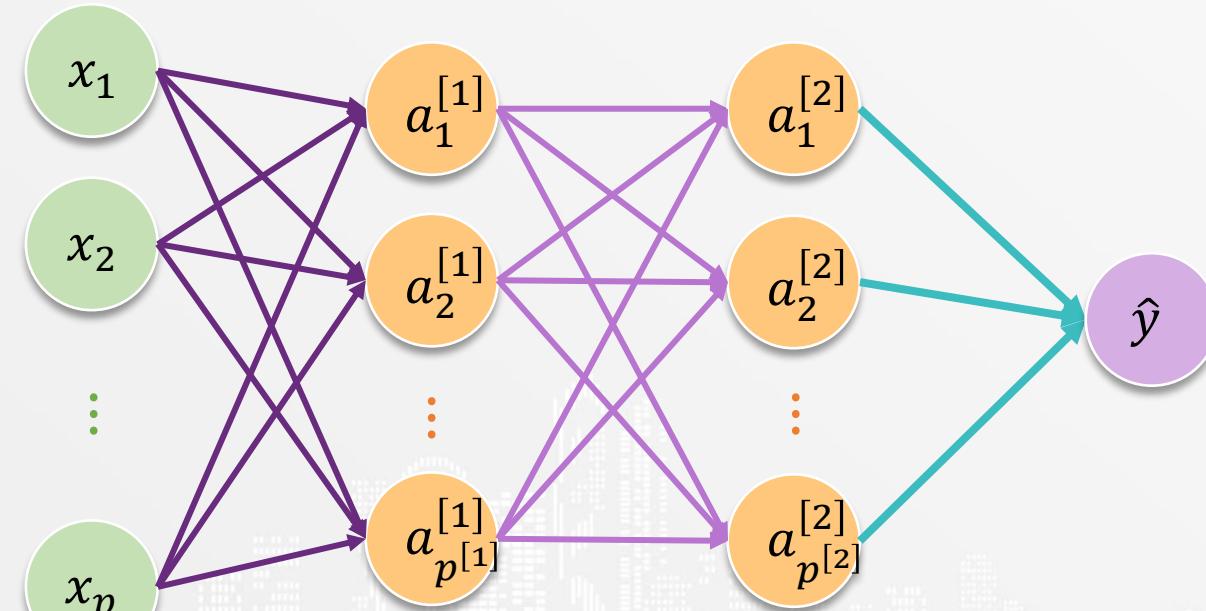
# Architecture of Deep Learning

## Binary Classification



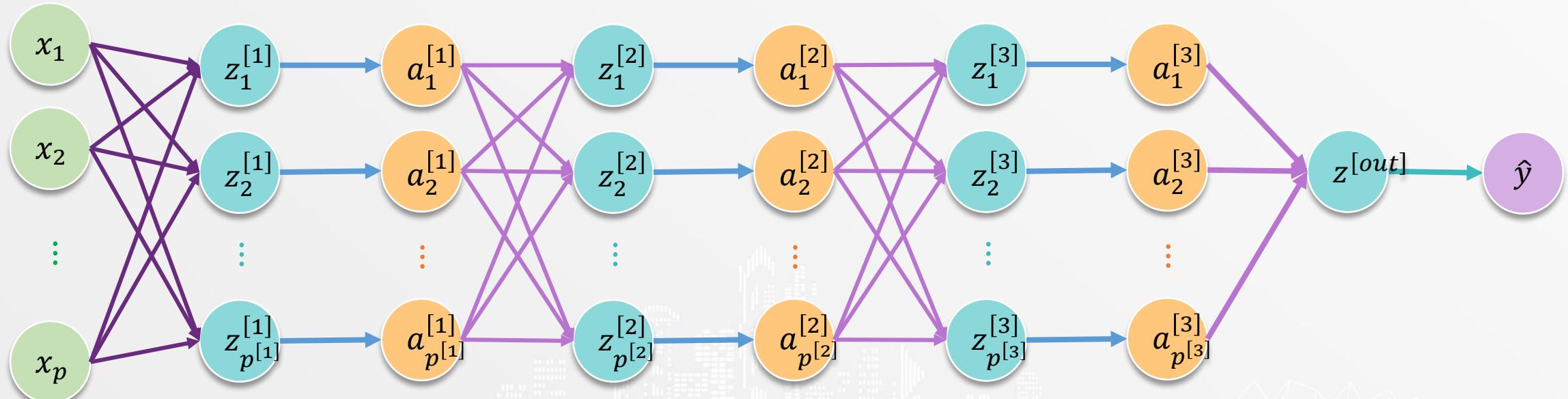
# Architecture of Deep Learning

## Binary Classification



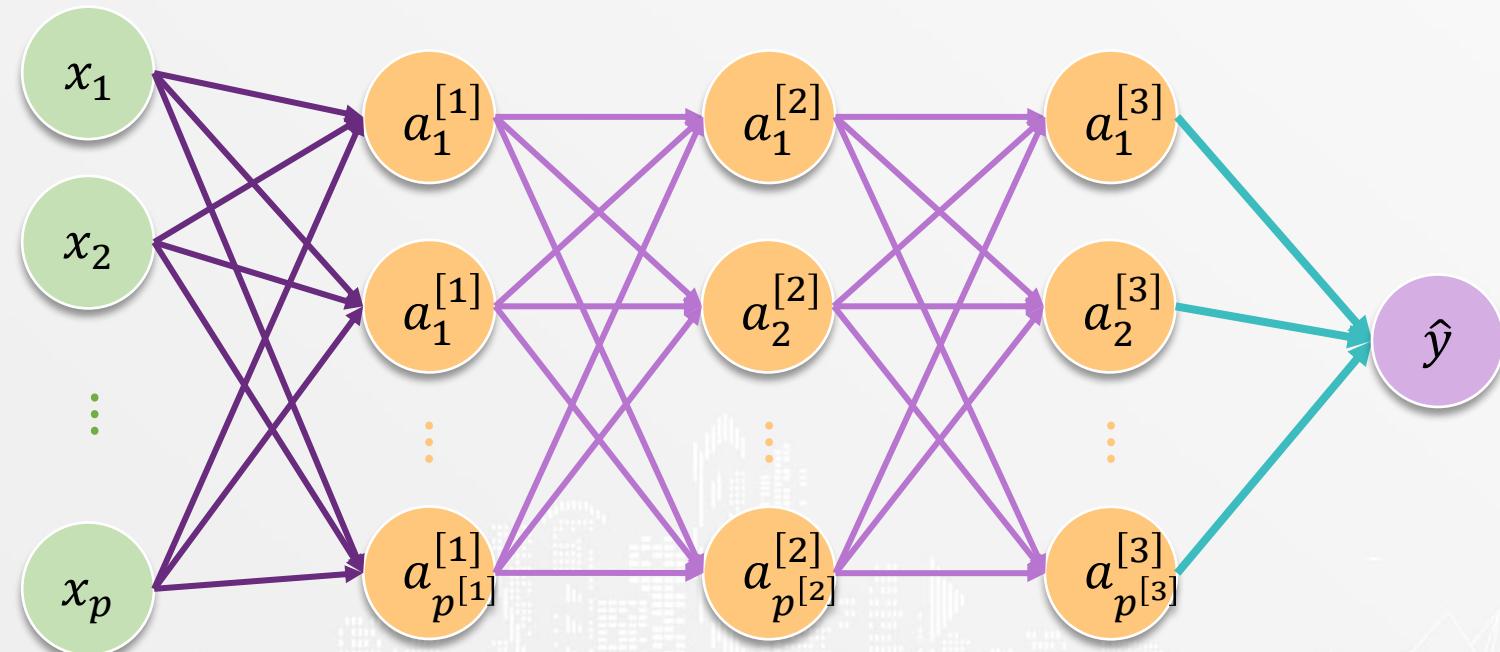
# Architecture of Deep Learning

## Binary Classification



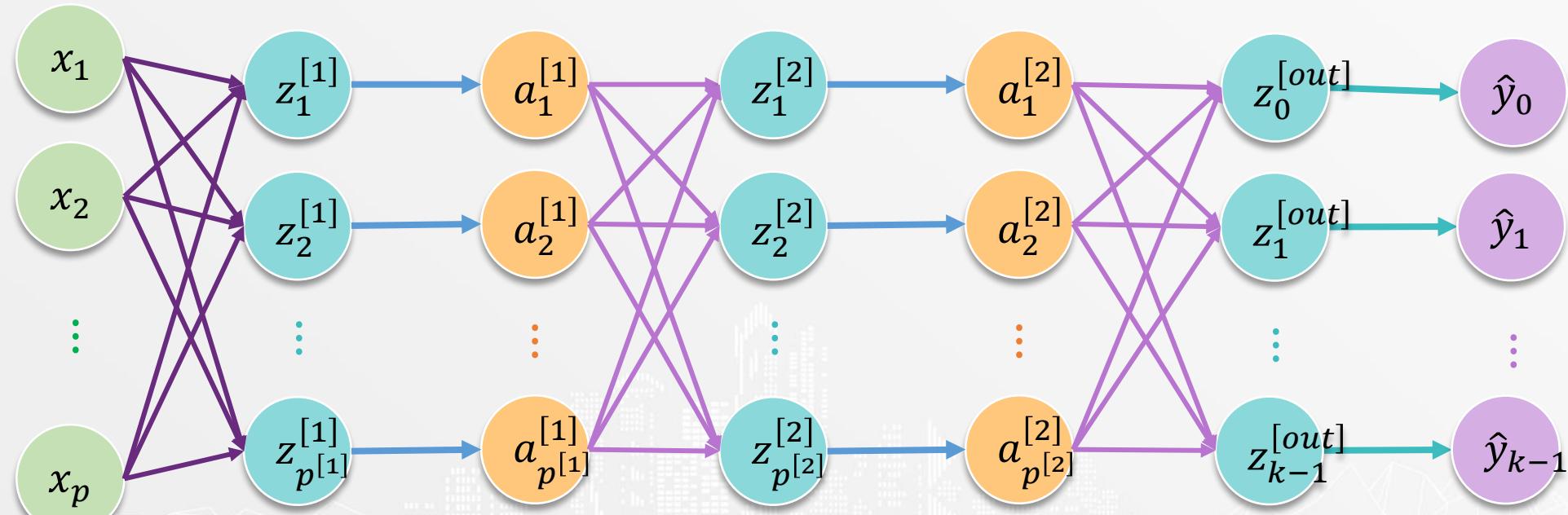
# Architecture of Deep Learning

## Binary Classification



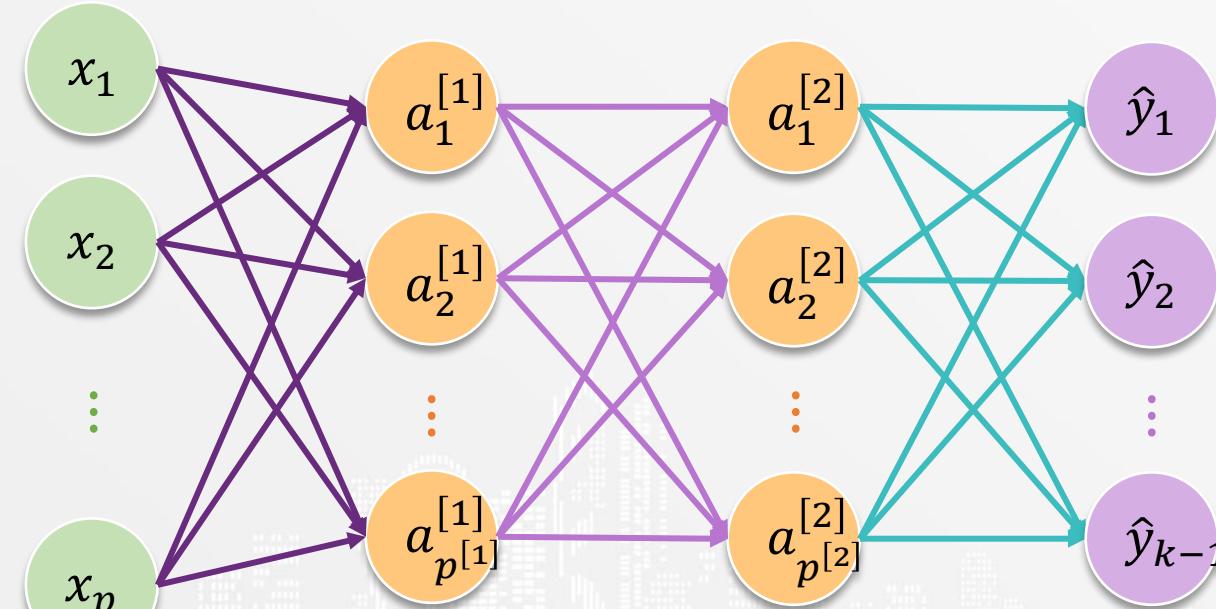
# Architecture of Deep Learning

## Multi-Class Classification



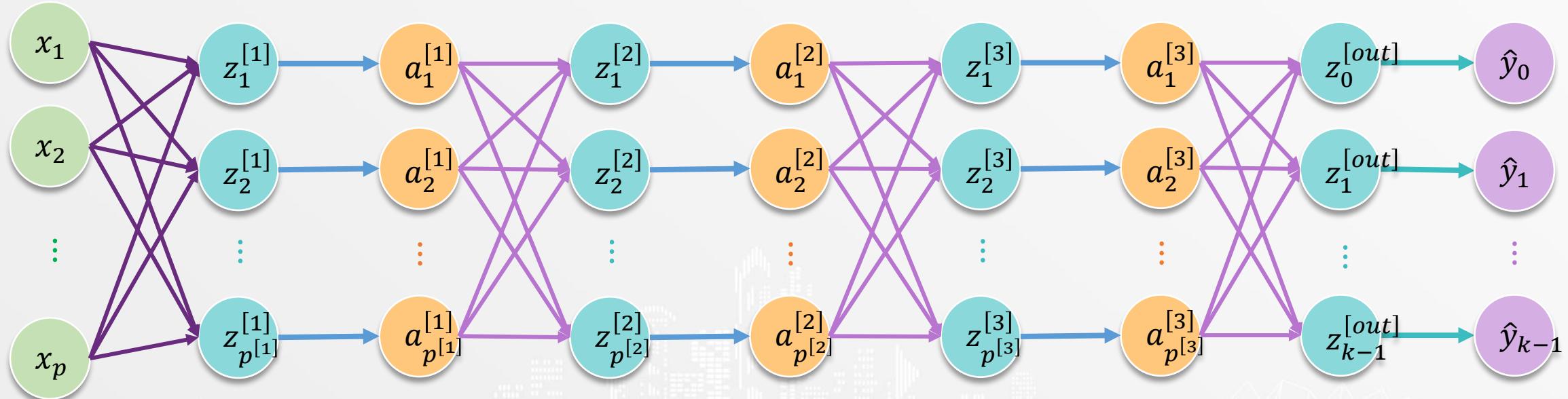
# Architecture of Deep Learning

## Multi-Class Classification



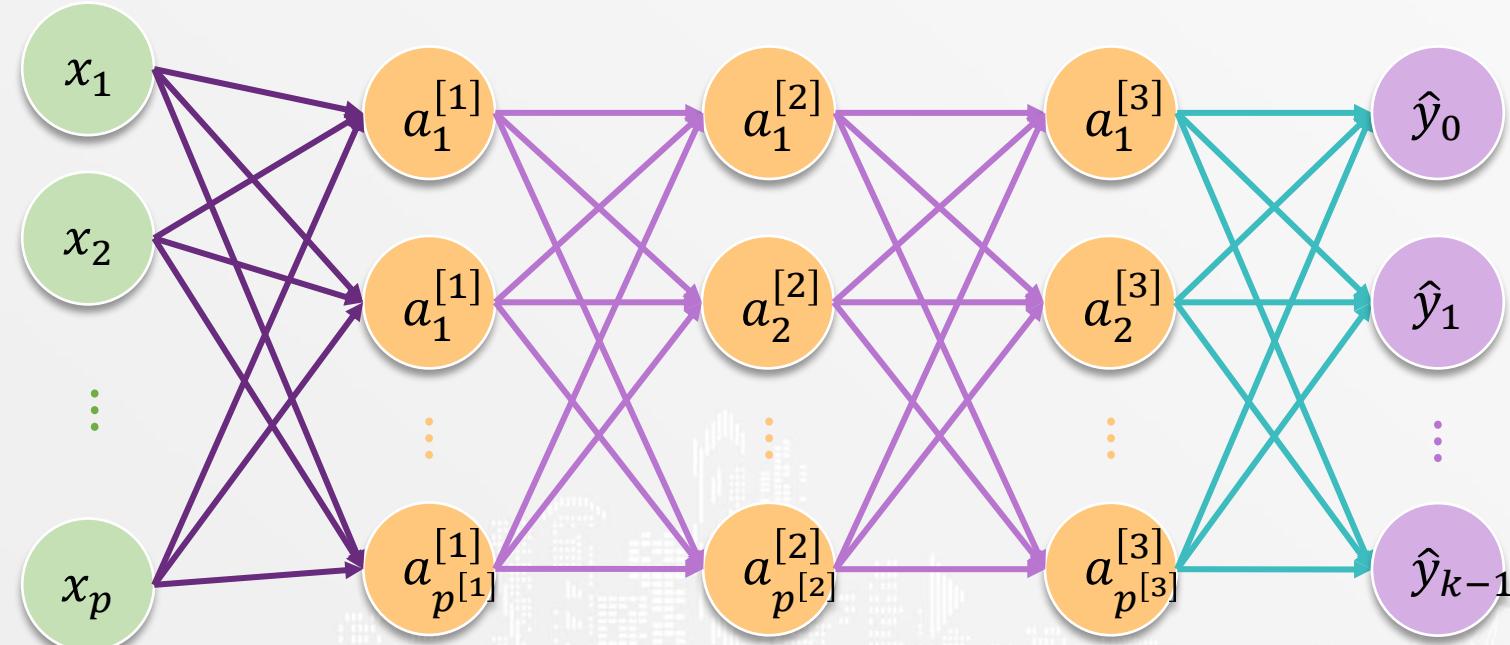
# Architecture of Deep Learning

## Multi-Class Classification



# Architecture of Deep Learning

## Multi-Class Classification



# Deep Learning



# Component of Deep Learning

# Component of Neural Network

Hidden Node

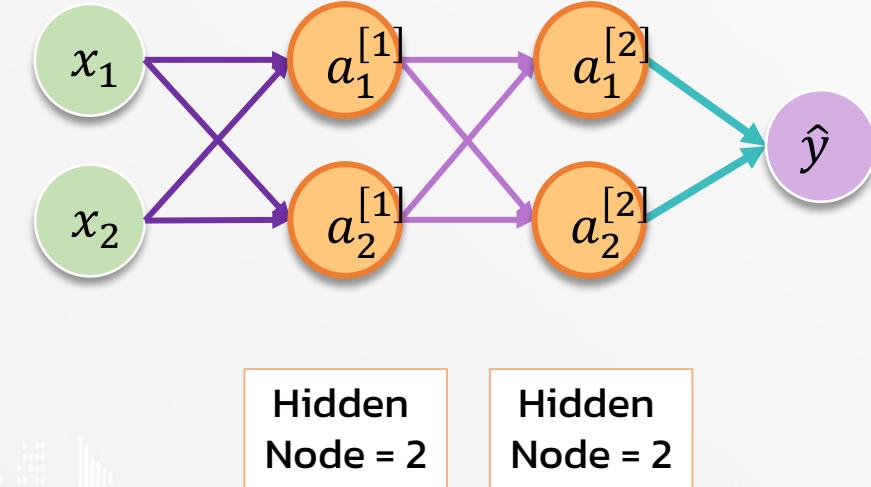
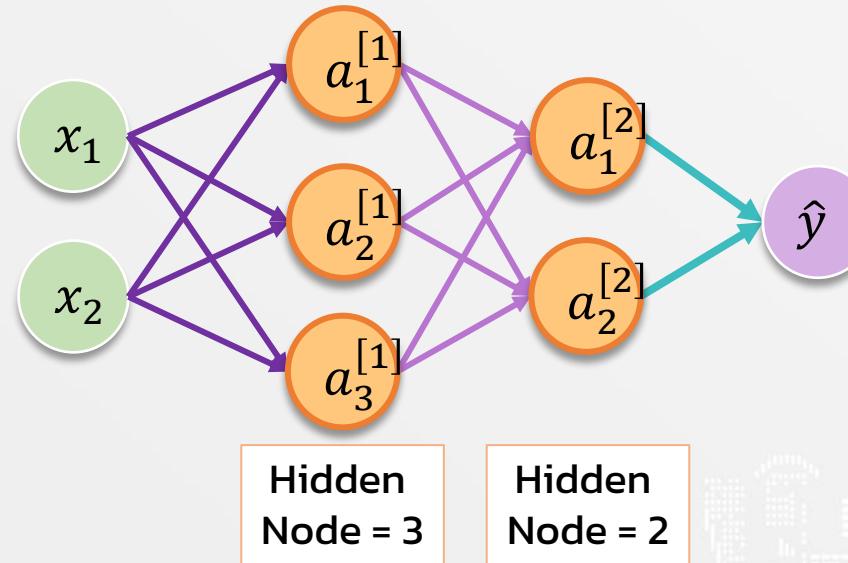
Hidden Layer

Weight & Bias

Activation  
Function

# Hidden Node

**Hidden Node** คือ จำนวนของ nonlinear function ที่ใช้



# Component of Neural Network

**Hidden Node**



**Hidden Layer**



**Weight & Bias**

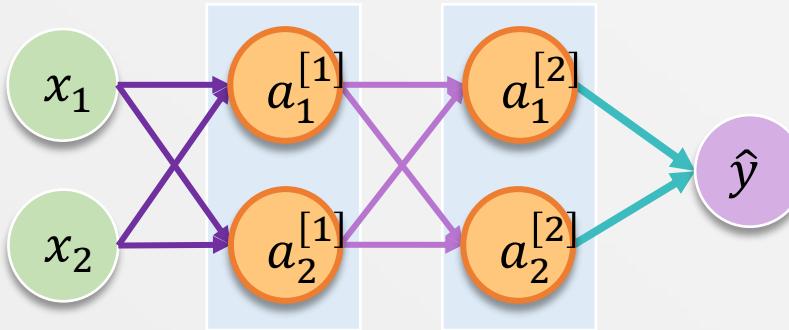


**Activation  
Function**

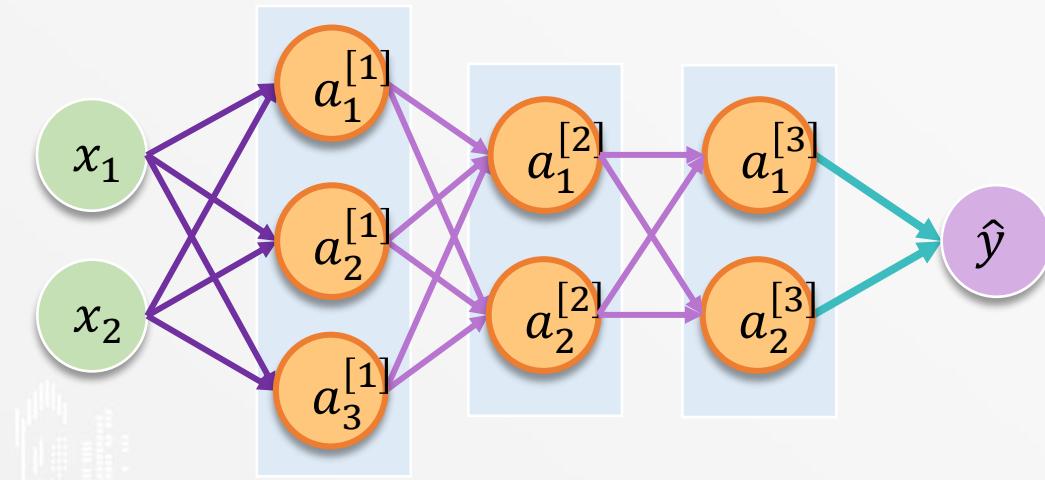


# Hidden Layer

**Hidden Layer** คือ ชั้นที่เก็บ hidden node



จำนวน Hidden Layer = 2



จำนวน Hidden Layer = 3

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



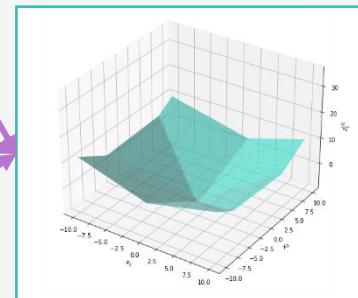
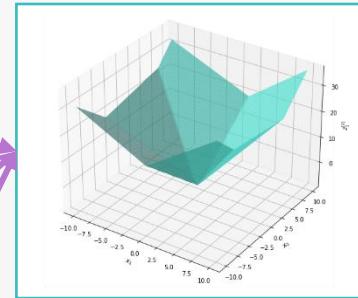
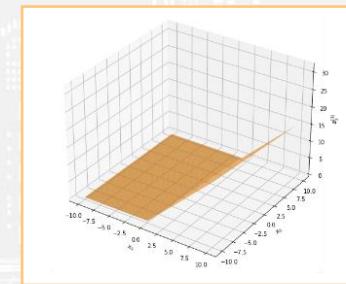
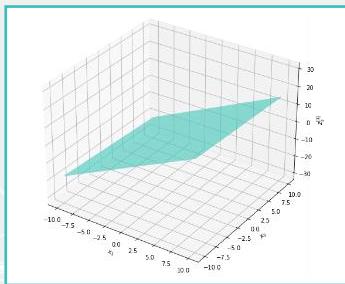
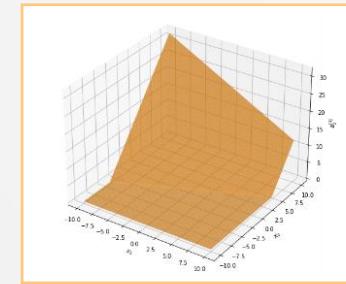
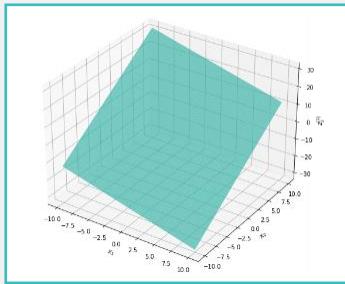
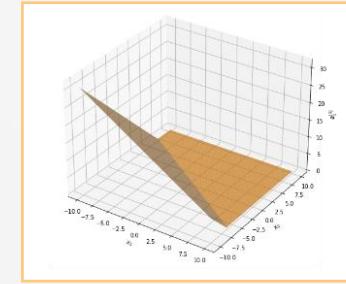
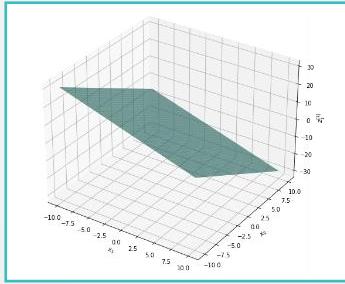
**Weight & Bias**



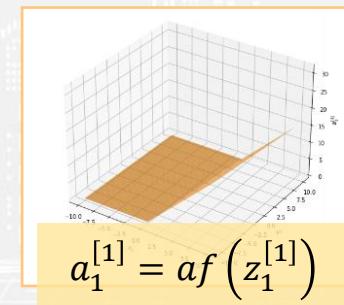
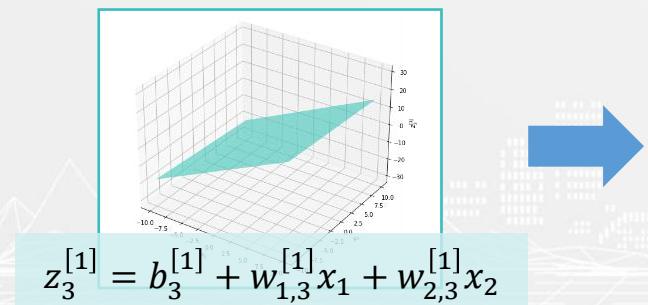
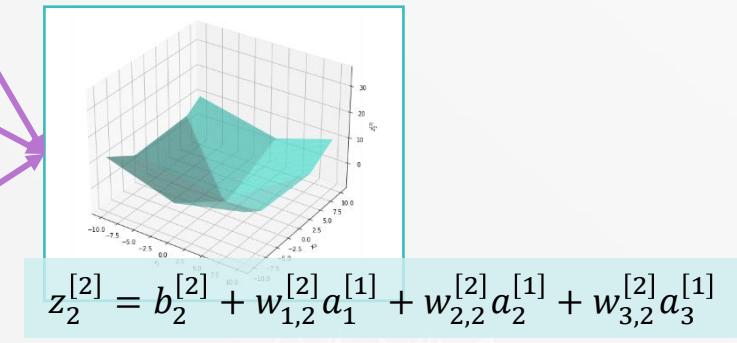
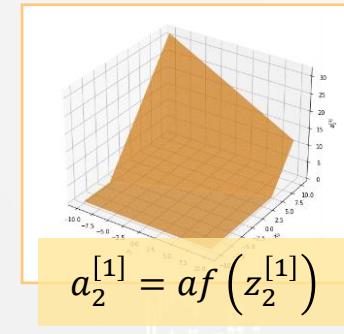
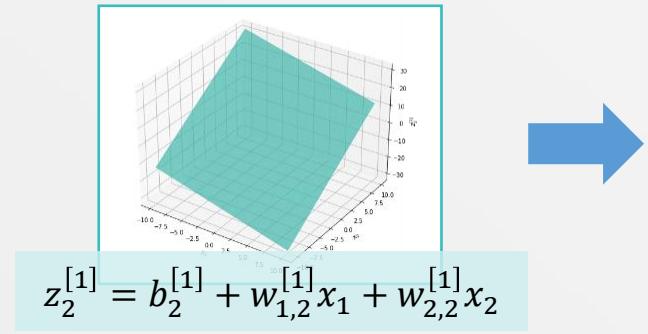
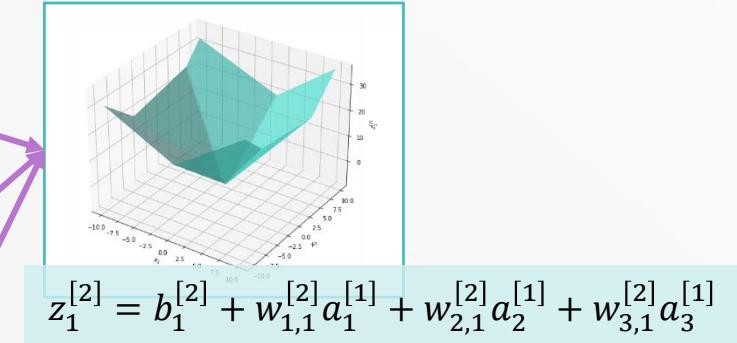
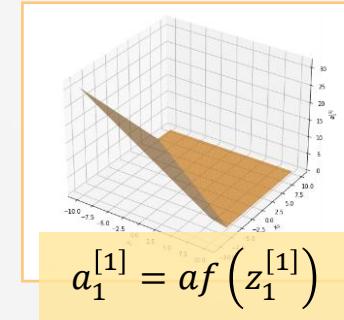
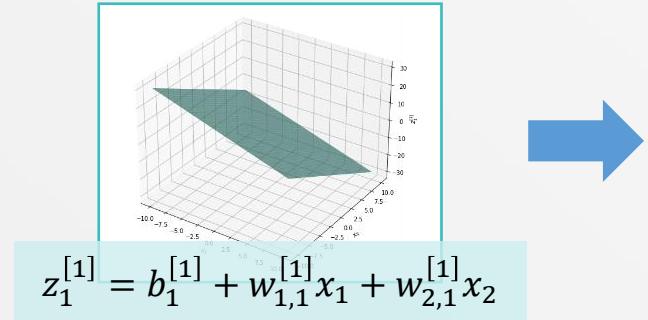
**Activation  
Function**



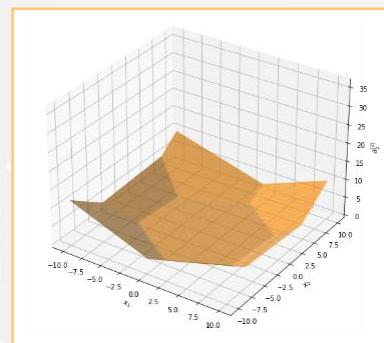
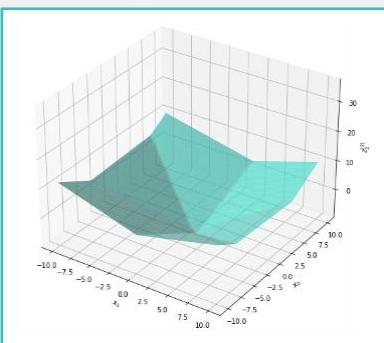
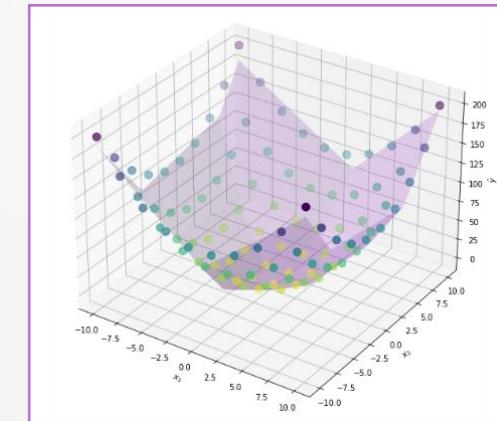
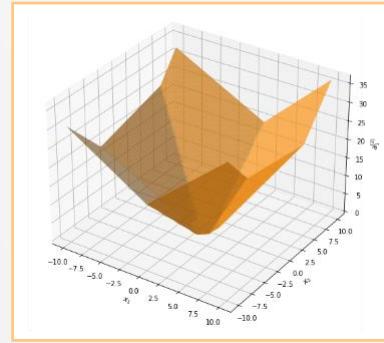
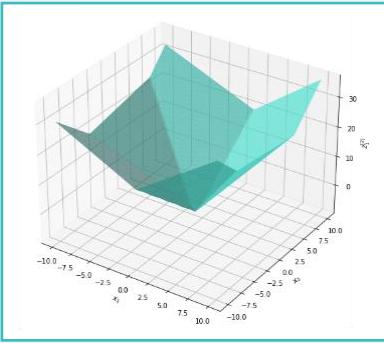
# Weight & Bias



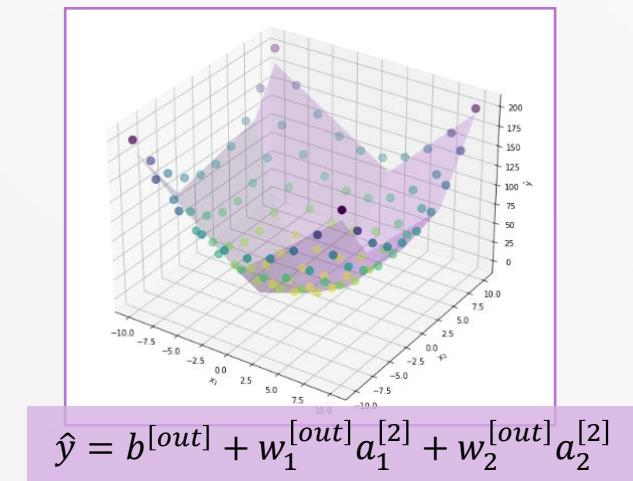
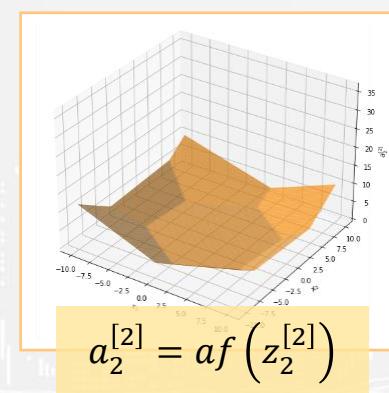
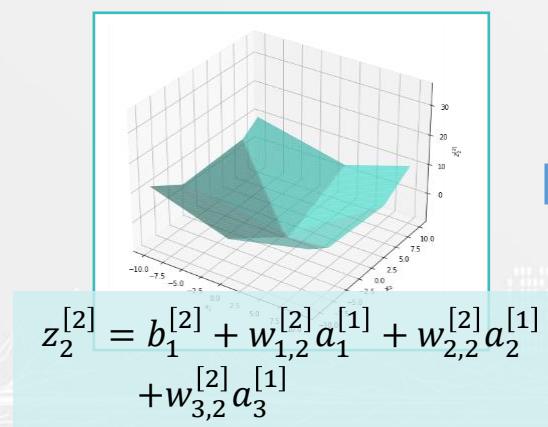
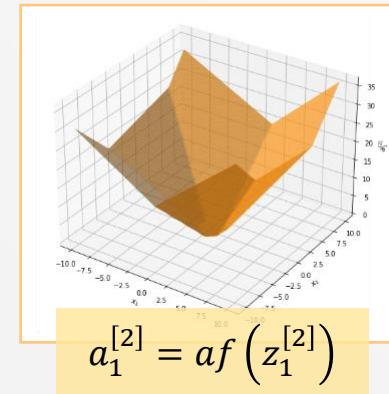
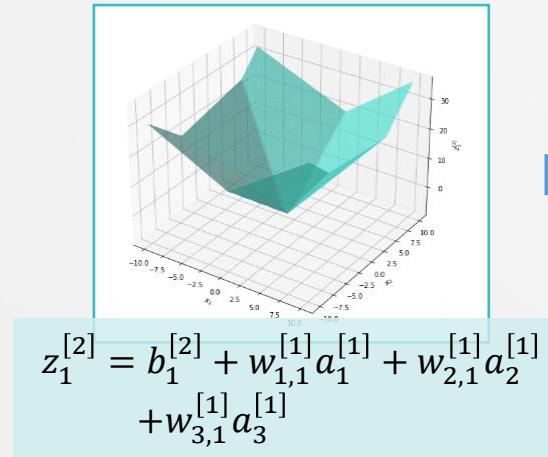
# Weight & Bias



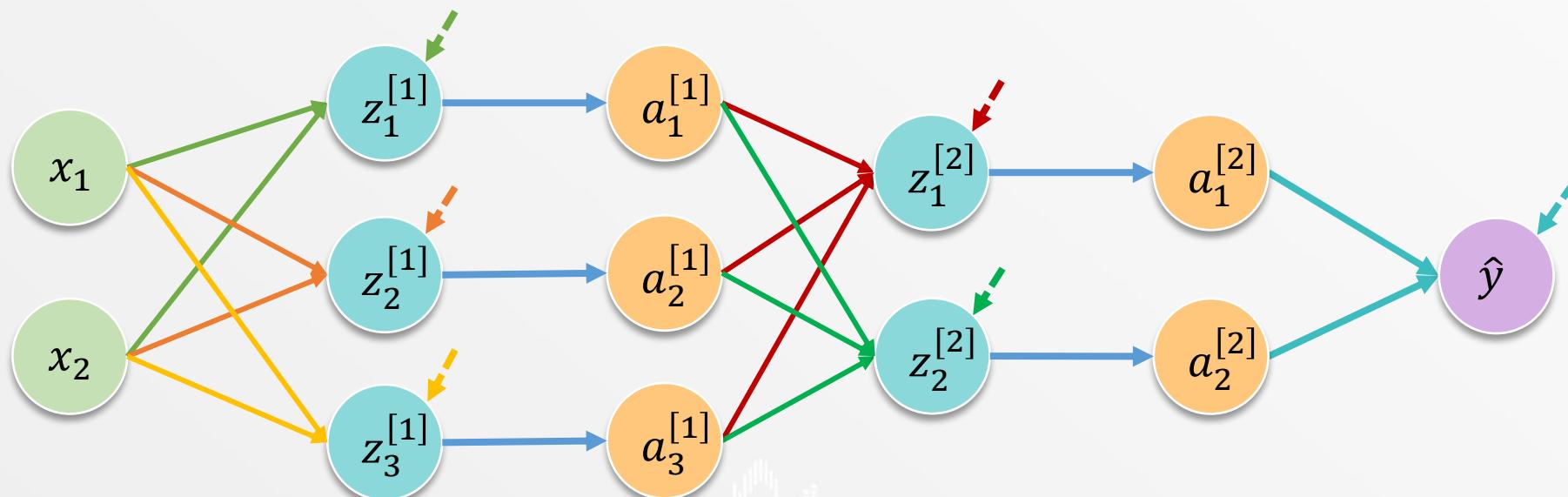
# Weight & Bias



# Weight & Bias



# Weight & Bias



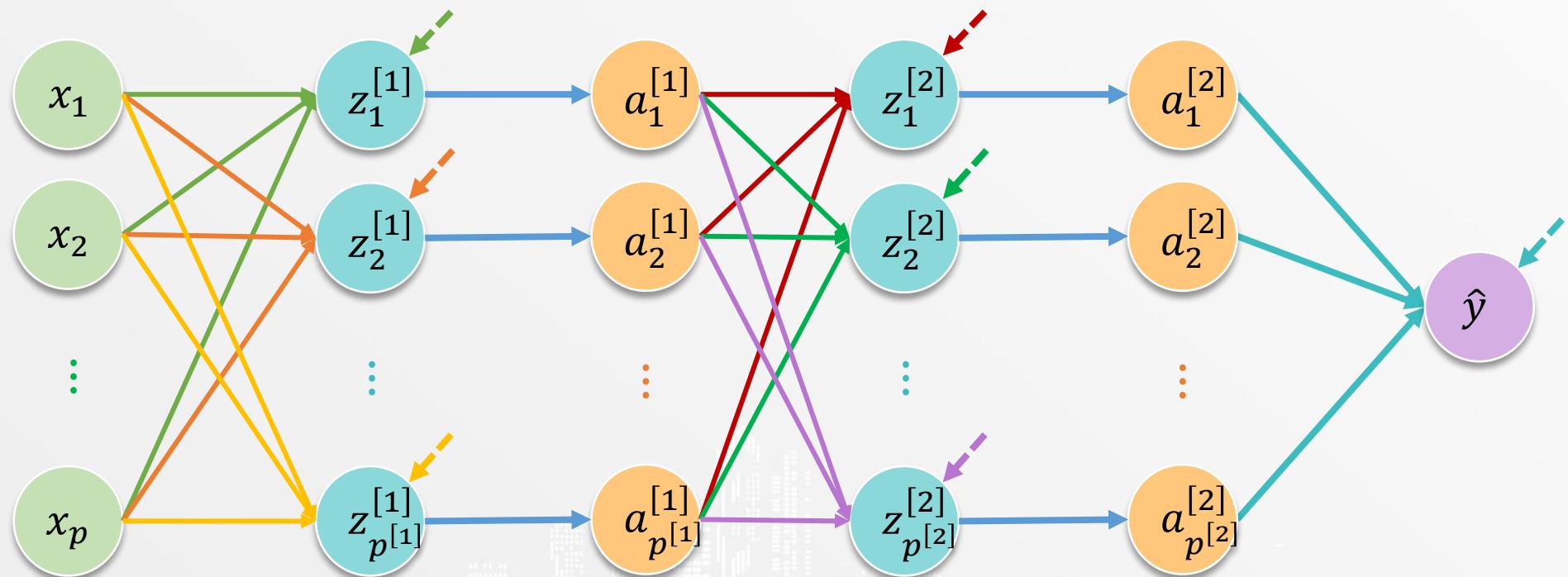
# Weight & Bias

$$\mathbf{b}^{[1]} = [b_1^{[1]} \quad b_2^{[1]} \quad b_3^{[1]}], \quad W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & w_{1,3}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & w_{2,3}^{[1]} \end{bmatrix}$$

$$\mathbf{b}^{[2]} = [b_1^{[2]} \quad b_2^{[2]}], \quad W^{[2]} = \begin{bmatrix} w_{1,1}^{[2]} & w_{1,2}^{[2]} \\ w_{2,1}^{[2]} & w_{2,2}^{[2]} \\ w_{3,1}^{[2]} & w_{3,2}^{[2]} \end{bmatrix}$$

$$\mathbf{b}^{[out]} = [b^{[out]}], \quad W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \end{bmatrix}$$

# Weight & Bias



# Weight & Bias

$$z_1^{[1]} = b_1^{[1]} + w_{1,1}^{[1]}x_1 + w_{2,1}^{[1]}x_2 + \cdots + w_{p,1}^{[1]}x_p$$

$$z_2^{[1]} = b_2^{[1]} + w_{1,2}^{[1]}x_1 + w_{2,2}^{[1]}x_2 + \cdots + w_{p,2}^{[1]}x_p$$

⋮

$$z_{p^{[1]}}^{[1]} = b_{p^{[1]}}^{[1]} + w_{1,p^{[1]}}^{[1]}x_1 + w_{2,p^{[1]}}^{[1]}x_2 + \cdots + w_{p,p^{[1]}}^{[1]}x_p$$

# Weight & Bias

$$a_1^{[1]} = af(z_1^{[1]})$$

$$a_2^{[1]} = af(z_2^{[1]})$$

⋮

$$a_{p^{[1]}}^{[1]} = af(z_{p^{[1]}}^{[1]})$$

# Weight & Bias

$$\mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & \dots & b_{p^{[1]}}^{[1]} \end{bmatrix}, \quad W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & \dots & w_{1,p^{[1]}}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & \dots & w_{2,p^{[1]}}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[1]} & w_{p,2}^{[1]} & \dots & w_{p,p^{[1]}}^{[1]} \end{bmatrix}$$

# Weight & Bias

$$z_1^{[2]} = b_1^{[2]} + w_{1,1}^{[2]} a_1^{[1]} + w_{2,1}^{[2]} a_2^{[1]} + \dots + w_{p^{[1]},1}^{[2]} a_{p^{[1]}}^{[1]}$$

$$z_2^{[2]} = b_2^{[2]} + w_{1,2}^{[2]} a_1^{[1]} + w_{2,2}^{[2]} a_2^{[1]} + \dots + w_{p^{[1]},2}^{[2]} a_{p^{[1]}}^{[1]}$$

⋮

$$z_{p^{[2]}}^{[2]} = b_{p^{[2]}}^{[2]} + w_{1,p^{[2]}}^{[2]} a_1^{[1]} + w_{2,p^{[2]}}^{[2]} a_2^{[1]} + \dots + w_{p^{[1]},p^{[2]}}^{[2]} a_{p^{[1]}}^{[1]}$$

# Weight & Bias

$$a_1^{[2]} = af(z_1^{[2]})$$

$$a_2^{[2]} = af(z_2^{[2]})$$

⋮

$$a_{p^{[2]}}^{[2]} = af(z_{p^{[2]}}^{[2]})$$

$$\hat{y} = b^{[out]} + w_1^{[out]} a_1^{[2]} + w_2^{[out]} a_2^{[2]} + \dots + w_{p^{[2]}}^{[out]} a_{p^{[2]}}^{[2]}$$

# Weight & Bias

$$\mathbf{b}^{[2]} = [b_1^{[2]} \quad b_2^{[2]} \quad \dots \quad b_{p^{[2]}}^{[2]}], \quad W^{[2]} = \begin{bmatrix} w_{1,1}^{[2]} & w_{1,2}^{[2]} & \dots & w_{1,p^{[2]}}^{[2]} \\ w_{2,1}^{[2]} & w_{2,2}^{[2]} & \dots & w_{2,p^{[2]}}^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p^{[1]},1}^{[2]} & w_{p^{[1]},2}^{[2]} & \dots & w_{p^{[1]},p^{[2]}}^{[2]} \end{bmatrix}$$

$$\mathbf{b}^{[out]} = [b^{[out]}], \quad W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \\ \vdots \\ w_{p^{[2]}}^{[out]} \end{bmatrix}$$

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



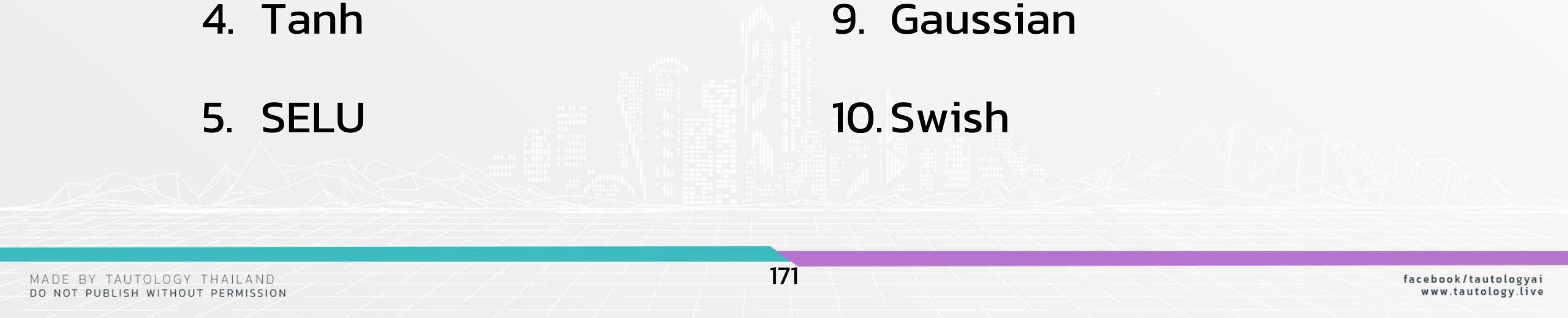
**Weight & Bias**



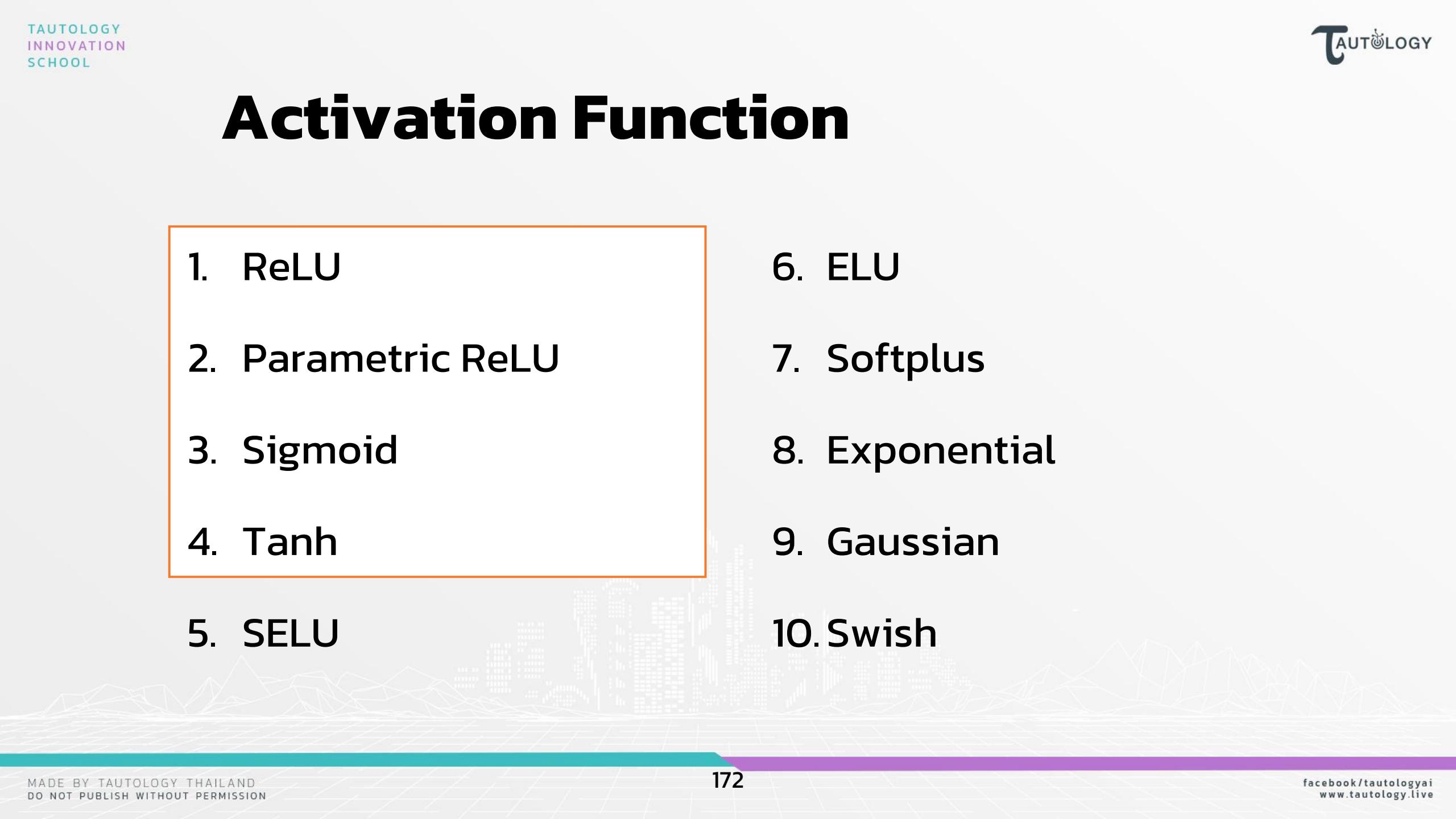
**Activation  
Function**



# Activation Function

- 
- 1. ReLU
  - 2. Parametric ReLU
  - 3. Sigmoid
  - 4. Tanh
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

# Activation Function

- 
- 1. ReLU
  - 2. Parametric ReLU
  - 3. Sigmoid
  - 4. Tanh
  - 5. SELU
  - 6. ELU
  - 7. Softplus
  - 8. Exponential
  - 9. Gaussian
  - 10. Swish

# Component of Neural Network

**Hidden Node**



**Hidden Layer**



**Weight & Bias**



**Activation  
Function**



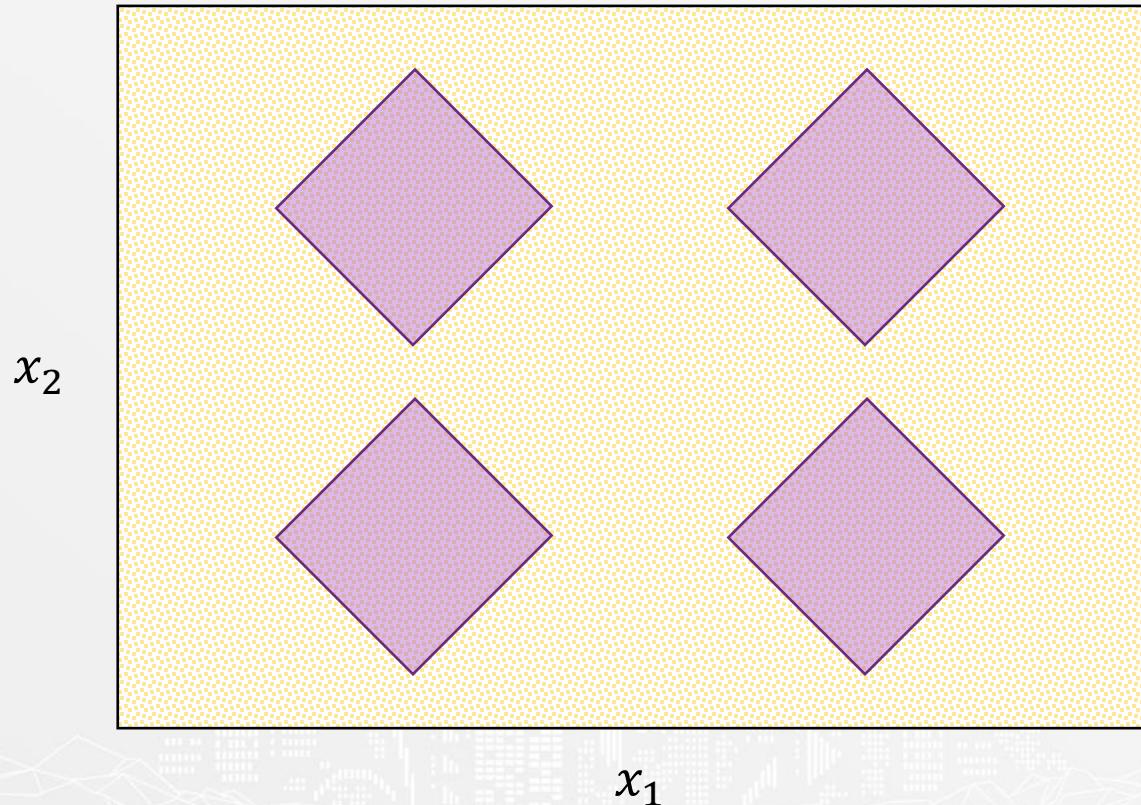
# Deep Learning



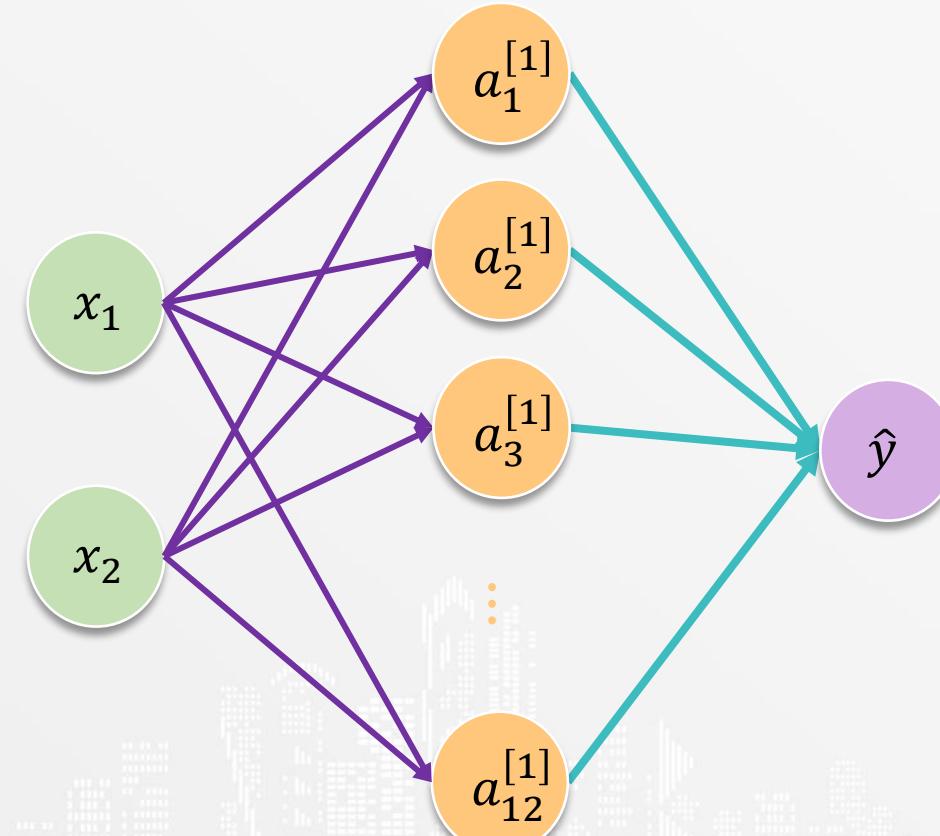


# Why we need Deep Learning?

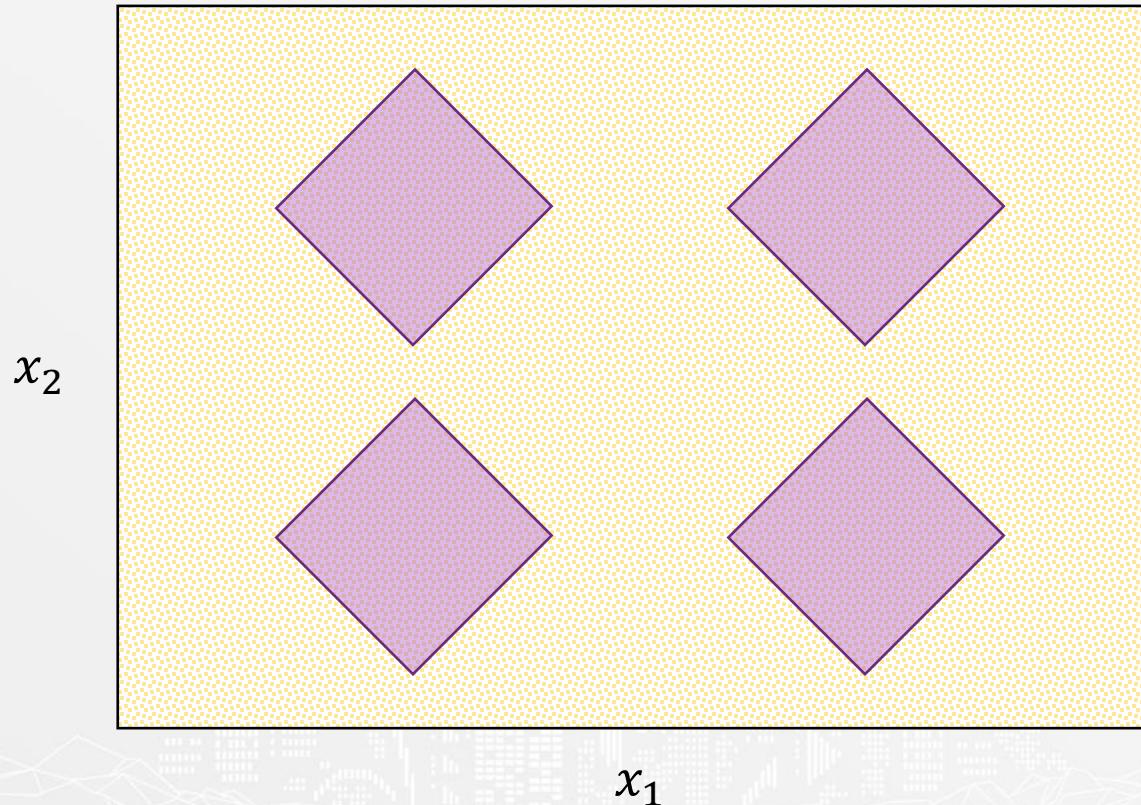
# Why we need Deep Learning?



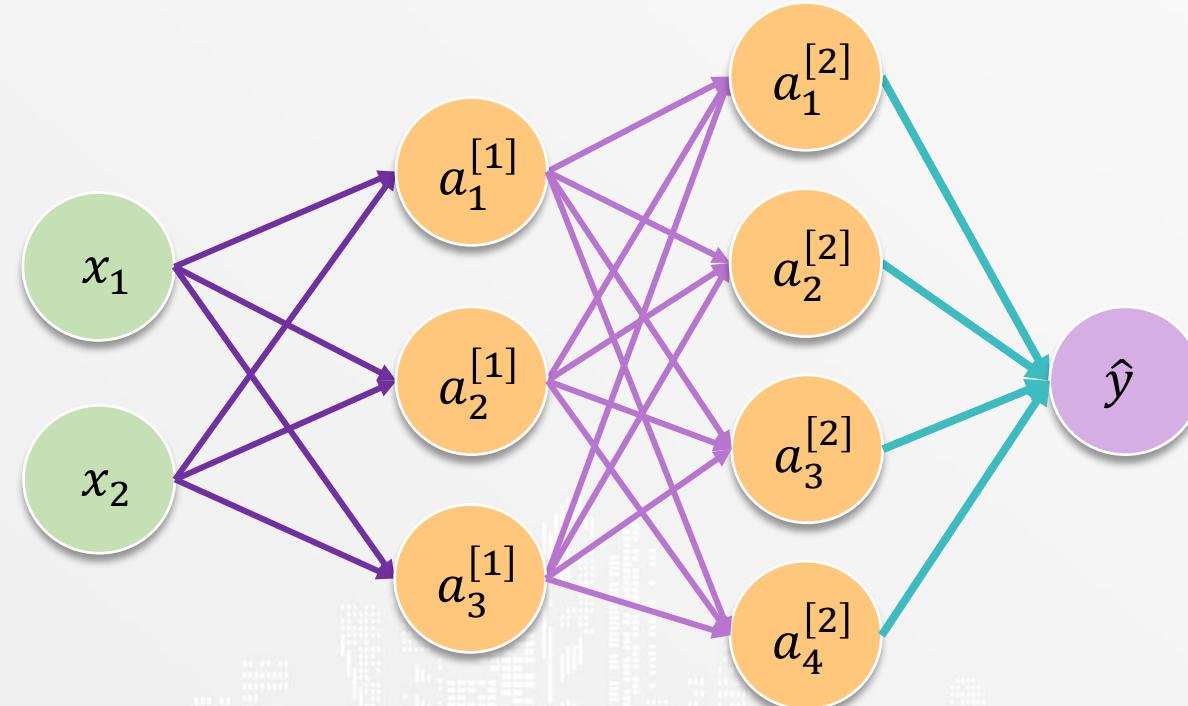
# Why we need Deep Learning?



# Why we need Deep Learning?



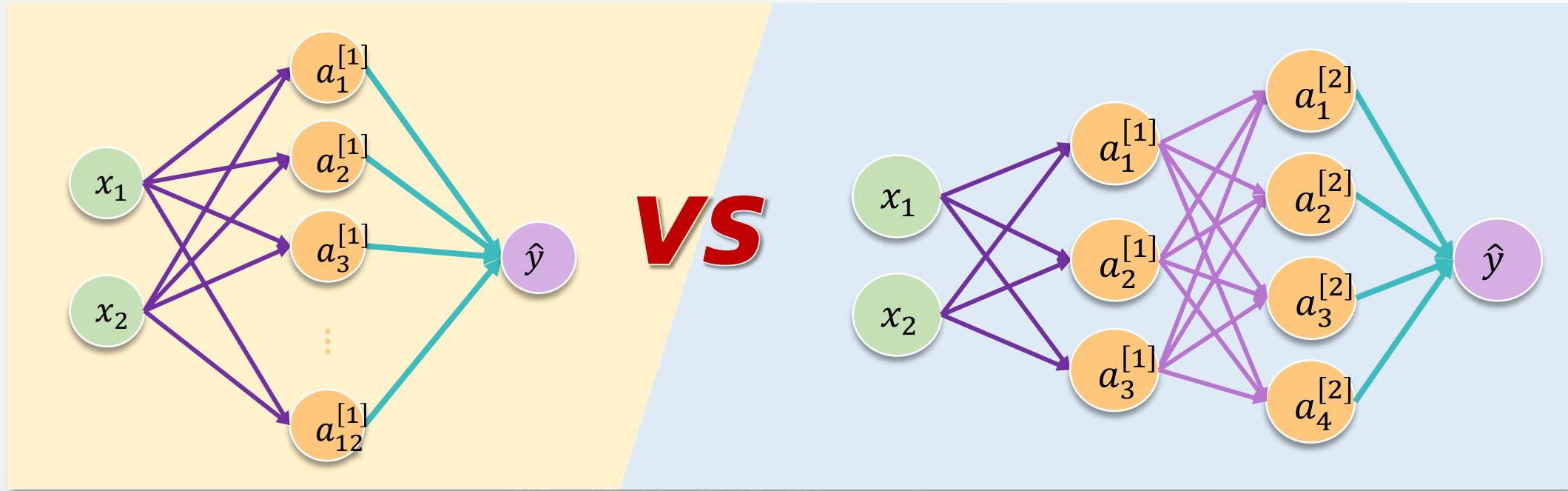
# Why we need Deep Learning?



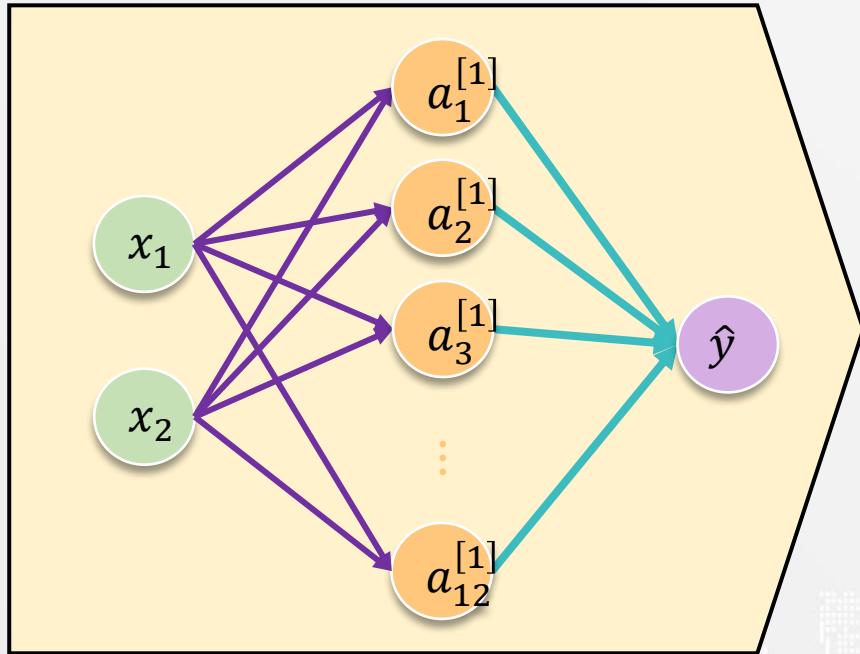
# Why we need Deep Learning?

**Computation cost** ที่ใช้ในการคำนวณ deep learning แปรผันตรงกับจำนวน connection ใน architecture

# Why we need Deep Learning?



# Why we need Deep Learning?

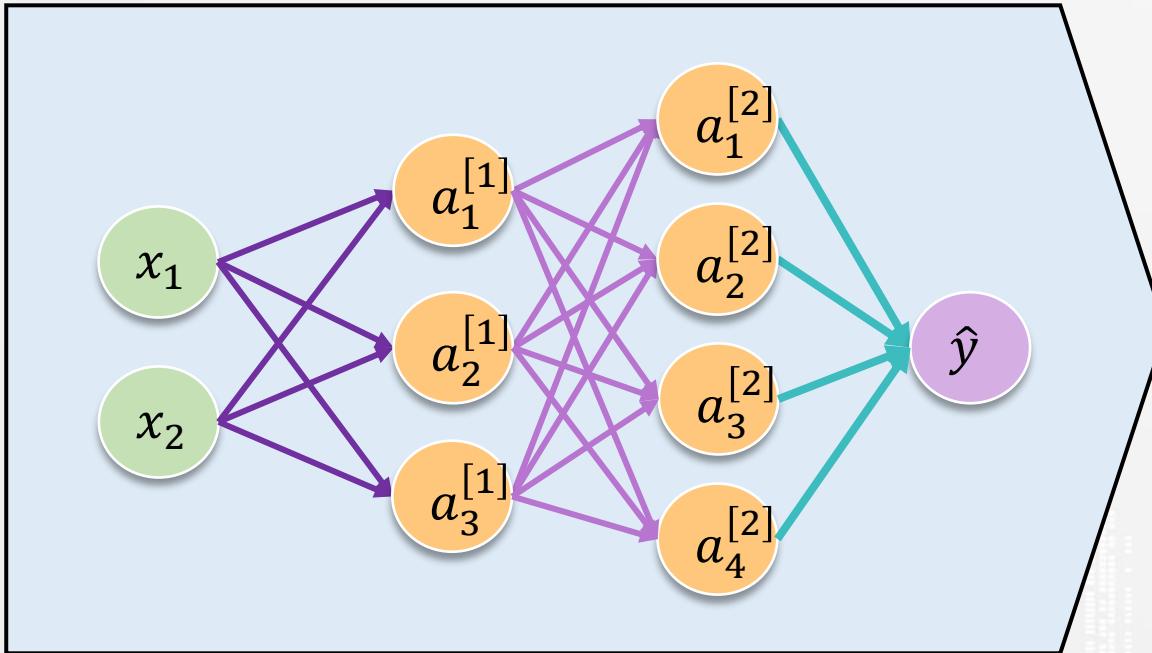


$$\text{Cost} \approx (2 \times 12) + (12 \times 1)$$

$$= 24 + 12$$

$$= 36$$

# Why we need Deep Learning?



$$\text{Cost} \approx (2 \times 3) + (3 \times 4) + (4 \times 1)$$

$$= 6 + 12 + 4$$

$$= 22$$

# Why we need Deep Learning?

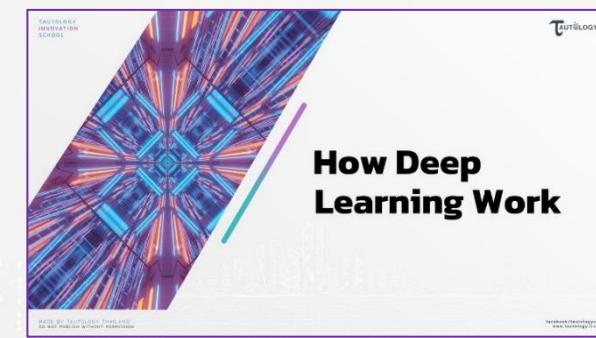


For more information



Deep Learning  
Interpretation

# Deep Learning

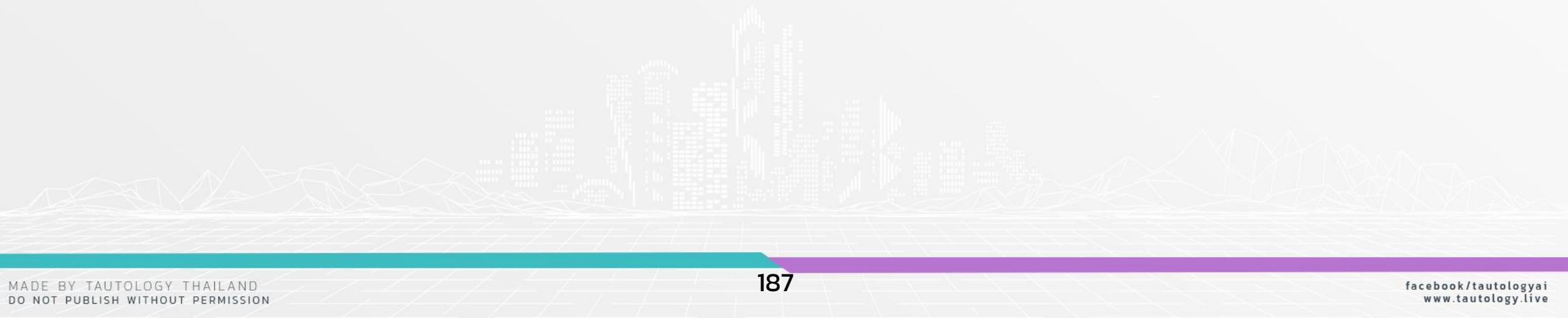


# How Deep Learning Work

# How Deep Learning Work

**Regression**

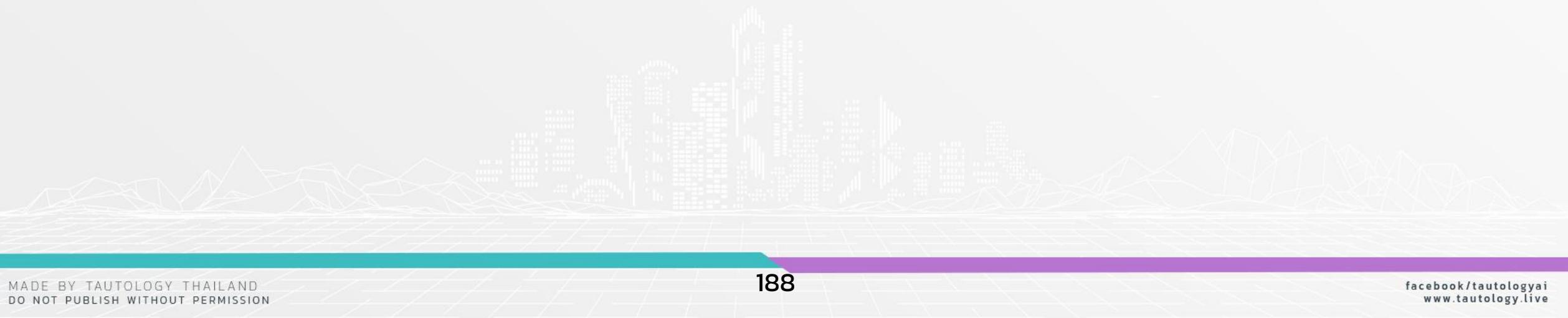
**Classification**



# How Deep Learning Work

**Regression**

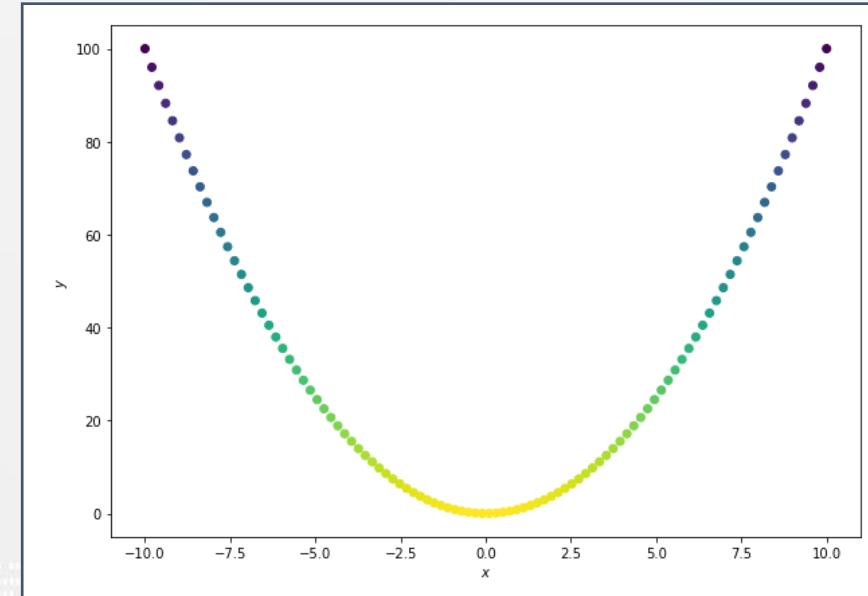
**Classification**



# How Deep Learning Work

Regression

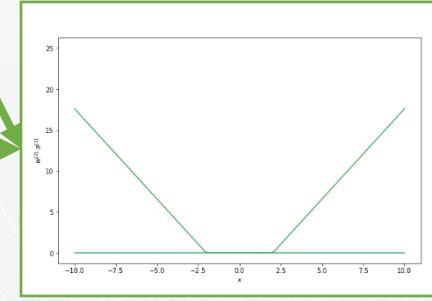
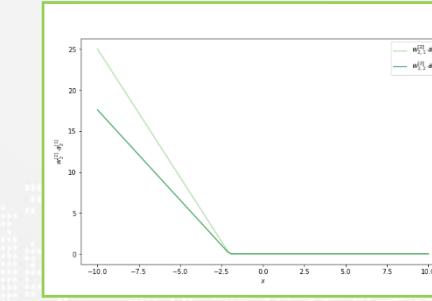
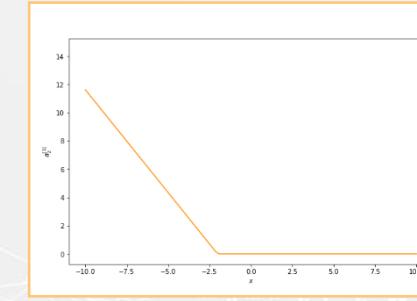
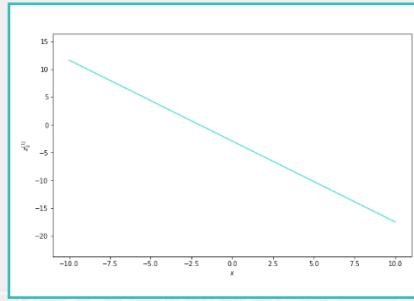
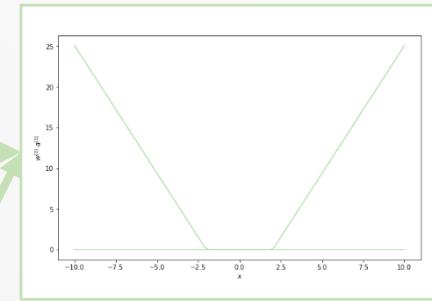
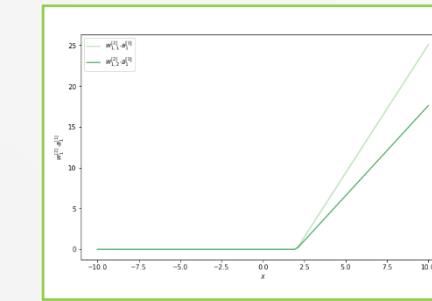
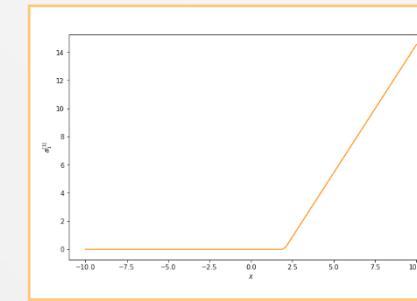
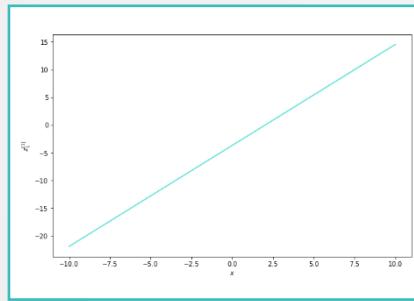
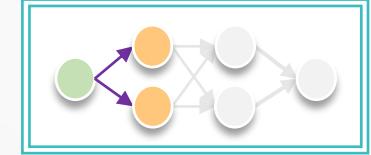
Example 1



# How Deep Learning Work

**Regression**

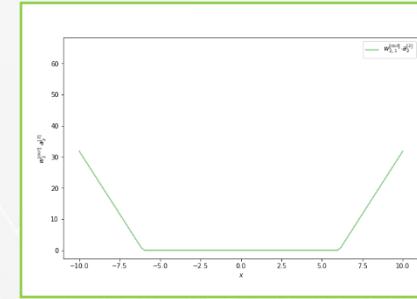
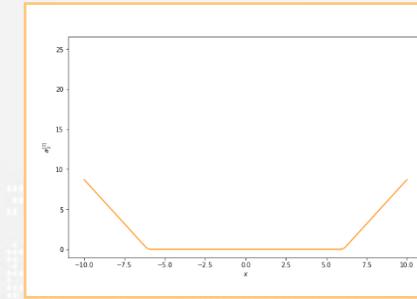
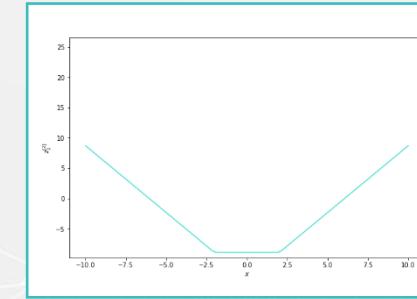
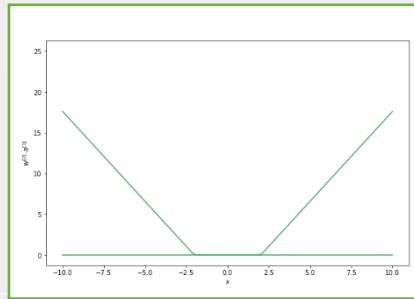
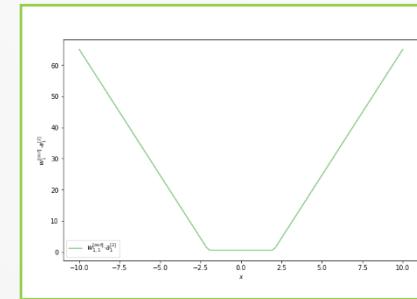
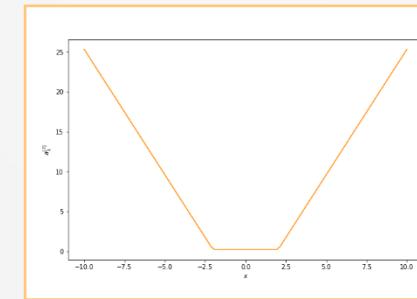
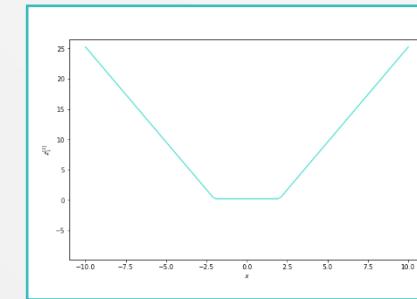
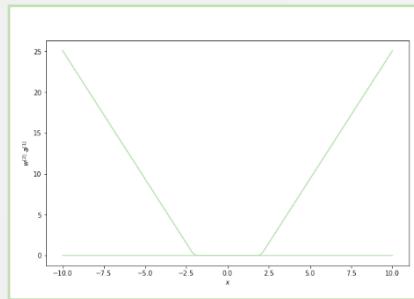
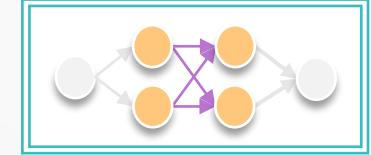
**Example 1**



# How Deep Learning Work

**Regression**

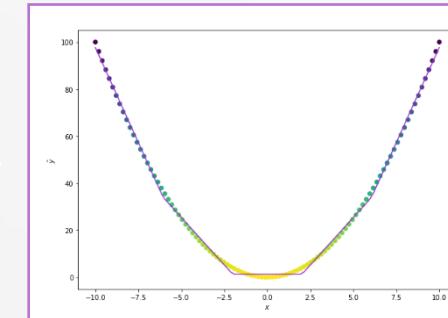
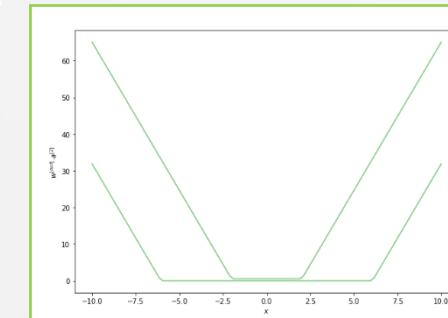
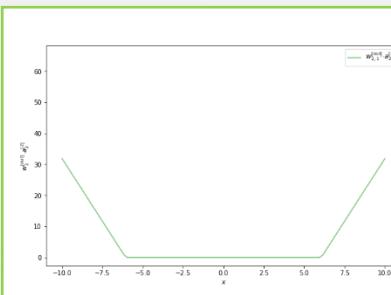
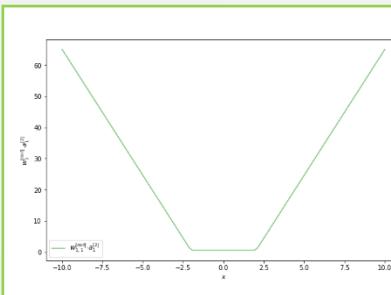
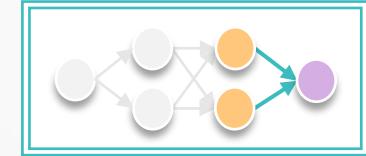
**Example 1**



# How Deep Learning Work

Regression

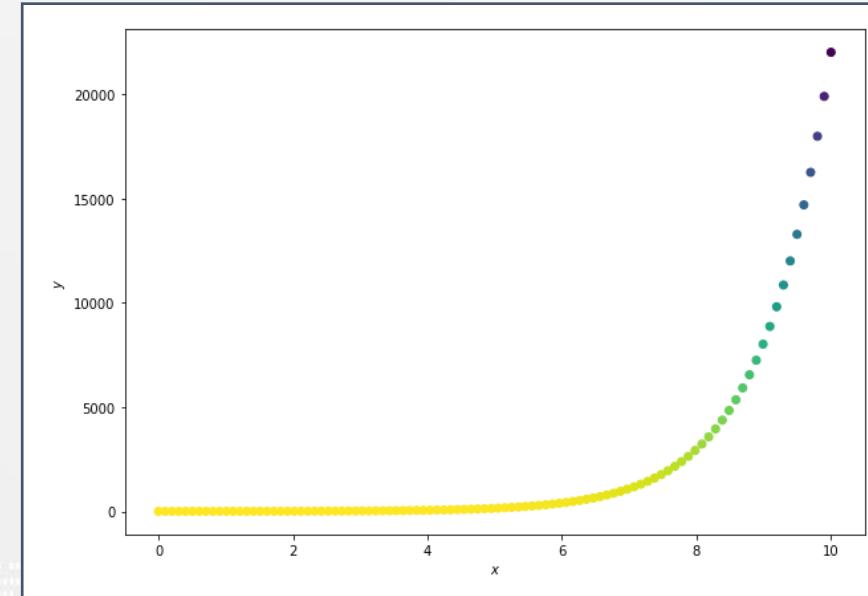
Example 1



# How Deep Learning Work

Regression

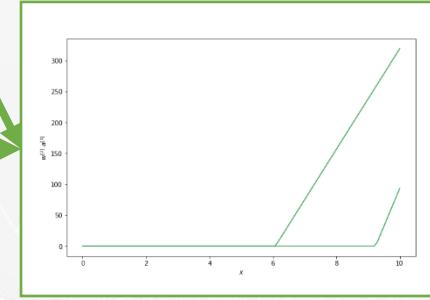
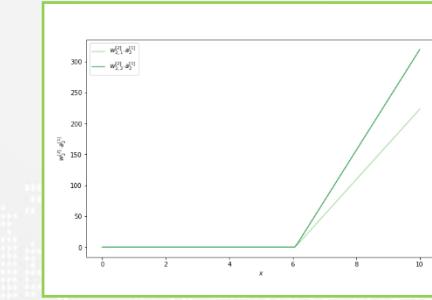
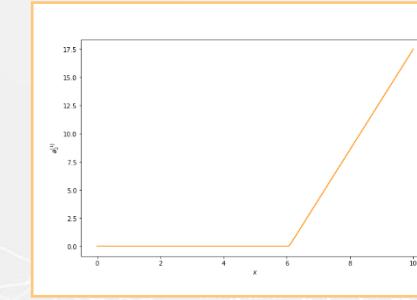
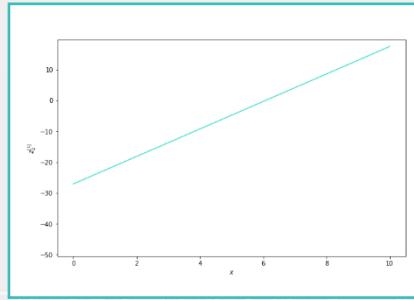
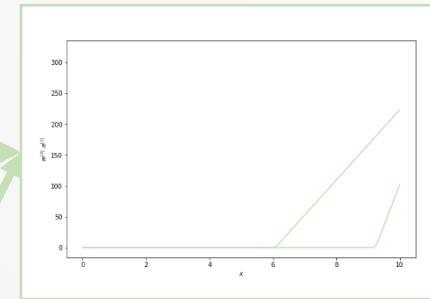
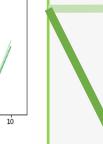
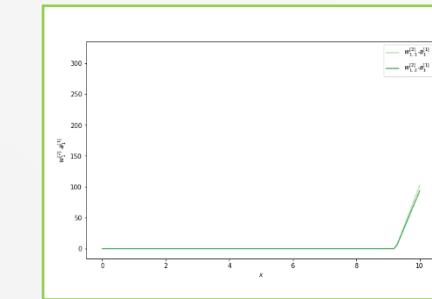
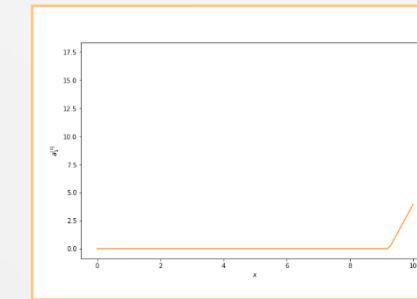
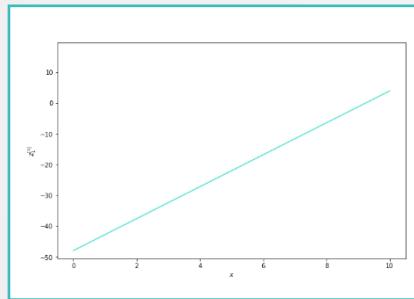
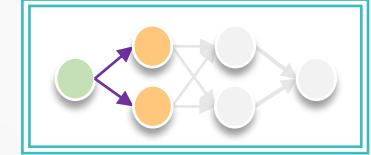
Example 2



# How Deep Learning Work

**Regression**

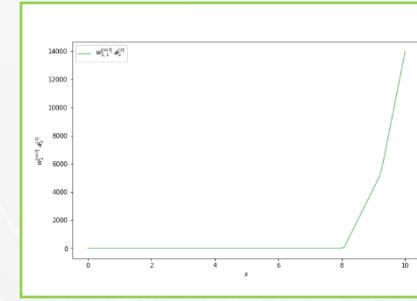
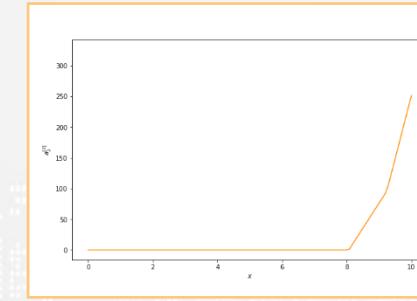
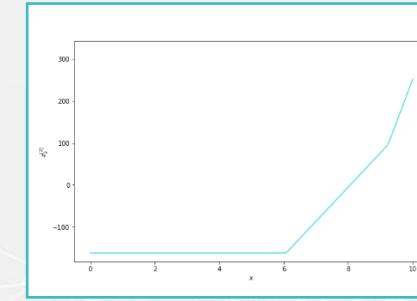
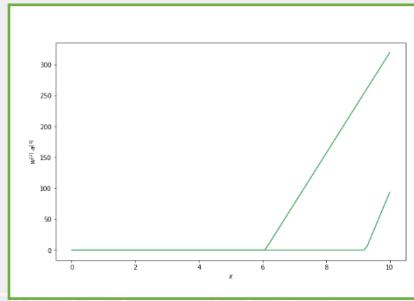
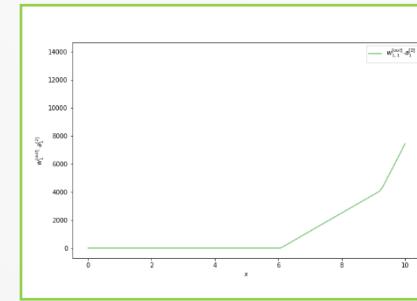
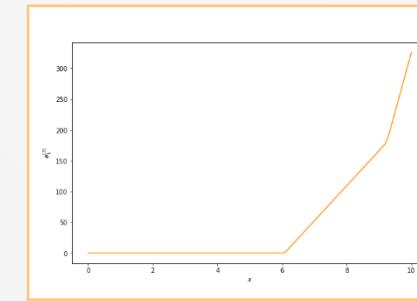
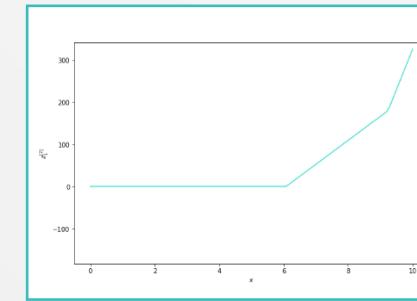
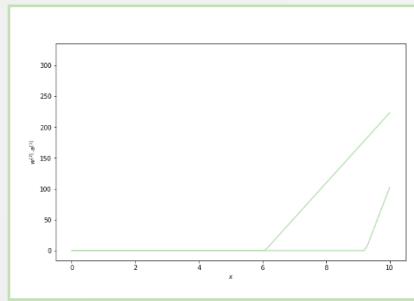
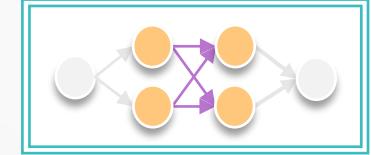
**Example 2**



# How Deep Learning Work

**Regression**

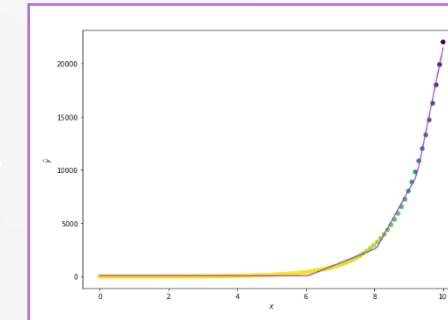
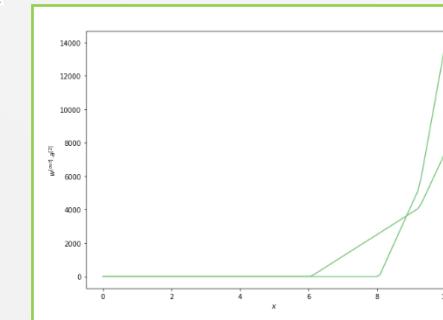
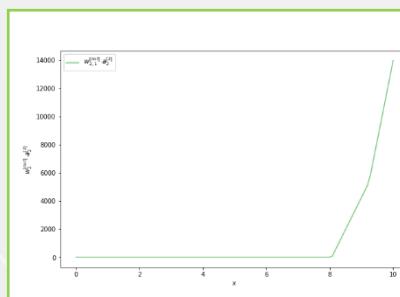
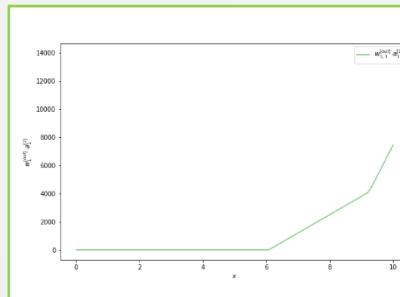
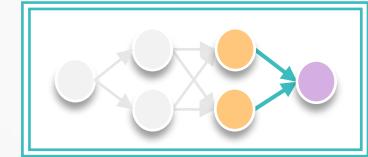
**Example 2**



# How Deep Learning Work

Regression

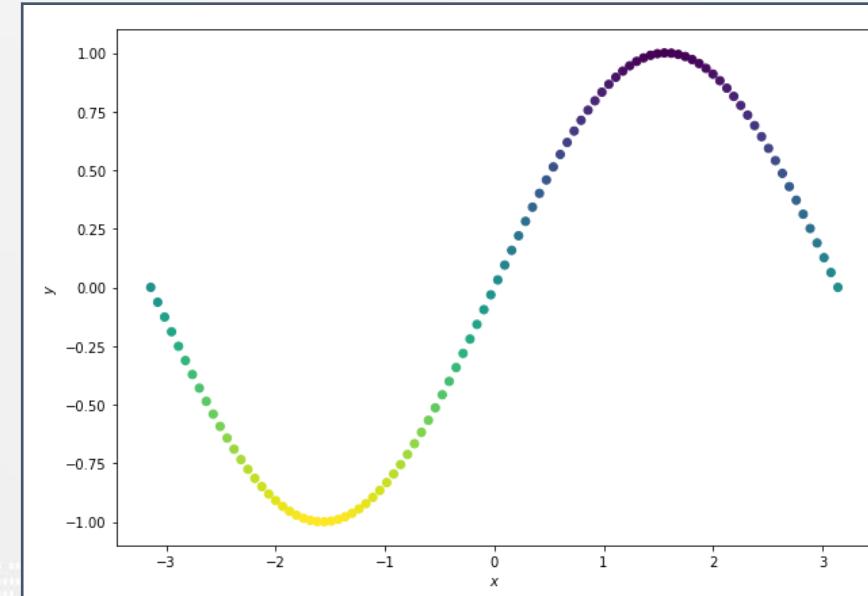
Example 2



# How Deep Learning Work

Regression

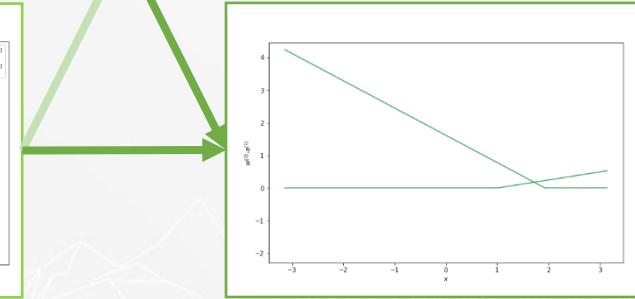
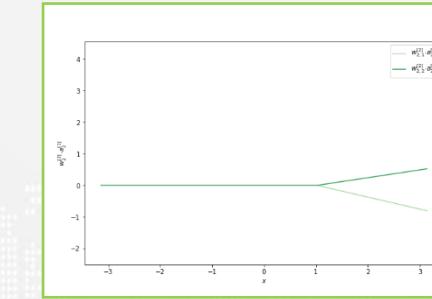
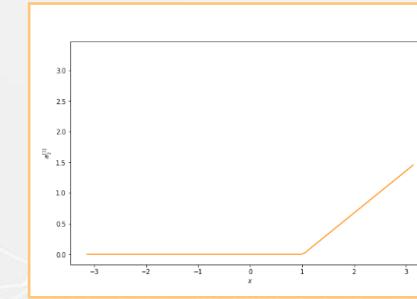
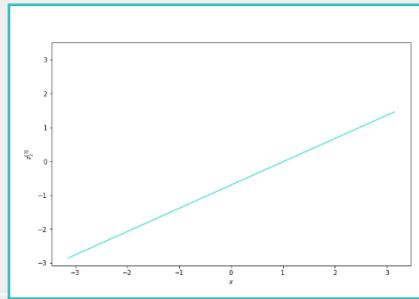
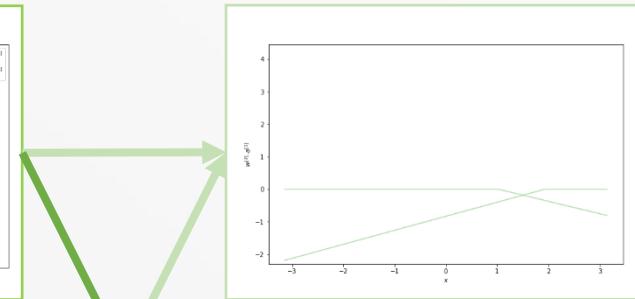
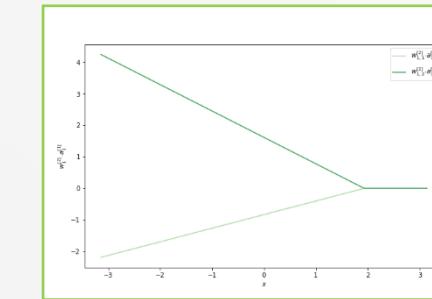
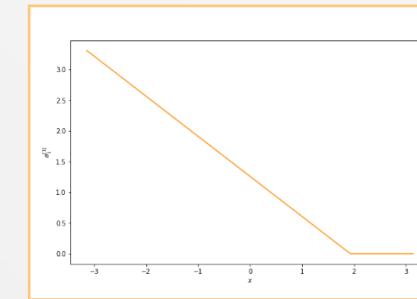
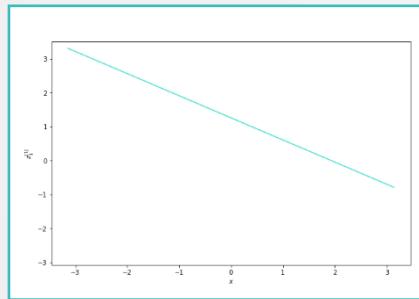
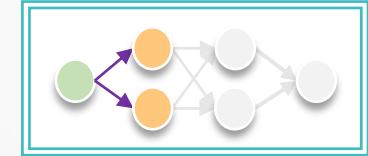
Example 3



# How Deep Learning Work

**Regression**

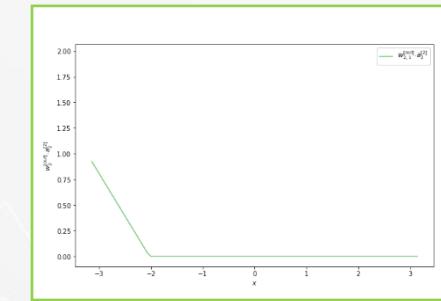
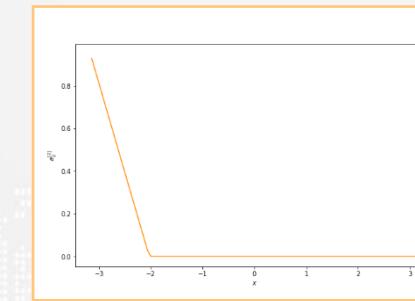
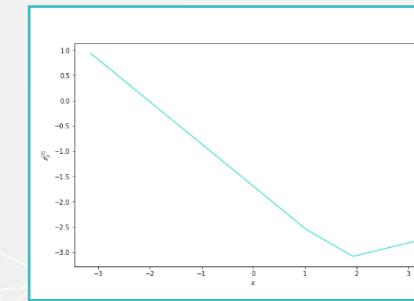
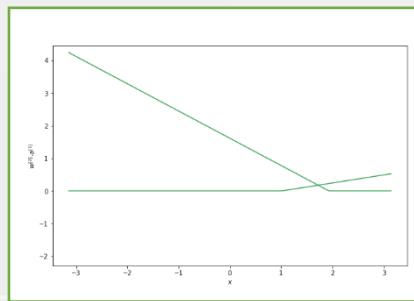
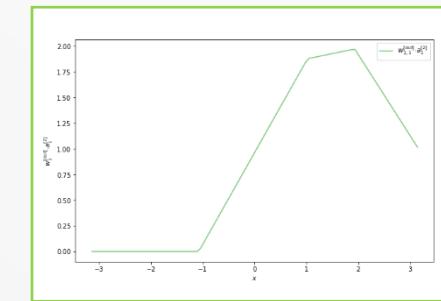
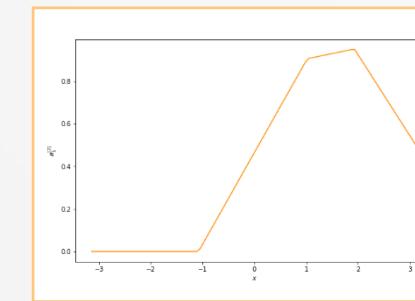
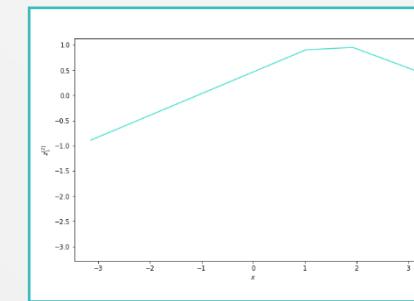
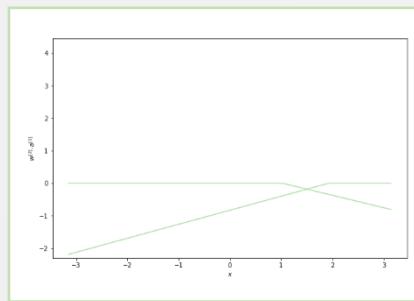
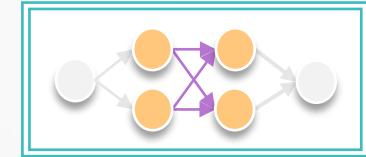
**Example 3**



# How Deep Learning Work

**Regression**

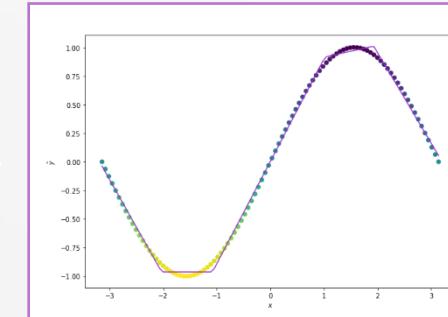
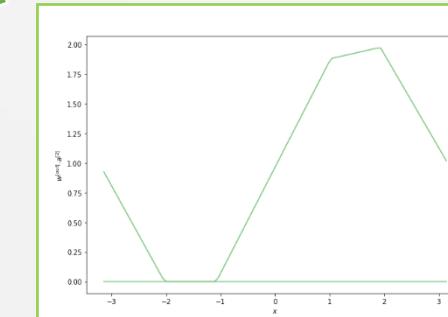
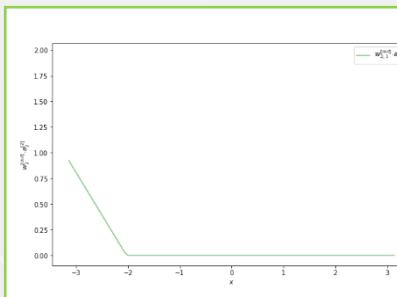
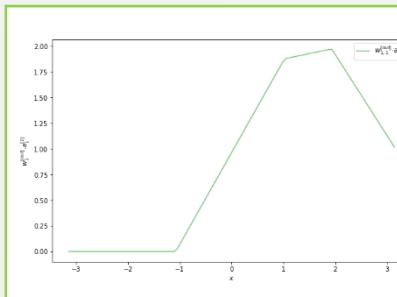
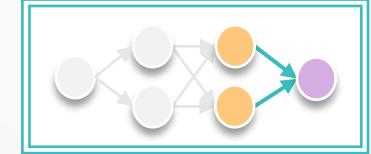
**Example 3**



# How Deep Learning Work

Regression

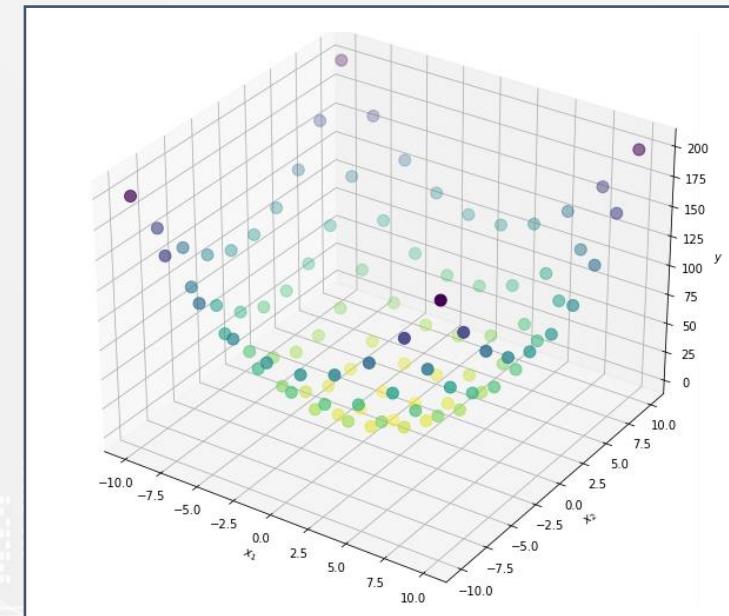
Example 3



# How Deep Learning Work

Regression

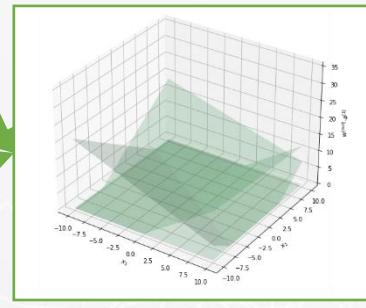
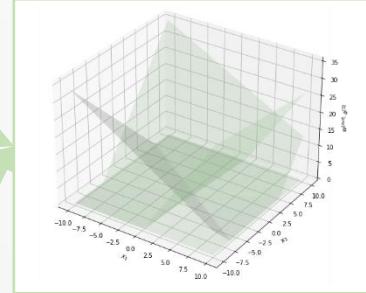
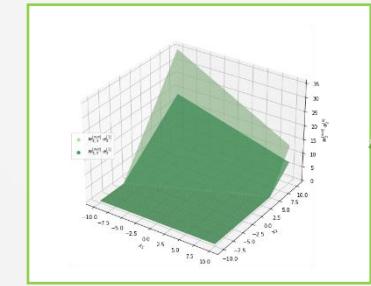
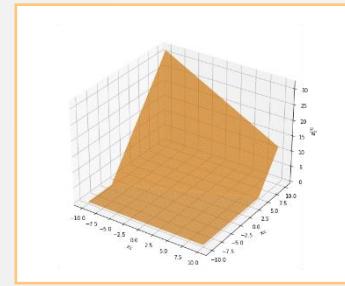
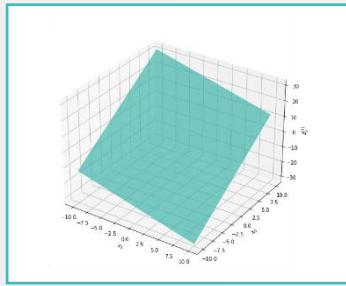
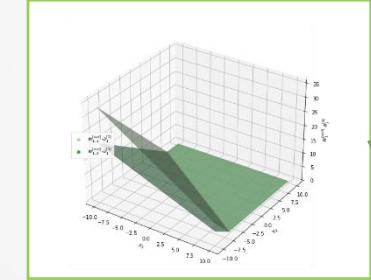
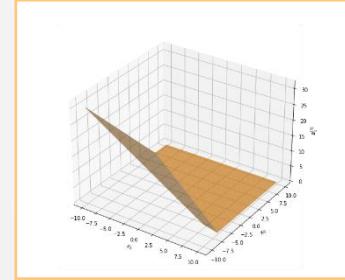
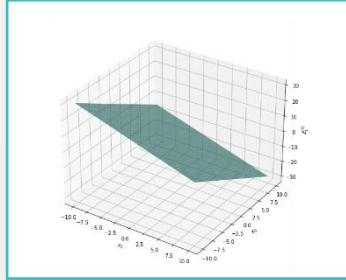
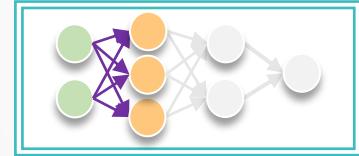
Example 4



# How Deep Learning Work

Regression

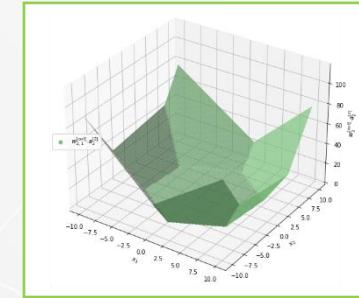
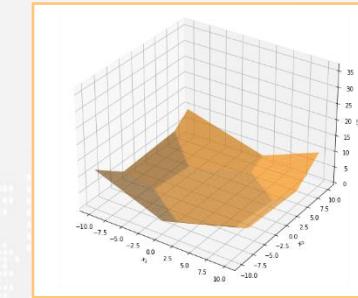
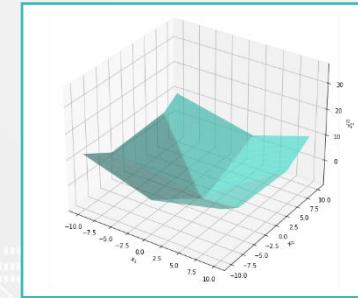
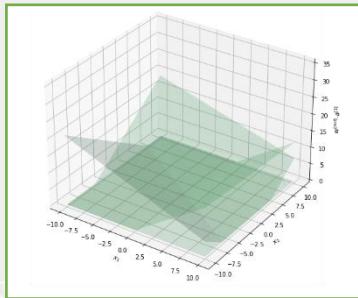
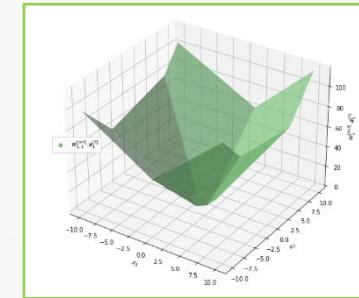
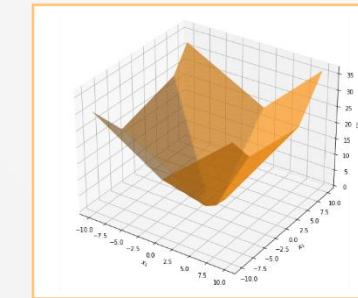
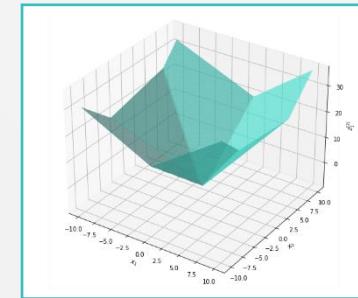
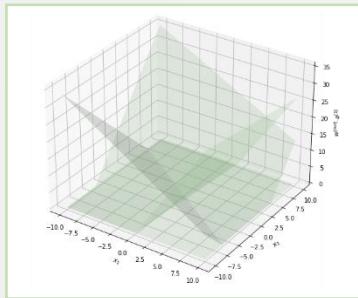
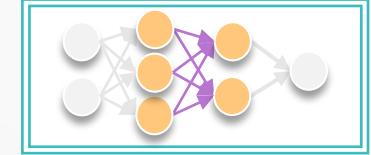
Example 4



# How Deep Learning Work

Regression

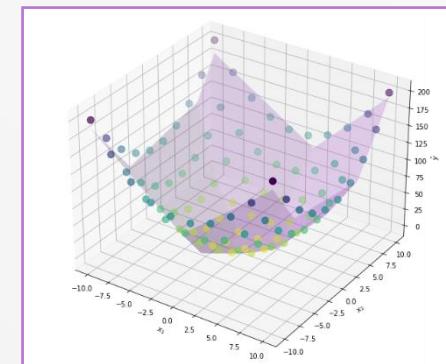
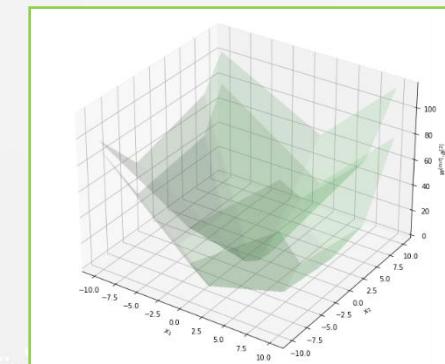
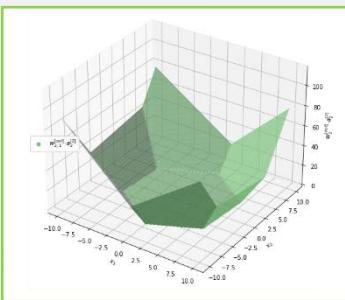
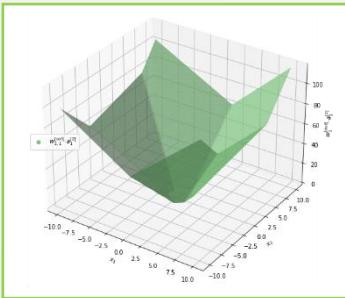
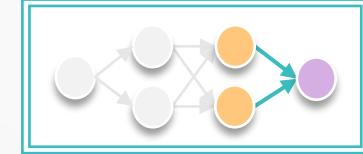
Example 4



# How Deep Learning Work

Regression

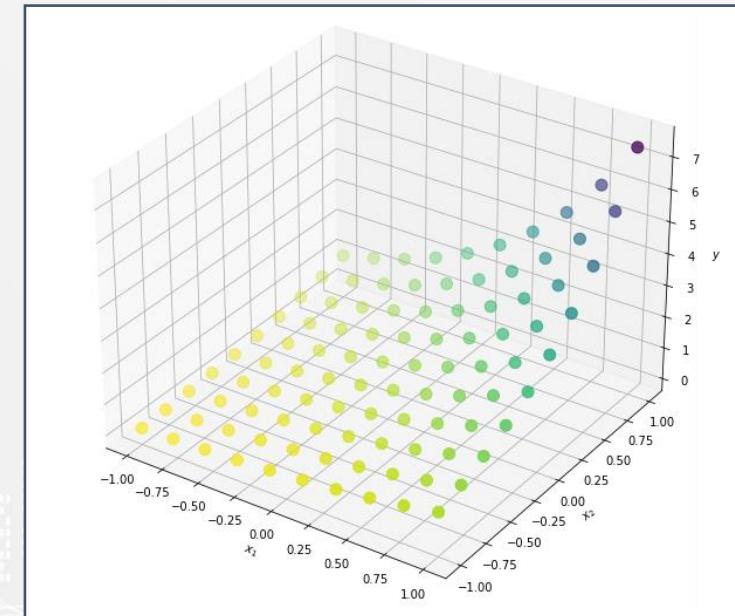
Example 4



# How Deep Learning Work

Regression

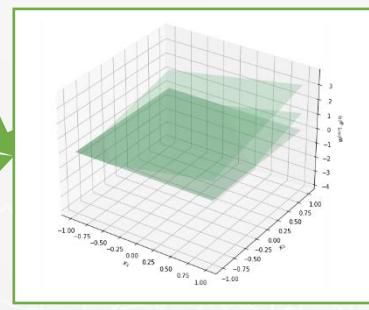
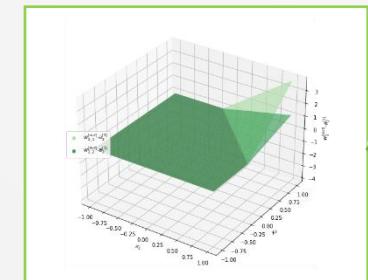
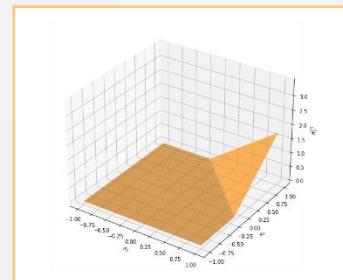
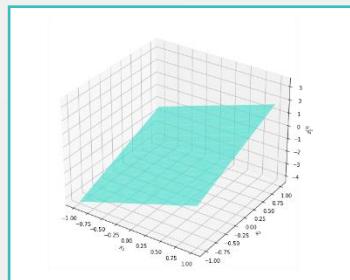
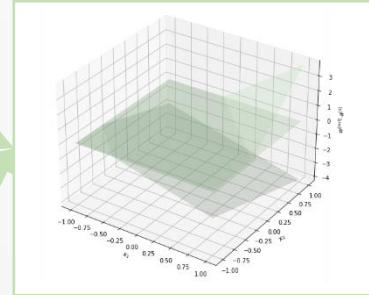
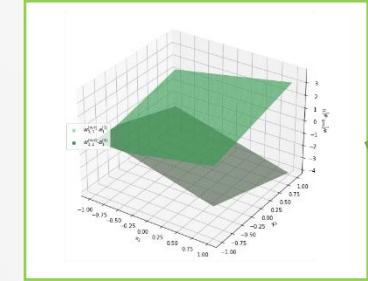
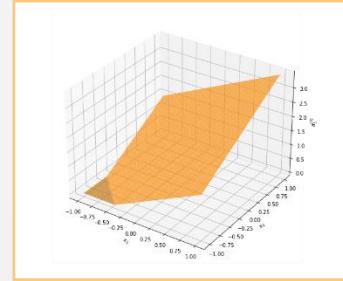
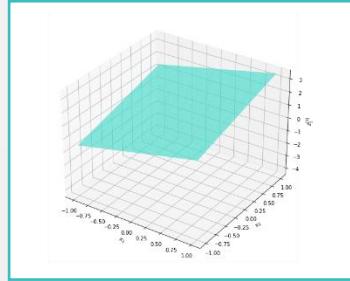
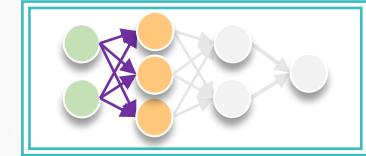
Example 5



# How Deep Learning Work

**Regression**

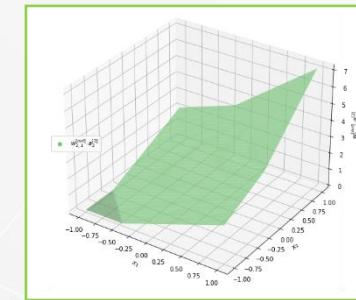
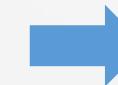
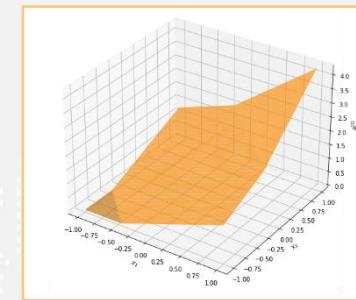
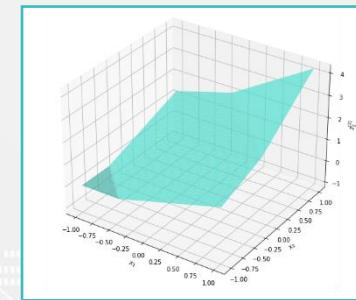
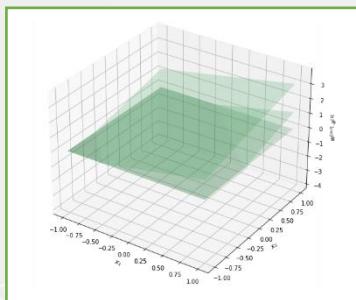
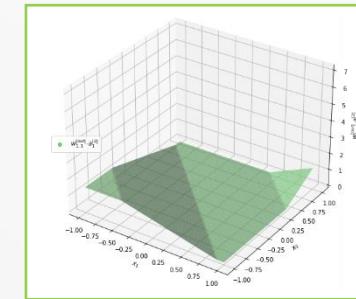
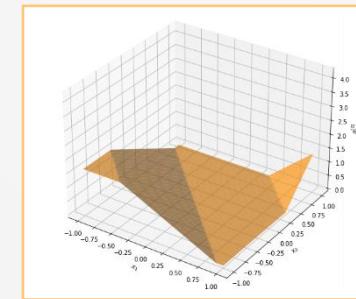
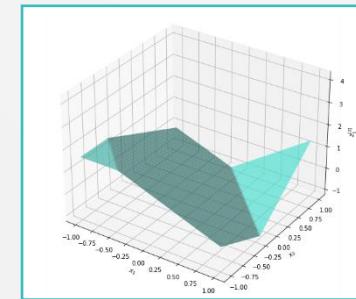
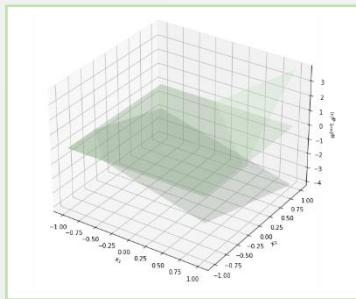
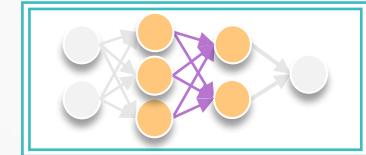
**Example 5**



# How Deep Learning Work

Regression

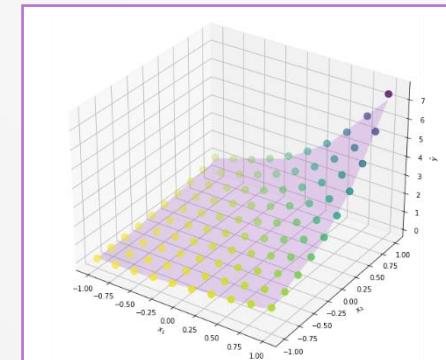
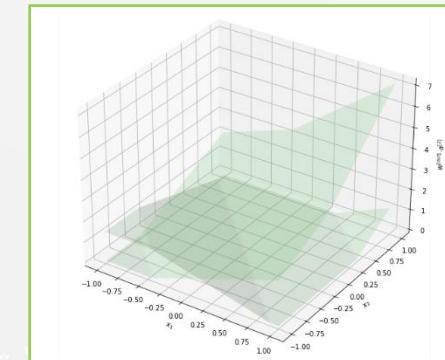
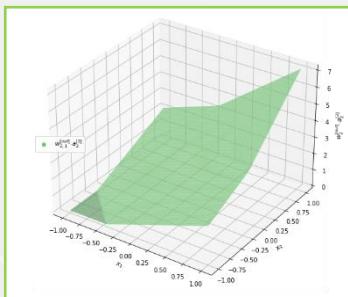
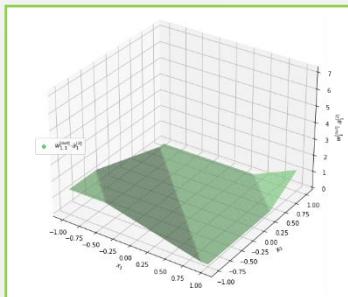
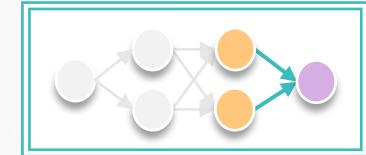
Example 5



# How Deep Learning Work

Regression

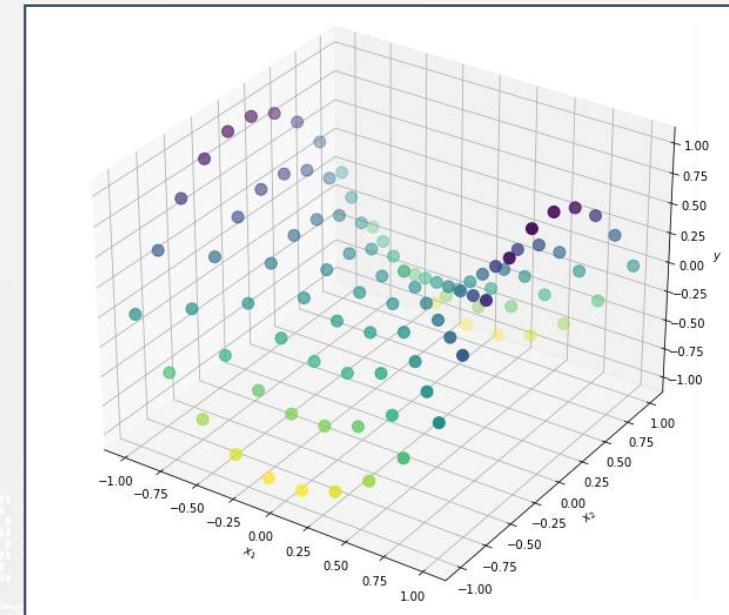
Example 5



# How Deep Learning Work

Regression

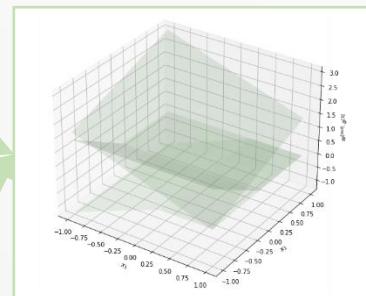
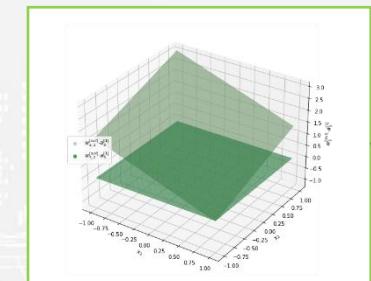
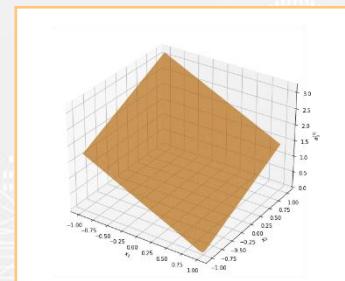
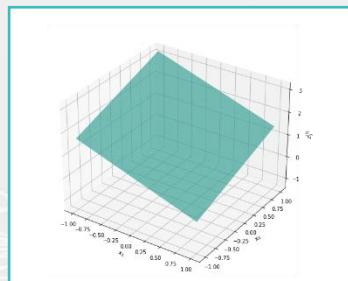
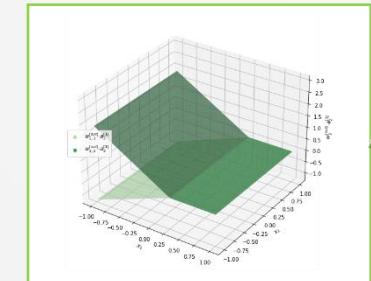
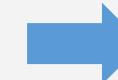
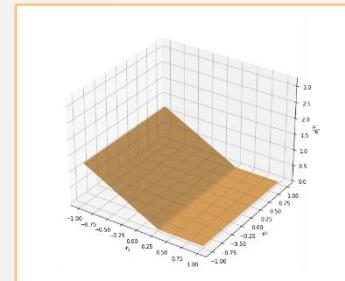
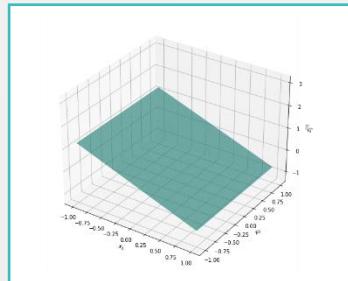
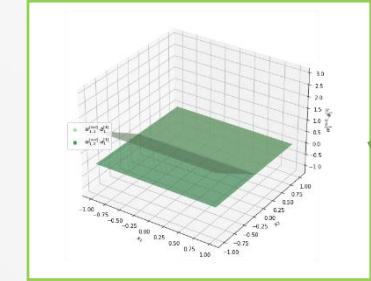
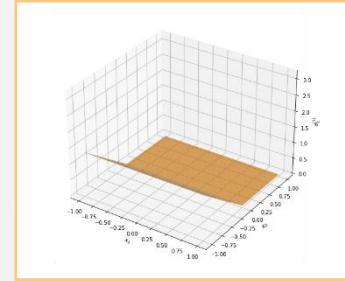
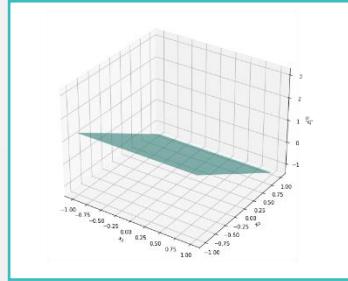
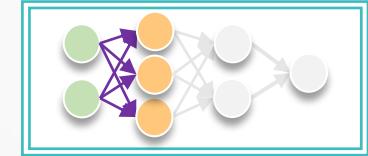
Example 6



# How Deep Learning Work

**Regression**

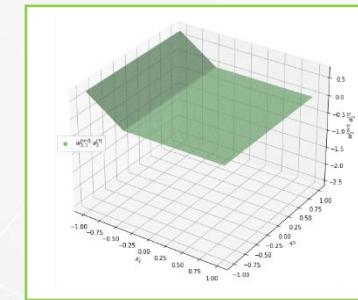
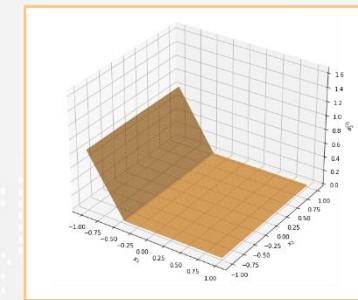
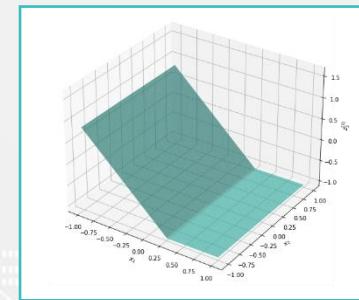
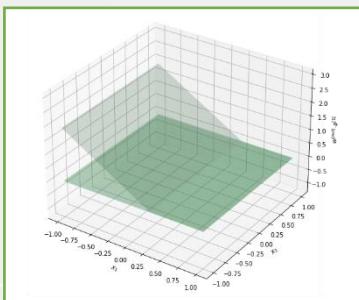
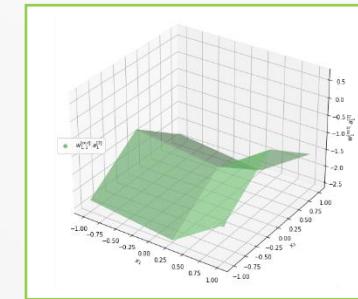
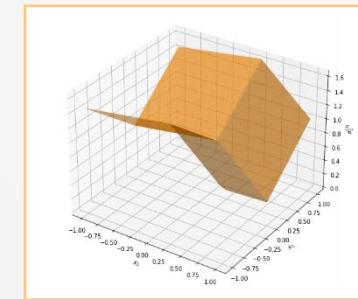
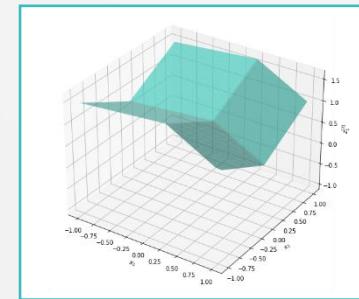
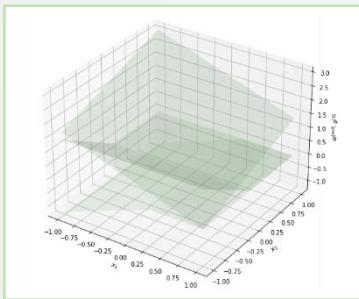
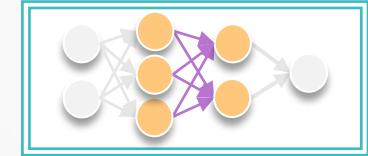
**Example 6**



# How Deep Learning Work

Regression

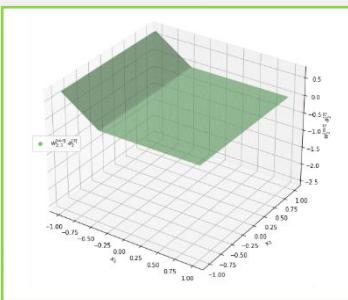
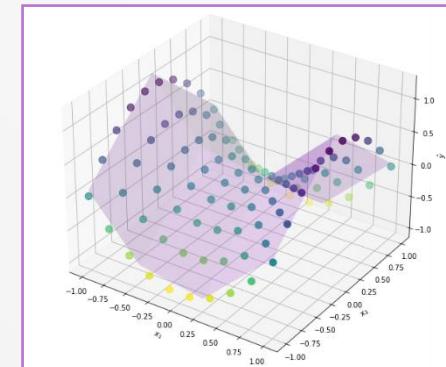
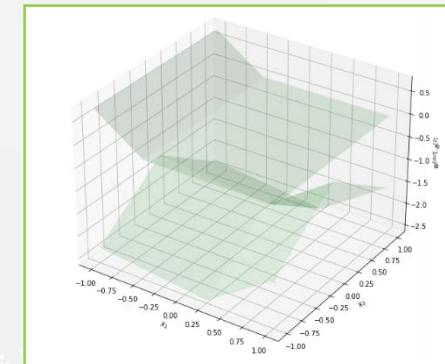
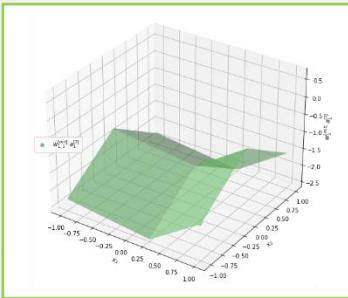
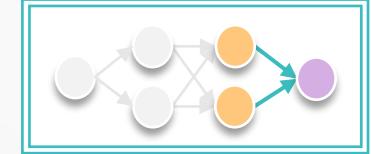
Example 6



# How Deep Learning Work

Regression

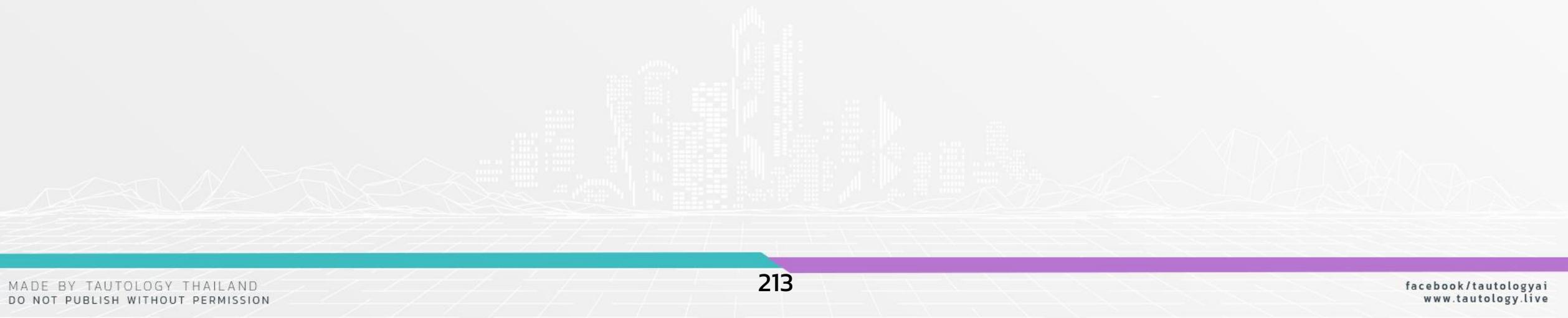
Example 6



# How Deep Learning Work

Regression

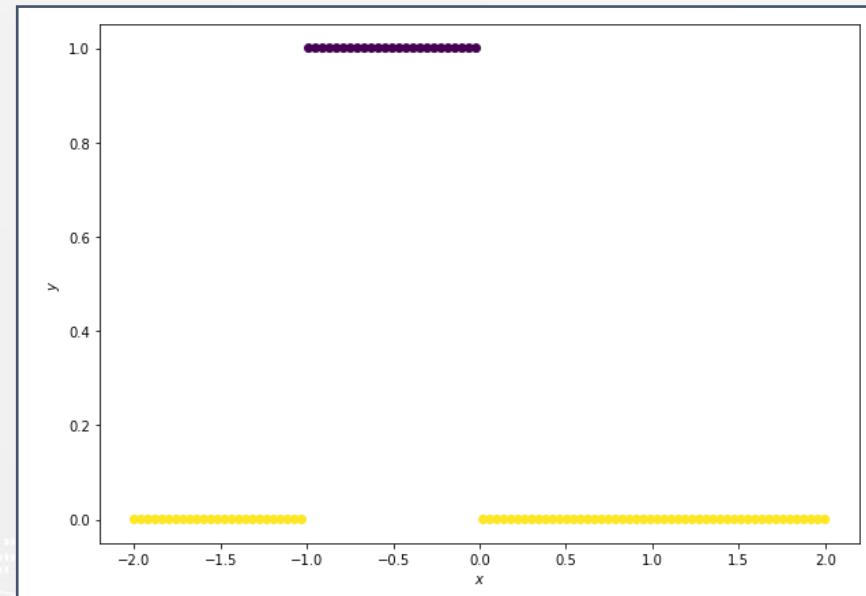
Classification



# How Deep Learning Work

## Classification

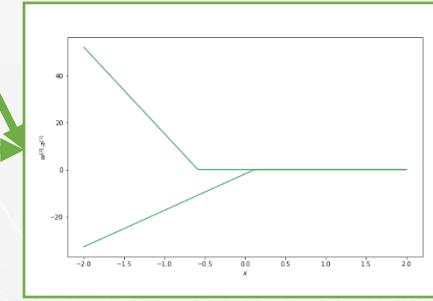
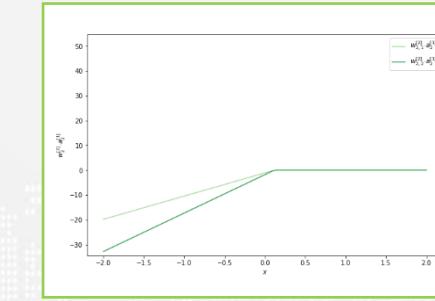
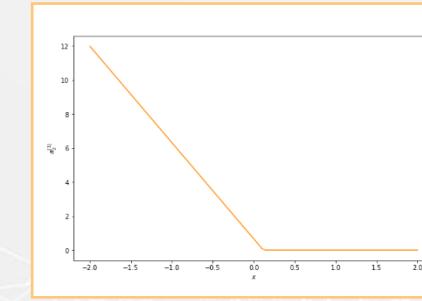
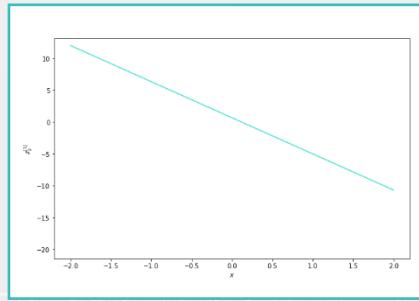
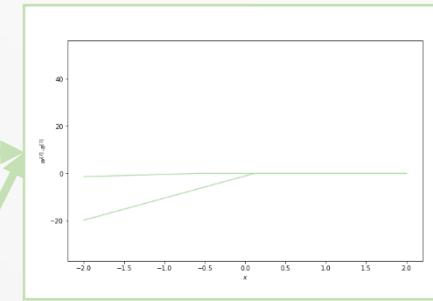
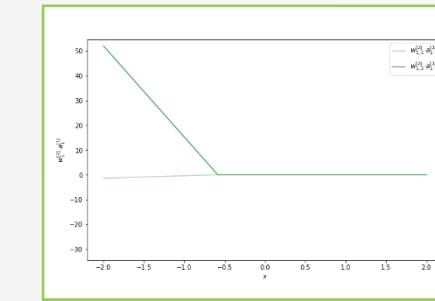
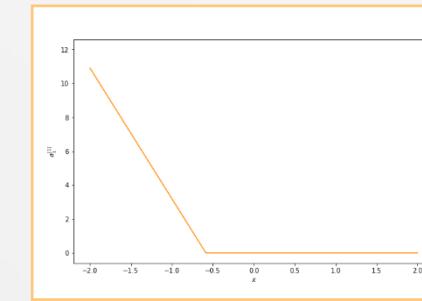
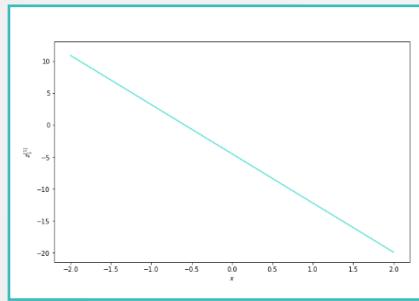
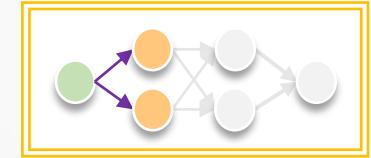
### Example 1: Binary Classification



# How Deep Learning Work

**Classification**

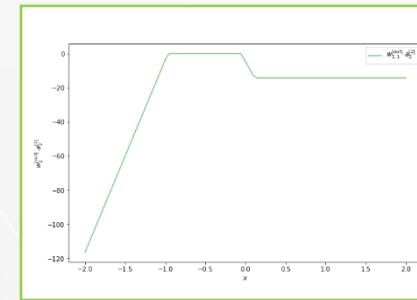
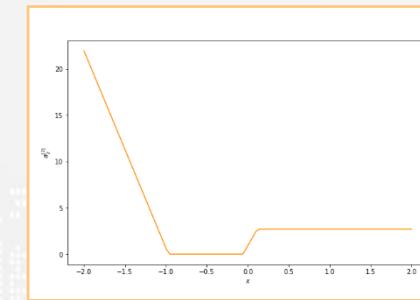
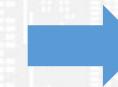
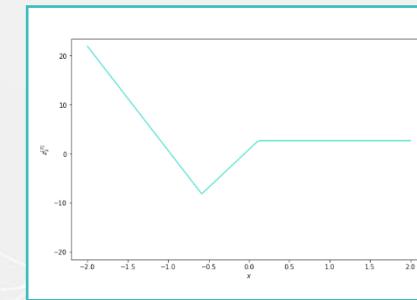
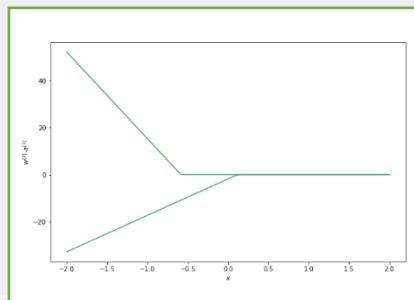
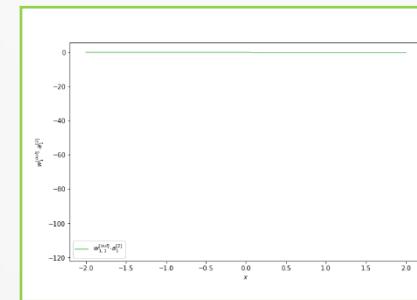
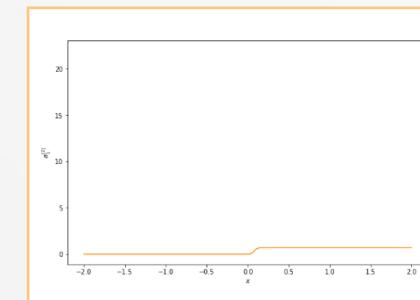
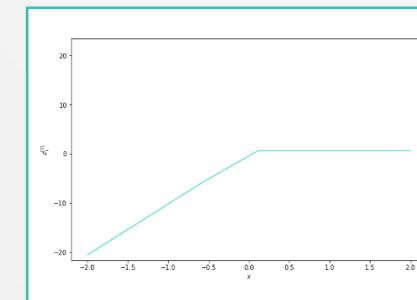
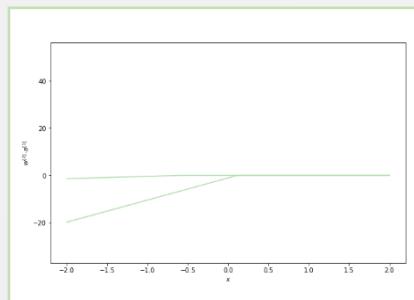
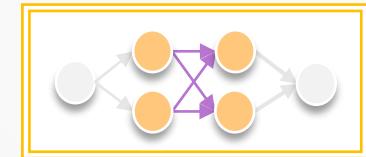
**Example 1**



# How Deep Learning Work

**Classification**

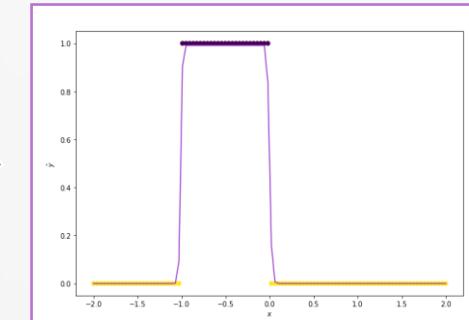
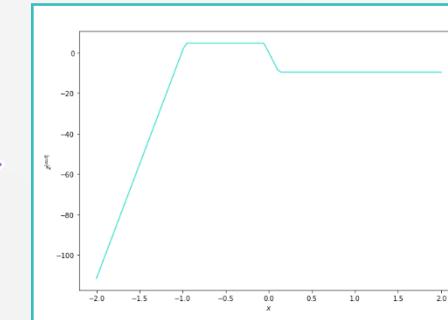
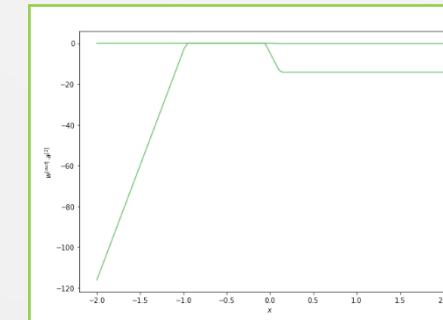
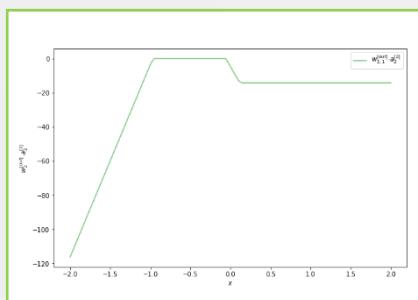
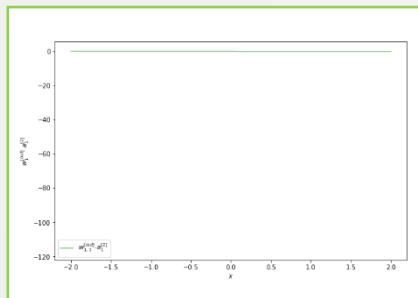
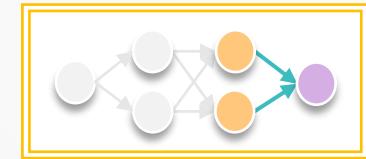
**Example 1**



# How Deep Learning Work

Classification

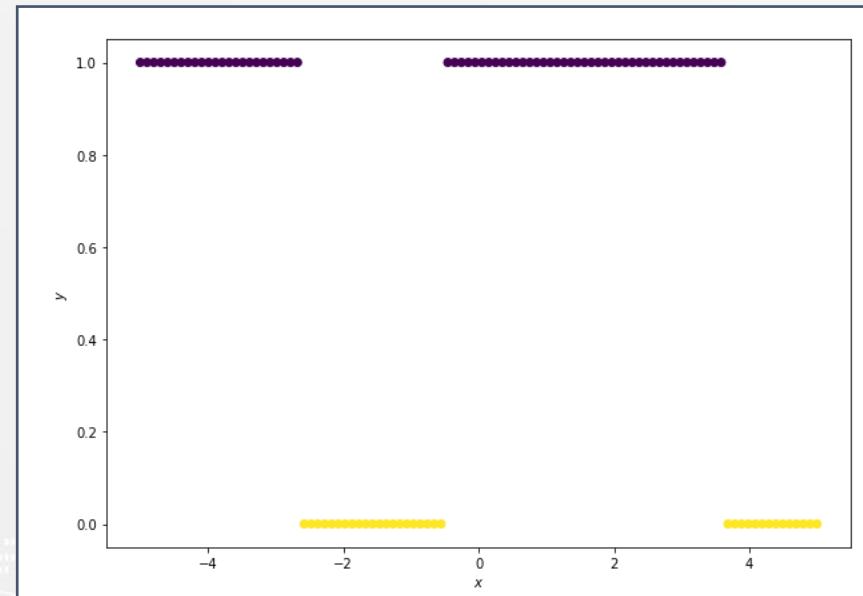
Example 1



# How Deep Learning Work

## Classification

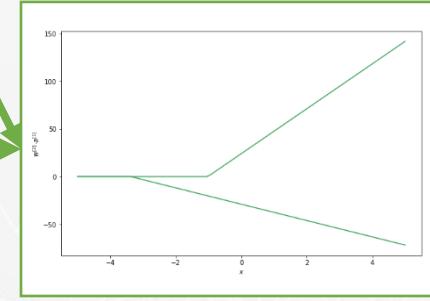
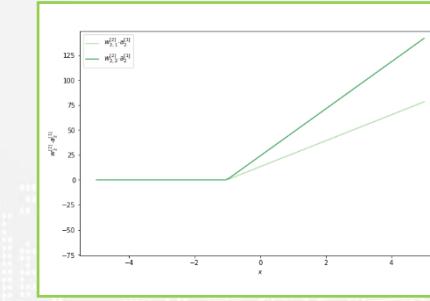
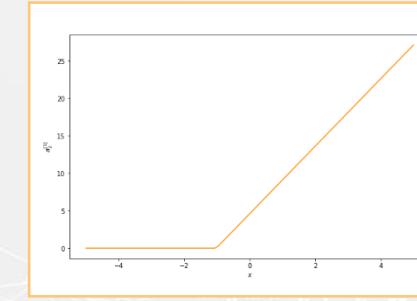
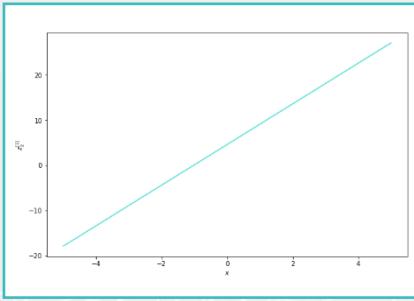
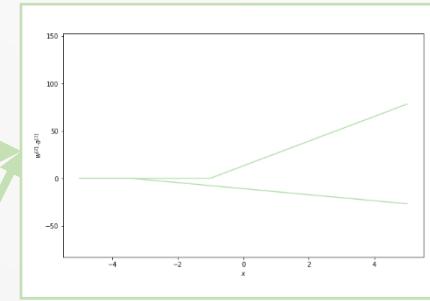
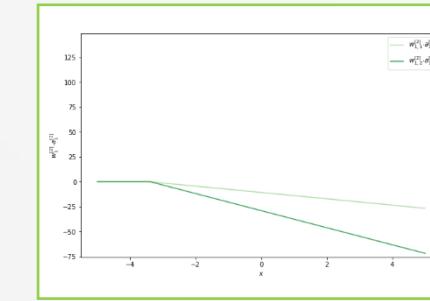
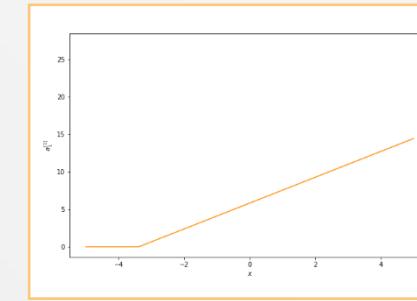
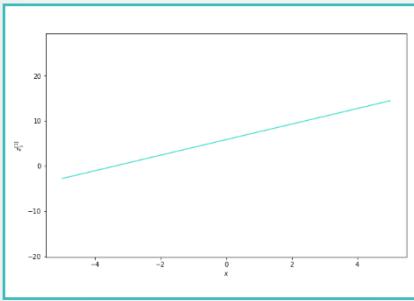
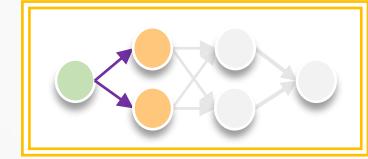
### Example 2 : Binary Classification



# How Deep Learning Work

**Classification**

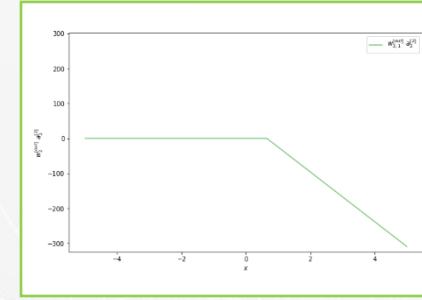
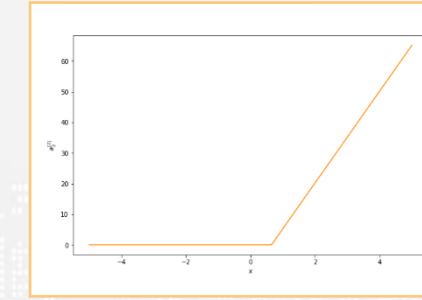
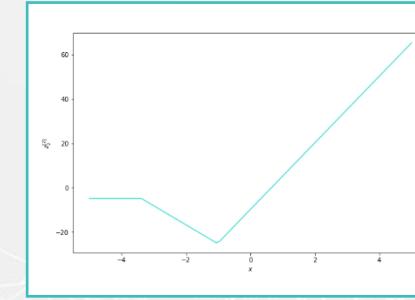
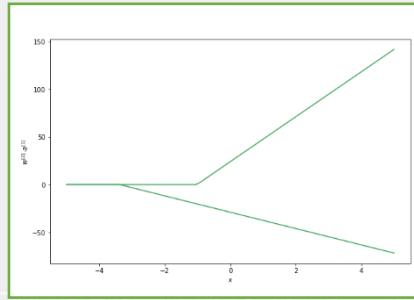
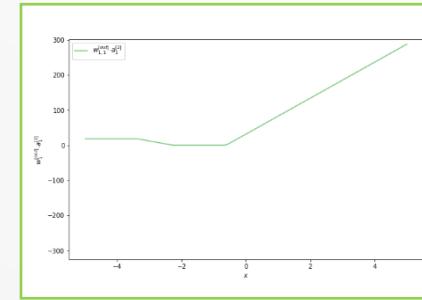
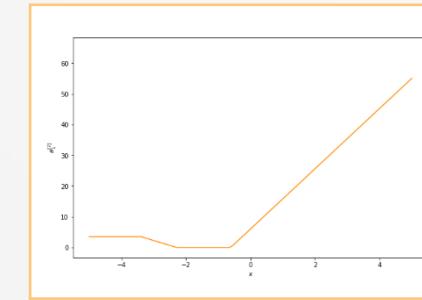
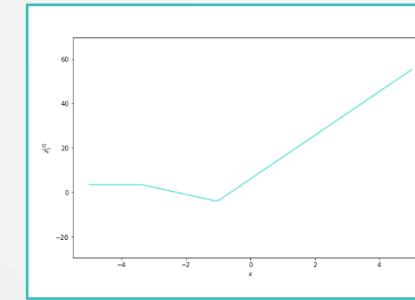
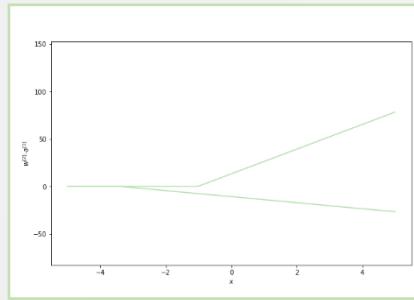
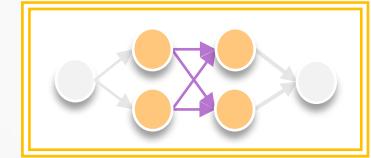
**Example 2**



# How Deep Learning Work

**Classification**

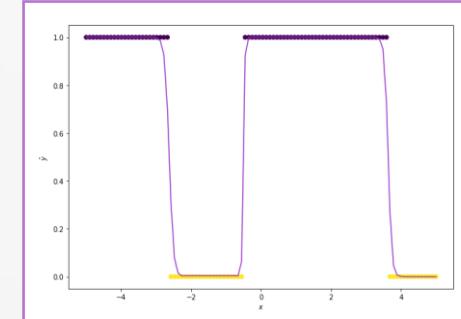
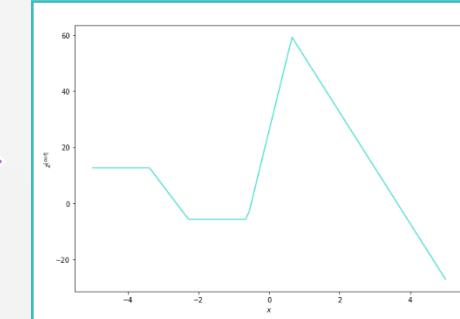
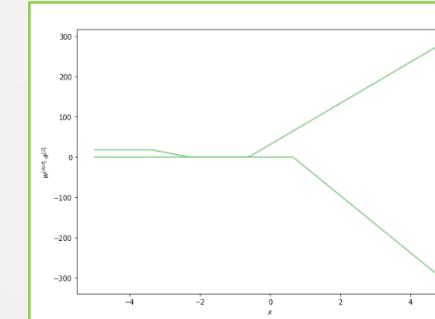
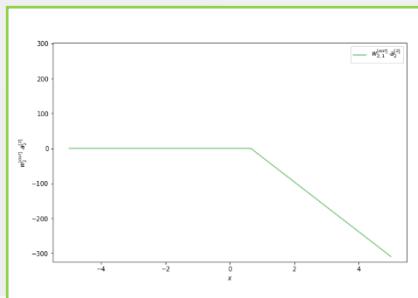
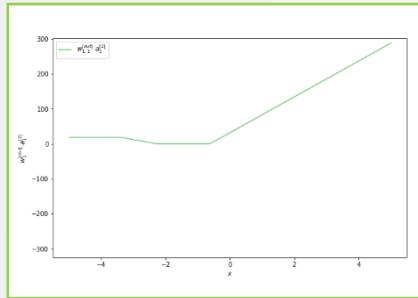
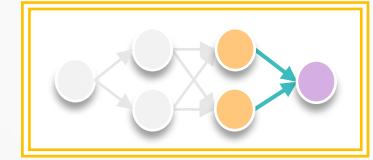
**Example 2**



# How Deep Learning Work

Classification

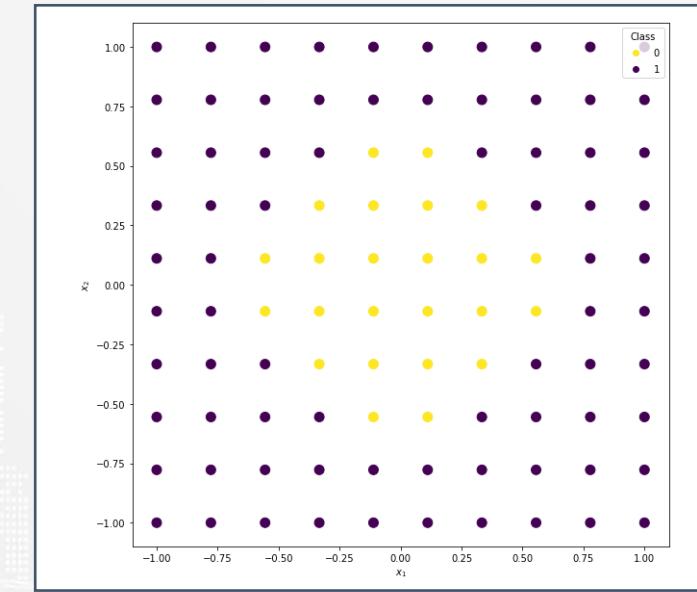
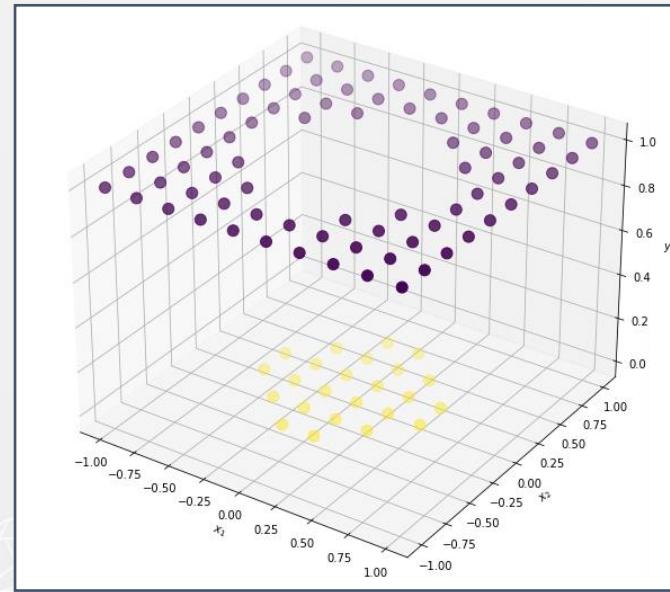
Example 2



# How Deep Learning Work

## Classification

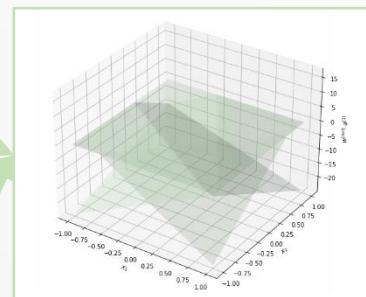
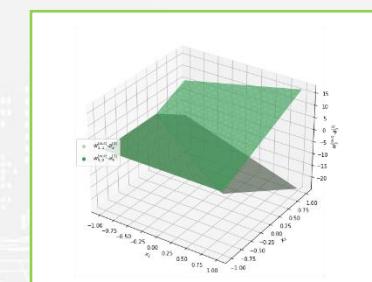
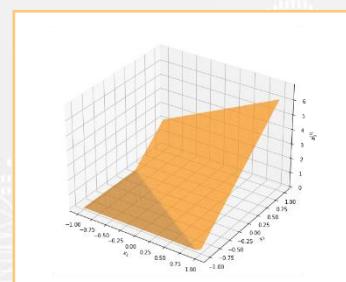
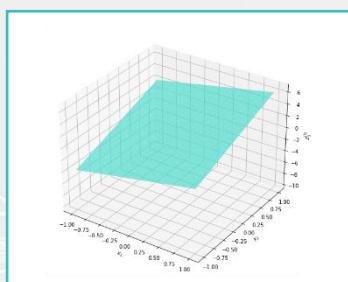
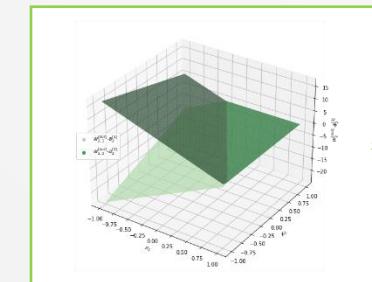
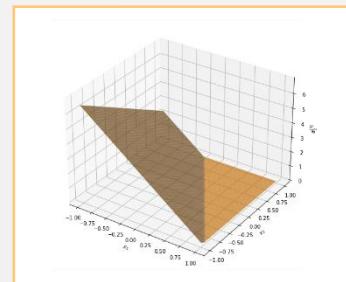
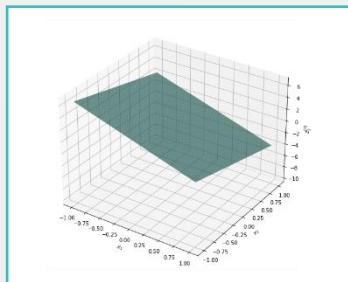
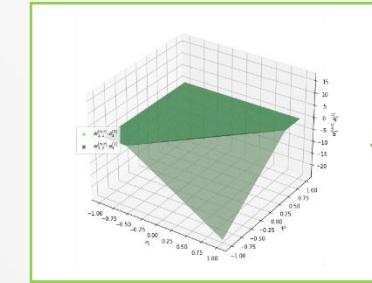
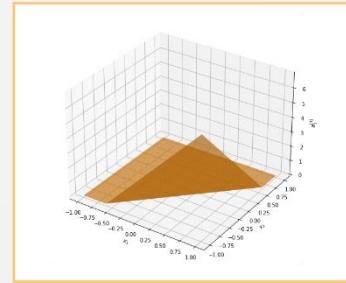
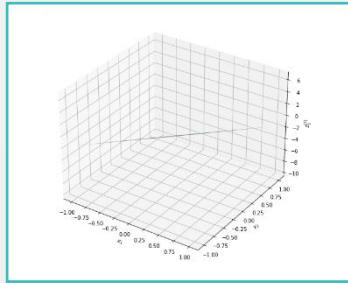
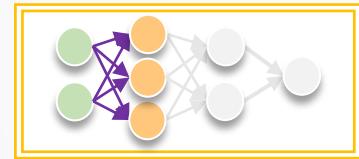
### Example 3 : Binary Classification



# How Deep Learning Work

## Classification

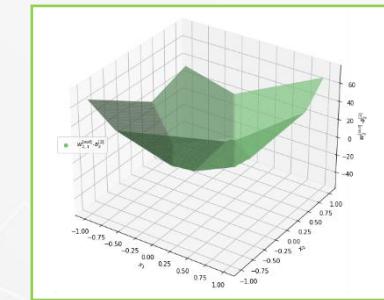
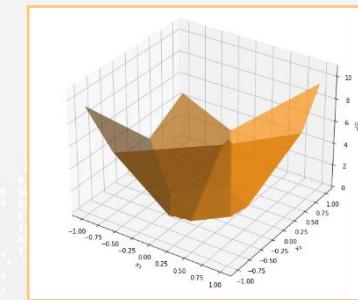
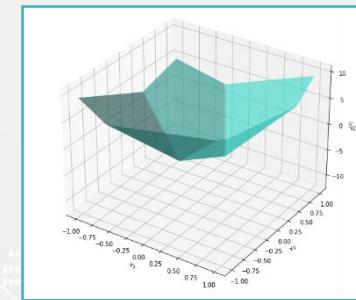
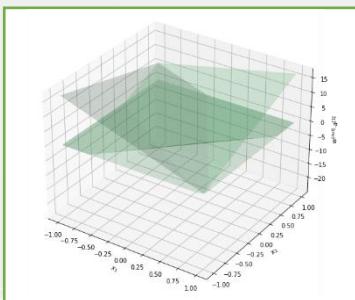
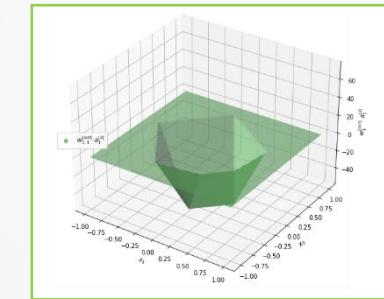
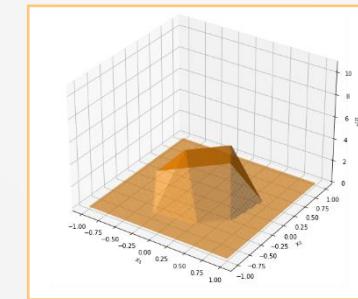
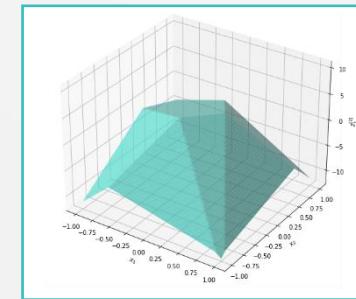
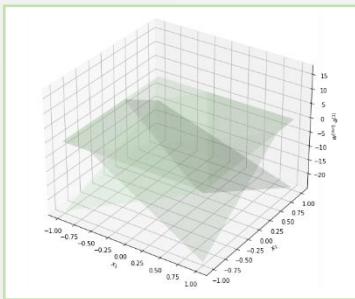
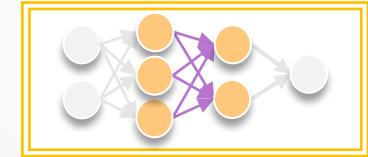
## Example 3



# How Deep Learning Work

Classification

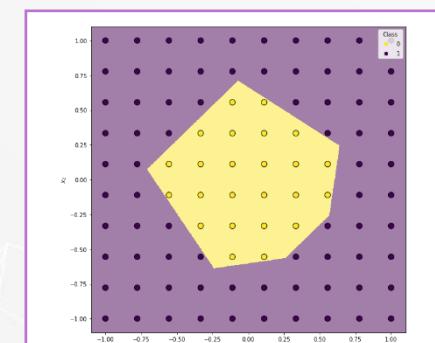
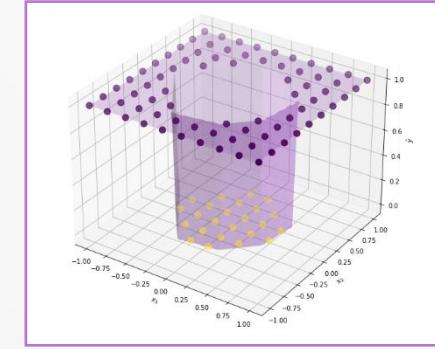
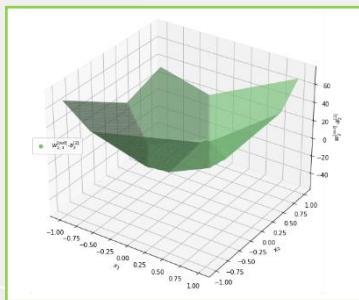
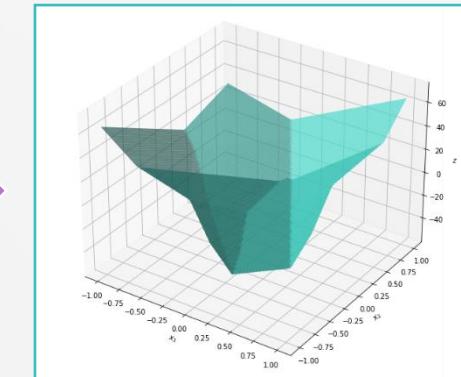
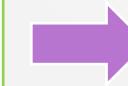
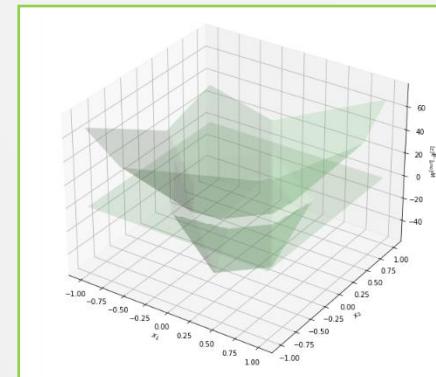
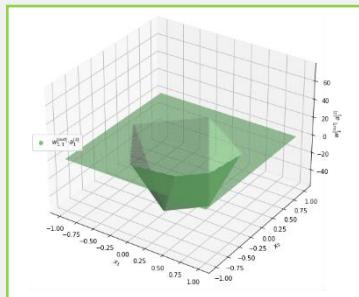
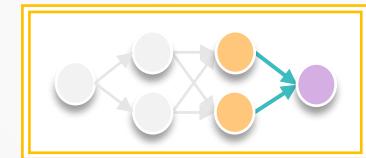
Example 3



# How Deep Learning Work

Classification

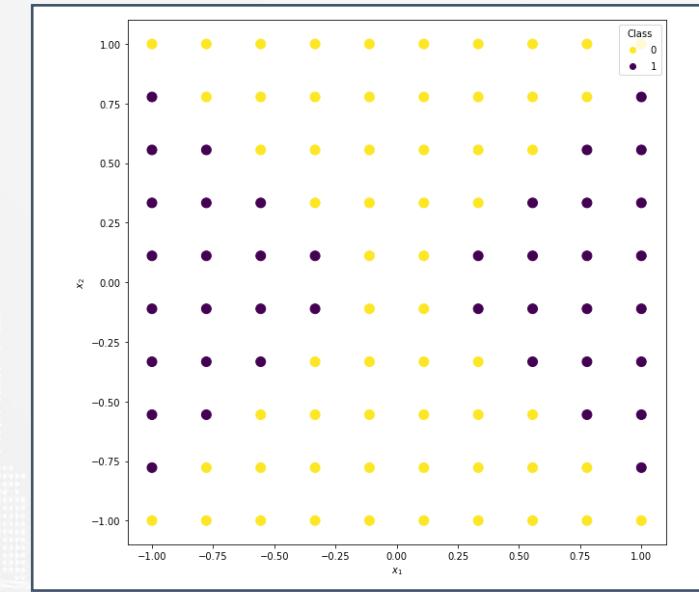
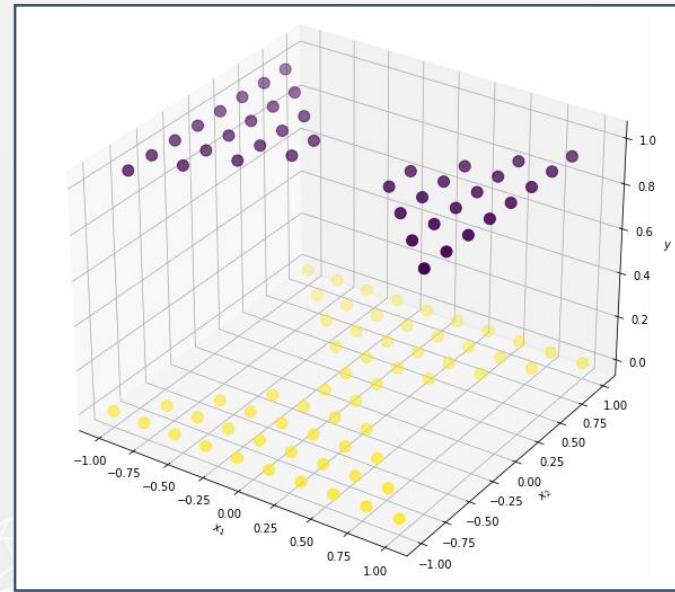
Example 3



# How Deep Learning Work

## Classification

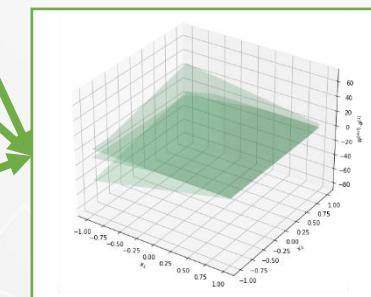
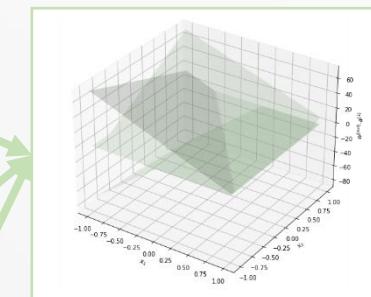
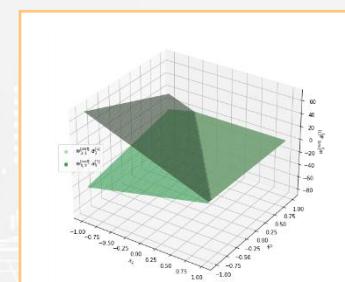
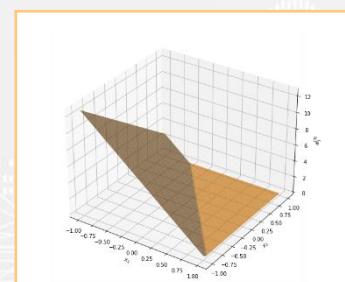
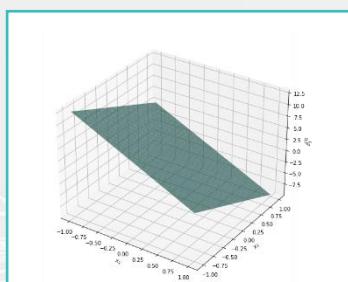
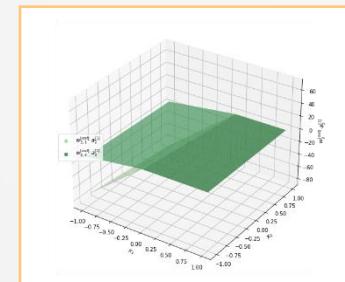
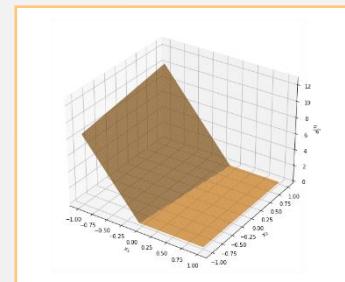
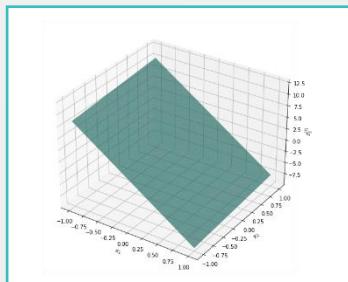
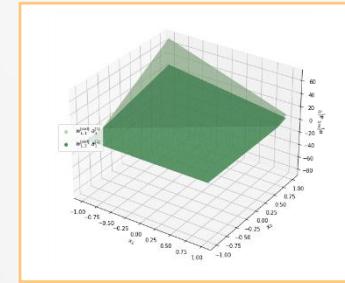
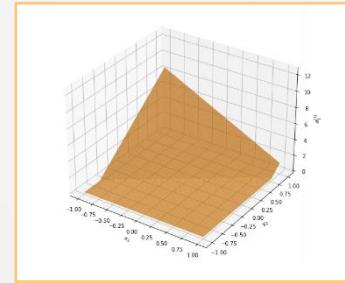
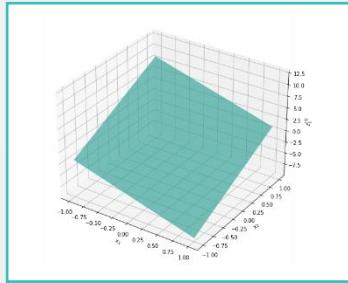
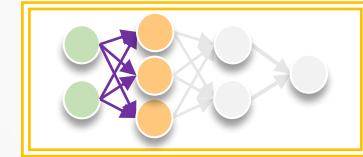
### Example 4 : Binary Classification



# How Deep Learning Work

## Classification

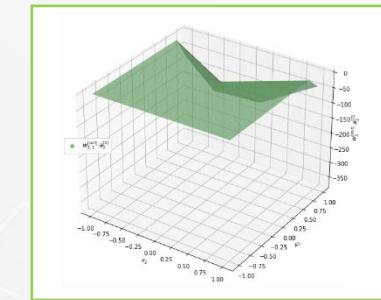
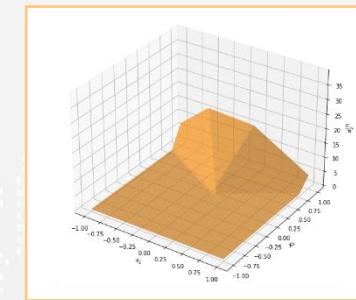
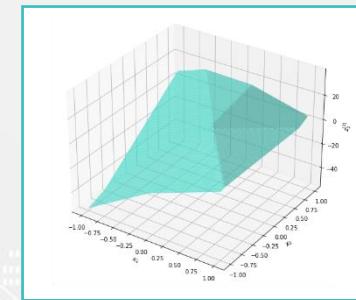
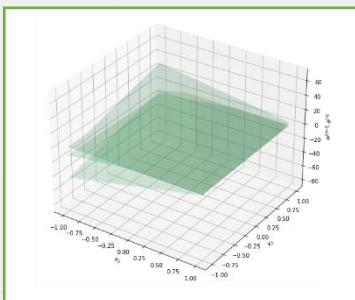
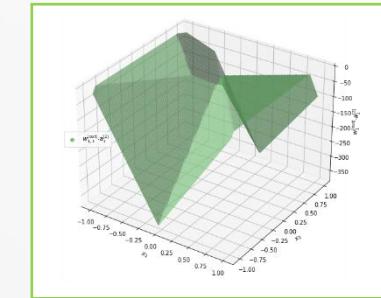
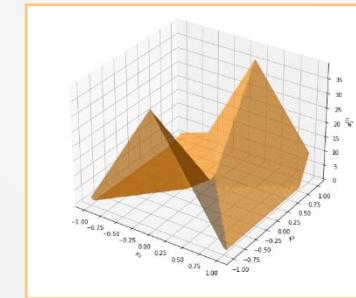
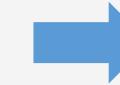
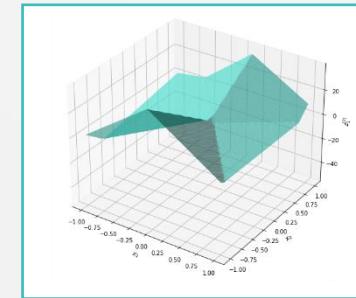
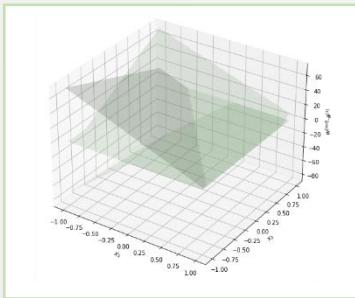
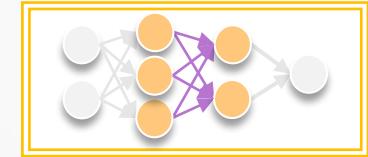
## Example 4



# How Deep Learning Work

Classification

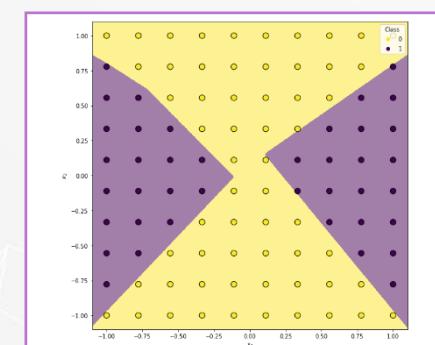
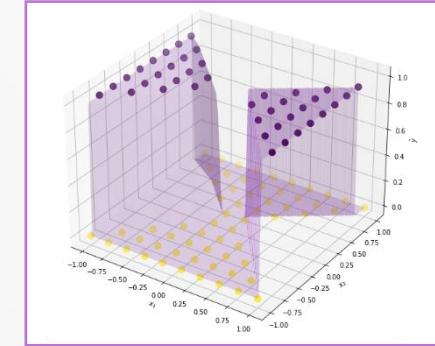
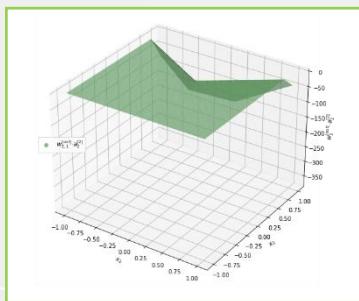
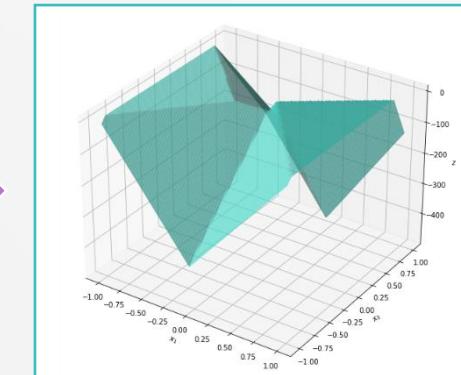
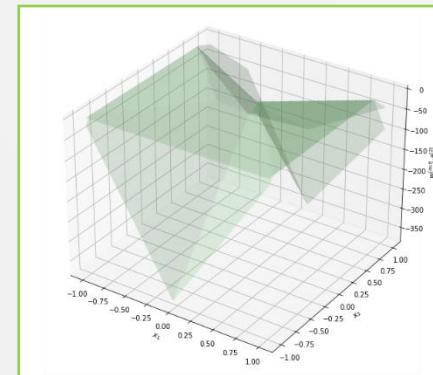
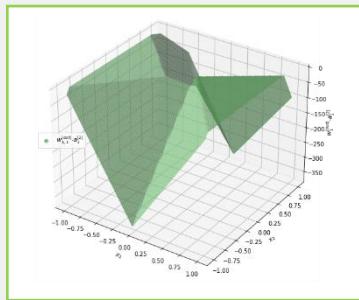
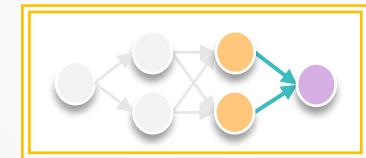
Example 4



# How Deep Learning Work

Classification

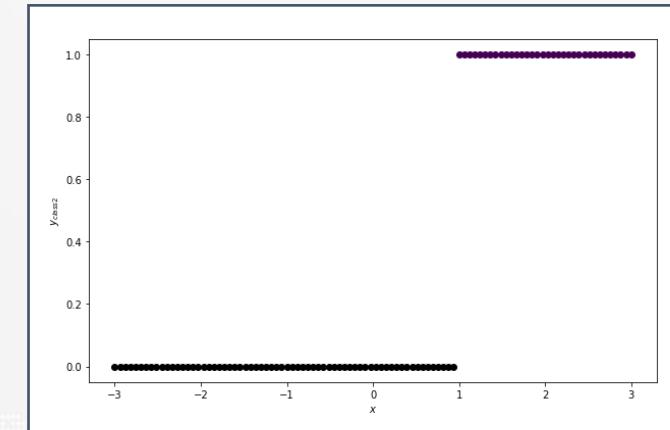
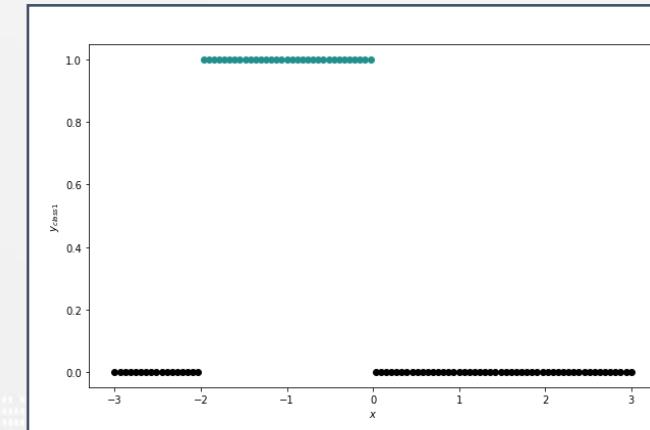
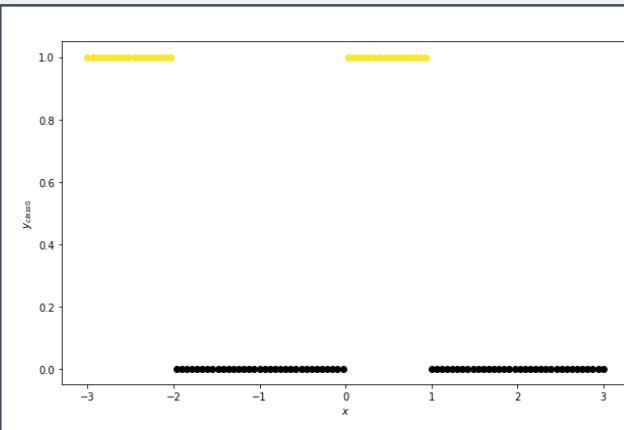
Example 4



# How Neural Network Work

## Classification

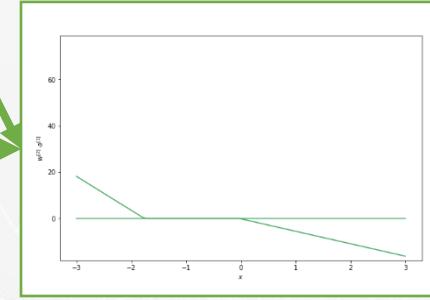
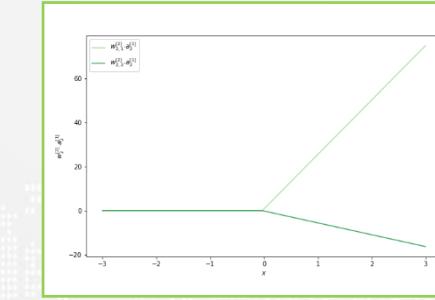
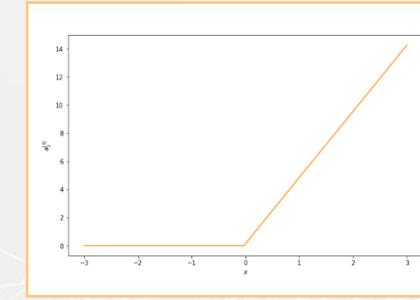
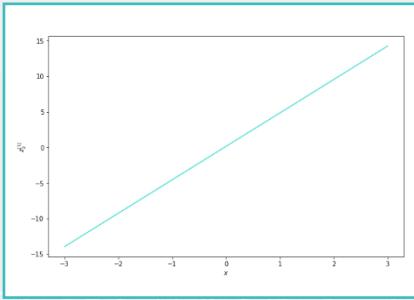
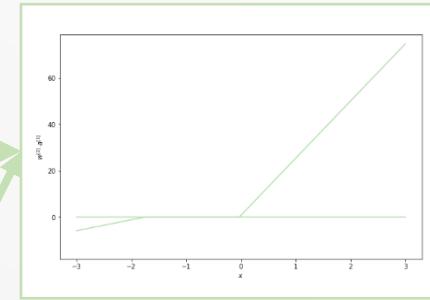
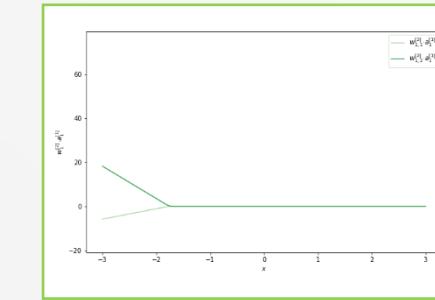
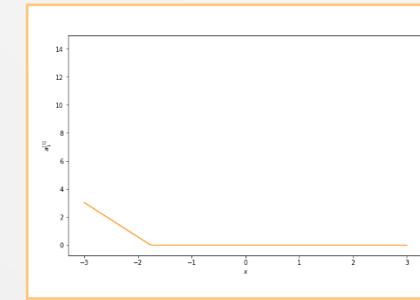
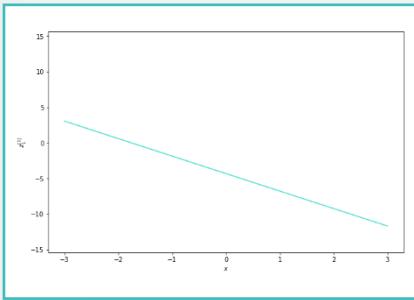
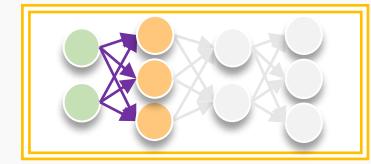
### Example 5 : Multi-Class Classification



# How Deep Learning Work

**Classification**

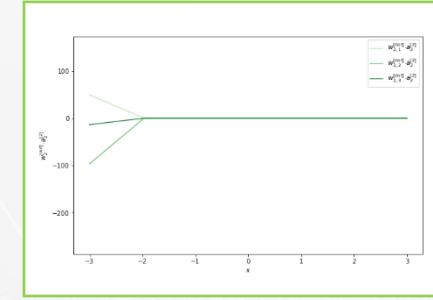
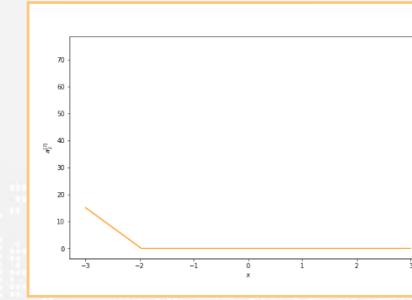
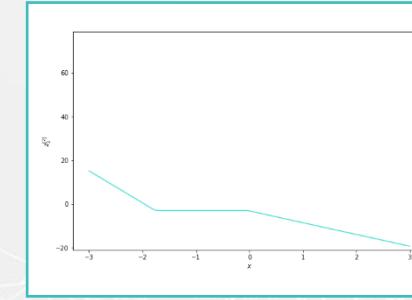
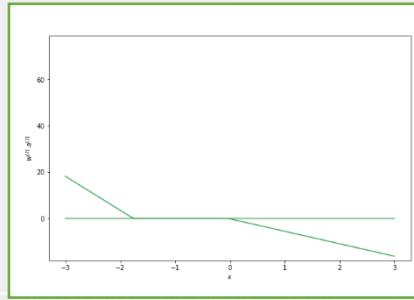
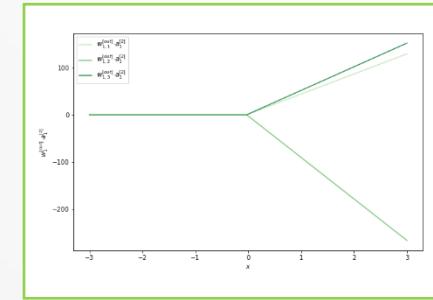
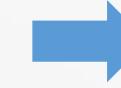
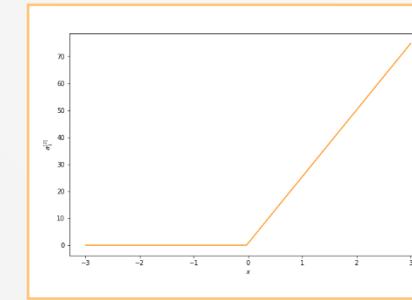
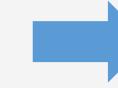
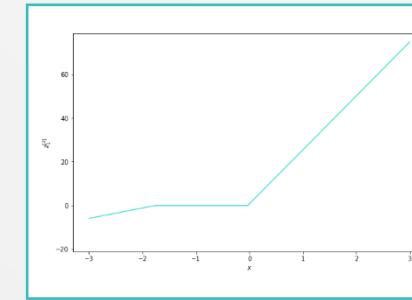
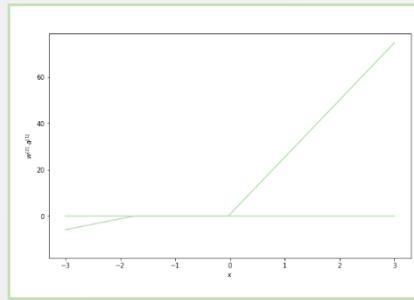
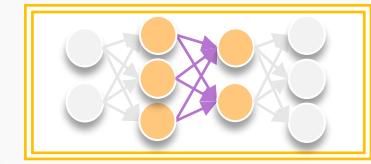
**Example 5**



# How Deep Learning Work

**Classification**

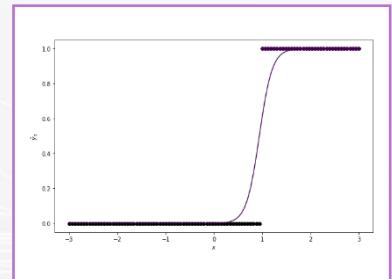
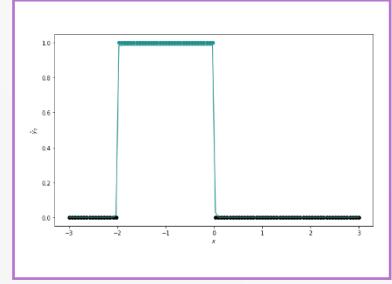
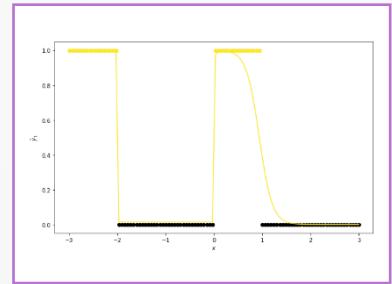
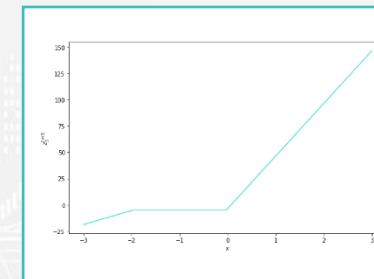
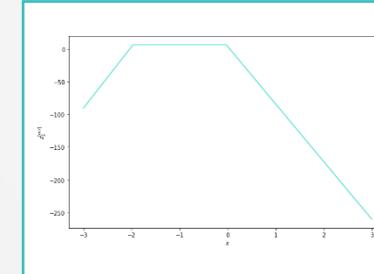
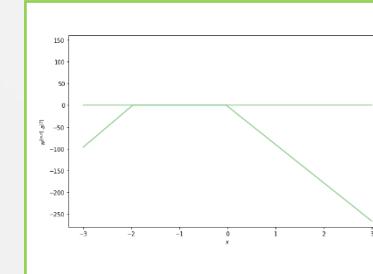
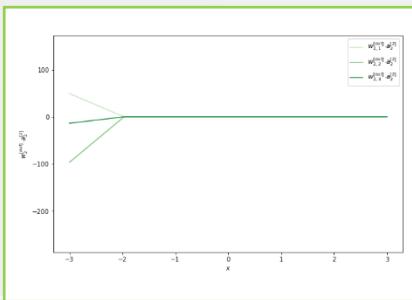
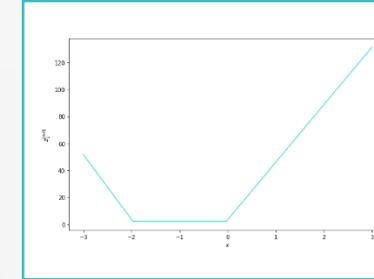
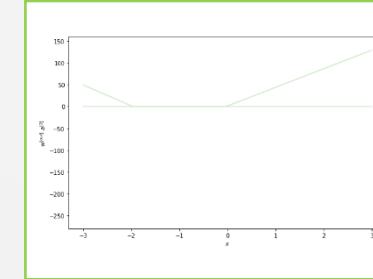
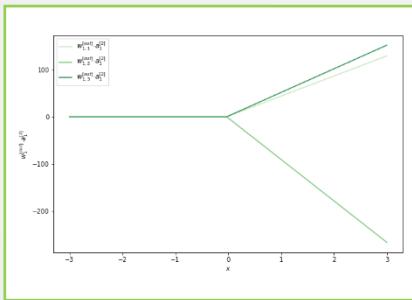
**Example 5**



# How Deep Learning Work

**Classification**

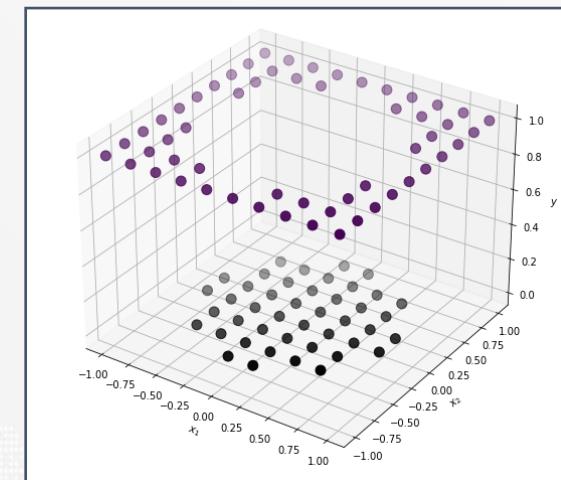
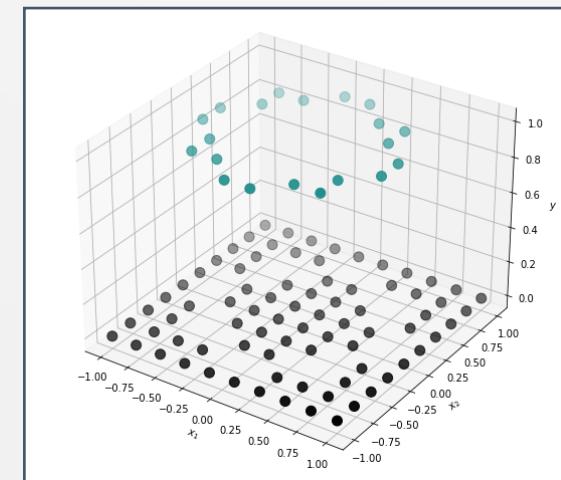
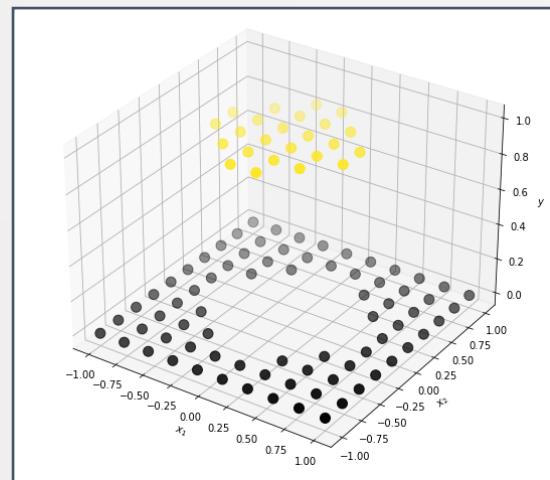
**Example 5**



# How Neural Network Work

## Classification

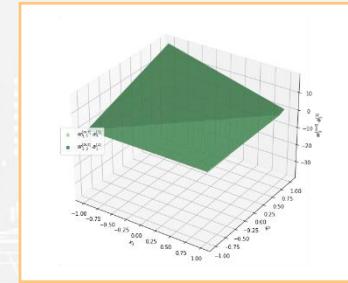
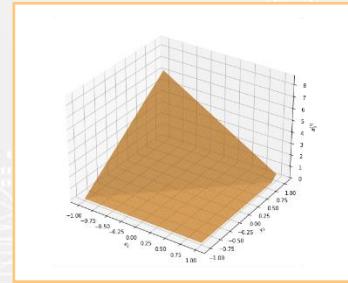
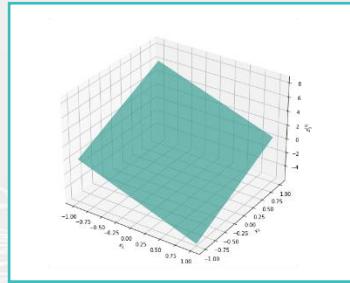
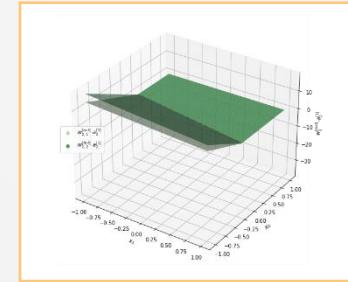
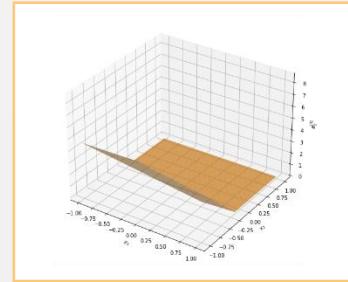
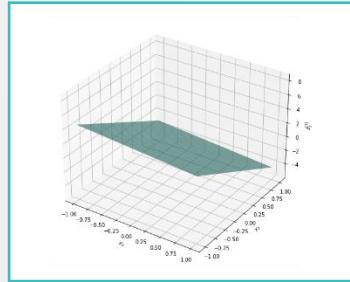
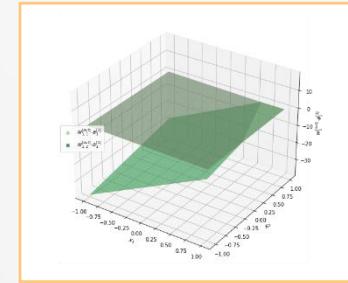
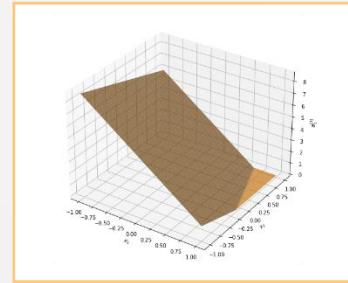
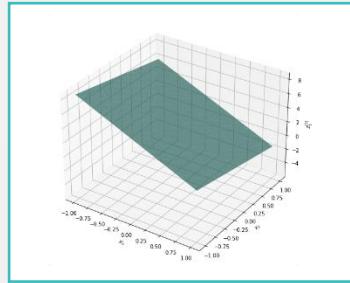
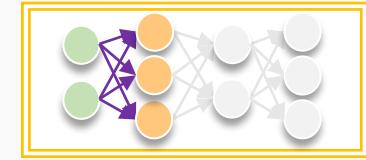
### Example 6 : Multi-Class Classification



# How Deep Learning Work

**Classification**

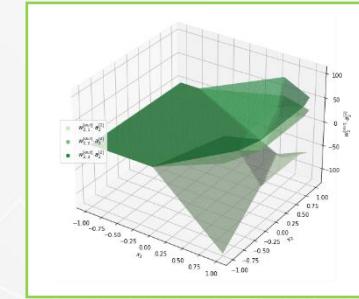
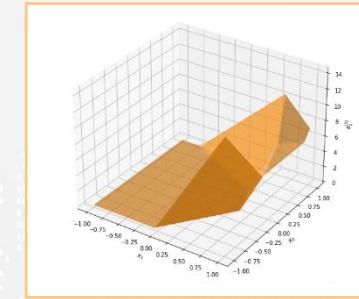
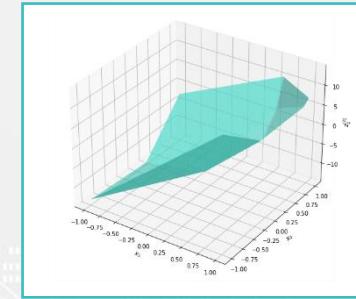
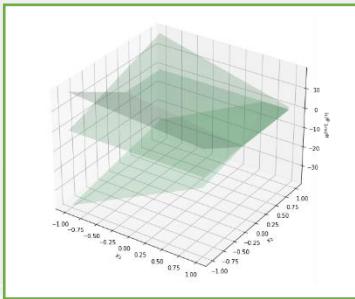
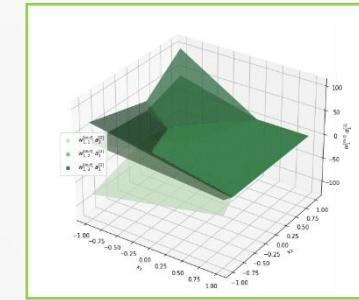
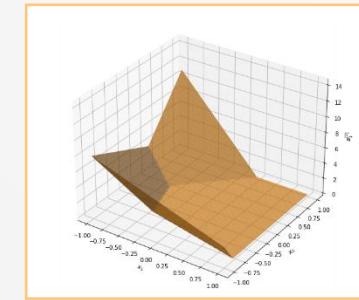
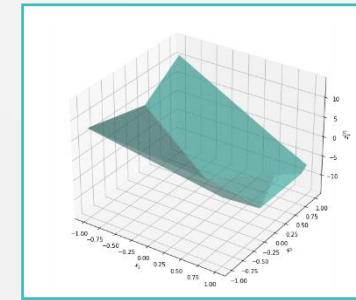
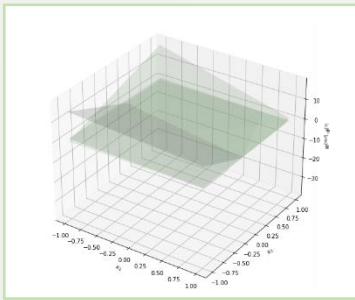
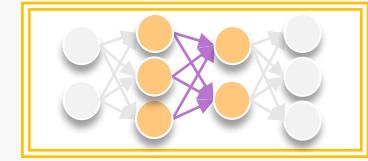
**Example 6**



# How Deep Learning Work

Classification

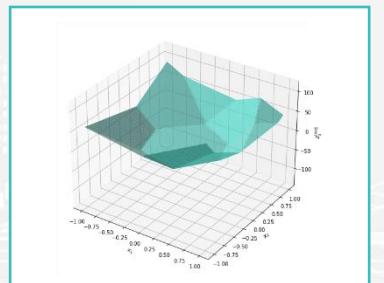
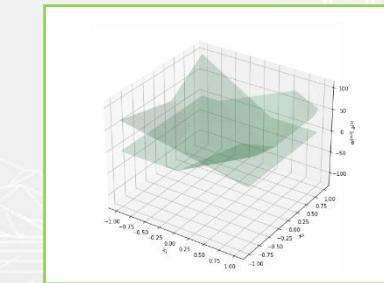
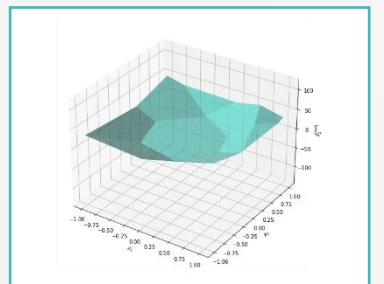
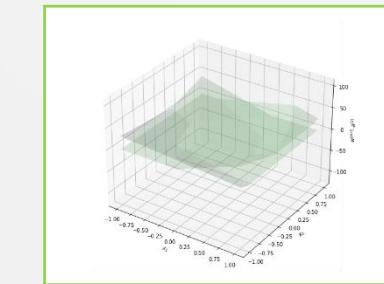
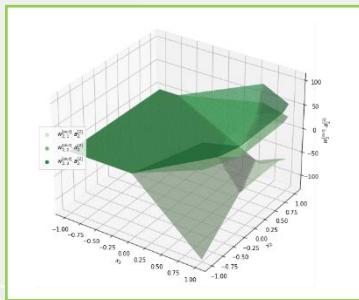
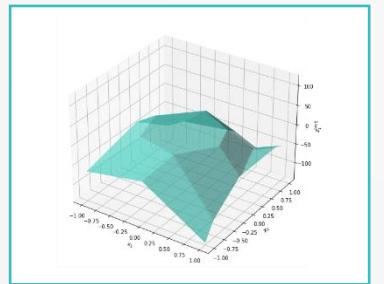
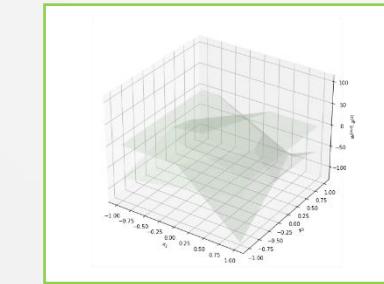
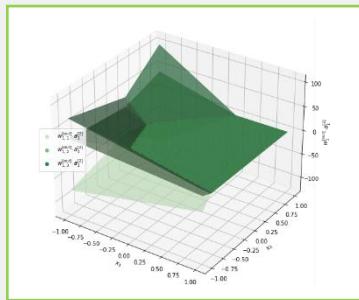
Example 6



# How Deep Learning Work

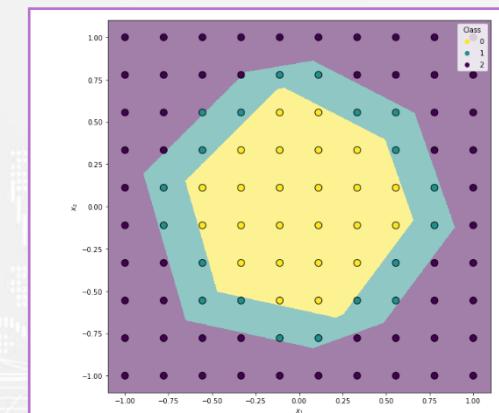
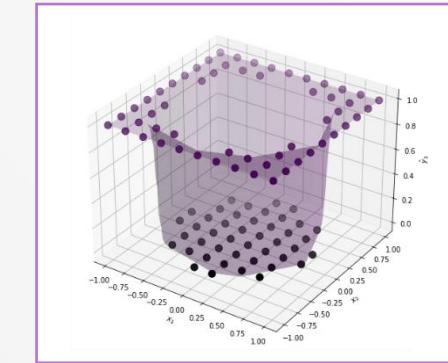
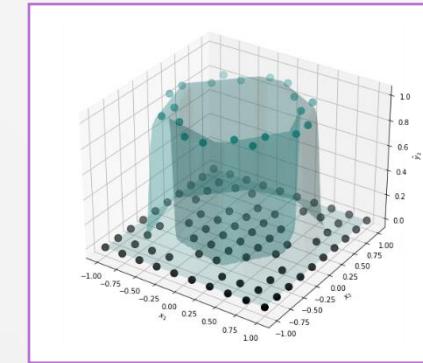
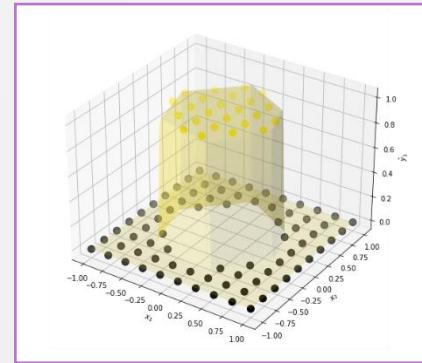
## Classification

## Example 6



# How Neural Network Work

## Classification



# Deep Learning



# Neural Network & Deep Learning

**Neural Network**



**Deep Learning**



**Deep Learning for  
Regression**



**Deep Learning for  
Classification**

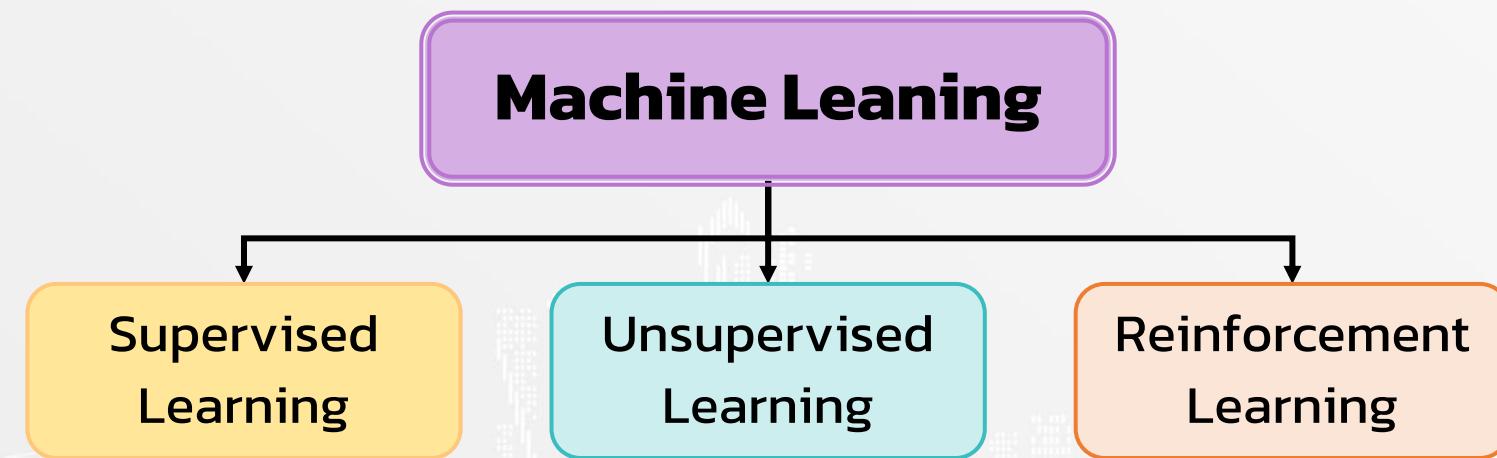


**Workshop**



# Deep Learning for Regression

**Deep Learning** เป็นหัวหนึ่งใน algorithm ประเภท  
supervised learning



# Concept of Supervised Learning

Data → Model → Prediction

# Deep Learning for Regression





# Data

# Data

Data Stating

Data  
Requirement

# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

- $n$  คือ จำนวน sample
- $p$  คือ จำนวน feature

# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

- $x_{2,3}$  คือ sample ที่ 2 feature ที่ 3
- $x_{3,p}$  คือ sample ที่ 3 feature ที่  $p$
- $x_{n,p}$  คือ sample ที่  $n$  feature ที่  $p$
- $y_2$  คือ target ของ sample ที่ 2
- $y_3$  คือ target ของ sample ที่ 3
- $y_n$  คือ target ของ sample ที่  $n$

# Data Stating

## Example

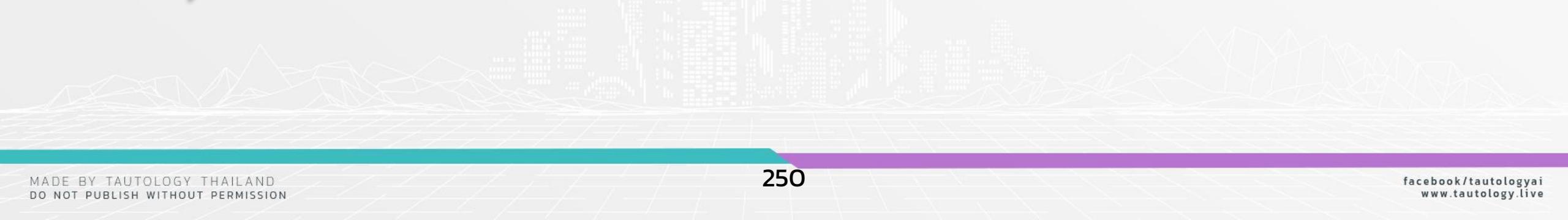
- เราต้องการจะพยากรณ์ราคาบ้าน โดยดูองค์ประกอบจากจำนวนห้องน้ำ, จำนวนห้องนอน, พื้นที่ของบ้าน, ราคาก่อตัวตารางวา

## Data

จำนวนห้องนอน (ห้อง)	จำนวนห้องน้ำ (ห้อง)	พื้นที่ของบ้าน (ตร.ว.)	ราคาก่อตัว (บาท/ตร.ว.)	ราคารายบ้าน (ล้านบาท)
2	2	70	25000	3.5
3	2	120	30000	5.2
1	1	50	20000	1.2
2	1	80	35000	4.0

# Data Stating

- ข้อมูลตามแนวแอกว คือ Sample



จำนวนห้องนอน (ห้อง)	จำนวนห้องน้ำ (ห้อง)	พื้นที่ของบ้าน (ตร.ว.)	ราคาที่ดิน (บาท/ ตร.ว.)	ราคารายบ้าน (ล้านบาท)
2	2	70	25000	3.5
3	2	120	30000	5.2
1	1	50	20000	1.2
2	1	80	35000	4.0

# Data Stating

- ข้อมูลตามแนวหลัก คือ Feature and Target
  - Feature (ตัวแปรต้น) คือ ข้อมูลที่ส่งผลให้เกิด target
  - Target (ตัวแปรตาม) คือ ข้อมูลที่เราสนใจจะพยากรณ์

Feature				Target
จำนวนห้องนอน (ห้อง)	จำนวนห้องน้ำ (ห้อง)	พื้นที่ของบ้าน (ตร.ว.)	ราคาที่ดิน (บาท/ตร.ว.)	ราคาขายบ้าน (ล้านบาท)
2	2	70	25000	3.5
3	2	120	30000	5.2
1	1	50	20000	1.2
2	1	80	35000	4.0

# Data Stating

- Feature and Target
  - เราสามารถแยก และปรับให้เป็น matrix ได้ดังนี้

$$X = \begin{bmatrix} 2 & 2 & 70 & 25000 \\ 3 & 2 & 120 & 30000 \\ 1 & 1 & 50 & 20000 \\ 2 & 1 & 80 & 35000 \end{bmatrix}$$

$$y = \begin{bmatrix} 3.5 \\ 5.2 \\ 1.2 \\ 4.0 \end{bmatrix}$$

# Data

**Data Stating**



**Data  
Requirement**



# Data Requirement

- ข้อมูลต้องอยู่ในรูปแบบของตาราง
- ข้อมูลต้องเป็น numerical

จำนวนห้องนอน (ห้อง)	จำนวนห้องน้ำ (ห้อง)	พื้นที่ของบ้าน (ตร.ว.)	ราคาที่ดิน (บาท/ตร.ว.)	ราคายาบ้าน (ล้านบาท)
2	2	70	25000	3.5
3	2	120	30000	5.2
1	1	50	20000	1.2
2	1	80	35000	4.0

# Data Requirement

- ตัวอย่างข้อมูลที่สามารถใช้งานได้เลย และยังไม่สามารถใช้งานได้

พื้นที่ของบ้าน (ตร.ว.)	ราคาที่ดิน (บาท/ตร.ว.)	ราคายาบ้าน (ล้านบาท)
70	25000	3.5
120	30000	5.2
50	20000	1.2
80	35000	4.0



exp (yr)	position	salary
1	secretary	26000
4	engineer	48000
3	accountant	41500
1	engineer	26500



# Data Requirement

- เราสามารถแปลงได้โดยสามารถใช้ความรู้ในส่วนของ Data Preparation

exp (yr)	position	salary
1	secretary	26000
4	engineer	48000
3	accountant	41500
1	engineer	26500



exp (yr)	accountant	engineer	secretary	salary
1	0	0	1	26000
4	0	1	0	48000
3	1	0	0	41500
1	0	1	0	26500

# Data Requirement



For more information



## Data Preparation

# Data

**Data Stating**



**Data  
Requirement**



# Deep Learning for Regression





# Model

# Model

Assumption

Real Face of the Model

Cost Function and Cost Landscape

How to Create Model

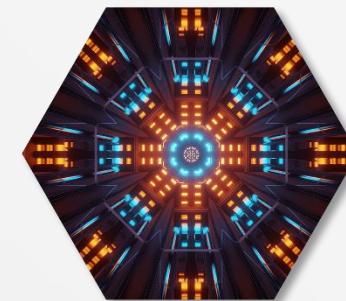
# Assumption

1. Non Linear Relationship
2. Normality of Residuals
3. Homoscedasticity
4. No Missing Features
5. No Multicollinearity

# Assumption



For more information



Model Improvement  
In DL101

# Model

**Assumption**



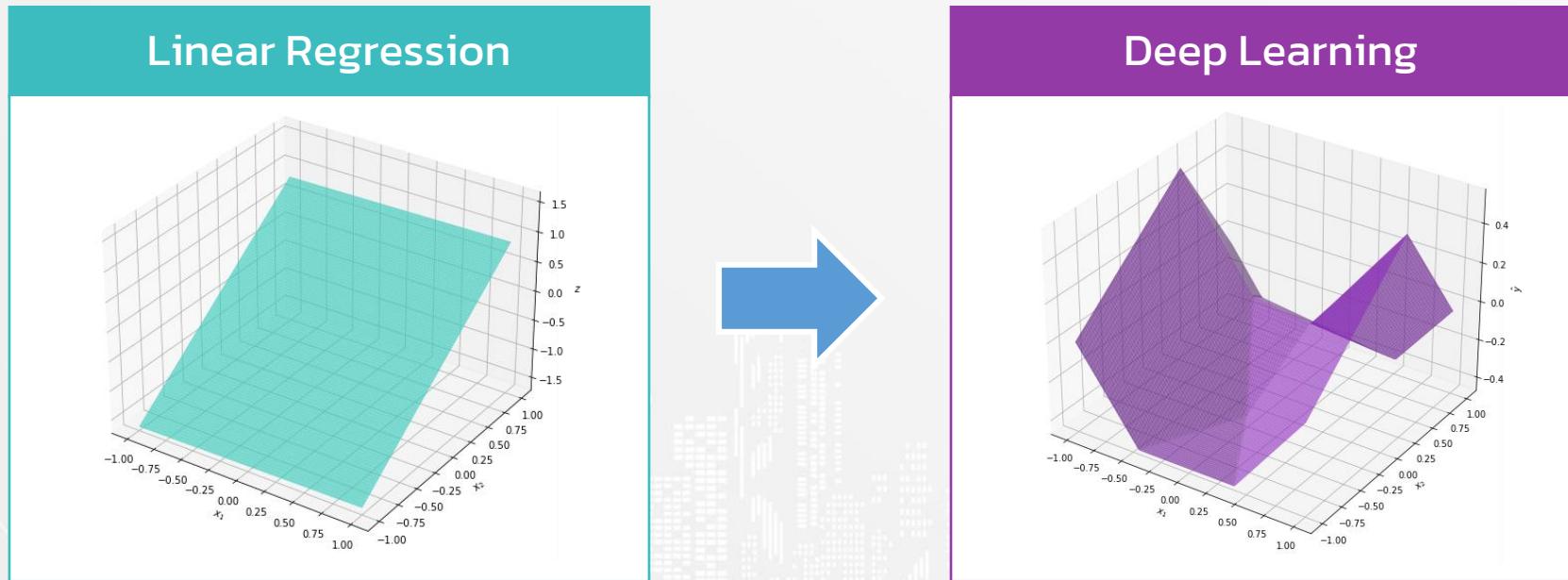
**Real Face of the Model**

**Cost Function and Cost Landscape**

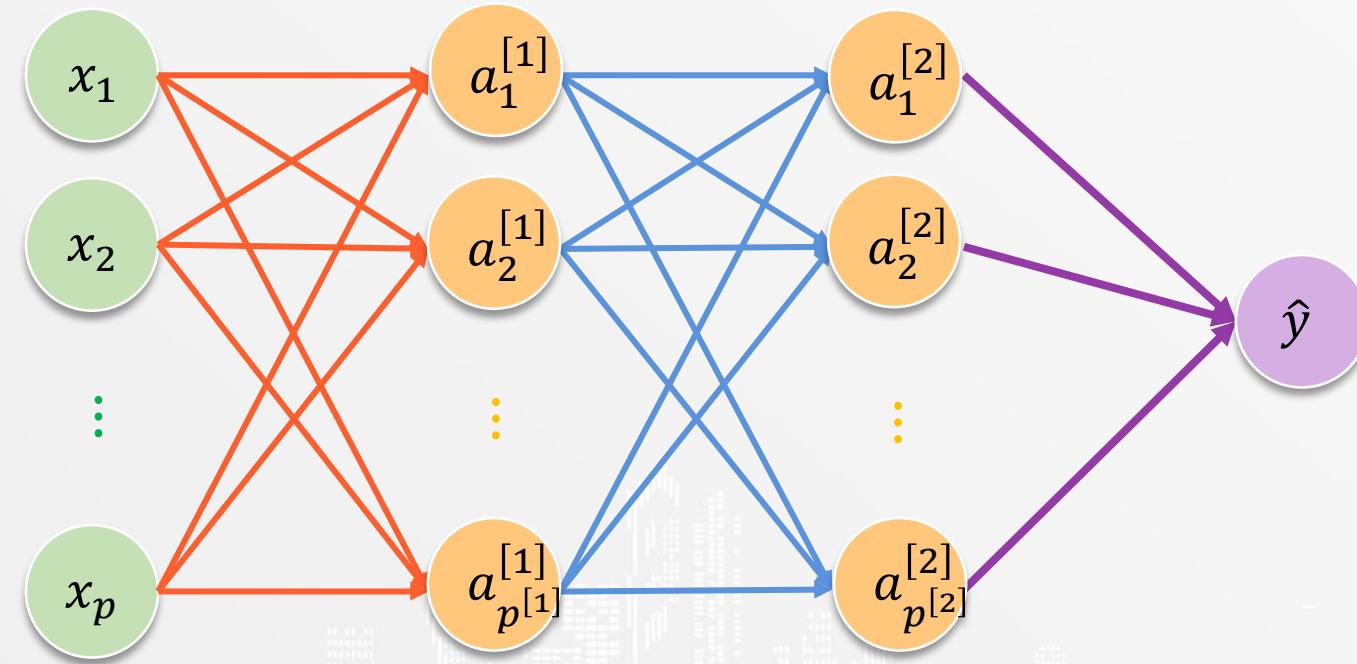
**How to Create Model**

# Real Face of the Model

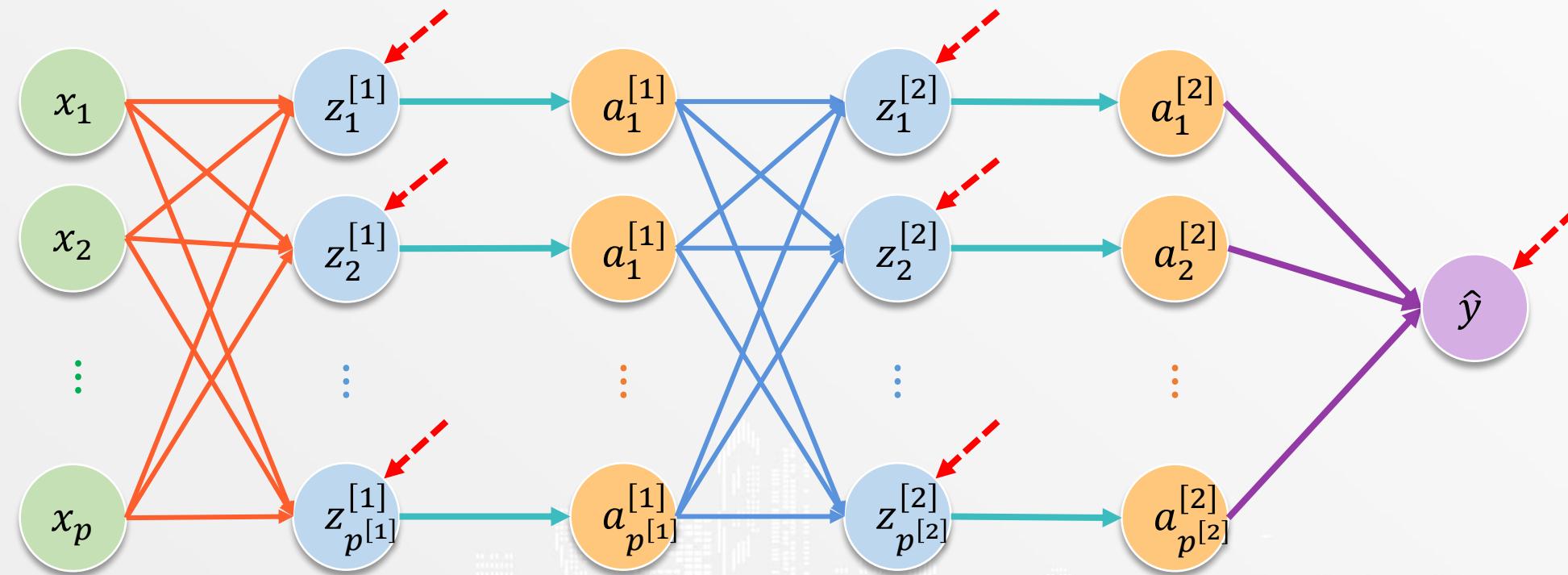
แนวคิดของ deep learning คือ การนำเอา hyperplane หัก ๆ มาประกอบกัน เพื่อให้สามารถประมาณ nonlinear function ที่ซับซ้อนได้



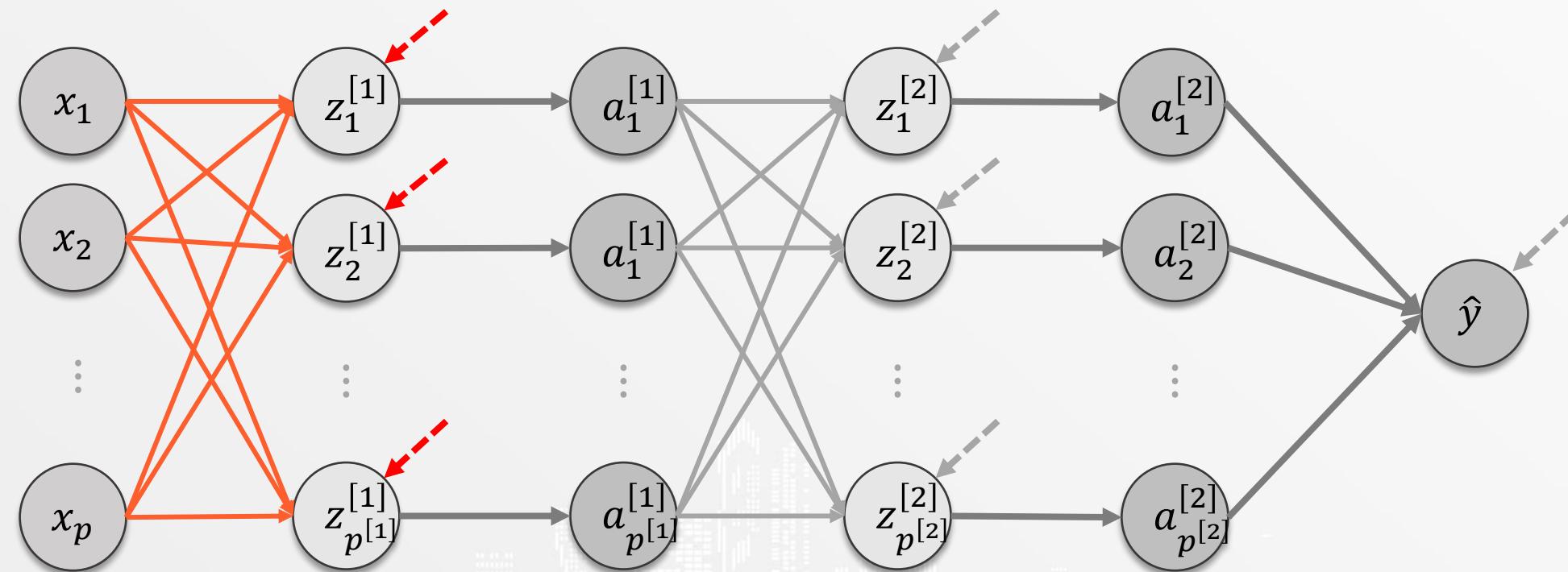
# Real Face of the Model



# Real Face of the Model



# Real Face of the Model

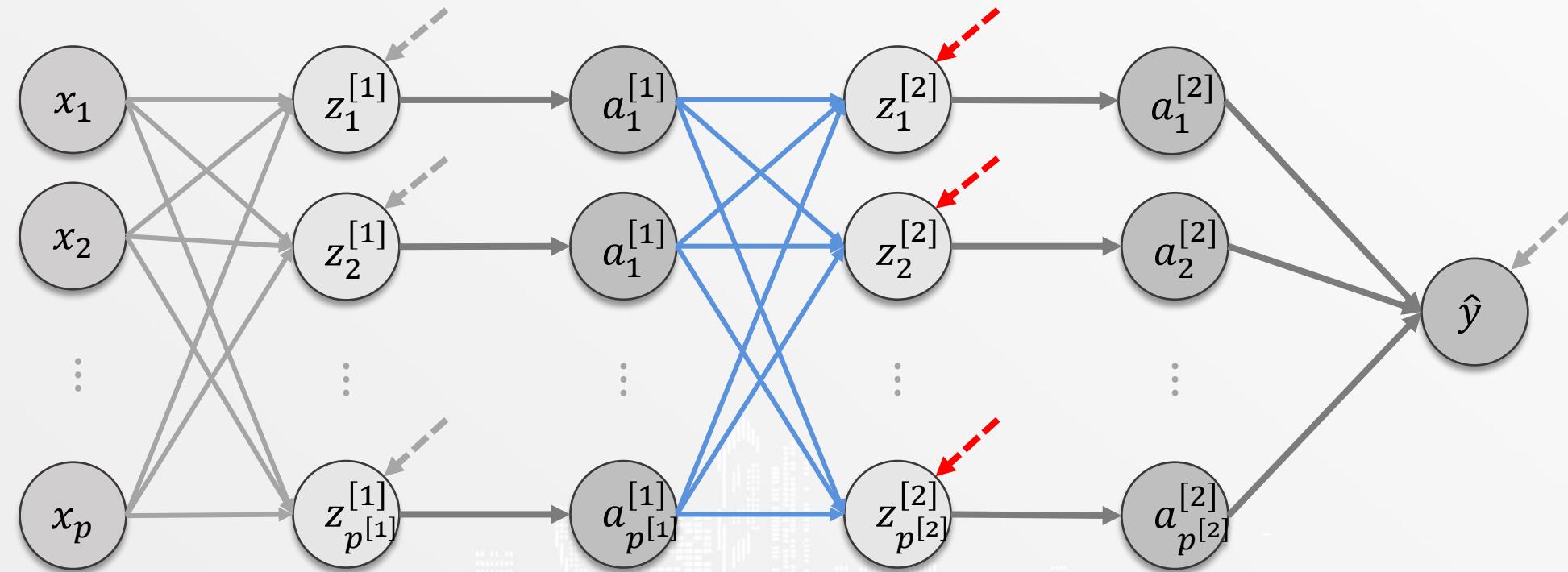


# Real Face of the Model

$$b^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & \dots & b_{p^{[1]}}^{[1]} \end{bmatrix}$$

$$W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & \dots & w_{1,p^{[1]}}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & \dots & w_{2,p^{[1]}}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[1]} & w_{p,2}^{[1]} & \dots & w_{p,p^{[1]}}^{[1]} \end{bmatrix}$$

# Real Face of the Model

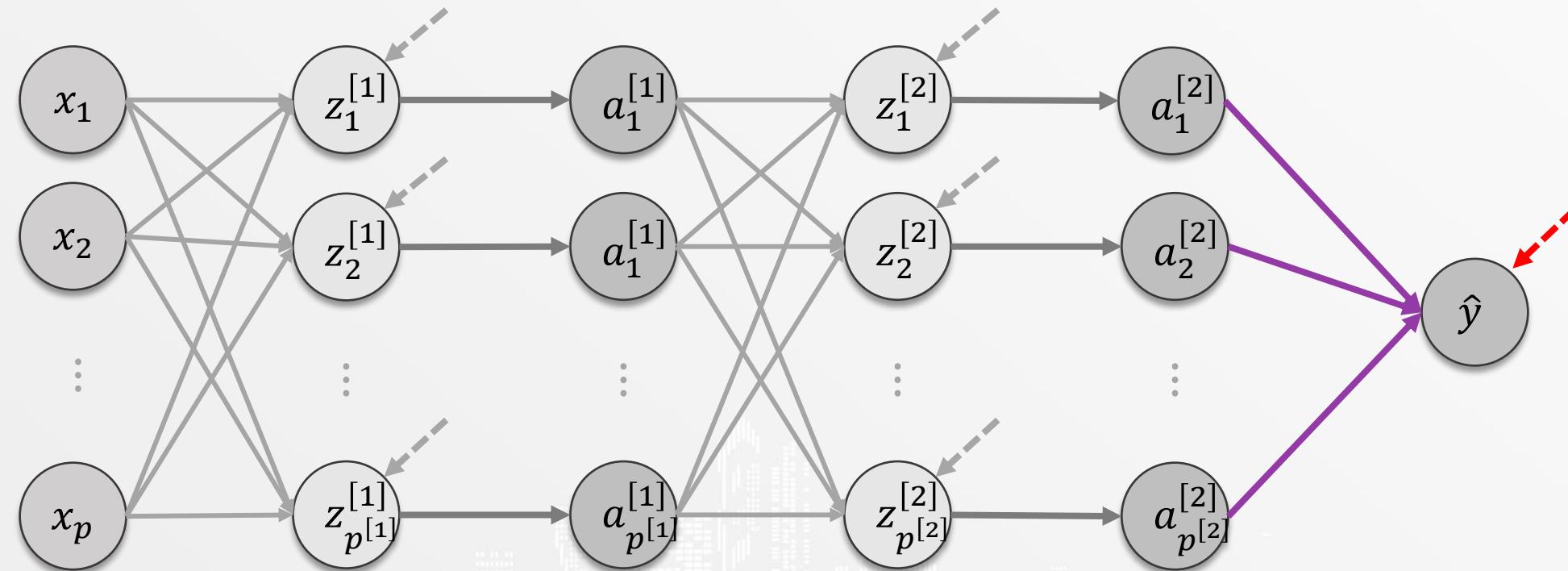


# Real Face of the Model

$$b^{[2]} = \begin{bmatrix} b_1^{[2]} & b_2^{[2]} & \dots & b_{p^{[2]}}^{[2]} \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} w_{1,1}^{[2]} & w_{1,2}^{[2]} & \dots & w_{1,p^{[2]}}^{[2]} \\ w_{2,1}^{[2]} & w_{2,2}^{[2]} & \dots & w_{2,p^{[2]}}^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[2]} & w_{p,2}^{[2]} & \dots & w_{p,p^{[2]}}^{[2]} \end{bmatrix}$$

# Real Face of the Model



# Real Face of the Model

$$b^{[out]} = [b^{[out]}]$$

$$W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \\ \vdots \\ w_{p^{[2]}}^{[out]} \end{bmatrix}$$

# Real Face of the Model

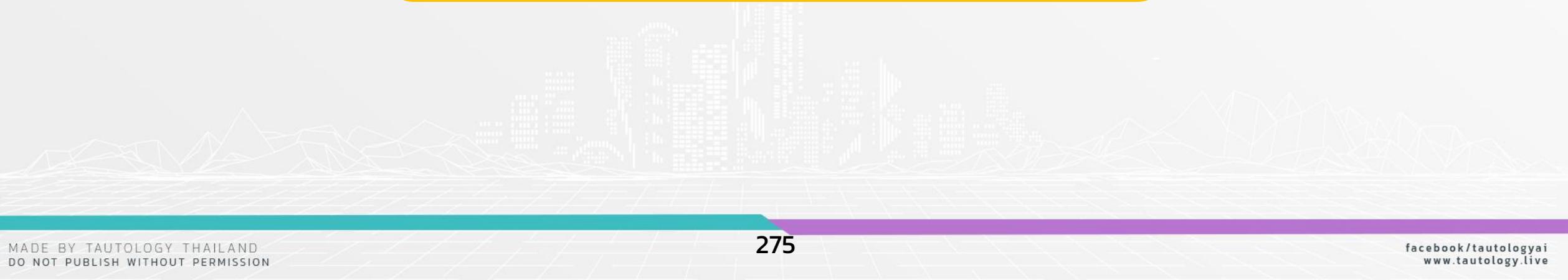
$b^{[1]}, b^{[2]}, b^{[out]}$

$W^{[1]}, W^{[2]}, W^{[out]}$

# Real Face of the Model

$b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$

$W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$



# Real Face of the Model

เราต้องการหา  $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ  $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$  ที่ทำให้ cost function ต่ำที่สุด

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model**



# Cost Function & Cost Landscape

**Cost function** ที่เราจะใช้ในการสร้าง model คือ

$$-\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

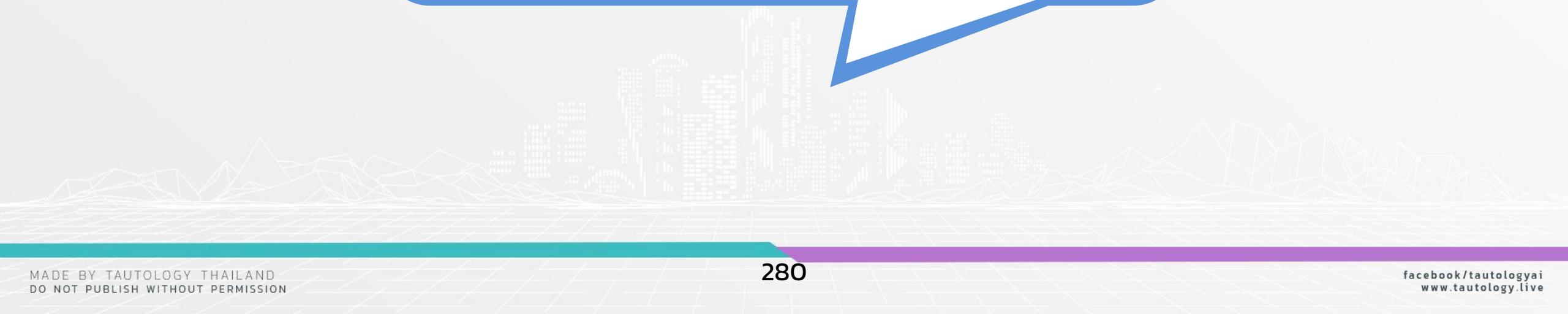
โดยสูตรข้างต้นมีชื่อว่า mean square error หรือ MSE

# Cost Function & Cost Landscape

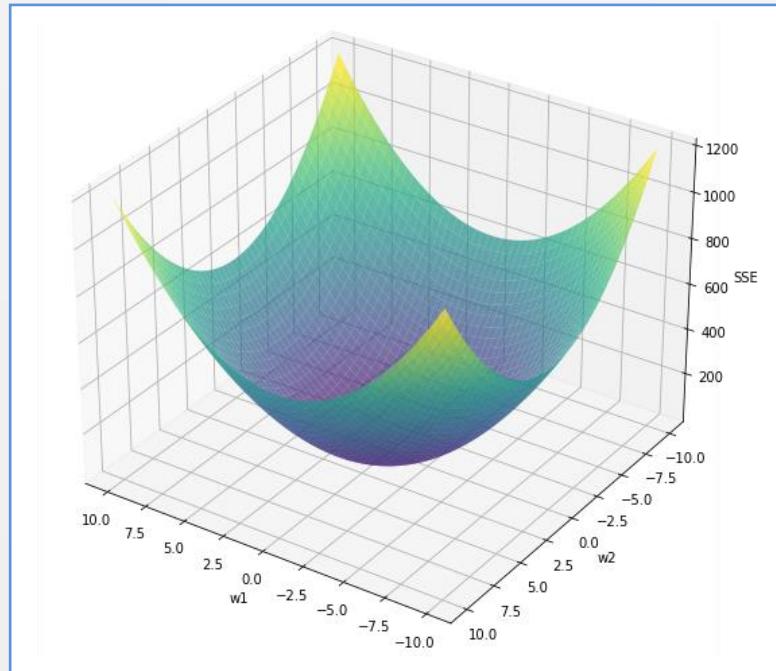
**Q :** ทำไมไม่ใช้ sum of squared error  
หรือ SSE เป็น cost function เมื่อตอน  
สร้าง linear regression

# Cost Function & Cost Landscape

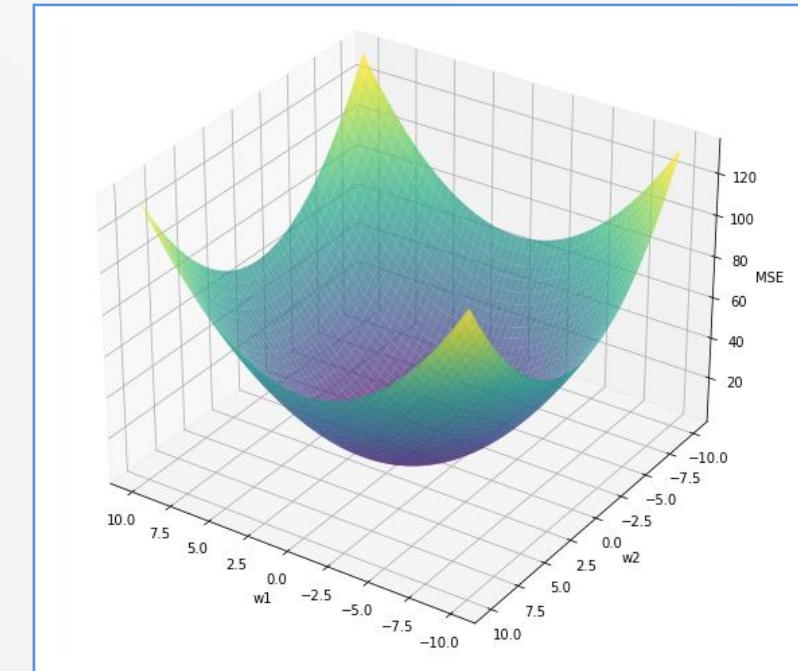
A: เนื่องจาก deep learning ใช้ gradient descent ในการสร้าง model ทำให้การใช้ค่าเฉลี่ยมีความหมายมากกว่า



# Cost Function & Cost Landscape

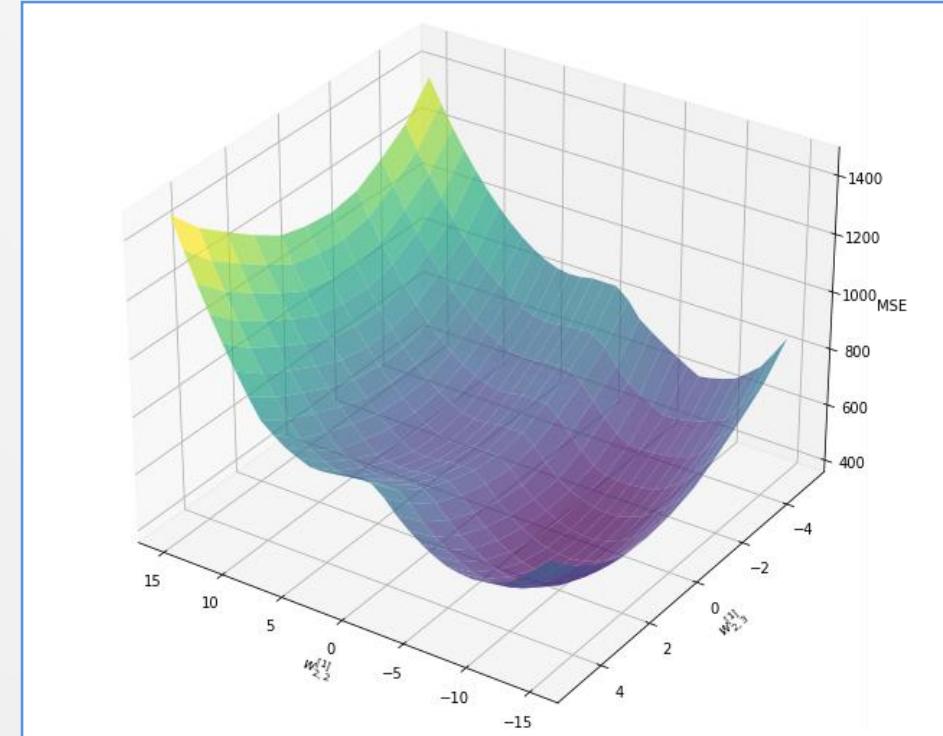


กราฟแสดง cost landscape ของ linear regression โดยที่ cost function เป็น SSE



กราฟแสดง cost landscape ของ linear regression โดยที่ cost function เป็น MSE

# Cost Function & Cost Landscape



กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น MSE

# Cost Function & Cost Landscape

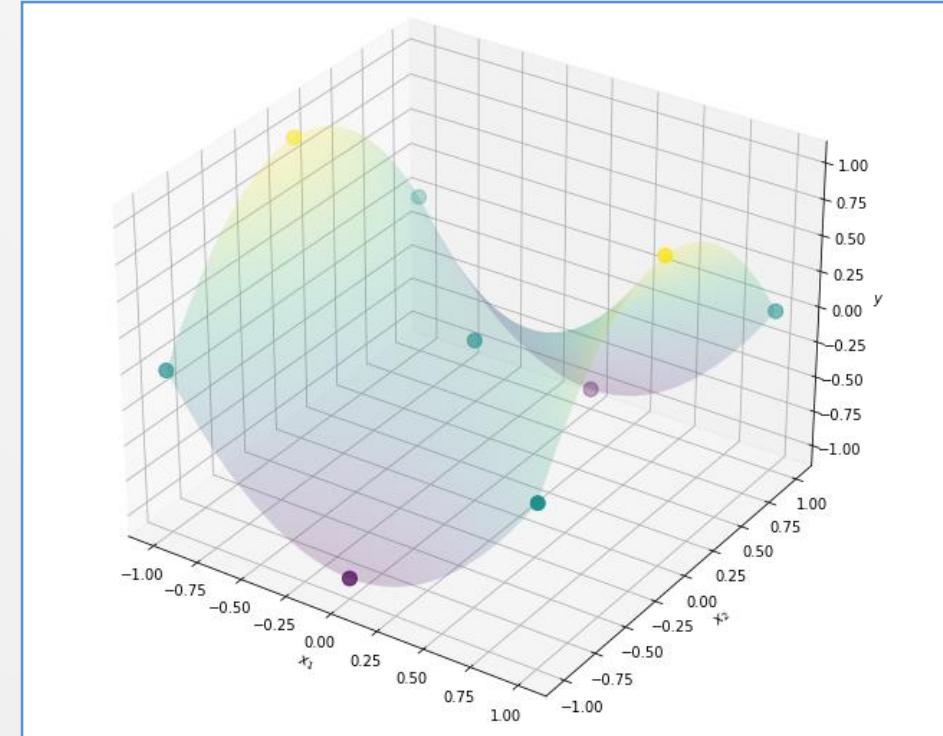
ตัวอย่างการ plot cost landscape  
ของ deep learning

$x_1$	$x_2$	y
-1	-1	2
0	-1	1
1	-1	2
-1	0	1
0	0	0

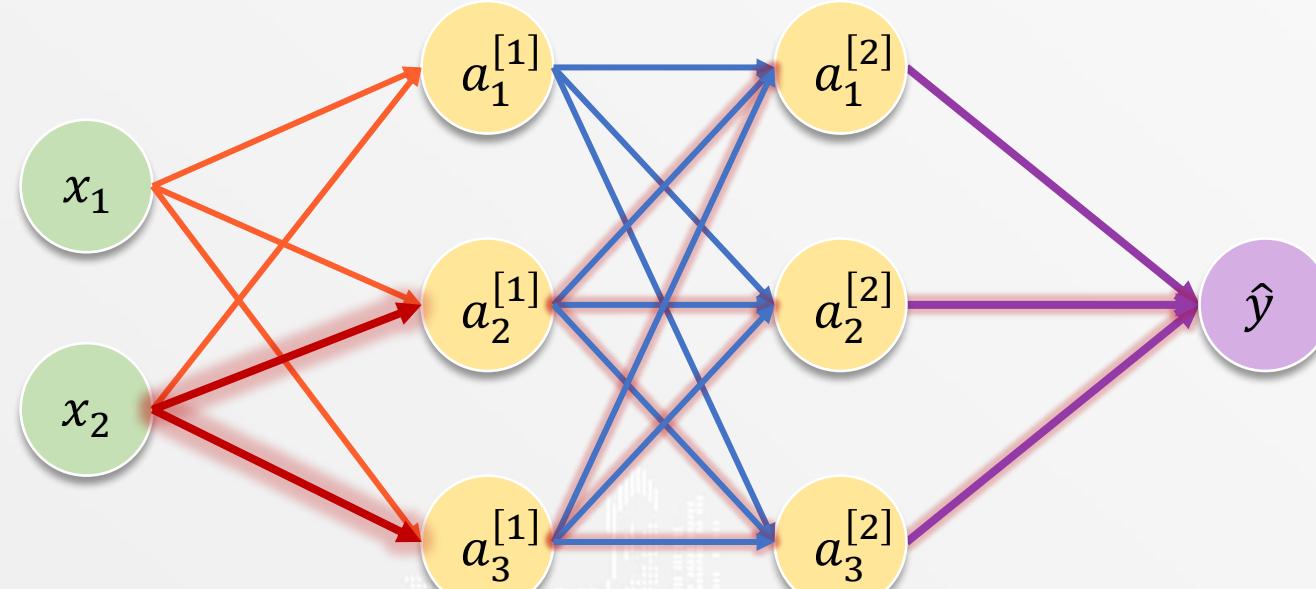
$x_1$	$x_2$	y
1	0	1
-1	1	2
0	1	1
1	1	0

ตารางแสดง toy dataset

# Cost Function & Cost Landscape



# Cost Function & Cost Landscape



# Cost Function & Cost Landscape

$$\begin{aligned}\hat{y} &= b^{[out]} + w_1^{[out]} a_1^{[1]} + w_2^{[out]} a_2^{[1]} + w_3^{[out]} a_3^{[1]} \\ a_1^{[1]} &= \text{ReLU} \left( b_1^{[1]} + w_{1,1}^{[1]} x_1 + w_{2,1}^{[1]} x_2 \right) \\ a_2^{[1]} &= \text{ReLU} \left( b_2^{[1]} + w_{1,2}^{[1]} x_1 + w_{2,2}^{[1]} x_2 \right) \\ a_3^{[1]} &= \text{ReLU} \left( b_3^{[1]} + w_{1,3}^{[1]} x_1 + w_{2,3}^{[1]} x_2 \right)\end{aligned}$$

# Cost Function & Cost Landscape

กำหนดให้

$$b^{[1]} = [-2 \quad -1 \quad 1]$$

$$W^{[1]} = \begin{bmatrix} -1 & 3 & -3 \\ 0 & ? & ? \end{bmatrix}$$

พิจารณา  $w_{2,2}^{[1]}, w_{2,3}^{[1]}$

# Cost Function & Cost Landscape

กำหนดให้

$$b^{[2]} = [-2 \quad -1 \quad -2]$$

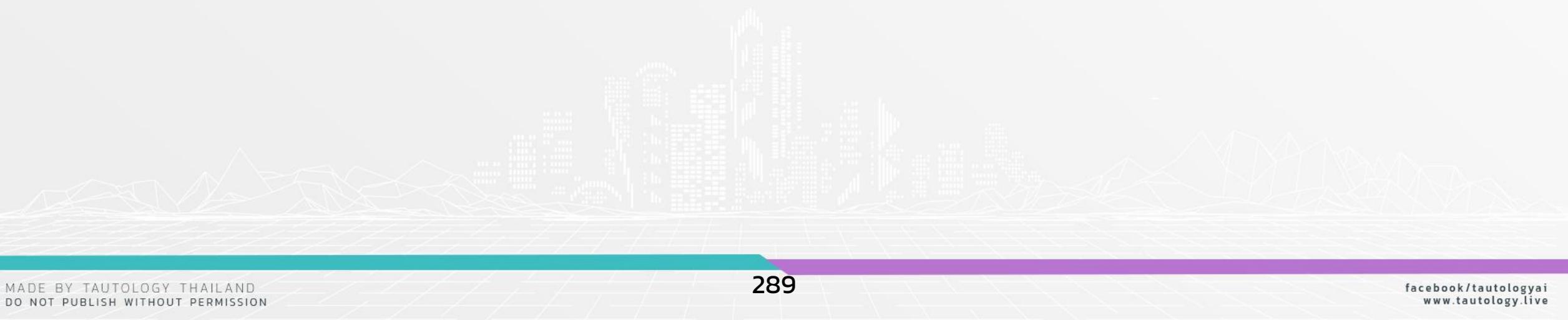
$$W^{[2]} = \begin{bmatrix} -1 & 1 & 0 \\ 1 & 1 & 0 \\ 3 & -2 & 0 \end{bmatrix}$$

# Cost Function & Cost Landscape

กำหนดให้

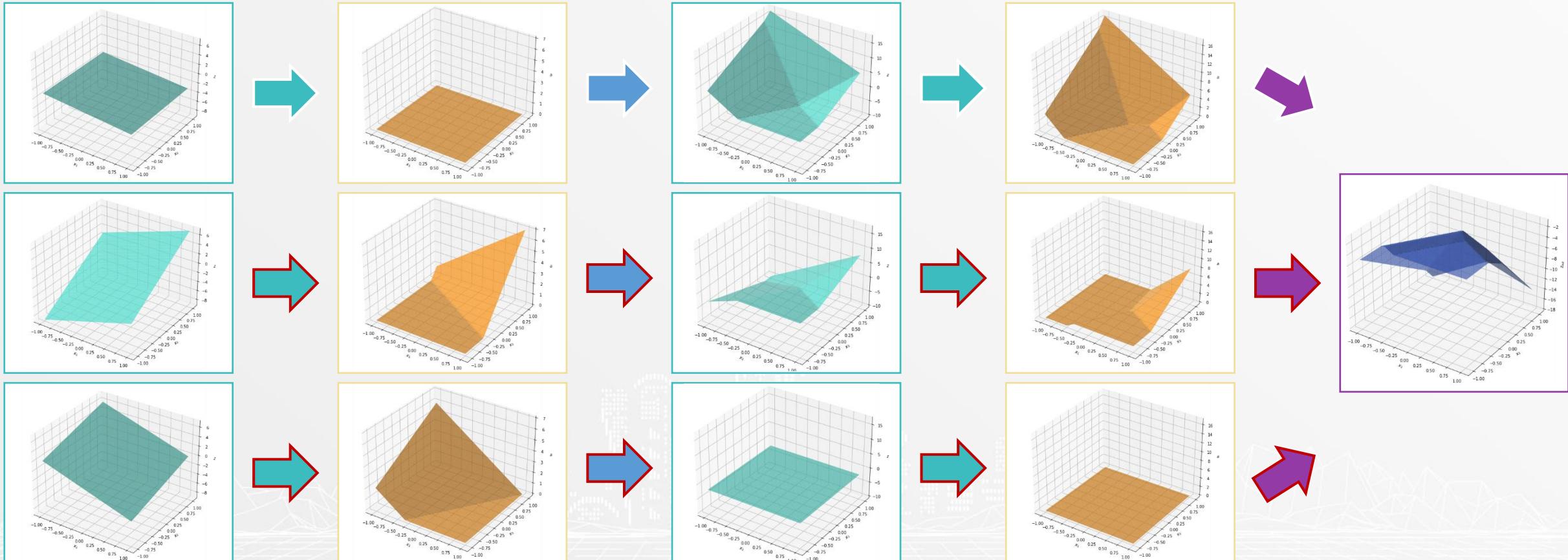
$$b^{[out]} = [-1]$$

$$W^{[out]} = \begin{bmatrix} -1 \\ -1 \\ -2 \end{bmatrix}$$



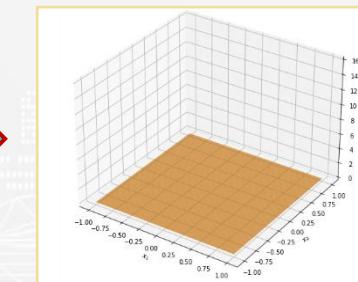
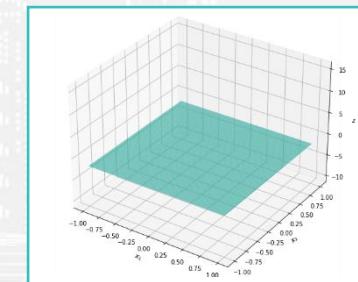
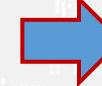
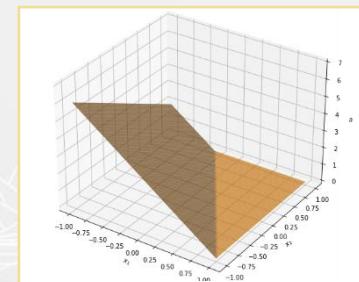
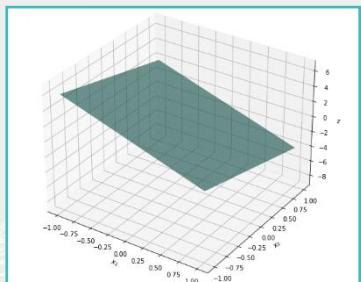
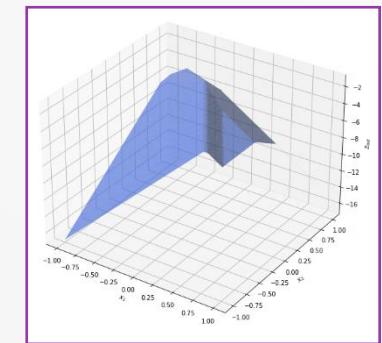
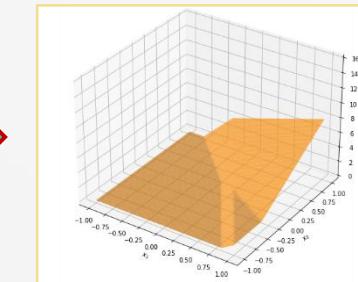
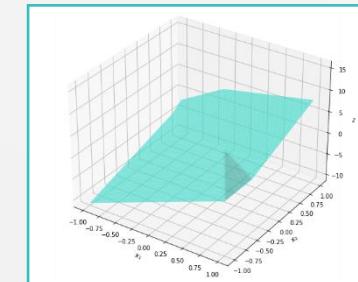
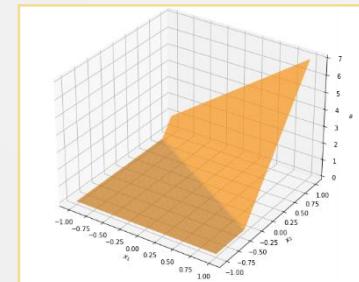
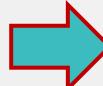
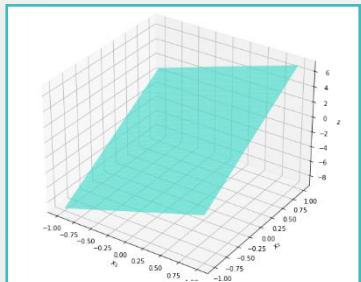
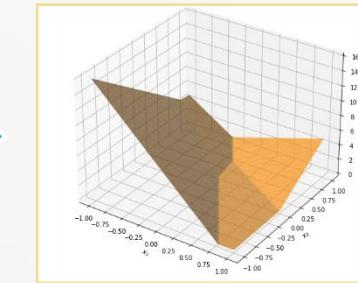
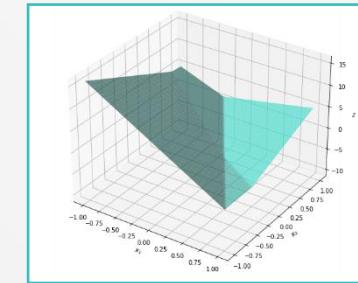
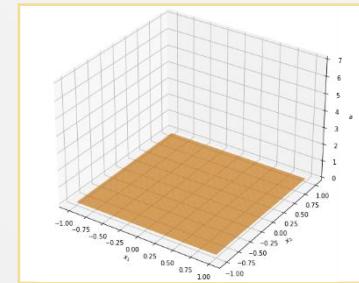
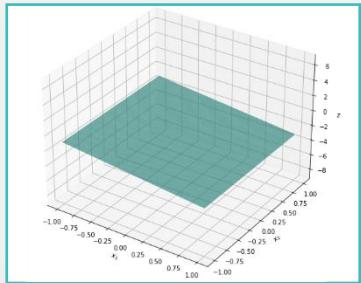
# Cost Function & Cost Landscape

$$w_{2,2}^{[1]} = 5, \quad w_{2,3}^{[1]} = 2$$



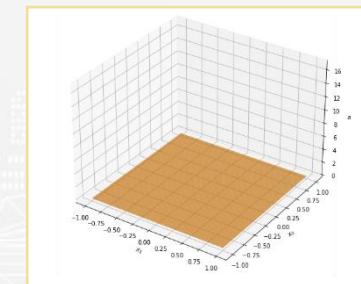
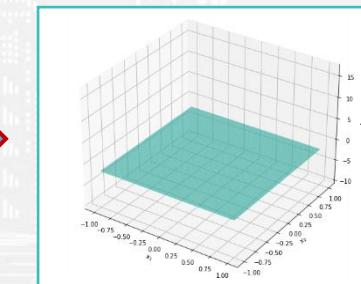
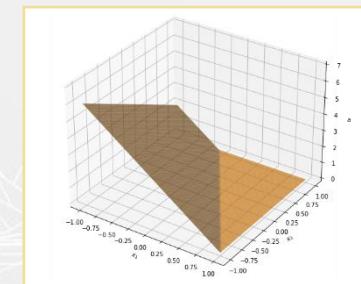
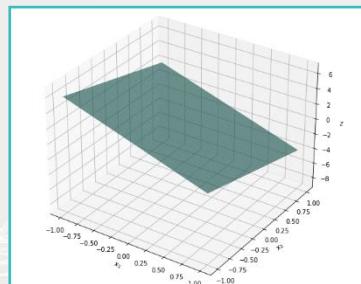
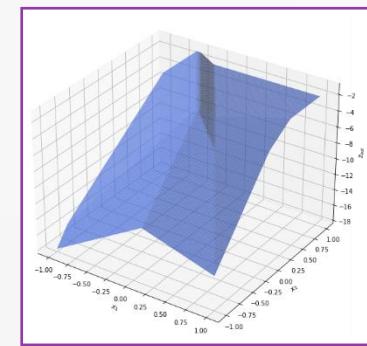
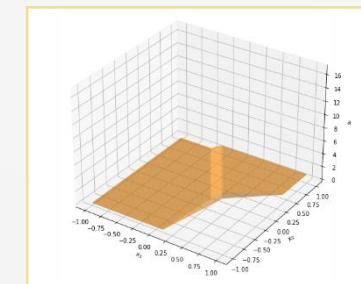
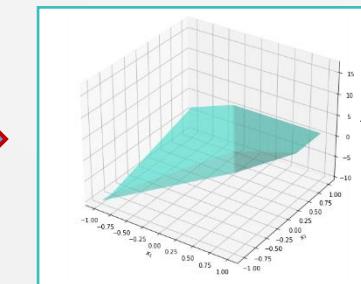
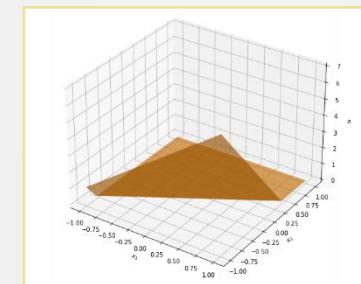
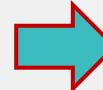
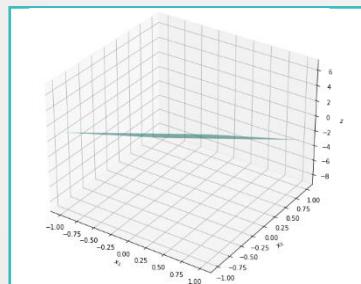
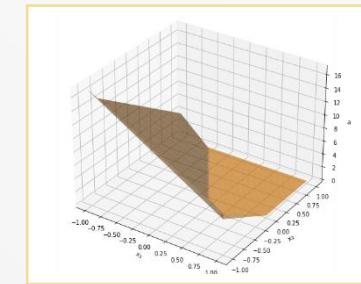
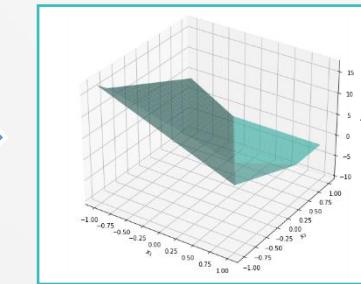
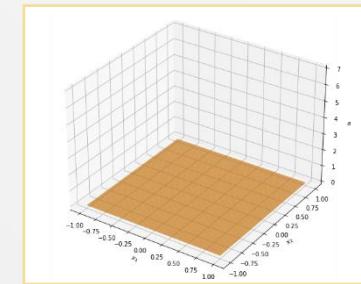
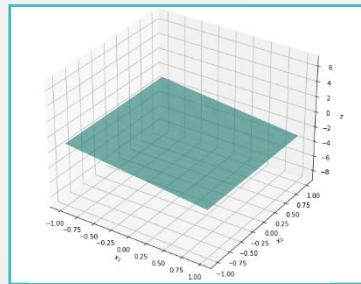
# Cost Function & Cost Landscape

$$w_{2,2}^{[1]} = 5, \quad w_{2,3}^{[1]} = -2$$



# Cost Function & Cost Landscape

$$w_{2,2}^{[1]} = -2, \quad w_{2,3}^{[1]} = 5$$



# Cost Function & Cost Landscape

ปรับเปลี่ยน  $\hat{y}$  กับ  $y$  เพื่อคำนวณ cost function ( $Cost = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$ )

$w_{2,2}^{[1]} = -15$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -14$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -13$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -12$ $w_{2,3}^{[1]} = -5$	...	$w_{2,2}^{[1]} = 12$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 13$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 14$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 15$ $w_{2,3}^{[1]} = 5$	
$\hat{y}_1$	-8	-8	-8	-8	...	-17	-17	-17	-17
$\hat{y}_2$	-13	-12	-11	-11	...	-8	-8	-8	-8
$\hat{y}_3$	-21	-20	-19	-18	...	-2	-2	-2	-2
$\hat{y}_4$	-11	-11	-11	-11		-11	-11	-11	-11
$\hat{y}_5$	-2	-2	-2	-2		-2	-2	-2	-2

# Cost Function & Cost Landscape

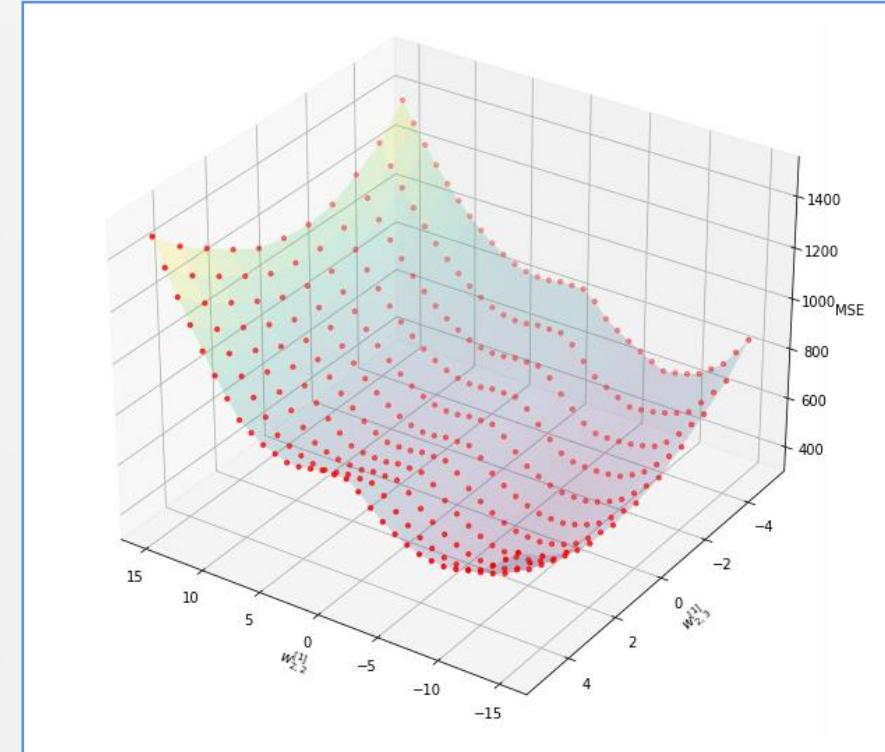
ปรับเปลี่ยน  $\hat{y}$  กับ  $y$  เพื่อคำนวณ cost function ( $Cost = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$ )

$w_{2,2}^{[1]} = -15$	$w_{2,2}^{[1]} = -14$	$w_{2,2}^{[1]} = -13$	$w_{2,2}^{[1]} = -12$	$\dots$	$w_{2,2}^{[1]} = 12$	$w_{2,2}^{[1]} = 13$	$w_{2,2}^{[1]} = 14$	$w_{2,2}^{[1]} = 15$
$w_{2,3}^{[1]} = -5$	$w_{2,3}^{[1]} = -5$	$w_{2,3}^{[1]} = -5$	$w_{2,3}^{[1]} = -5$		$w_{2,3}^{[1]} = 5$	$w_{2,3}^{[1]} = 5$	$w_{2,3}^{[1]} = 5$	$w_{2,3}^{[1]} = 5$
$\hat{y}_6$	-4	-4	-4	-4	-4	-4	-4	-4
$\hat{y}_7$	-5	-5	-5	-5	-10	-11	-12	-13
$\hat{y}_8$	-2	-2	-2	-2	-10	-11	-12	-13
$\hat{y}_9$	-2	-2	-2	-2	-10	-11	-12	-13

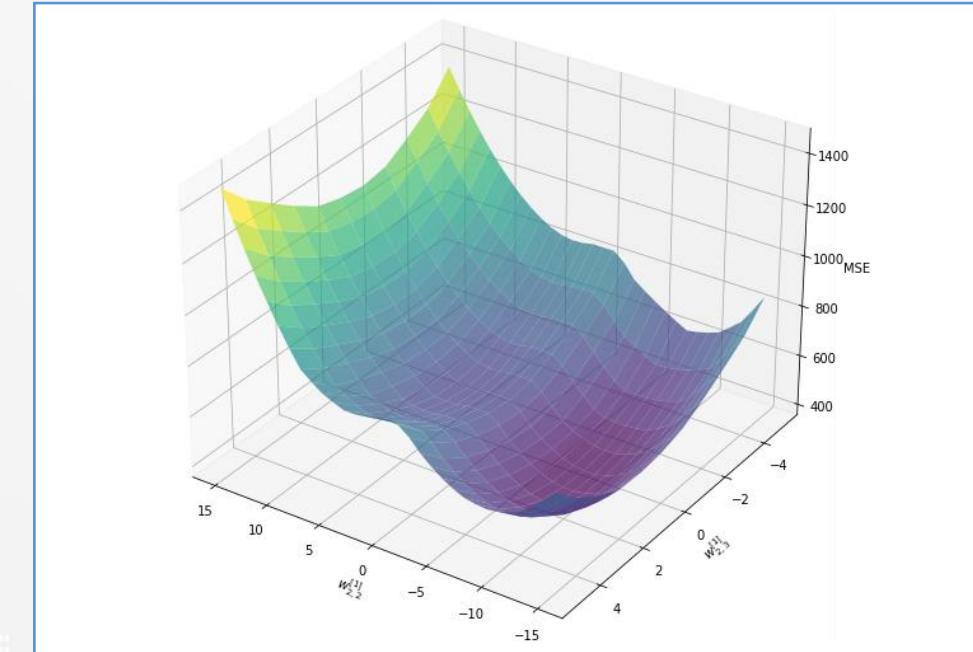
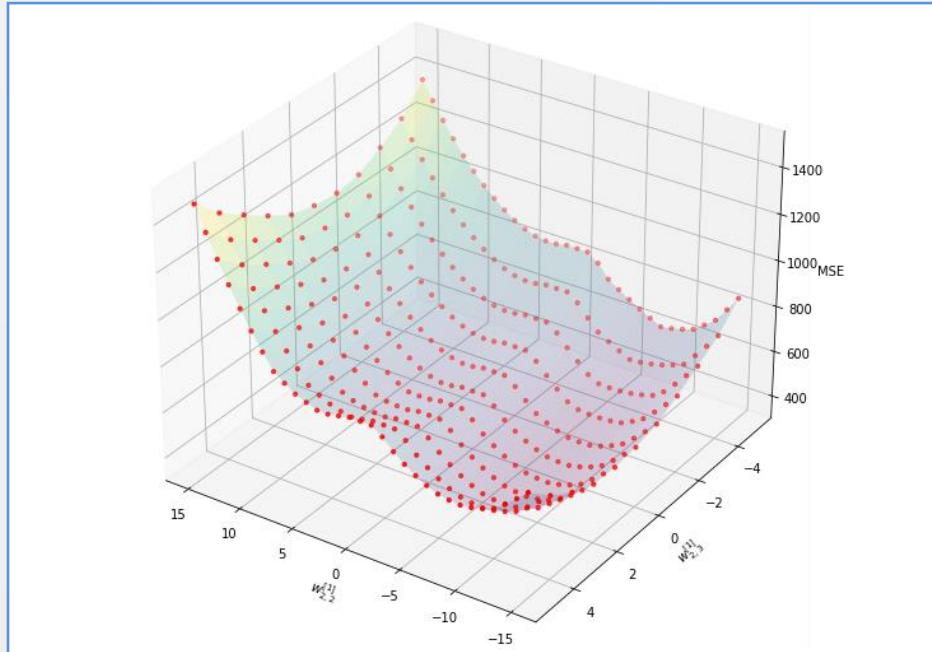
# Cost Function & Cost Landscape

$w_{2,2}^{[1]} = -15$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -14$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -13$ $w_{2,3}^{[1]} = -5$	$w_{2,2}^{[1]} = -12$ $w_{2,3}^{[1]} = -5$	...	$w_{2,2}^{[1]} = 12$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 13$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 14$ $w_{2,3}^{[1]} = 5$	$w_{2,2}^{[1]} = 15$ $w_{2,3}^{[1]} = 5$	
<i>Cost</i>	19170.0	1768.5	1629.0	1545.75	...	2659.5	2864.25	3082.5	3314.25

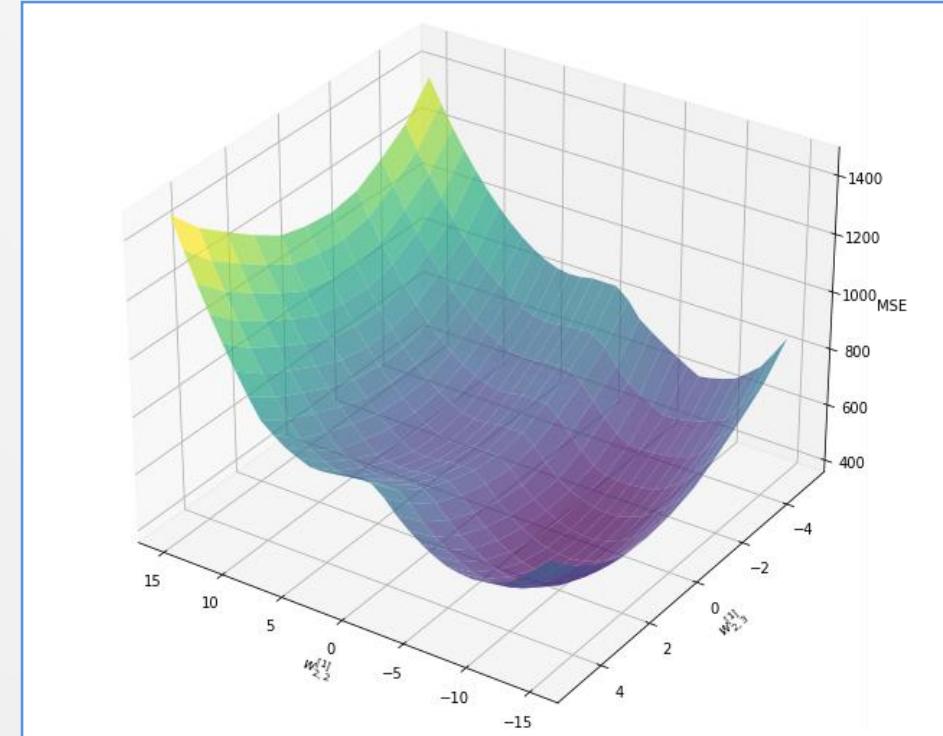
# Cost Function & Cost Landscape



# Cost Function & Cost Landscape

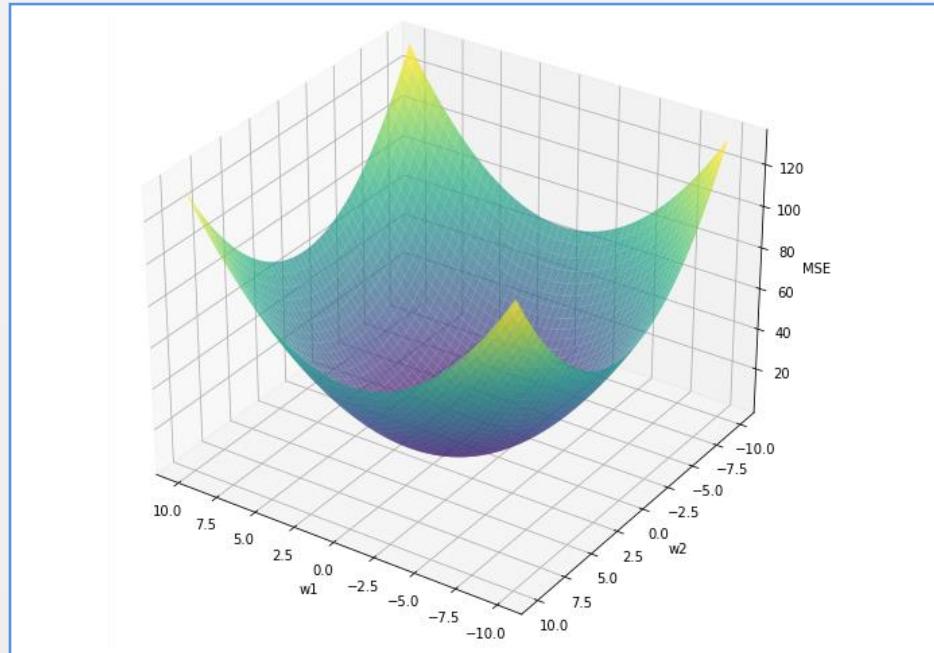


# Cost Function & Cost Landscape

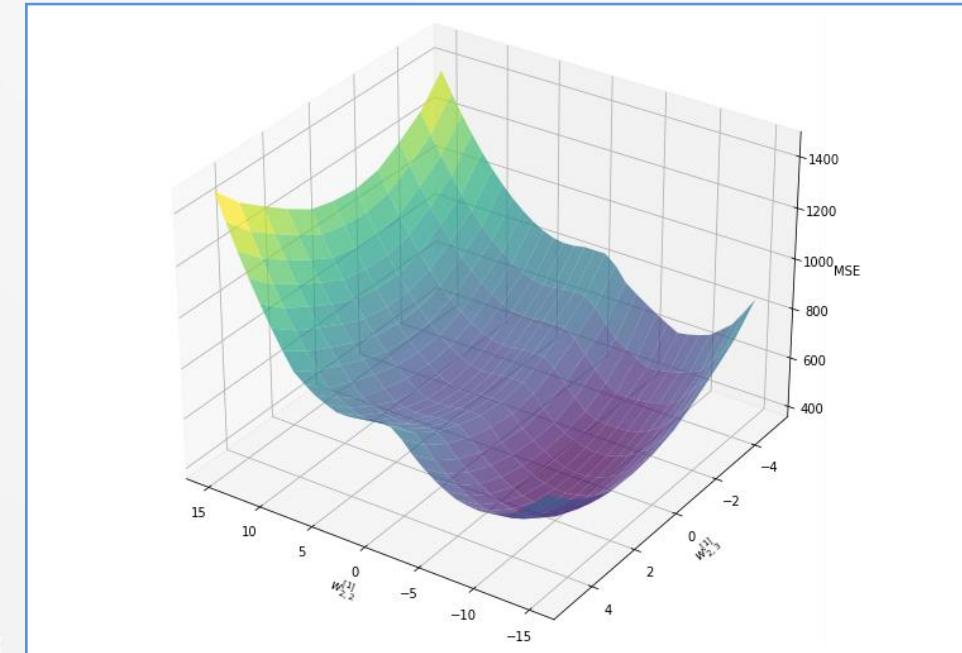


กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น MSE

# Cost Function & Cost Landscape



VS

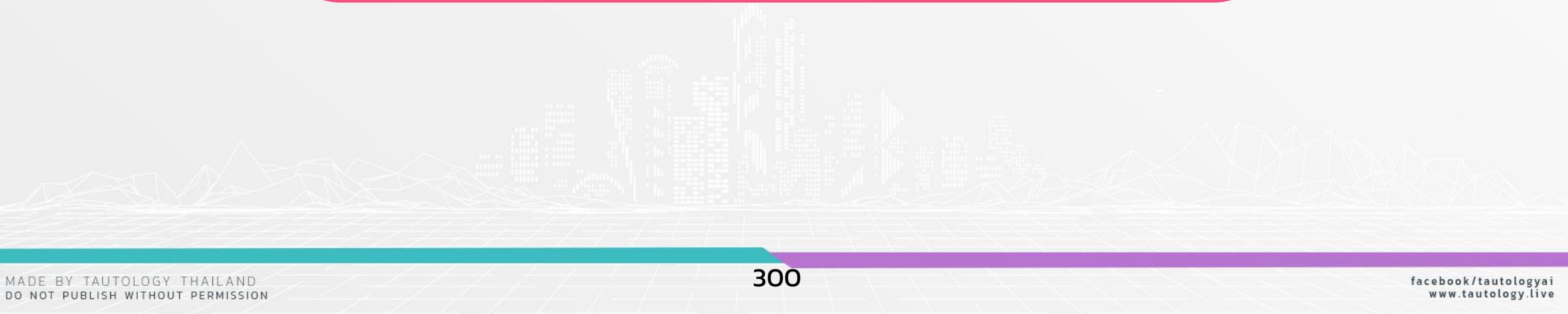


กราฟแสดง cost landscape ของ linear regression โดยที่ cost function เป็น MSE

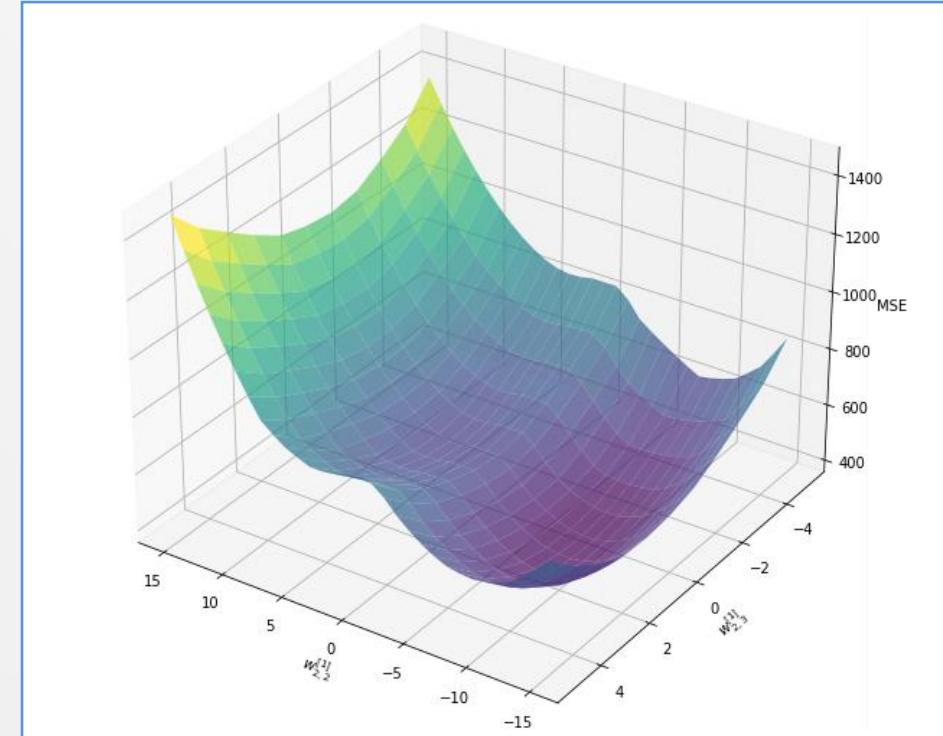
กราฟแสดง cost landscape ของ deep learning โดยที่ cost function เป็น MSE

# Cost Function & Cost Landscape

Cost function ของ deep learning เป็น non-convex  
ซึ่งมีจุดต่ำสุดหลายตำแหน่ง



# Cost Function & Cost Landscape



กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น MSE

# Cost Function & Cost Landscape



For more information



Improvement of  
Deep Learning

# Model

**Assumption**



**Real Face of the Model**



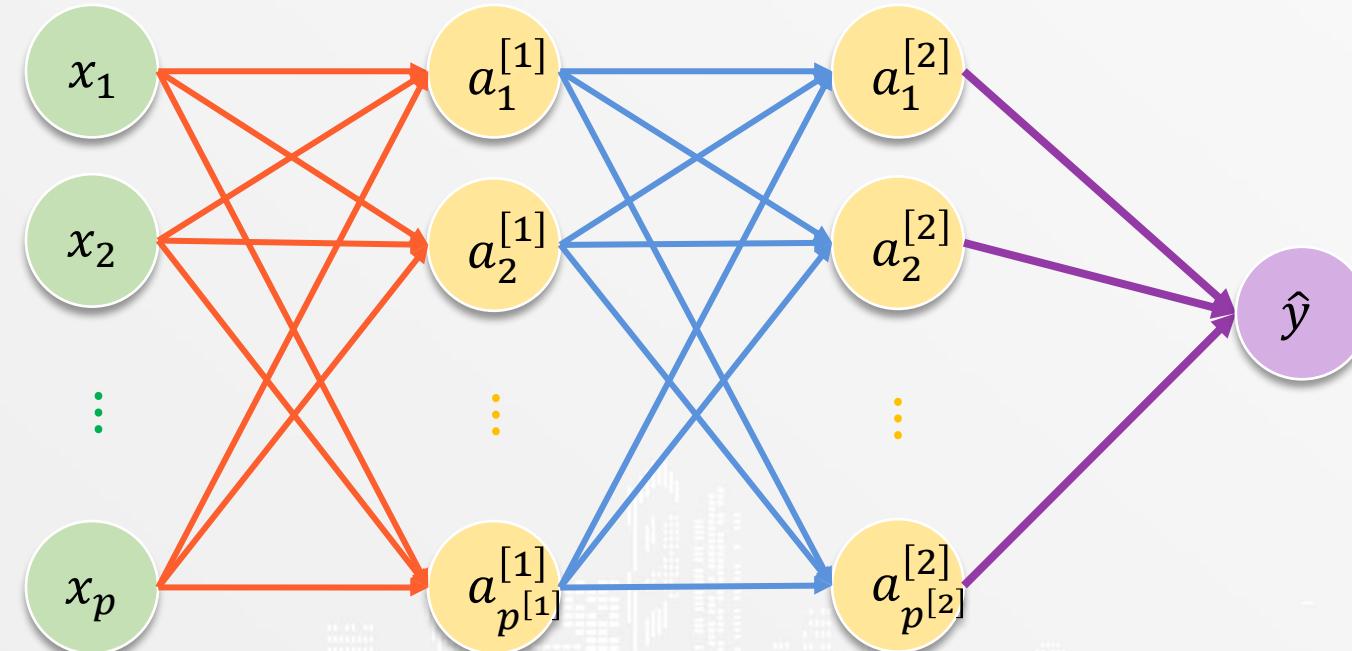
**Cost Function and Cost Landscape**



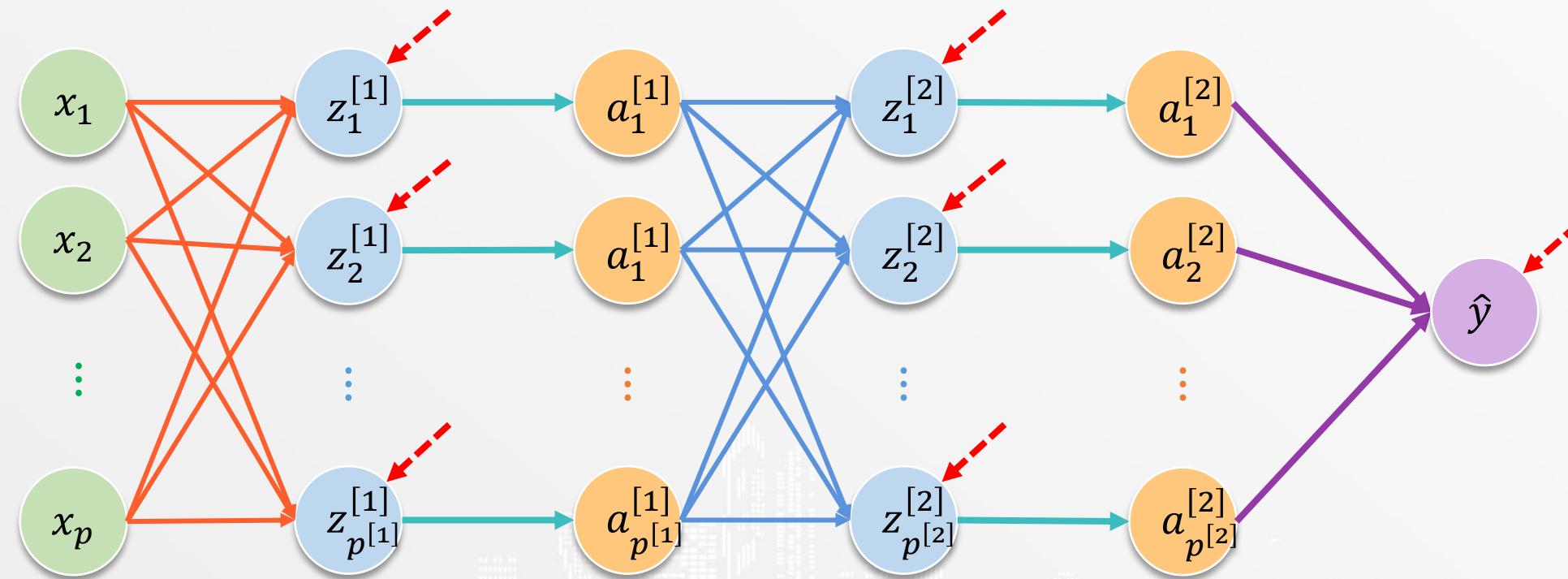
**How to Create Model**



# How to Create Model



# How to Create Model



# How to Create Model

เราต้องการหา  $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ  $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$  ที่ทำให้ cost function ต่ำที่สุด

# How to Create Model

**Cost function** กี่เราจะใช้ในการสร้าง model คือ

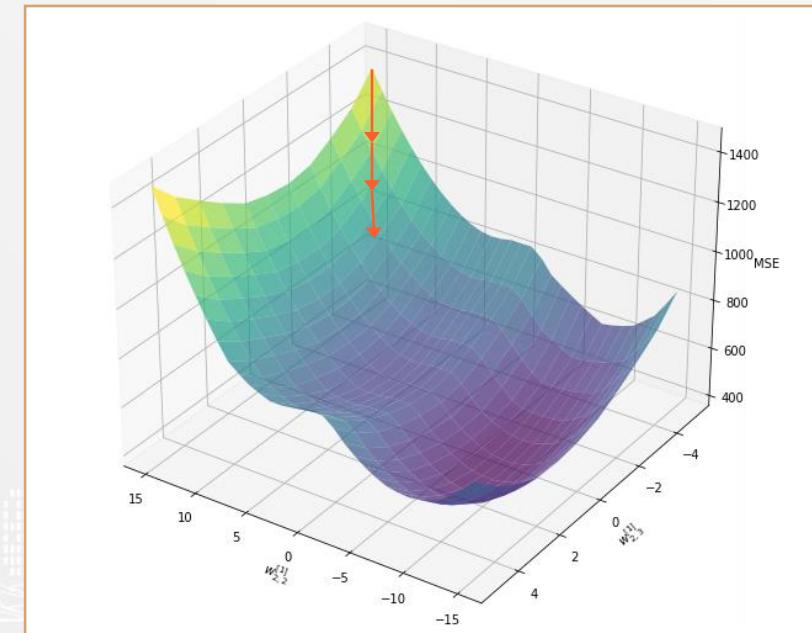
$$-\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

โดยสูตรข้างต้นมีชื่อว่า mean square error หรือ MSE

# How to Create Model

เครื่องมือที่เราจะใช้ในการหาค่าตอบ คือ

“gradient descent”



# How to Create Model

## Equation of Gradient Descent

$$W = W - \alpha \nabla Cost$$

โดย ◆  $\alpha$  คือ ค่าที่ใช้ควบคุม step size ของ  $W$

# How to Create Model

$$b^{[1]} = b^{[1]} - \alpha \nabla Cost$$

$$b^{[2]} = b^{[2]} - \alpha \nabla Cost$$

⋮

$$b^{[L]} = b^{[L]} - \alpha \nabla Cost$$

$$b^{[out]} = b^{[out]} - \alpha \nabla Cost$$

# How to Create Model

$$W^{[1]} = W^{[1]} - \alpha \nabla Cost$$

$$W^{[2]} = W^{[2]} - \alpha \nabla Cost$$

⋮

$$W^{[L]} = W^{[L]} - \alpha \nabla Cost$$

$$W^{[out]} = W^{[out]} - \alpha \nabla Cost$$

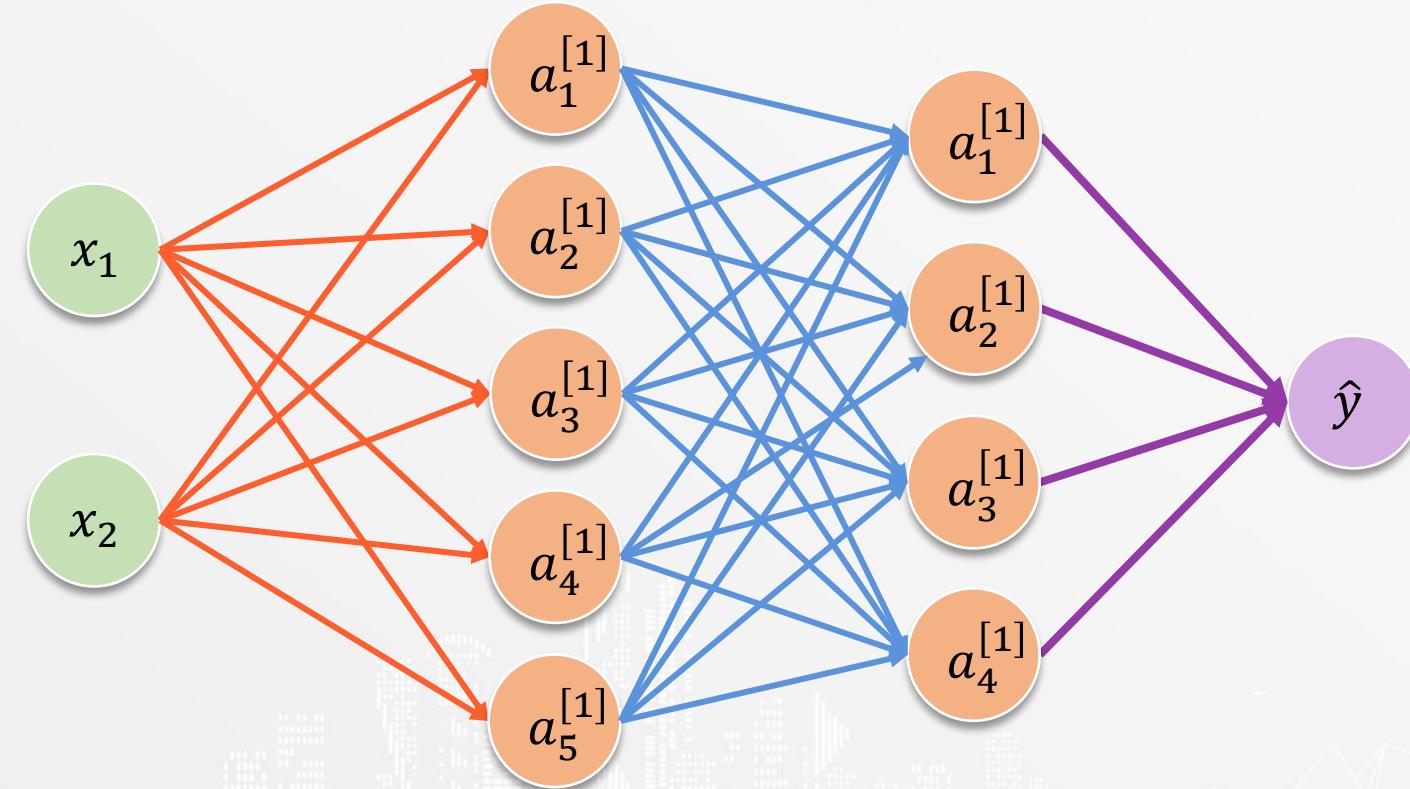
# How to Create Model

**ตัวอย่างการหา weight สำหรับ deep learning**

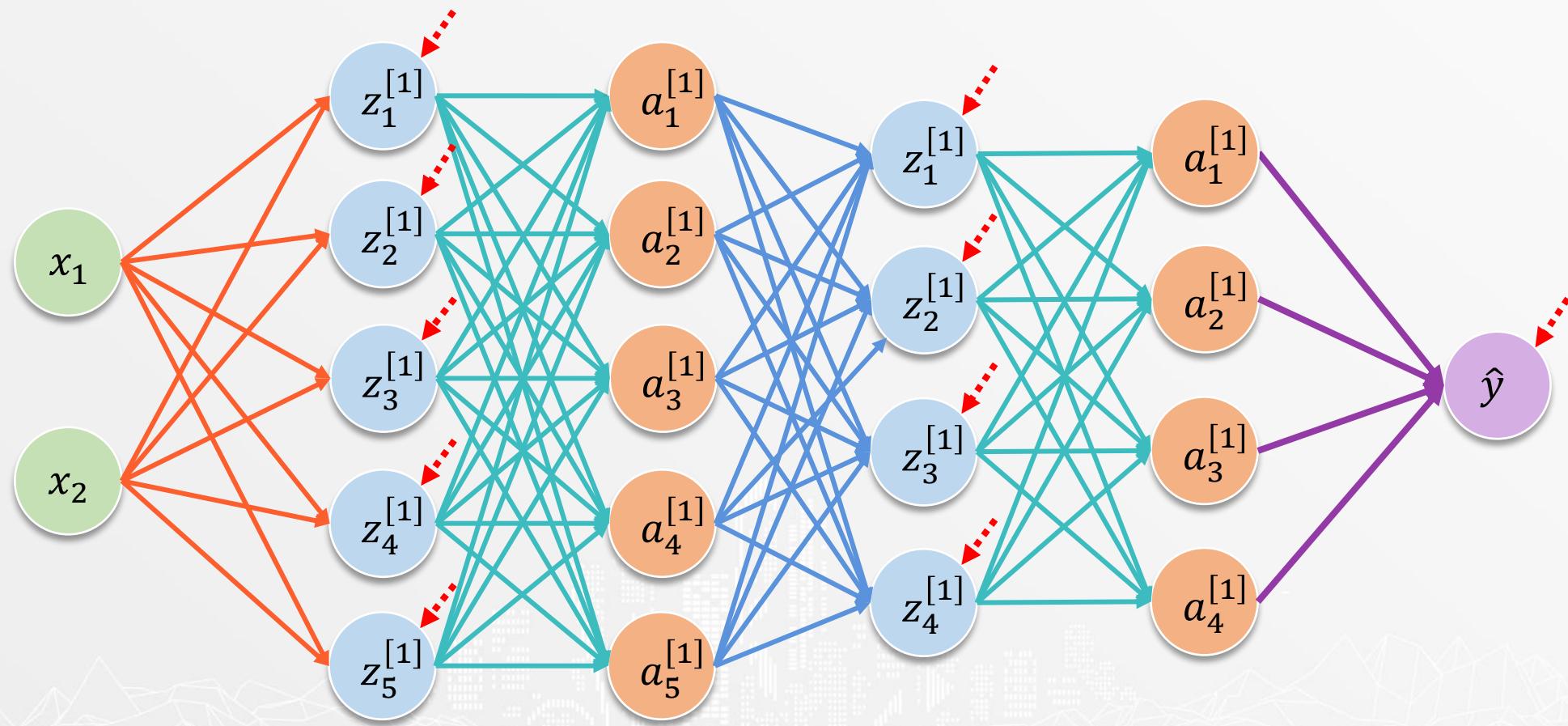
$x_1$	$x_2$	y
2	1	9
2	-1	1
-2	2	6
1	2	11
-2	3	9
2	0	5
-1	-1	-2
-2	1	3
0	0	2

$x_1$	$x_2$	y
1	-1	0
-1	0	1
-1	1	4
1	3	15
2	2	13
2	3	17
1	1	7
0	2	9
-1	3	11

# How to Create Model



# How to Create Model



# How to Create Model

$$X = \begin{bmatrix} 2 & 1 \\ 2 & -1 \\ -2 & 2 \\ 1 & 2 \\ \vdots & \vdots \\ 0 & 2 \\ -1 & 3 \end{bmatrix}, \quad y = \begin{bmatrix} 9 \\ 1 \\ 6 \\ 11 \\ \vdots \\ 9 \\ 11 \end{bmatrix}$$

# How to Create Model

```
1 reg = MLPRegressor(  
2     hidden_layer_sizes=(5, 4),  
3     activation='relu',  
4     solver='sgd',  
5     alpha=0,  
6     learning_rate_init=0.01,  
7     max_iter=10000,  
8     momentum=0  
9 )  
10  
11 reg.fit(X, y)
```

# How to Create Model



Code for this section



Open File  
**Regression/Model Creation (sklearn).ipynb**

# How to Create Model

$$b^{[1]} = [0.461 \quad 1.079 \quad -0.924 \quad -0.729 \quad -0.393]$$

$$W^{[1]} = \begin{bmatrix} 0.724 & -0.03 & -0.565 & -0.549 & 0.458 \\ 0.831 & 1.11 & 0.207 & 0.48 & 0.436 \end{bmatrix}$$

# How to Create Model

$$b^{[2]} = [0 \quad 0 \quad -0.73 \quad -0.42]$$

$$W^{[2]} = \begin{bmatrix} 0.369 & 0.663 & 0.624 & 0.912 \\ 0.695 & 0.672 & 0.363 & 0.331 \\ -0.323 & -0.146 & 0.364 & 0.767 \\ 0.224 & 0.395 & -0.557 & -0.836 \\ 0.592 & 0.794 & -0.765 & 0.106 \end{bmatrix}$$

# How to Create Model

$$b^{[out]} = [-1.128]$$

$$W^{[out]} = \begin{bmatrix} 0.877 \\ 1.084 \\ 0.142 \\ 0.789 \end{bmatrix}$$

# How to Create Model

$$a_1^{[1]} = \text{ReLU}(0.461 + 0.724x_1 + 0.663x_2)$$

$$a_2^{[1]} = \text{ReLU}(1.079 - 0.03x_1 + 1.11x_2)$$

$$a_3^{[1]} = \text{ReLU}(-0.924 - 0.565x_1 + 0.207x_2)$$

$$a_4^{[1]} = \text{ReLU}(-0.729 - 0.549x_1 + 0.48x_2)$$

$$a_5^{[1]} = \text{ReLU}(-0.393 + 0.458x_1 + 0.436x_2)$$

# How to Create Model

$$a_1^{[2]} = \text{ReLU}(0 + 0.369a_1^{[1]} + 0.695a_2^{[1]} - 0.323a_3^{[1]} + 0.224a_4^{[1]} + 0.592a_5^{[1]})$$

$$a_2^{[2]} = \text{ReLU}(0 + 0.663a_1^{[1]} + 0.672a_2^{[1]} - 0.146a_3^{[1]} + 0.395a_4^{[1]} + 0.794a_5^{[1]})$$

$$a_3^{[2]} = \text{ReLU}(-0.73 + 0.624a_1^{[1]} + 0.323a_2^{[1]} + 0.364a_3^{[1]} - 0.554a_4^{[1]} - 0.765a_5^{[1]})$$

$$a_4^{[2]} = \text{ReLU}(-0.42 + 0.912a_1^{[1]} + 0.331a_2^{[1]} + 0.767a_3^{[1]} - 0.836a_4^{[1]} + 0.106a_5^{[1]})$$

# How to Create Model

$$\hat{y} = -1.128 + 0.877a_1^{[3]} + 1.084a_2^{[3]} + 0.142a_3^{[3]} + 0.789a_4^{[3]}$$

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model**

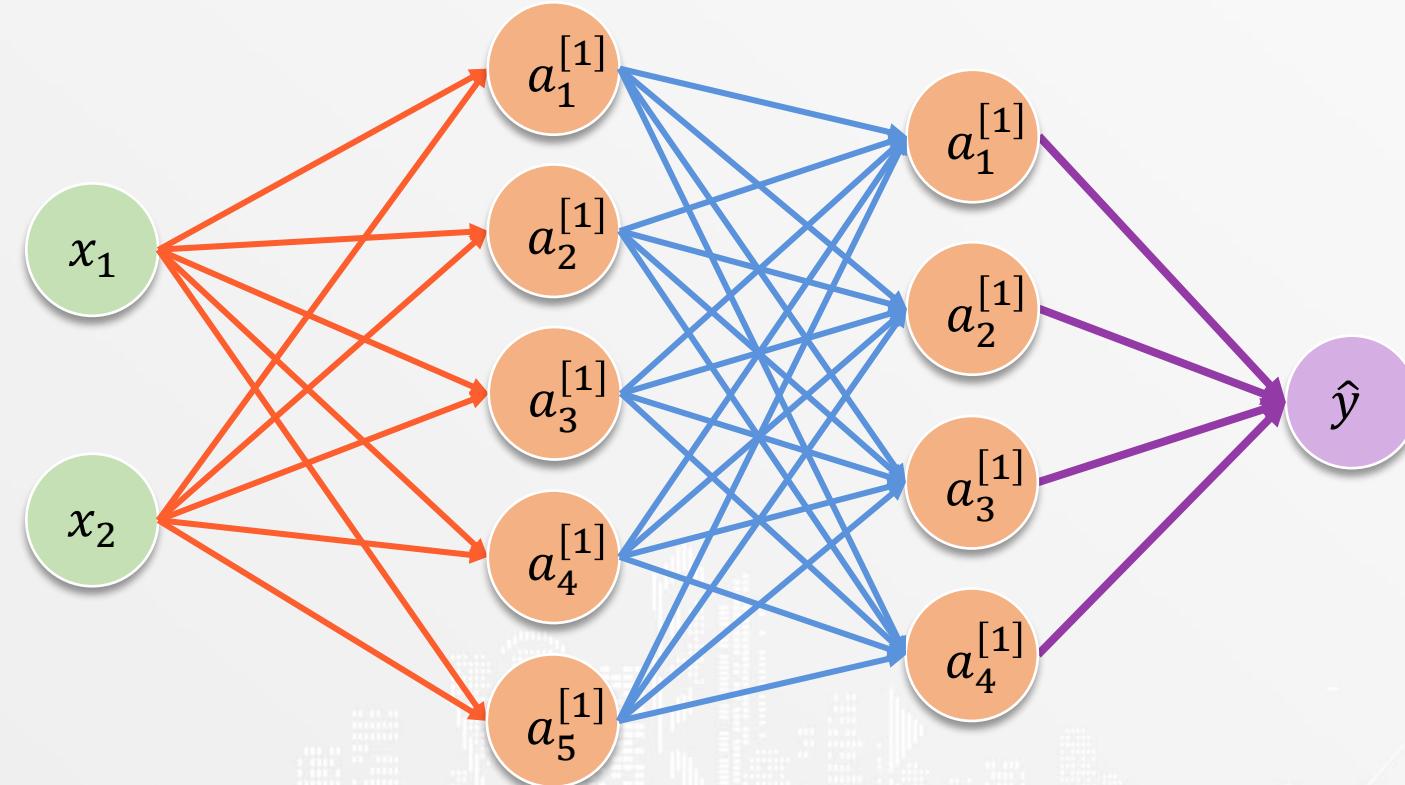


# Deep Learning for Regression

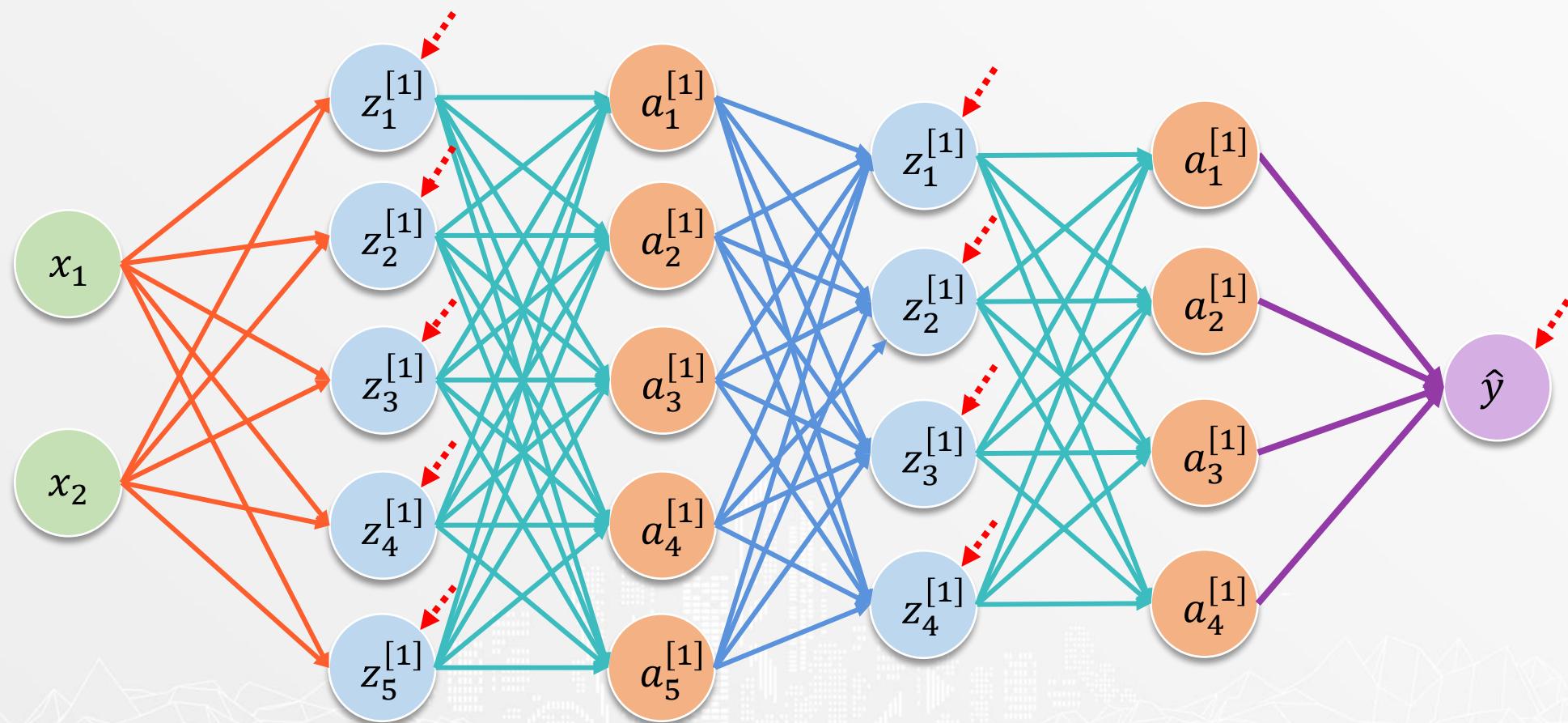


# Prediction

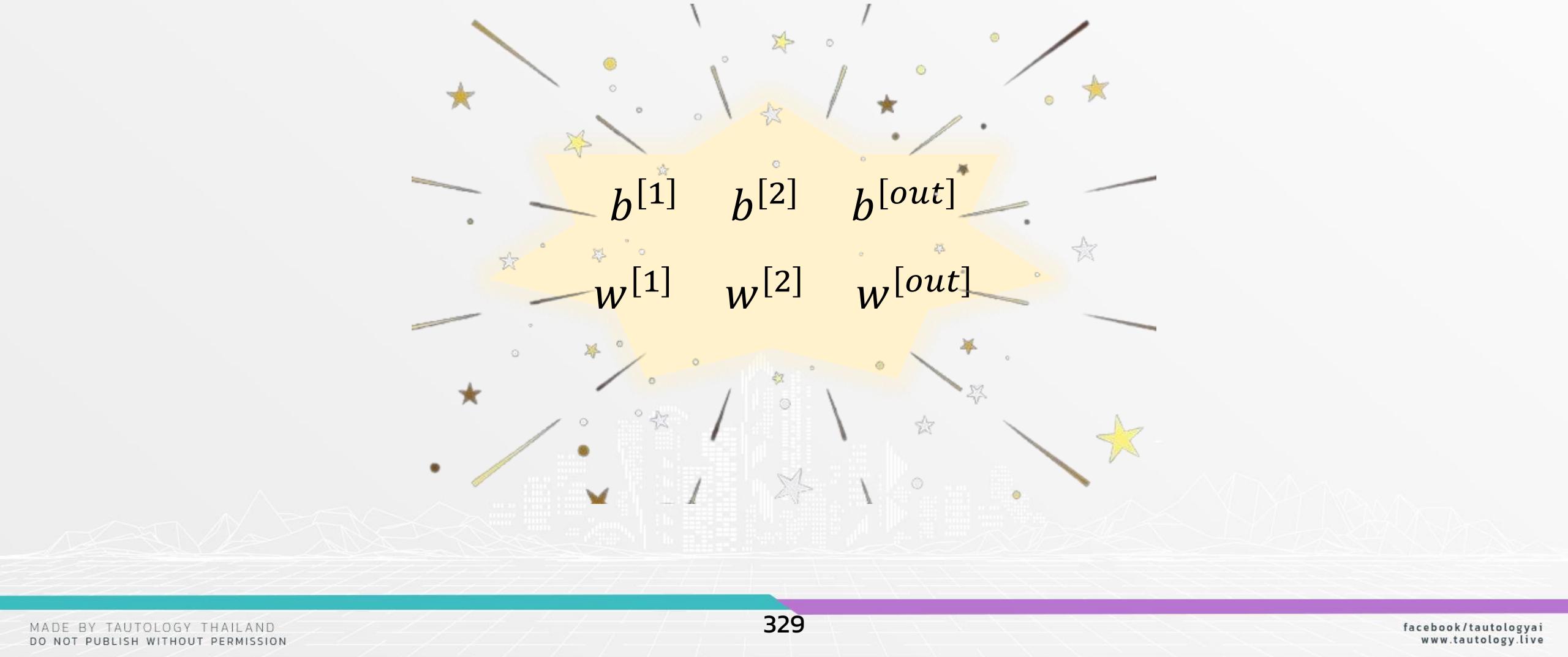
# Prediction



# Prediction

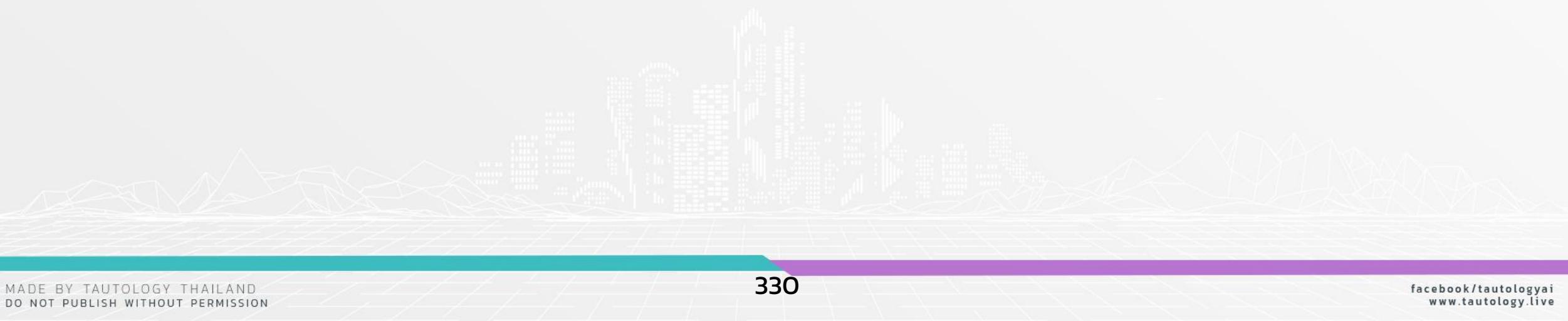


# Prediction



# Prediction

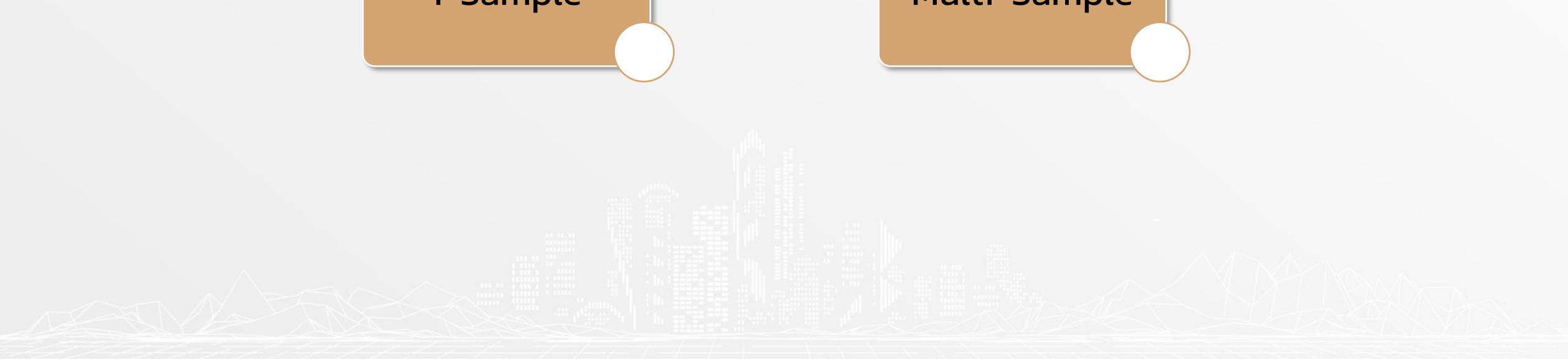
- 1-Sample
- Multi-Sample



# Prediction

1-Sample

Multi-Sample



# 1-Sample

ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
0	3



$\hat{y}$
?

# 1-Sample

- สมมติว่า weight ของปัจจัยหนึ่งที่เรามาได้คือ

$$b^{[1]} = [0.461 \quad 1.079 \quad -0.924 \quad -0.729 \quad -0.393]$$

$$W^{[1]} = \begin{bmatrix} 0.724 & -0.03 & -0.565 & -0.549 & 0.458 \\ 0.831 & 1.11 & 0.207 & 0.48 & 0.436 \end{bmatrix}$$

# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$b^{[2]} = [0 \quad 0 \quad -0.73 \quad -0.42]$$

$$W^{[2]} = \begin{bmatrix} 0.369 & 0.663 & 0.624 & 0.912 \\ 0.695 & 0.672 & 0.363 & 0.331 \\ -0.323 & -0.146 & 0.364 & 0.767 \\ 0.224 & 0.395 & -0.557 & -0.836 \\ 0.592 & 0.794 & -0.765 & 0.106 \end{bmatrix}$$

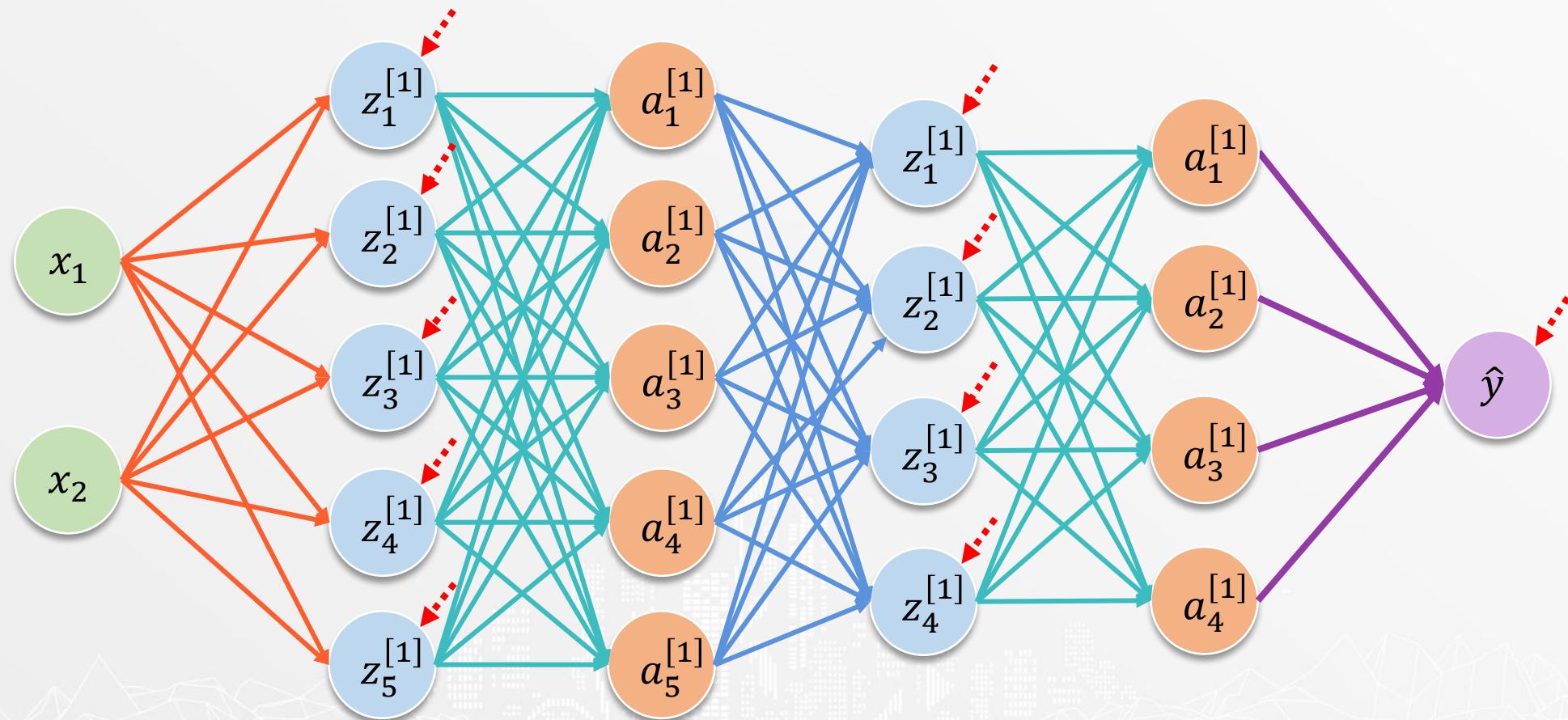
# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$b^{[out]} = [-1.128]$$

$$W^{[out]} = \begin{bmatrix} 0.877 \\ 1.084 \\ 0.142 \\ 0.789 \end{bmatrix}$$

# 1-Sample



# 1-Sample

$$a_1^{[1]} = \text{ReLU}(0.461 + 0.724x_1 + 0.831x_2) = 2.955$$

$$a_2^{[1]} = \text{ReLU}(1.079 - 0.03x_1 + 1.11x_2) = 4.409$$

$$a_3^{[1]} = \text{ReLU}(-0.924 - 0.565x_1 + 0.207x_2) = 0$$

$$a_4^{[1]} = \text{ReLU}(-0.729 - 0.549x_1 + 0.48x_2) = 0.711$$

$$a_5^{[1]} = \text{ReLU}(-0.393 + 0.458x_1 + 0.436x_2) = 0.915$$

# 1-Sample

$$a_1^{[2]} = \text{ReLU} \left( 0 + 0.369a_1^{[1]} + 0.695a_2^{[1]} - 0.323a_3^{[1]} + 0.224a_4^{[1]} + 0.592a_5^{[1]} \right) = 4.856$$

$$a_2^{[2]} = \text{ReLU} \left( 0 + 0.663a_1^{[1]} + 0.672a_2^{[1]} - 0.146a_3^{[1]} + 0.395a_4^{[1]} + 0.794a_5^{[1]} \right) = 5.929$$

$$a_3^{[2]} = \text{ReLU} \left( -0.73 + 0.624a_1^{[1]} + 0.323a_2^{[1]} + 0.364a_3^{[1]} - 0.554a_4^{[1]} - 0.765a_5^{[1]} \right) = 1.618$$

$$a_4^{[2]} = \text{ReLU} \left( -0.42 + 0.912a_1^{[1]} + 0.331a_2^{[1]} + 0.767a_3^{[1]} - 0.836a_4^{[1]} + 0.106a_5^{[1]} \right) = 4.077$$

# 1-Sample

$$\begin{aligned}\hat{y} &= -1.128 + 0.877a_1^{[3]} + 1.084a_2^{[3]} + 0.142a_3^{[3]} + 0.789a_4^{[3]} \\ &= 13.004\end{aligned}$$

# 1-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
0	3



$\hat{y}$
13.004

# Prediction

**1-Sample**



**Multi-Sample**



# Multi-Sample

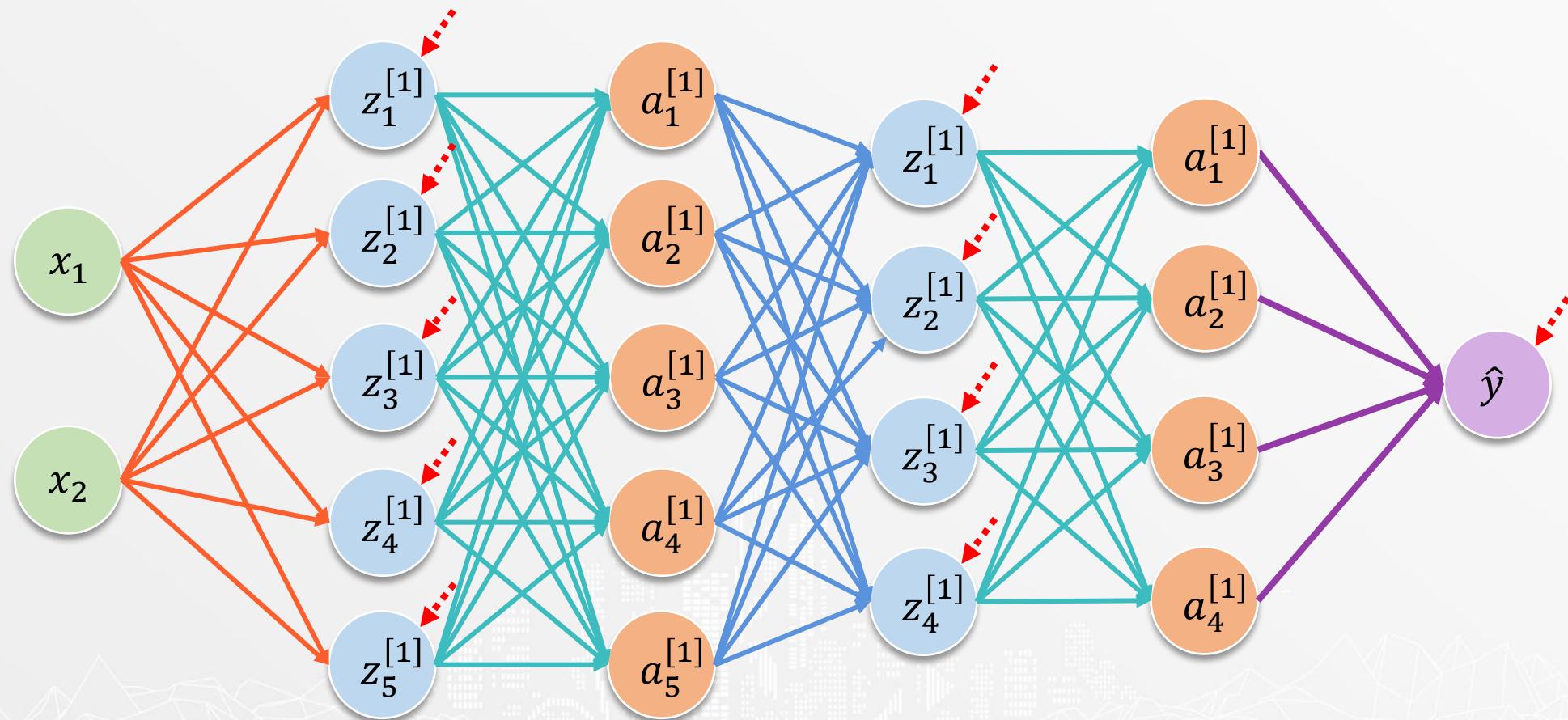
ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
0	3
-2	-1
-2	1
0	-1



$\hat{y}$
?
?
?
?

# Multi-Sample



# Multi-Sample

$$X \rightarrow Z^{[1]} \rightarrow A^{[1]} \rightarrow Z^{[2]} \rightarrow A^{[2]} \rightarrow \hat{y}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + b^{[1]}$$

$$\begin{aligned} Z^{[1]} &= \begin{bmatrix} 0 & 3 \\ -2 & -1 \\ -2 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0.724 & -0.03 & -0.565 & -0.549 & 0.458 \\ 0.831 & 1.11 & 0.207 & 0.48 & 0.436 \end{bmatrix} + [0.461 \quad 1.079 \quad -0.924 \quad -0.729 \quad -0.393] \\ &= \begin{bmatrix} 2.494 & 3.329 & 0.62 & 1.439 & 1.307 \\ -2.28 & -1.05 & 0.924 & 0.619 & -1.351 \\ -0.813 & 1.11 & 0.207 & 0.48 & 0.436 \\ -0.831 & -1.11 & -0.207 & -0.48 & -0.436 \end{bmatrix} + \begin{bmatrix} 0.462 & 1.079 & -0.924 & -0.729 & -0.393 \\ 0.462 & 1.079 & -0.924 & -0.729 & -0.393 \\ 0.462 & 1.079 & -0.924 & -0.729 & -0.393 \\ 0.462 & 1.079 & -0.924 & -0.729 & -0.393 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + b^{[1]}$$

$$Z^{[1]} = \begin{bmatrix} 2.956 & 4.408 & -0.304 & 0.71 & 0.914 \\ -1.818 & 0.029 & 0 & -0.11 & -1.744 \\ 1.293 & 2.189 & -0.717 & -0.249 & -0.043 \\ -0.369 & -0.031 & -1.131 & -1.209 & -0.829 \end{bmatrix}$$

# Multi-Sample

$$A^{[1]} = \text{ReLU}(Z^{[1]})$$

$$A^{[1]} = \text{ReLU} \begin{pmatrix} 2.956 & 4.408 & -0.304 & 0.71 & 0.914 \\ -1.818 & 0.029 & 0 & -0.11 & -1.744 \\ 1.293 & 2.189 & -0.717 & -0.249 & -0.043 \\ -0.369 & -0.031 & -1.131 & -1.209 & -0.829 \end{pmatrix}$$

$$= \begin{bmatrix} 2.956 & 4.408 & 0 & 0.71 & 0.914 \\ 0 & 0.029 & 0 & 0 & 0 \\ 1.293 & 2.189 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

# Multi-Sample

$$Z^{[2]} = A^{[1]}W^{[2]} + b^{[2]}$$

$$\begin{aligned} Z^{[2]} &= \begin{bmatrix} 2.956 & 4.408 & 0 & 0.71 & 0.914 \\ 0 & 0.029 & 0 & 0 & 0 \\ 1.293 & 2.189 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.369 & 0.663 & 0.624 & 0.912 \\ 0.695 & 0.672 & 0.363 & 0.331 \\ -0.323 & -0.146 & 0.364 & 0.767 \\ 0.224 & 0.395 & -0.557 & -0.836 \\ 0.592 & 0.794 & -0.765 & 0.106 \end{bmatrix} + [0 \ 0 \ -0.73 \ -0.42] \\ &= \begin{bmatrix} 4.853 & 5.93 & 2.347 & 3.656 \\ 0.02 & 0.019 & 0.011 & 0.01 \\ 2.023 & 2.363 & 1.567 & 1.908 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & -0.73 & -0.42 \\ 0 & 0 & -0.73 & -0.42 \\ 0 & 0 & -0.73 & -0.42 \\ 0 & 0 & -0.73 & -0.42 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[2]} = A^{[1]}W^{[2]} + b^{[2]}$$

$$Z^{[2]} = \begin{bmatrix} 4.853 & 5.93 & 1.617 & 4.076 \\ 0.02 & 0.019 & -0.719 & 0.43 \\ 2.023 & 2.363 & 0.837 & 2.328 \\ 0 & 0 & -0.73 & 0.42 \end{bmatrix}$$

# Multi-Sample

$$A^{[2]} = \text{ReLU}(Z^{[2]})$$

$$A^{[2]} = \text{ReLU} \left( \begin{bmatrix} 4.853 & 5.93 & 1.617 & 4.076 \\ 0.02 & 0.019 & -0.719 & 0.43 \\ 2.023 & 2.363 & 0.837 & 2.328 \\ 0 & 0 & -0.73 & 0.42 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 4.853 & 5.93 & 1.617 & 4.076 \\ 0.02 & 0.019 & 0 & 0.43 \\ 2.023 & 2.363 & 0.837 & 2.328 \\ 0 & 0 & 0 & 0.42 \end{bmatrix}$$

# Multi-Sample

$$\hat{\mathbf{y}} = A^{[2]}W^{[out]} + b^{[out]}$$

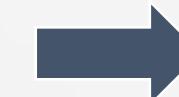
$$\hat{\mathbf{y}} = \begin{bmatrix} 4.853 & 5.93 & 1.617 & 4.076 \\ 0.02 & 0.019 & 0 & 0.43 \\ 2.023 & 2.363 & 0.837 & 2.328 \\ 0 & 0 & 0 & 0.42 \end{bmatrix} \begin{bmatrix} 0.877 \\ 1.084 \\ 0.142 \\ 0.789 \end{bmatrix} + [-1.128]$$

$$= \begin{bmatrix} 14.131 \\ 0.377 \\ 6.292 \\ 0.331 \end{bmatrix} + \begin{bmatrix} -1.128 \\ -1.128 \\ -1.128 \\ -1.128 \end{bmatrix} = \begin{bmatrix} 13.003 \\ -0.751 \\ 5.164 \\ -0.797 \end{bmatrix}$$

# Multi-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
0	3
-2	-1
-2	1
0	-1



$\hat{y}$
13.003
-0.751
5.164
-0.797

# Multi-Sample

```
1 reg.predict(X)
```

```
array([13.00237175, -0.75090167,  5.1624261 , -0.79658003])
```

# Multi-Sample



Code for this section



Open File  
**Regression/Model Creation (sklearn).ipynb**

# Prediction

**1-Sample**



**Multi-Sample**



# Deep Learning for Regression



# Neural Network & Deep Learning

**Neural Network**



**Deep Learning**



**Deep Learning for  
Regression**



**Deep Learning for  
Classification**



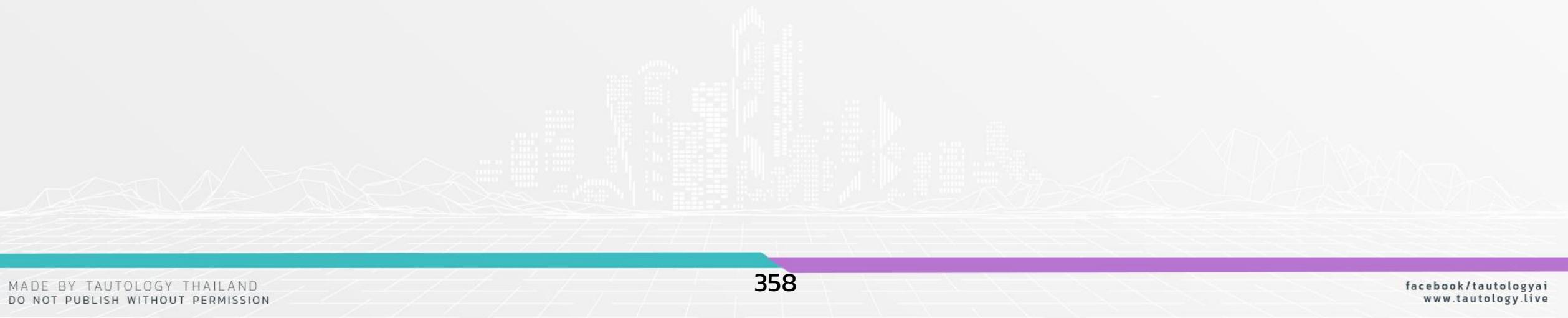
**Workshop**



# Deep Learning for Classification

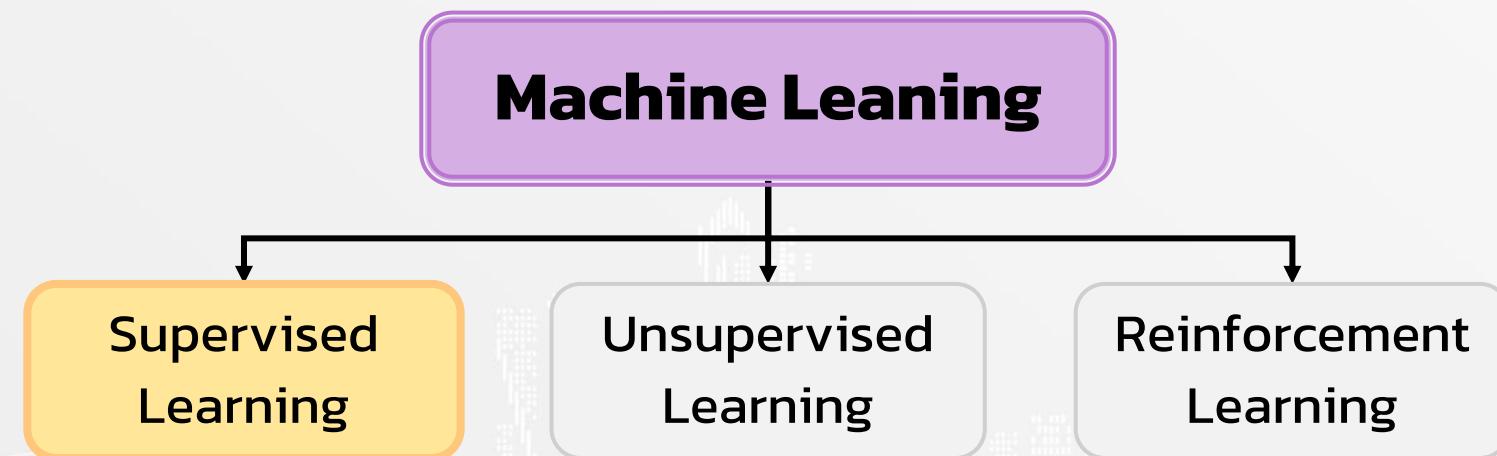
Deep Learning for  
Binary Classification

Deep Learning for  
Multi-Class Classification



# Deep Learning for Binary Classification

**Deep Learning** เป็นหนึ่งใน algorithm ประเภท  
supervised learning



# Concept of Supervised Learning

Data → Model → Prediction

# Deep Learning for Binary Classification



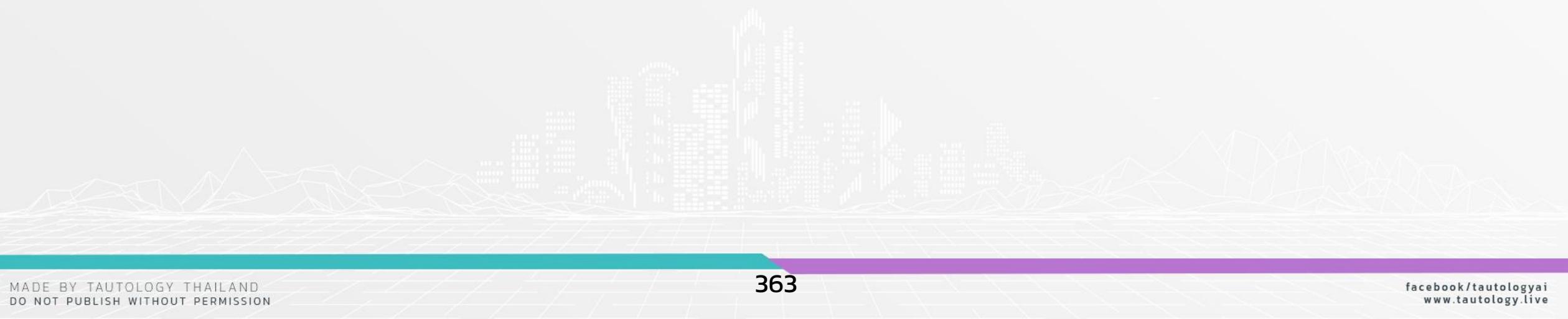
# Data

# Data

Data Stating

Data  
Requirement

More about  
Target



# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

# Data Stating

The diagram illustrates a data matrix represented as a table. The columns are labeled  $x_1, x_2, x_3, \dots, x_p$  and the rows are labeled  $y_1, y_2, y_3, \dots, y_n$ . Orange arrows point downwards from the column labels to the first four columns of the matrix. A green arrow points downwards from the row label  $y$  to the last column of the matrix. The matrix itself contains entries  $x_{i,j}$  for each row  $i$  and column  $j$ .

$x_1$	$x_2$	$x_3$	$\cdots$	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$\cdots$	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$\cdots$	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$\cdots$	$x_{3,p}$	$y_3$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$\cdots$	$x_{n,p}$	$y_n$

- ♦  $n$  คือ จำนวน sample
- ♦  $p$  คือ จำนวน feature

# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	⋮	⋮	⋮
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

- $x_{2,3}$  คือ sample ที่ 2 feature ที่ 3
- $x_{3,p}$  คือ sample ที่ 3 feature ที่  $p$
- $x_{n,p}$  คือ sample ที่  $n$  feature ที่  $p$
- $y_2$  คือ target ของ sample ที่ 2
- $y_3$  คือ target ของ sample ที่ 3
- $y_n$  คือ target ของ sample ที่  $n$

# Data Stating

## Example

- เราต้องการจำแนกผู้ป่วยโรคหัวใจ โดยพิจารณาจาก อายุ เพศ ความดันโลหิต คอลเลสเตอรอล

## Data

อายุ	เพศ	ความดันโลหิต (mmHg)	คอลเลสเตอรอล (mg/dL)	เป็นโรคหัวใจ
42	0	120	209	1
57	1	150	168	1
58	1	128	259	0
59	0	174	249	0

# Data Stating

- ข้อมูลตามแนวแอกว คือ Sample



อายุ	เพศ	ความดันโลหิต (mmHg)	คอเลสเตอรอล (mg/dL)	เป็นโรคหัวใจ
42	0	120	209	1
57	1	150	168	1
58	1	128	259	0
59	0	174	249	0

# Data Stating

- ข้อมูลตามแนวหลัก คือ Feature and Target
  - Feature (ตัวแปรต้น) คือ ข้อมูลที่ส่งผลให้เกิด target
  - Target (ตัวแปรตาม) คือ ข้อมูลที่เราสนใจจะพยากรณ์

Feature				Target
อายุ	เพศ	ความดันโลหิต (mmHg)	คอเลสเตอรอล (mg/dL)	เป็นโรคหัวใจ
42	0	120	209	1
57	1	150	168	1
58	1	128	259	0
59	0	174	249	0

# Data Stating

- Feature and Target
  - เราสามารถแยก และปรับให้เป็น matrix ได้ดังนี้

$$X = \begin{bmatrix} 42 & 0 & 120 & 209 \\ 57 & 1 & 150 & 168 \\ 58 & 1 & 128 & 259 \\ 59 & 0 & 174 & 249 \end{bmatrix}$$

$$y = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

# Data

**Data Stating**



**Data  
Requirement**

**More about  
Target**

# Data Requirement

- ข้อมูลต้องอยู่ในรูปแบบของตาราง**
- ข้อมูลต้องเป็น numerical**

อายุ	เพศ	ความดันโลหิต (mmHg)	คอลเลสเตอรอล (mg/dL)	เป็นโรคหัวใจ
42	0	120	209	1
57	1	150	168	1
58	1	128	259	0
59	0	174	249	0

# Data Requirement

- ตัวอย่างข้อมูลที่สามารถใช้งานได้เลย และยังไม่สามารถใช้งานได้

อายุ	เพศ	เป็นโรคหัวใจ
42	0	1
57	1	1
58	1	0
59	0	0



อายุ	เพศ	เป็นโรคหัวใจ
42	female	1
57	male	1
58	male	0
59	female	0



# Data Requirement

- เราสามารถแปลงได้โดยสามารถใช้ความรู้ในส่วนของ Data Preparation

อายุ	เพศ	เป็นโรคหัวใจ
42	female	1
57	male	1
58	male	0
59	female	0



อายุ	female	male	เป็นโรคหัวใจ
42	1	0	1
57	0	1	1
58	0	1	0
59	1	0	0

# Data Requirement



For more information



## Data Preparation

# Data

**Data Stating**



**Data  
Requirement**



**More about  
Target**



# More about Target

อายุ	female	male	เป็นโรคหัวใจ
42	1	0	True
57	0	1	True
58	0	1	False
59	1	0	False



อายุ	female	male	เป็นโรคหัวใจ
42	1	0	1
57	0	1	1
58	0	1	0
59	1	0	0

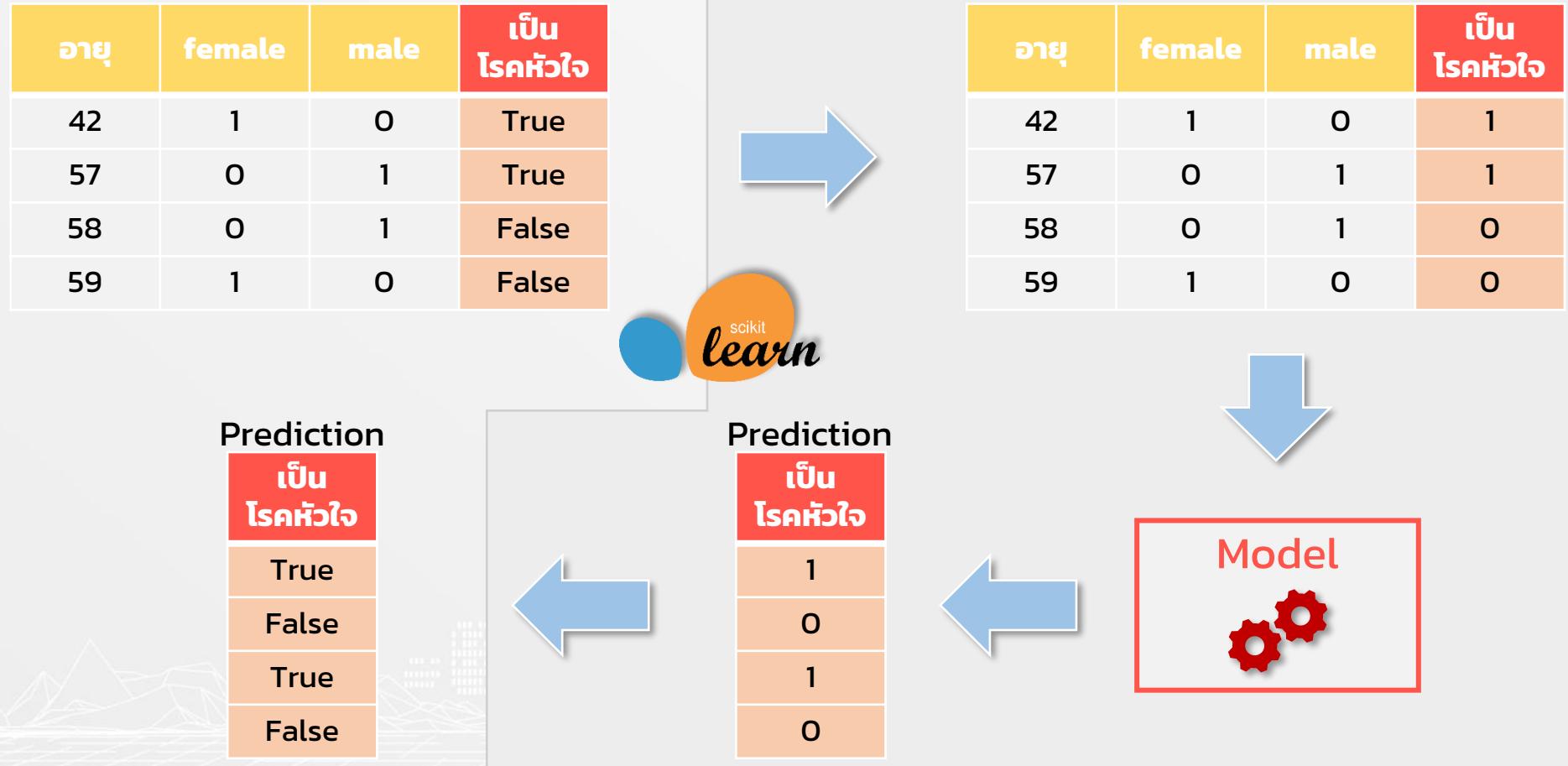


# More about Target

อายุ	female	male	เป็นโรคหัวใจ
42	1	0	True
57	0	1	True
58	0	1	False
59	1	0	False

อายุ	female	male	เป็นโรคหัวใจ
42	1	0	1
57	0	1	1
58	0	1	0
59	1	0	0

# More about Target



# Data

**Data Stating**



**Data  
Requirement**



**More about  
Target**



# Deep Learning for Binary Classification



# Model

# Model

Assumption

Real Face of the  
Model

Cost Function and  
Cost Landscape

How to Create  
Model (Math)

How to Create  
Model (Code)

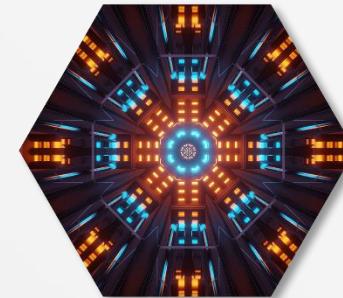
# Assumption

1. Nonlinear Relationship
2. Normality of Residuals
3. Homoscedasticity
4. No Missing Features
5. No Multicollinearity

# Assumption



For more information



Model Improvement  
In DL101

# Model

**Assumption**



Real Face of the  
Model

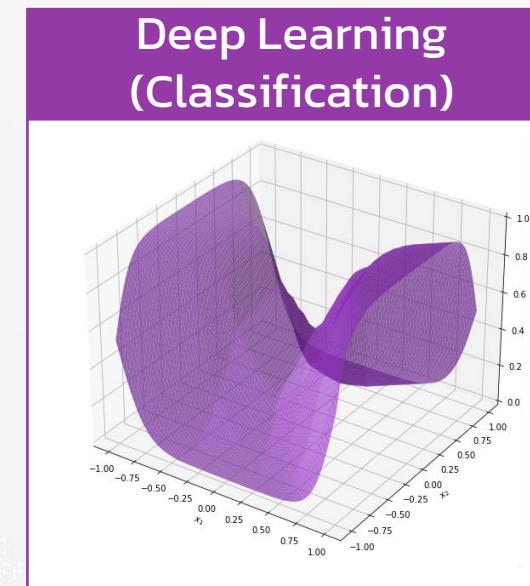
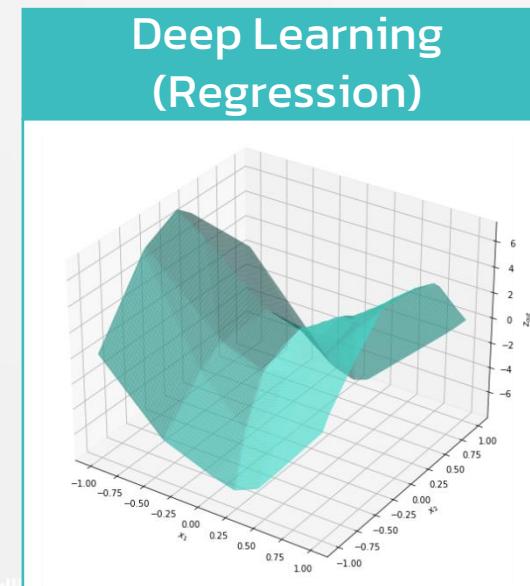
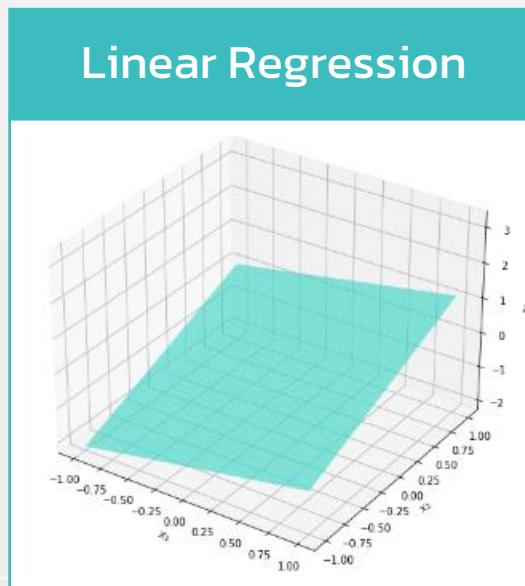
Cost Function and  
Cost Landscape

How to Create  
Model (Math)

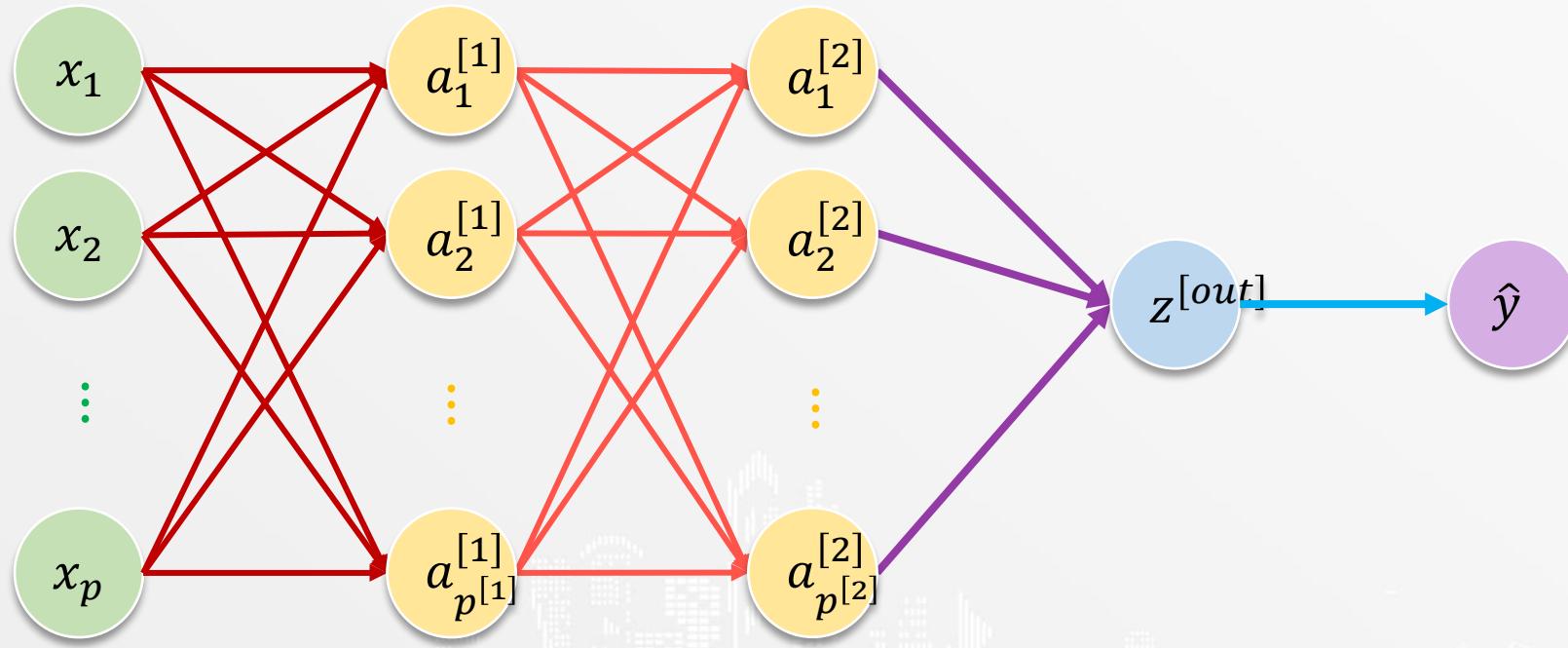
How to Create  
Model (Code)

# Real Face of the Model

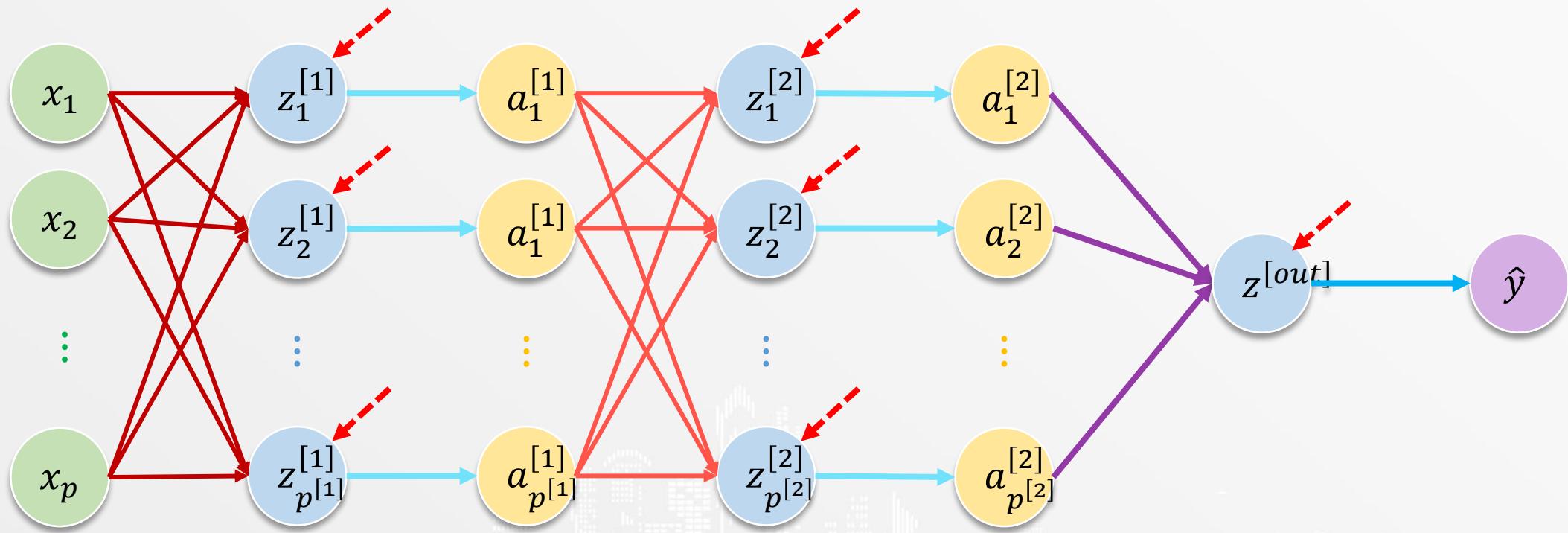
แนวคิดของ deep learning คือการนำ nonlinear function มาประกอบกันเพื่อประเมิน nonlinear function ที่ซับซ้อนได้



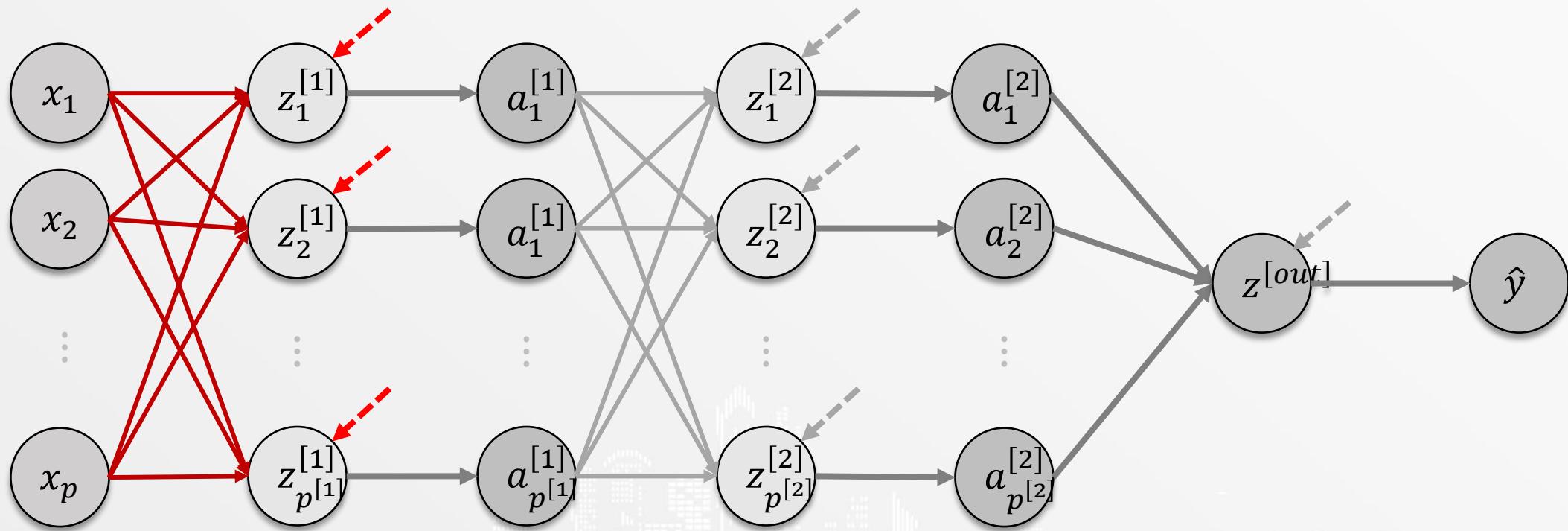
# Real Face of the Model



# Real Face of the Model



# Real Face of the Model

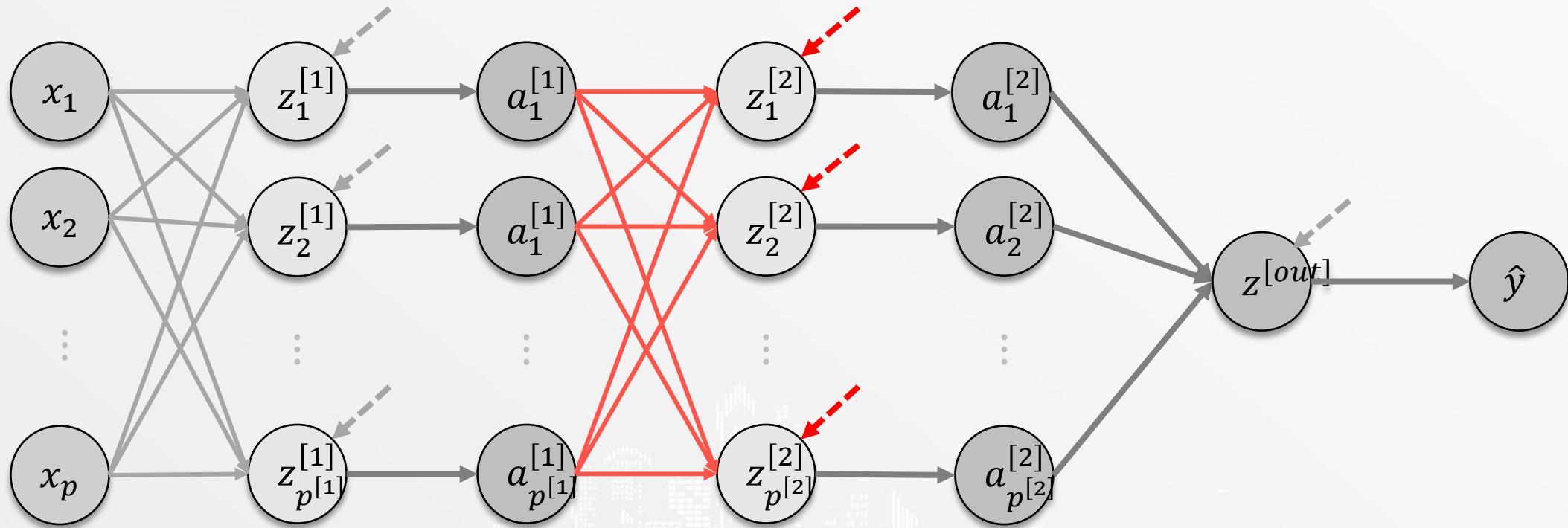


# Real Face of the Model

$$\mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & \dots & b_{p^{[1]}}^{[1]} \end{bmatrix}$$

$$W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & \dots & w_{1,p^{[1]}}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & \dots & w_{2,p^{[1]}}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[1]} & w_{p,2}^{[1]} & \dots & w_{p,p^{[1]}}^{[1]} \end{bmatrix}$$

# Real Face of the Model

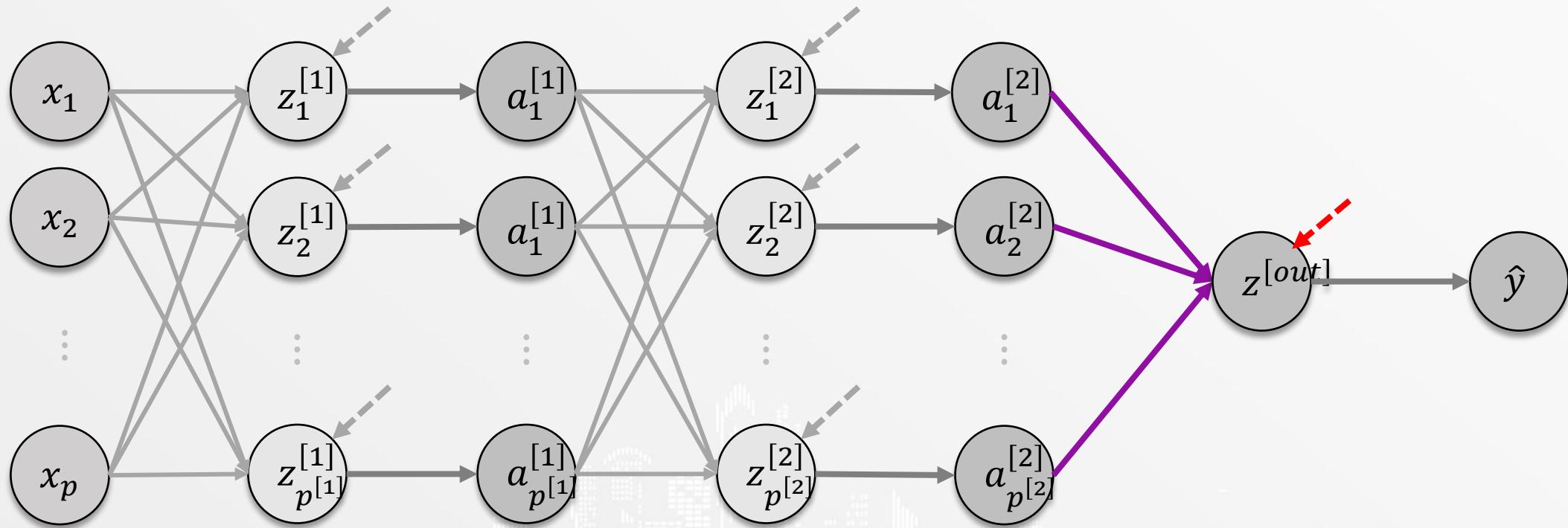


# Real Face of the Model

$$\mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} & b_2^{[2]} & \dots & b_{p^{[2]}}^{[2]} \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} w_{1,1}^{[2]} & w_{1,2}^{[2]} & \dots & w_{1,p^{[2]}}^{[2]} \\ w_{2,1}^{[2]} & w_{2,2}^{[2]} & \dots & w_{2,p^{[2]}}^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p^{[1]},1}^{[2]} & w_{p^{[1]},2}^{[2]} & \dots & w_{p^{[1]},p^{[2]}}^{[2]} \end{bmatrix}$$

# Real Face of the Model



# Real Face of the Model

$$\mathbf{b}^{[out]} = [b^{[out]}]$$

$$W^{[out]} = \begin{bmatrix} w_1^{[out]} \\ w_2^{[out]} \\ \vdots \\ w_{p^{[2]}}^{[out]} \end{bmatrix}$$

# Real Face of the Model

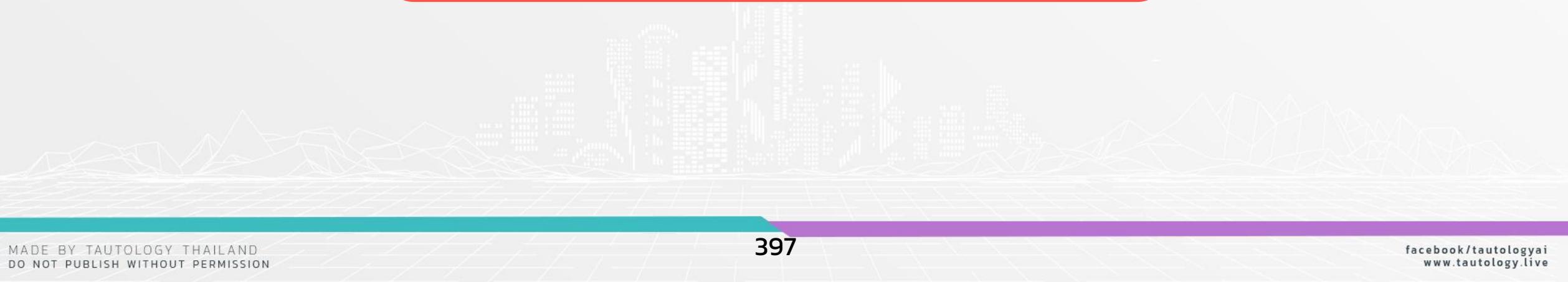
$\mathbf{b}^{[1]}, \mathbf{b}^{[2]}, \mathbf{b}^{[out]}$

$W^{[1]}, W^{[2]}, W^{[out]}$

# Real Face of the Model

$b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$

$W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$



# Real Face of the Model



“ เราต้องการหา  
 $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$   
ที่ทำให้ cost function ต่ำที่สุด ”

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



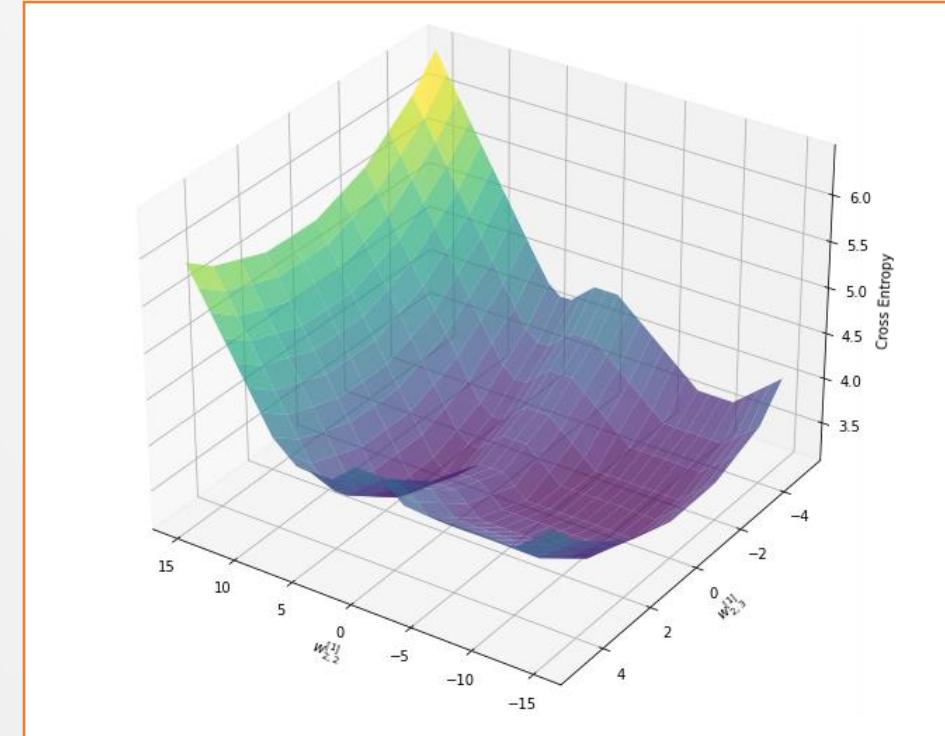
# Cost Function & Cost Landscape

**Cost function** กี่เราจะใช้ในการสร้าง model คือ

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

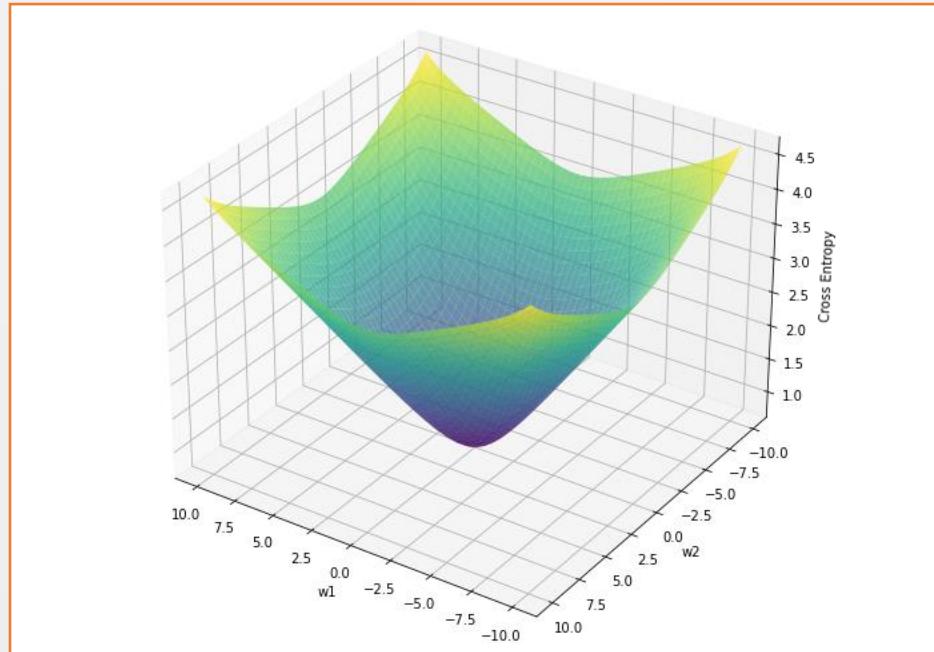
โดยสูตรข้างต้นมีชื่อว่า Cross Entropy

# Cost Function & Cost Landscape

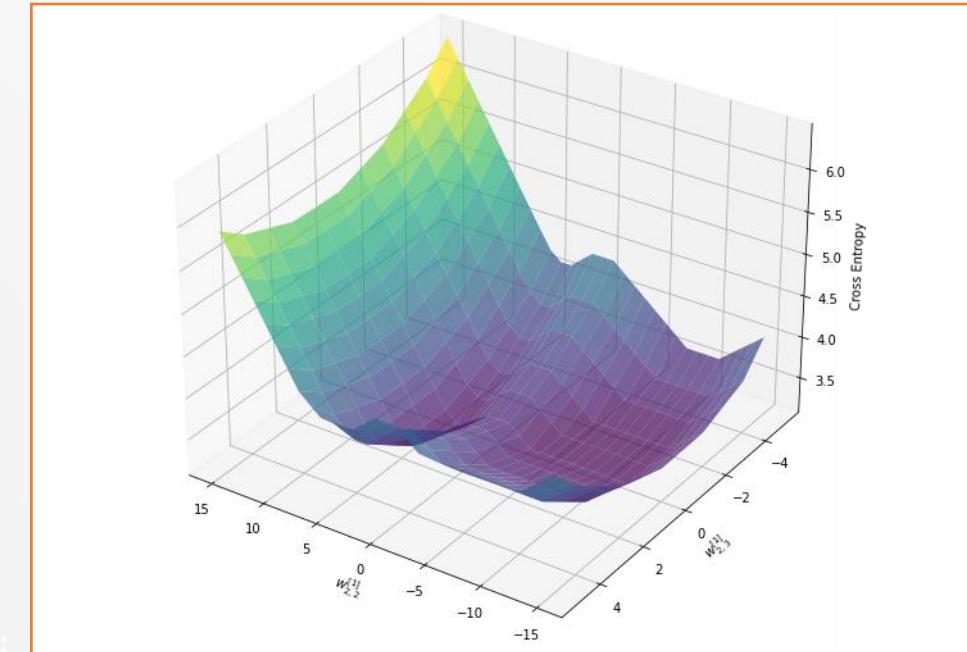


กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape



VS



กราฟแสดง cost landscape ของ logistic regression โดยที่ cost function เป็น cross entropy

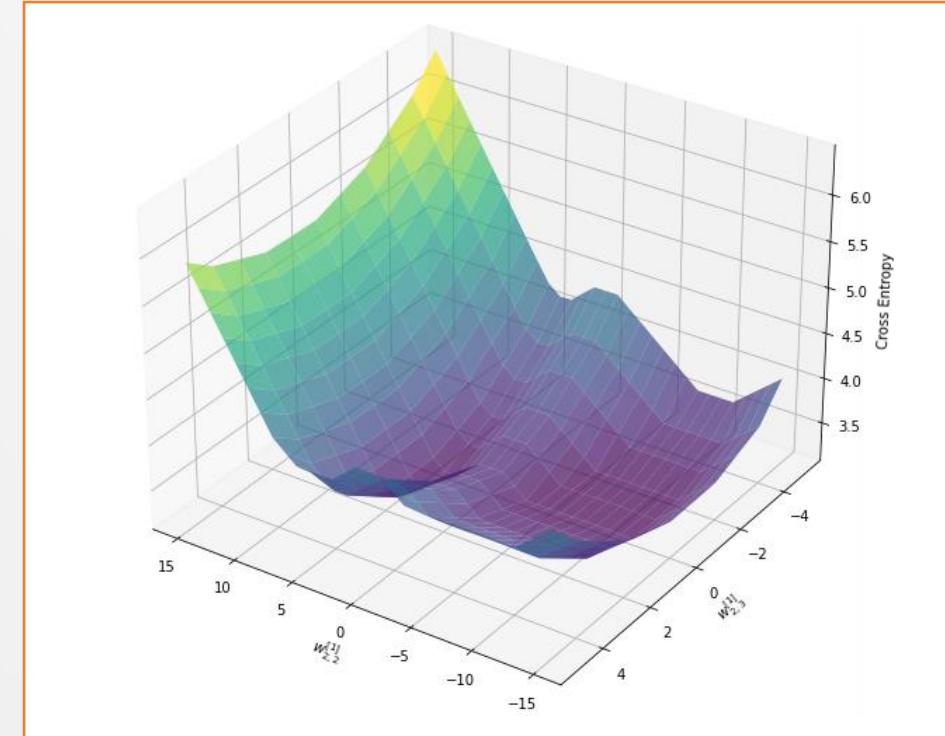
กราฟแสดง cost landscape ของ deep learning โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape

“**Cost landscape** ของ deep learning เป็น non-convex ซึ่งมีจุดต่ำสุดหลายตำแหน่ง”



# Cost Function & Cost Landscape



กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape



For more information



Improvement of  
Deep Learning

# Model

**Assumption**



**Real Face of the Model**



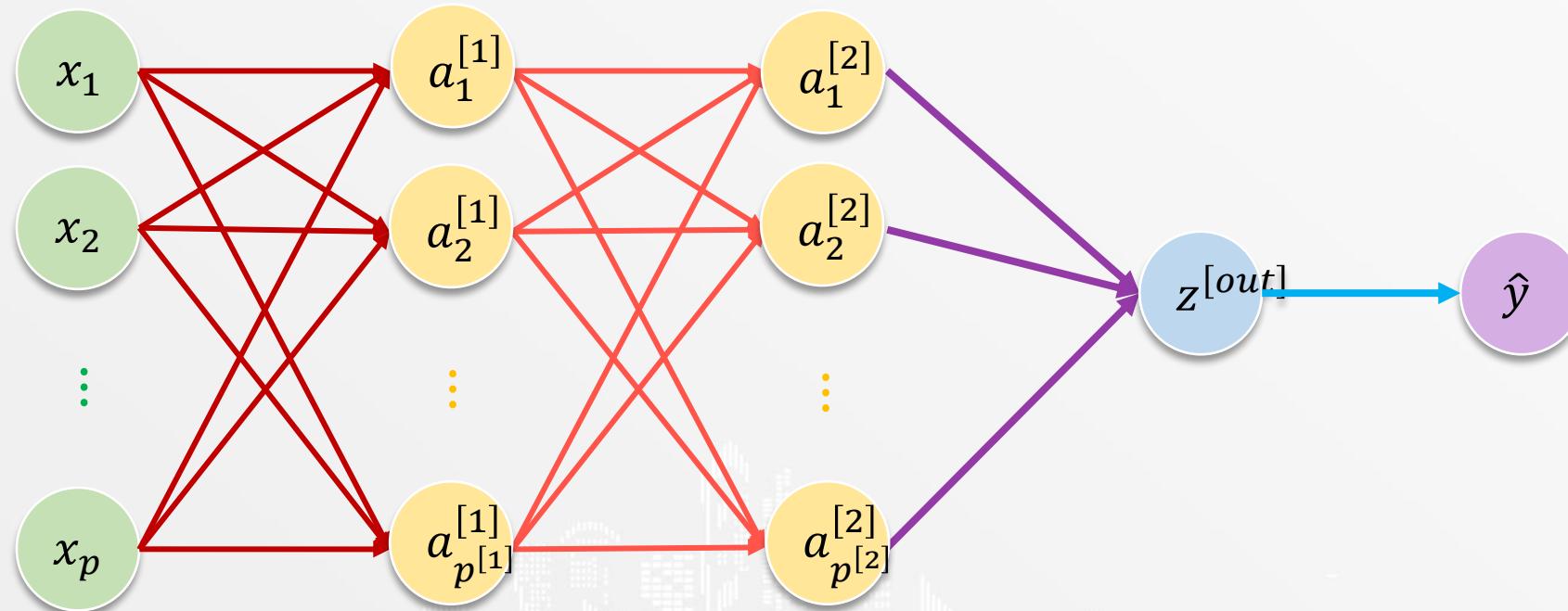
**Cost Function and Cost Landscape**



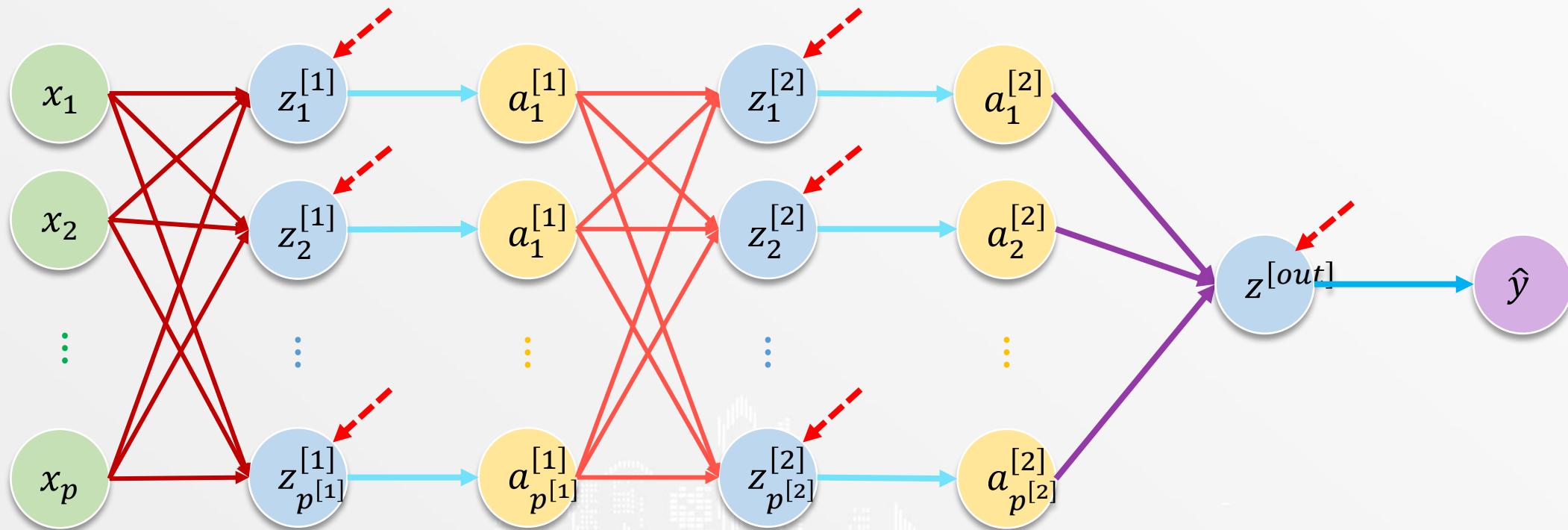
**How to Create Model (Math)**

**How to Create Model (Code)**

# How to Create Model (Math)



# How to Create Model (Math)



# How to Create Model (Math)



“ เราต้องการหา  
 $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ  
 $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$   
ที่ทำให้ cost function ต่ำที่สุด ”

# How to Create Model (Math)

**Cost function** กี่เราจะใช้ในการสร้าง model คือ

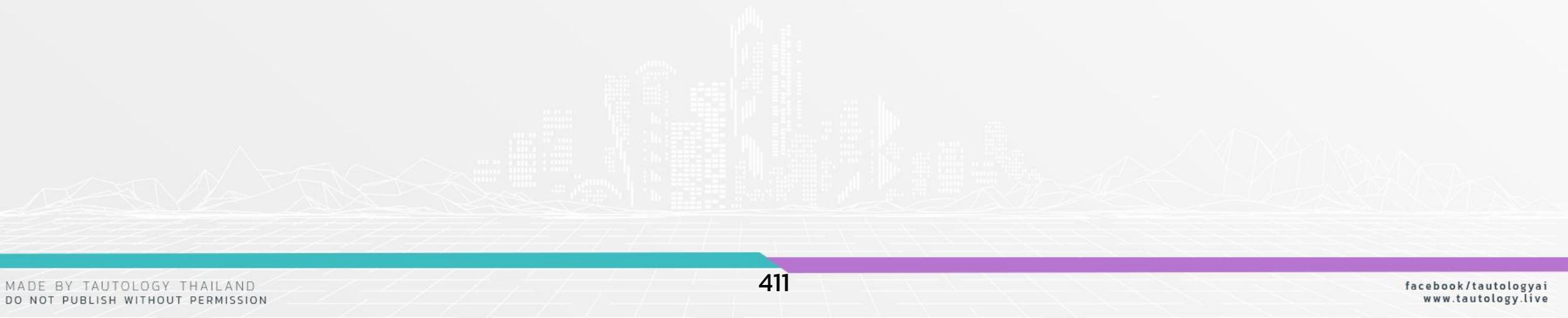
$$-\frac{1}{n} \sum_{i=1}^n (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

โดยสูตรข้างต้นมีชื่อว่า Cross Entropy

# How to Create Model (Math)

เครื่องมือที่เราจะใช้ในการหาค่าตอบ คือ

**“ Gradient Descent ”**



# How to Create Model (Math)

## Equation of Gradient Descent

$$W = W - \alpha \nabla Cost$$

โดย ◆  $\alpha$  คือ ค่าที่ใช้ควบคุม step size ของ  $W$

# How to Create Model (Math)

$$\mathbf{b}^{[1]} = \mathbf{b}^{[1]} - \alpha \nabla Cost$$

$$\mathbf{b}^{[2]} = \mathbf{b}^{[2]} - \alpha \nabla Cost$$

⋮

$$\mathbf{b}^{[L]} = \mathbf{b}^{[L]} - \alpha \nabla Cost$$

$$\mathbf{b}^{[out]} = \mathbf{b}^{[out]} - \alpha \nabla Cost$$

# How to Create Model (Math)

$$W^{[1]} = W^{[1]} - \alpha \nabla Cost$$

$$W^{[2]} = W^{[2]} - \alpha \nabla Cost$$

⋮

$$W^{[L]} = W^{[L]} - \alpha \nabla Cost$$

$$W^{[out]} = W^{[out]} - \alpha \nabla Cost$$

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



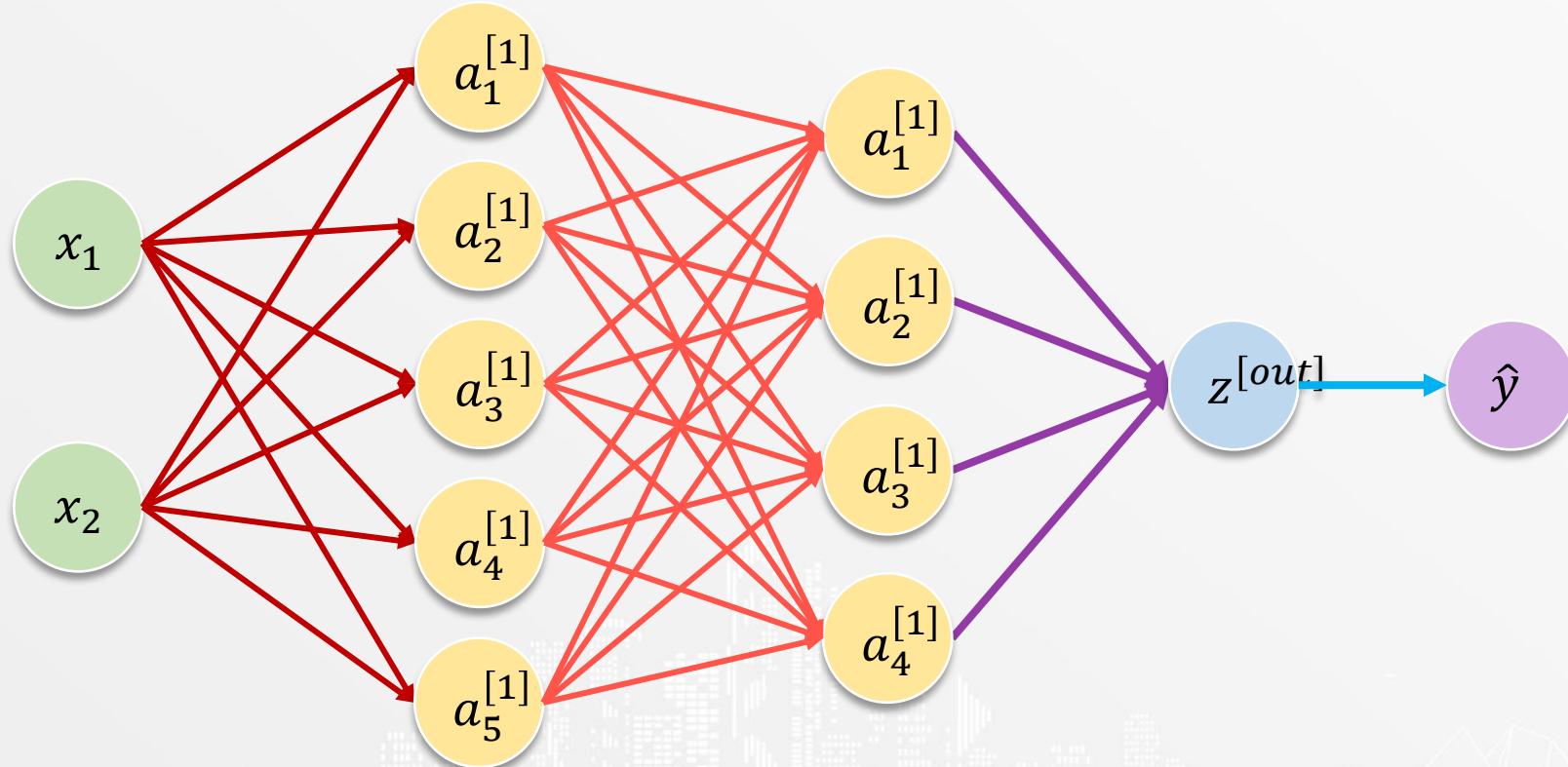
# How to Create Model (Code)

ตัวอย่างการหา weight สำหรับ deep learning

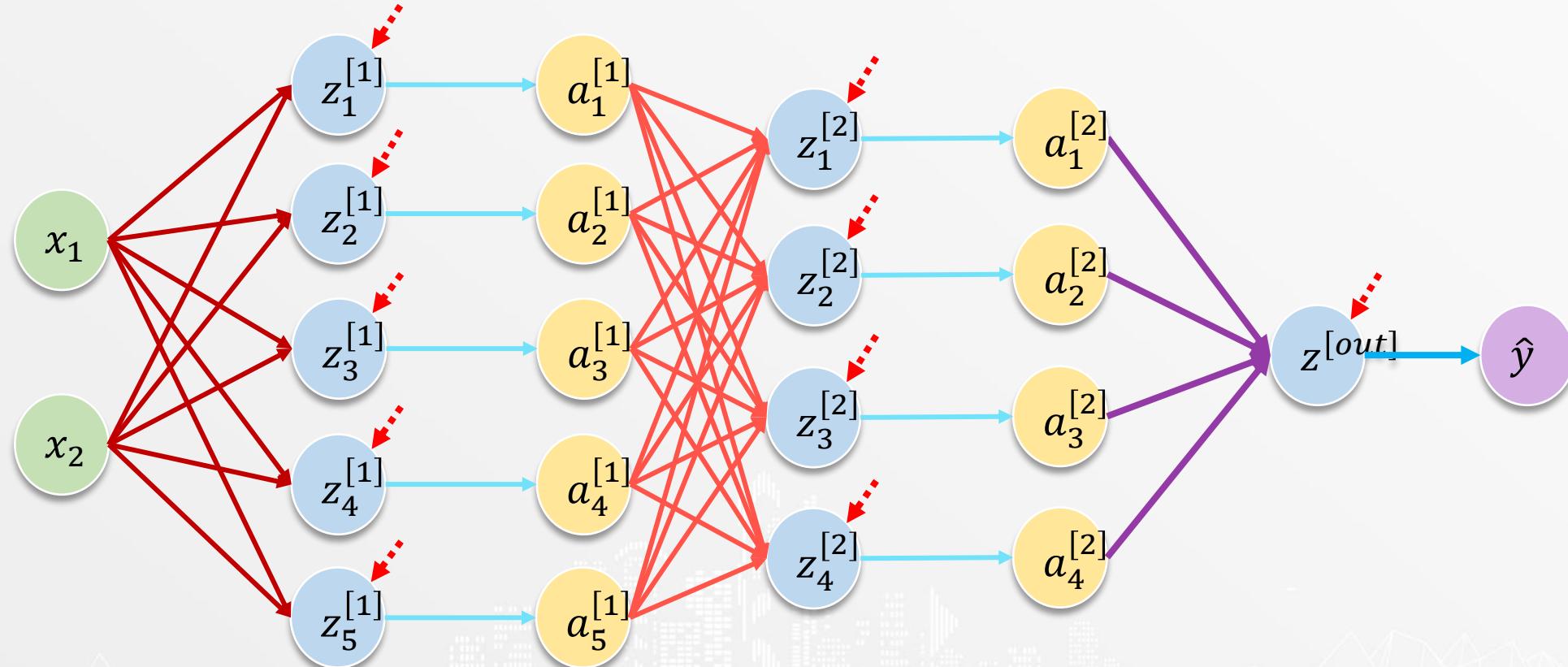
$x_1$	$x_2$	y
2	1	B
2	-1	B
-2	2	A
1	2	B
-2	3	B
2	0	B
-1	-1	A
-2	1	A
0	0	A

$x_1$	$x_2$	y
1	-1	A
-1	0	A
-1	1	A
1	3	B
2	2	B
2	3	B
1	1	B
0	2	B
-1	3	B

# How to Create Model (Code)



# How to Create Model (Code)



# How to Create Model (Code)

$$X = \begin{bmatrix} 2 & 1 \\ 2 & -1 \\ -2 & 2 \\ 1 & 2 \\ \vdots & \vdots \\ 0 & 2 \\ -1 & 3 \end{bmatrix}, \quad y = \begin{bmatrix} B \\ B \\ A \\ B \\ B \\ B \\ B \end{bmatrix}$$

# How to Create Model (Code)

```
1 clf = MLPClassifier(  
2     hidden_layer_sizes=(5, 4),  
3     activation='relu',  
4     solver='sgd',  
5     alpha=0,  
6     learning_rate_init=1,  
7     max_iter=1000,  
8     momentum=0  
9 )  
10  
11 clf.fit(X, y)
```

# How to Create Model (Code)



Code for this section



Open File

**Binary Classification/Model Creation (sklearn).ipynb**

# How to Create Model (Code)

$$\mathbf{b}^{[1]} = [0.002 \quad -0.596 \quad 0.007 \quad 1.308 \quad 0.69]$$

$$W^{[1]} = \begin{bmatrix} 1.321 & -0.377 & -0.273 & -1.071 & 0.769 \\ 1.324 & 0.685 & 0.7 & -0.928 & -0.737 \end{bmatrix}$$

# How to Create Model (Code)

$$\mathbf{b}^{[2]} = [-0.028 \quad -0.138 \quad 0.696 \quad 1.072]$$

$$W^{[2]} = \begin{bmatrix} -0.763 & -0.585 & 1.35 & -0.848 \\ -0.472 & -0.124 & 0.475 & 0.414 \\ -0.458 & -0.431 & 0.313 & -0.459 \\ 0.58 & 0.161 & -0.95 & 1.393 \\ -0.746 & -0.718 & 0.408 & 0.099 \end{bmatrix}$$

# How to Create Model (Code)

$$\mathbf{b}^{[out]} = [0.403]$$

$$W^{[out]} = \begin{bmatrix} -0.096 \\ 1.065 \\ 1.682 \\ -1.779 \end{bmatrix}$$

# How to Create Model (Code)

$$a_1^{[1]} = \text{ReLU}(0.002 + 1.321x_1 + 1.324x_2)$$

$$a_2^{[1]} = \text{ReLU}(-0.596 - 0.377x_1 + 0.685x_2)$$

$$a_3^{[1]} = \text{ReLU}(0.007 - 0.273x_1 + 0.7x_2)$$

$$a_4^{[1]} = \text{ReLU}(1.308 - 1.071x_1 - 0.928x_2)$$

$$a_5^{[1]} = \text{ReLU}(0.69 + 0.769x_1 - 0.737x_2)$$

# How to Create Model (Code)

$$a_1^{[2]} = \text{ReLU} \left( -0.028 - 0.763a_1^{[1]} - 0.472a_2^{[1]} - 0.458a_3^{[1]} + 0.58a_4^{[1]} - 0.746a_5^{[1]} \right)$$

$$a_2^{[2]} = \text{ReLU} \left( -0.138 - 0.585a_1^{[1]} - 0.124a_2^{[1]} - 0.431a_3^{[1]} + 0.161a_4^{[1]} - 0.718a_5^{[1]} \right)$$

$$a_3^{[2]} = \text{ReLU} \left( 0.696 + 1.35a_1^{[1]} + 0.475a_2^{[1]} + 0.313a_3^{[1]} - 0.95a_4^{[1]} + 0.408a_5^{[1]} \right)$$

$$a_4^{[2]} = \text{ReLU} \left( 1.072 - 0.848a_1^{[1]} + 0.414a_2^{[1]} - 0.459a_3^{[1]} + 1.393a_4^{[1]} + 0.099a_5^{[1]} \right)$$

# How to Create Model (Code)

$$z^{[out]} = 0.403 - 0.096a_1^{[3]} + 1.065a_2^{[3]} + 1.682a_3^{[3]} - 1.779a_4^{[3]}$$

$$\hat{y} = \sigma(z^{[out]}) = \frac{1}{1 + e^{-z^{[out]}}}$$

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



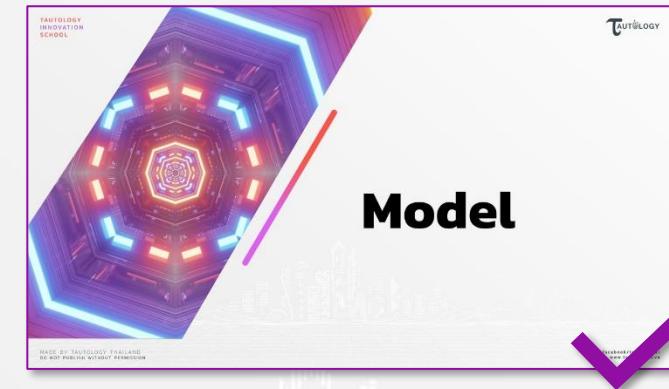
**How to Create Model (Math)**



**How to Create Model (Code)**

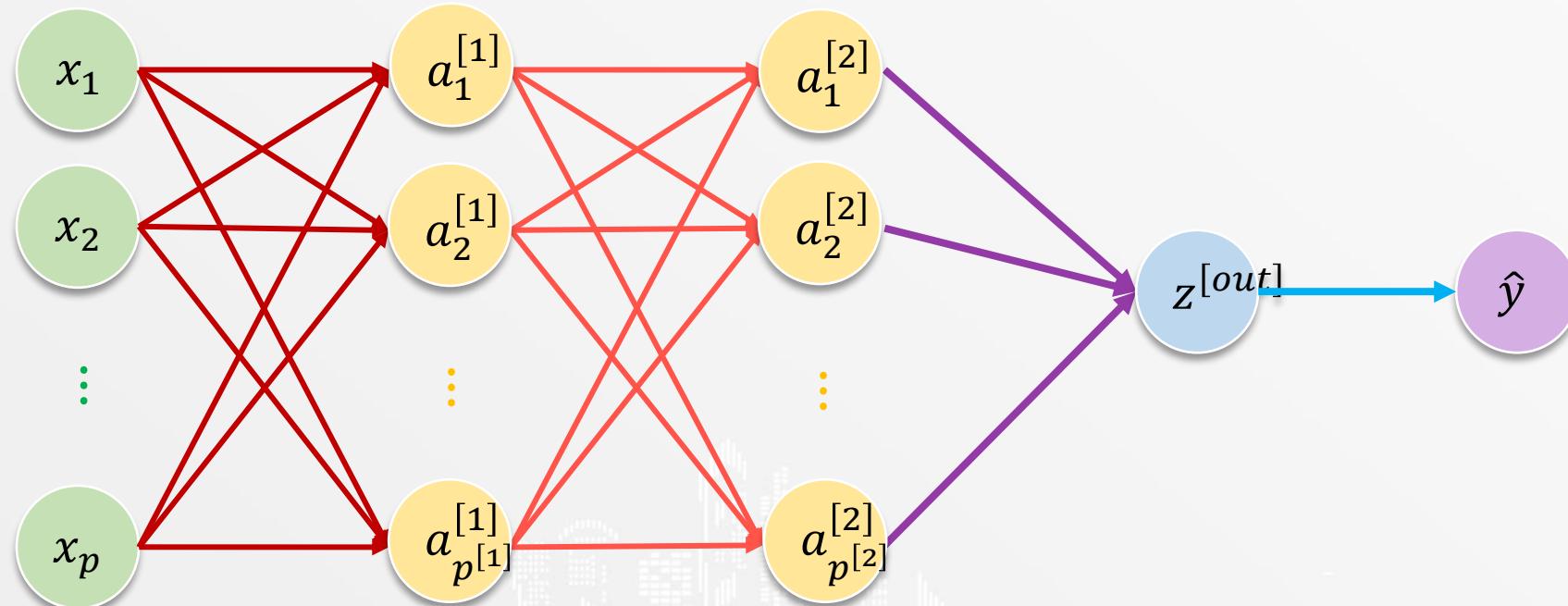


# Deep Learning for Binary Classification

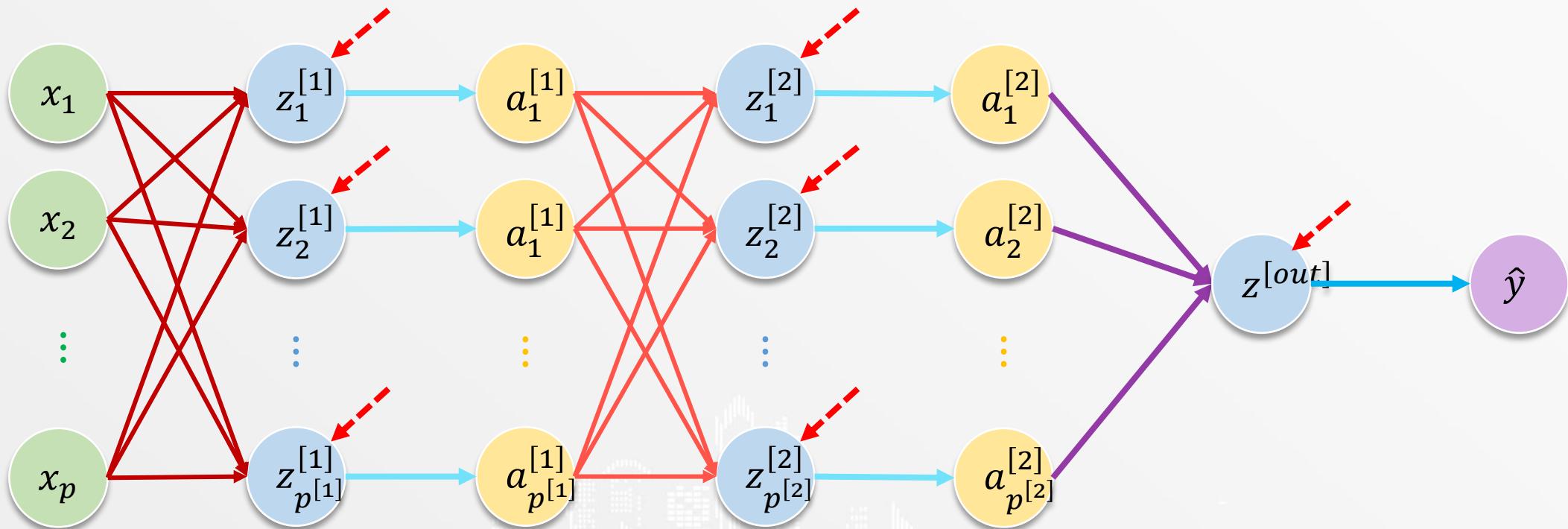


# Prediction

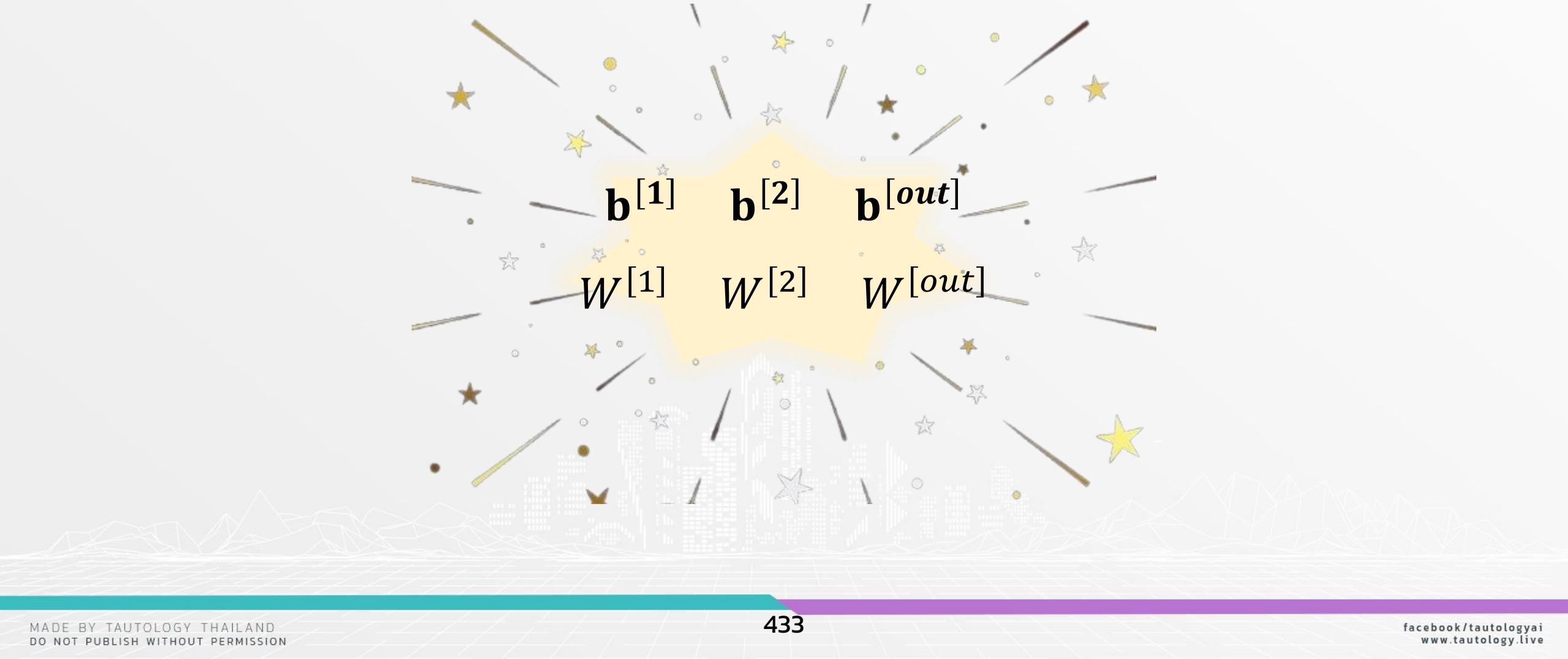
# Prediction



# Prediction



# Prediction

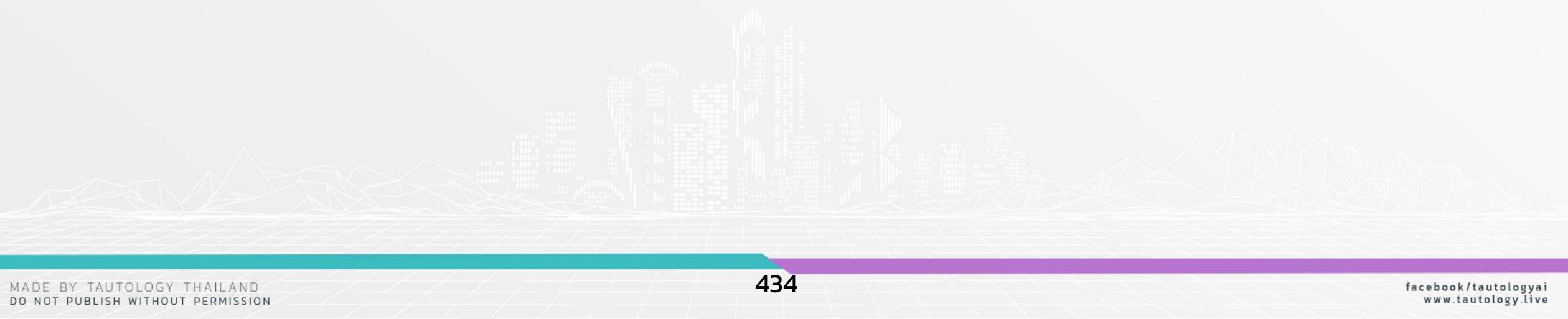


# Prediction

1-Sample

Multi-Sample

Code



# 1-Sample

ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
0	3



$\hat{y}$
?

# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$\mathbf{b}^{[1]} = [0.002 \quad -0.596 \quad 0.007 \quad 1.308 \quad 0.69]$$

$$W^{[1]} = \begin{bmatrix} 1.321 & -0.377 & -0.273 & -1.071 & 0.769 \\ 1.324 & 0.685 & 0.7 & -0.928 & -0.737 \end{bmatrix}$$

# 1-Sample

- สมมติว่า weight ของปัจจัยหนึ่งที่เราหมายได้คือ

$$\mathbf{b}^{[2]} = [-0.028 \quad -0.138 \quad 0.696 \quad 1.072]$$

$$W^{[2]} = \begin{bmatrix} -0.763 & -0.585 & 1.35 & -0.848 \\ -0.472 & -0.124 & 0.475 & 0.414 \\ -0.458 & -0.431 & 0.313 & -0.459 \\ 0.58 & 0.161 & -0.95 & 1.393 \\ -0.746 & -0.718 & 0.408 & 0.099 \end{bmatrix}$$

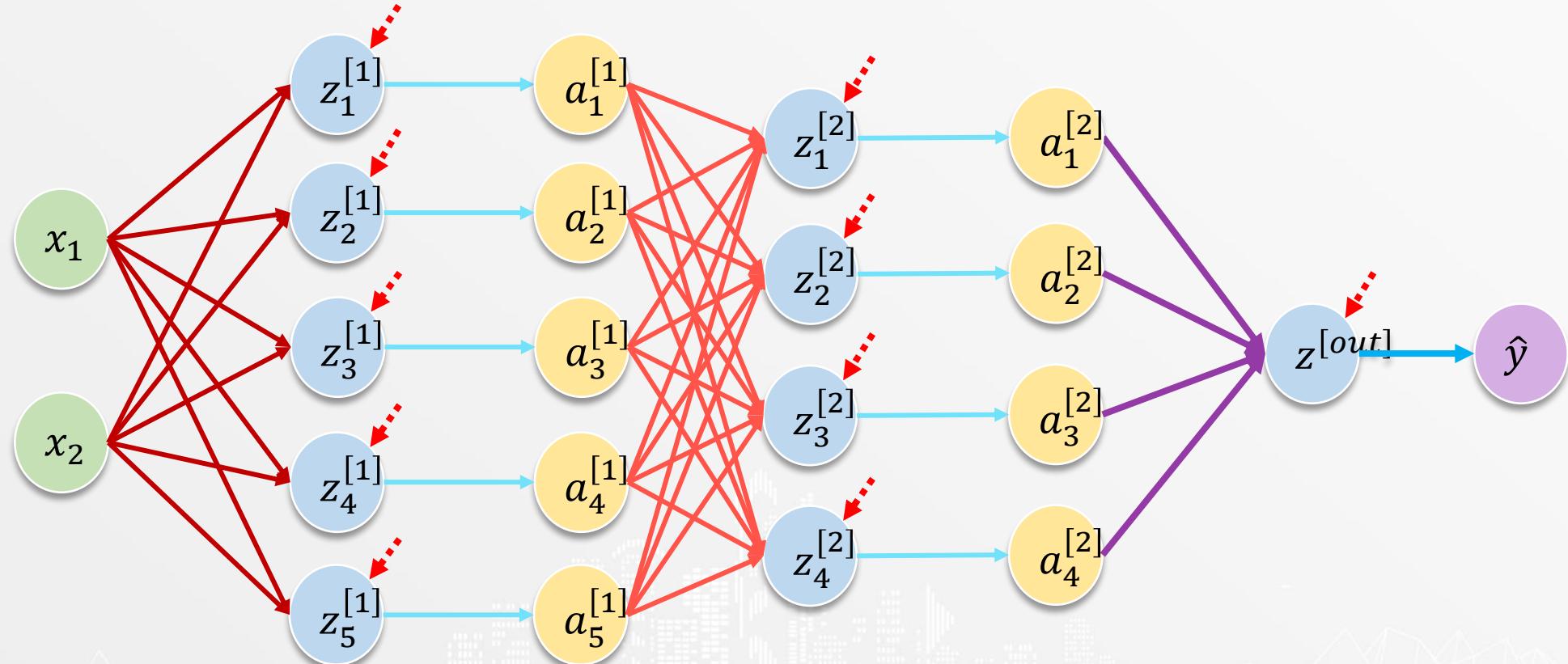
# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$\mathbf{b}^{[out]} = [0.403]$$

$$W^{[out]} = \begin{bmatrix} -0.096 \\ 1.065 \\ 1.682 \\ -1.779 \end{bmatrix}$$

# 1-Sample



# 1-Sample

$$a_1^{[1]} = \text{ReLU}(0.002 + 1.321x_1 + 1.324x_2) = 3.974$$

$$a_2^{[1]} = \text{ReLU}(-0.596 - 0.377x_1 + 0.685x_2) = 1.459$$

$$a_3^{[1]} = \text{ReLU}(0.007 - 0.273x_1 + 0.7x_2) = 2.107$$

$$a_4^{[1]} = \text{ReLU}(1.308 - 1.071x_1 - 0.928x_2) = 0$$

$$a_5^{[1]} = \text{ReLU}(0.69 + 0.769x_1 - 0.737x_2) = 0$$

# 1-Sample

$$a_1^{[2]} = \text{ReLU}(-0.028 - 0.763a_1^{[1]} - 0.472a_2^{[1]} - 0.458a_3^{[1]} + 0.58a_4^{[1]} - 0.746a_5^{[1]}) = 0$$

$$a_2^{[2]} = \text{ReLU}(-0.138 - 0.585a_1^{[1]} - 0.124a_2^{[1]} - 0.431a_3^{[1]} + 0.161a_4^{[1]} - 0.718a_5^{[1]}) = 0$$

$$a_3^{[2]} = \text{ReLU}(0.696 + 1.35a_1^{[1]} + 0.475a_2^{[1]} + 0.313a_3^{[1]} - 0.95a_4^{[1]} + 0.408a_5^{[1]}) = 7.143$$

$$a_4^{[2]} = \text{ReLU}(1.072 - 0.848a_1^{[1]} + 0.414a_2^{[1]} - 0.459a_3^{[1]} + 1.393a_4^{[1]} + 0.099a_5^{[1]}) = 0$$

# 1-Sample

$$z^{[out]} = 0.403 - 0.096a_1^{[3]} + 1.065a_2^{[3]} + 1.682a_3^{[3]} - 1.779a_4^{[3]} = 12.066$$

$$\hat{y} = \sigma(z^{[out]}) = \frac{1}{1 + e^{-z^{[out]}}} = \frac{1}{1 + e^{-12.066}} = 1 \Rightarrow \textcolor{red}{B^*}$$

0  $\Rightarrow$  A  
1  $\Rightarrow$  B

# 1-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
0	3



$\hat{y}$
$B$

# Prediction

1-Sample



Multi-Sample



Code



# Multi-Sample

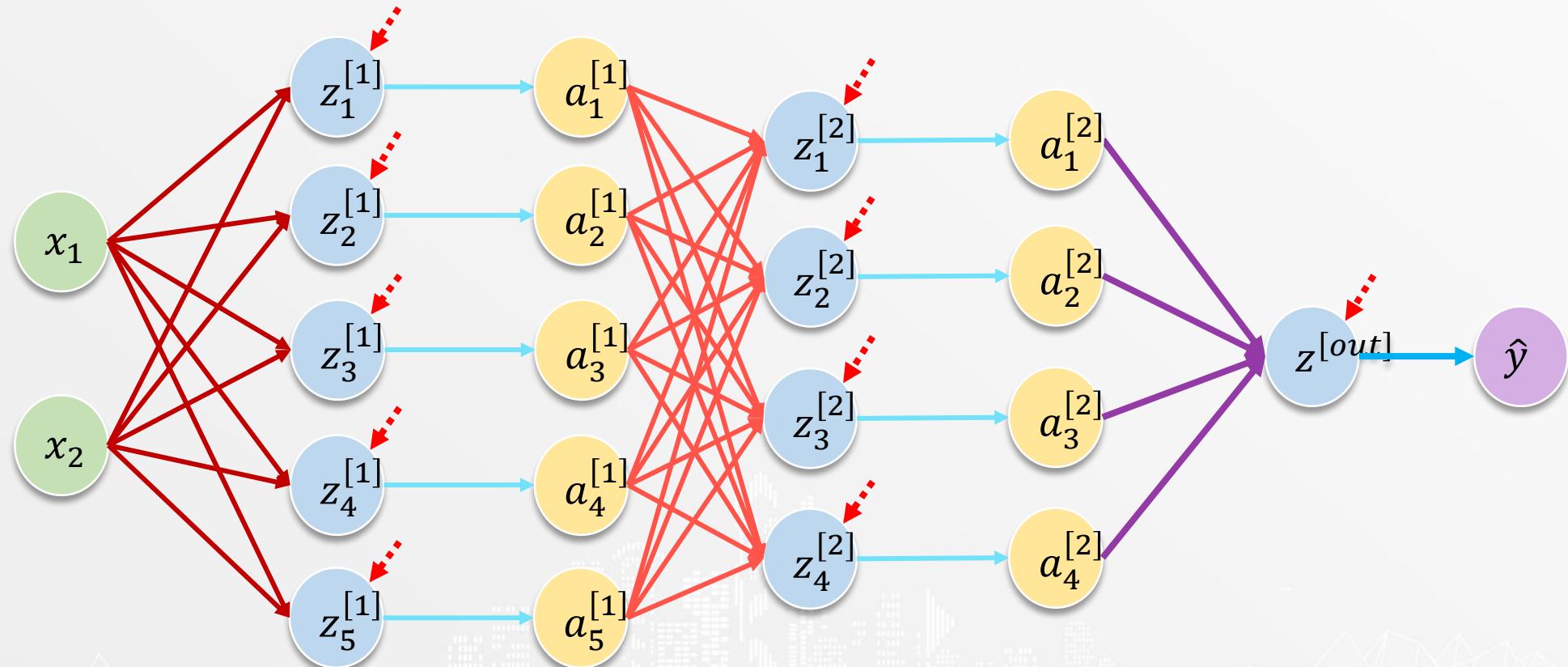
ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
0	3
-2	-1
-2	1
0	-1



$\hat{y}$
?
?
?
?

# Multi-Sample



# Multi-Sample

$$X \rightarrow Z^{[1]} \rightarrow A^{[1]} \rightarrow Z^{[2]} \rightarrow A^{[2]} \rightarrow z^{[out]} \rightarrow \hat{y}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + \mathbf{b}^{[1]}$$

$$Z^{[1]} = \begin{bmatrix} 0 & 3 \\ -2 & -1 \\ -2 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1.321 & -0.377 & -0.273 & -1.071 & 0.769 \\ 1.324 & 0.685 & 0.7 & -0.928 & -0.737 \end{bmatrix} + [0.002 \quad -0.596 \quad 0.007 \quad 1.308 \quad 0.69]$$

$$= \begin{bmatrix} 3.971 & 2.054 & 2.099 & -2.783 & -2.21 \\ -3.965 & 0.068 & -0.155 & 3.07 & -0.801 \\ 1.324 & 0.685 & 0.7 & -0.928 & -0.737 \\ -1.324 & -0.685 & -0.7 & 0.928 & 0.737 \end{bmatrix} + \begin{bmatrix} 0.002 & -0.596 & 0.007 & 1.308 & 0.69 \\ 0.002 & -0.596 & 0.007 & 1.308 & 0.69 \\ 0.002 & -0.596 & 0.007 & 1.308 & 0.69 \\ 0.002 & -0.596 & 0.007 & 1.308 & 0.69 \end{bmatrix}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + \mathbf{b}^{[1]}$$

$$Z^{[1]} = \begin{bmatrix} 3.973 & 1.458 & 2.106 & -1.475 & -1.52 \\ -3.963 & -0.528 & -0.148 & 4.378 & -0.111 \\ 1.326 & 0.089 & 0.707 & 0.38 & -0.047 \\ -1.322 & -1.281 & -0.693 & 2.236 & 1.427 \end{bmatrix}$$

# Multi-Sample

$$A^{[1]} = \text{ReLU}(Z^{[1]})$$

$$A^{[1]} = \text{ReLU} \begin{pmatrix} 3.973 & 1.458 & 2.106 & -1.475 & -1.52 \\ -3.963 & -0.528 & -0.148 & 4.378 & -0.111 \\ 1.326 & 0.089 & 0.707 & 0.38 & -0.047 \\ -1.322 & -1.281 & -0.693 & 2.236 & 1.427 \end{pmatrix}$$

$$= \begin{bmatrix} 3.973 & 1.458 & 2.106 & 0 & 0 \\ 0 & 0 & 0 & 4.378 & 0 \\ 1.326 & 0.089 & 0.707 & 0.38 & 0 \\ 0 & 0 & 0 & 2.236 & 1.427 \end{bmatrix}$$

# Multi-Sample

$$Z^{[2]} = A^{[1]}W^{[2]} + \mathbf{b}^{[2]}$$

$$\begin{aligned} Z^{[2]} &= \begin{bmatrix} 3.973 & 1.458 & 2.106 & 0 & 0 \\ 0 & 0 & 0 & 4.378 & 0 \\ 1.326 & 0.089 & 0.707 & 0.38 & 0 \\ 0 & 0 & 0 & 2.236 & 1.427 \end{bmatrix} \begin{bmatrix} -0.763 & -0.585 & 1.35 & -0.848 \\ -0.472 & -0.124 & 0.475 & 0.414 \\ -0.458 & -0.431 & 0.313 & -0.459 \\ 0.58 & 0.161 & -0.95 & 1.393 \\ -0.746 & -0.718 & 0.408 & 0.099 \end{bmatrix} \\ &\quad + [-0.028 \quad -0.138 \quad 0.696 \quad 1.072] \\ &= \begin{bmatrix} -4.685 & -3.414 & 6.715 & -3.733 \\ 2.537 & 0.706 & -4.157 & 6.097 \\ -1.157 & -1.031 & 1.693 & -0.883 \\ 0.231 & -0.663 & -1.542 & 3.255 \end{bmatrix} + \begin{bmatrix} -0.028 & -0.138 & 0.696 & 1.072 \\ -0.028 & -0.138 & 0.696 & 1.072 \\ -0.028 & -0.138 & 0.696 & 1.072 \\ -0.028 & -0.138 & 0.696 & 1.072 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[2]} = A^{[1]}W^{[2]} + \mathbf{b}^{[2]}$$

$$Z^{[2]} = \begin{bmatrix} -4.713 & -3.552 & 7.411 & -2.661 \\ 2.509 & 0.568 & -3.461 & 7.169 \\ -1.185 & -1.169 & 2.389 & 0.189 \\ 0.203 & -0.801 & -0.846 & 4.327 \end{bmatrix}$$

# Multi-Sample

$$A^{[2]} = \text{ReLU}(z^{[2]})$$

$$A^{[2]} = \text{ReLU} \left( \begin{bmatrix} -4.713 & -3.552 & 7.411 & -2.661 \\ 2.509 & 0.568 & -3.461 & 7.169 \\ -1.185 & -1.169 & 2.389 & 0.189 \\ 0.203 & -0.801 & -0.846 & 4.327 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 0 & 0 & 7.411 & 0 \\ 2.509 & 0.568 & 0 & 7.169 \\ 0 & 0 & 2.389 & 0.189 \\ 0.203 & 0 & 0 & 4.327 \end{bmatrix}$$

# Multi-Sample

$$\mathbf{z}^{[out]} = A^{[2]}W^{[out]} + \mathbf{b}^{[out]}$$

$$\mathbf{z}^{[out]} = \begin{bmatrix} 0 & 0 & 7.411 & 0 \\ 2.509 & 0.568 & 0 & 7.169 \\ 0 & 0 & 2.389 & 0.189 \\ 0.203 & 0 & 0 & 4.327 \end{bmatrix} \begin{bmatrix} -0.096 \\ 1.065 \\ 1.682 \\ -1.779 \end{bmatrix} + [0.403]$$

$$= \begin{bmatrix} 12.462 \\ -12.388 \\ 3.681 \\ -7.717 \end{bmatrix} + \begin{bmatrix} 0.403 \\ 0.403 \\ 0.403 \\ 0.403 \end{bmatrix} = \begin{bmatrix} 12.059 \\ -12.791 \\ 3.278 \\ -8.12 \end{bmatrix}$$

# Multi-Sample

$$\hat{\mathbf{y}} = \sigma(\mathbf{z}^{[out]})$$

$$\hat{\mathbf{y}} = \sigma \begin{pmatrix} 12.059 \\ -12.791 \\ 3.278 \\ -8.12 \end{pmatrix}$$

$$= \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \rightarrow \boxed{\begin{bmatrix} B \\ A \\ B \\ A \end{bmatrix}}^*$$

0  $\Rightarrow$  A  
1  $\Rightarrow$  B

# Multi-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
0	3
-2	-1
-2	1
0	-1



$\hat{y}$
B
A
B
A

# Prediction

1-Sample



Multi-Sample



Code



# Code

```
1 | clf.predict(X)
```

```
array(['B', 'A', 'B', 'A'], dtype='<U1')
```

# Code



Code for this section



Open File

**Binary Classification/Model Creation (sklearn).ipynb**

# Prediction

1-Sample



Multi-Sample



Code



# Deep Learning for Binary Classification



# Deep Learning for Classification

**Deep Learning for  
Binary Classification**

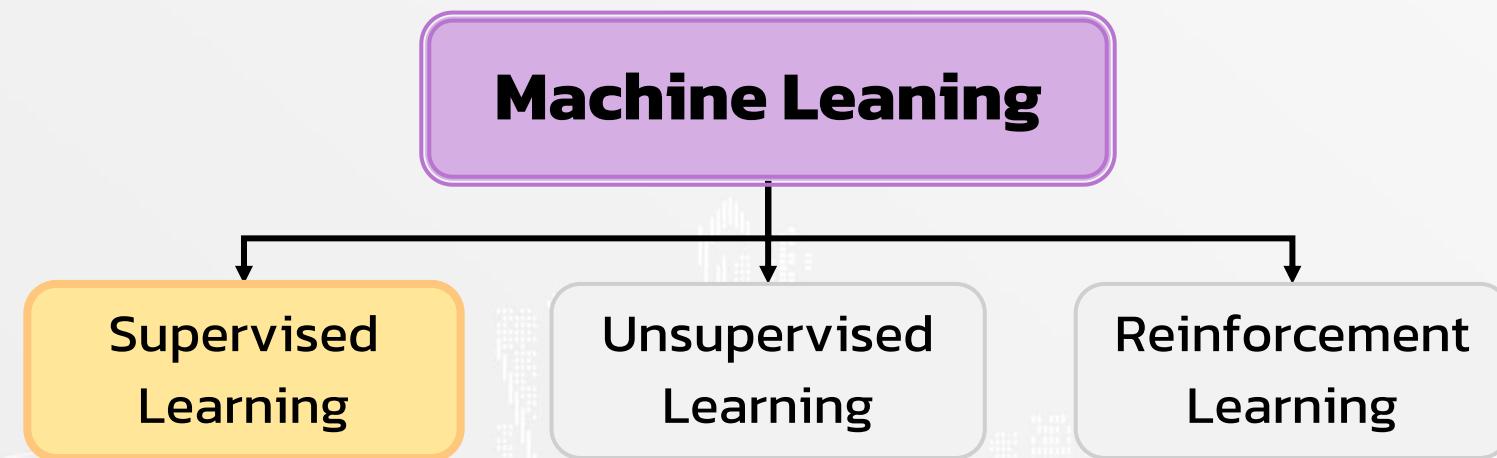


**Deep Learning for  
Multi-Class Classification**



# Deep Learning for Multi-Class Classification

Deep Learning เป็นหนึ่งใน algorithm ประเภท  
supervised learning



# Concept of Supervised Learning

Data → Model → Prediction

# Deep Learning for Multi-Class Classification





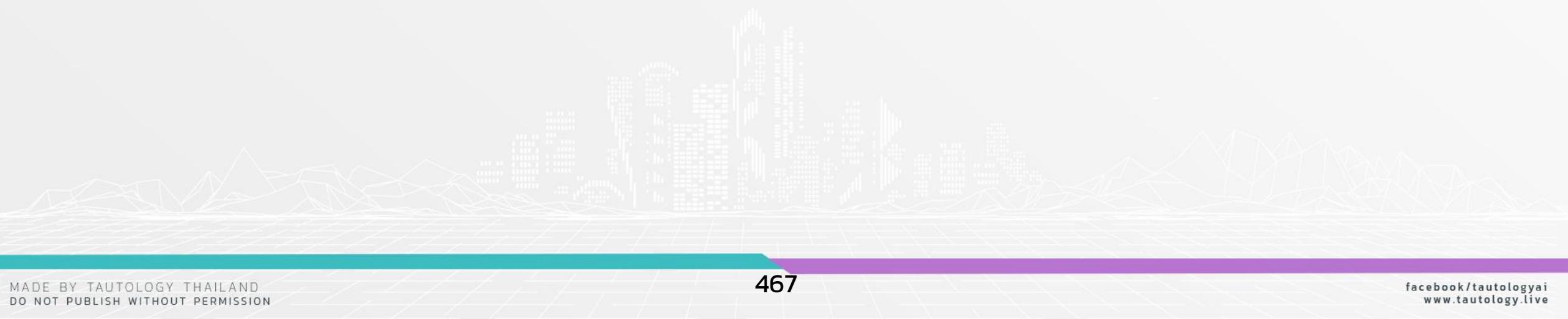
# Data

# Data

Data Stating

Data  
Requirement

More about  
Target



# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

# Data Stating

The diagram illustrates a data matrix represented as a table. The columns are labeled  $x_1, x_2, x_3, \dots, x_p$  and the rows are labeled  $y_1, y_2, y_3, \dots, y_n$ . The matrix contains elements  $x_{i,j}$  where  $i$  is the row index and  $j$  is the column index. The matrix is surrounded by green arrows pointing to the left, indicating the width of the data (number of features). Above the matrix, five purple arrows point downwards, indicating the height of the data (number of samples).

$x_1$	$x_2$	$x_3$	$\cdots$	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$\cdots$	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$\cdots$	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$\cdots$	$x_{3,p}$	$y_3$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	$\cdots$	$x_{n,p}$	$y_n$

- ◇  $n$  คือ จำนวน sample
- ◇  $p$  คือ จำนวน feature

# Data Stating

$x_1$	$x_2$	$x_3$	...	$x_p$	$y$
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	...	$x_{1,p}$	$y_1$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	...	$x_{2,p}$	$y_2$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	...	$x_{3,p}$	$y_3$
:	:	:	..	:	:
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	...	$x_{n,p}$	$y_n$

- $x_{2,3}$  คือ sample ที่ 2 feature ที่ 3
- $x_{3,p}$  คือ sample ที่ 3 feature ที่  $p$
- $x_{n,p}$  คือ sample ที่  $n$  feature ที่  $p$
- $y_2$  คือ target ของ sample ที่ 2
- $y_3$  คือ target ของ sample ที่ 3
- $y_n$  คือ target ของ sample ที่  $n$

# Data Stating

## Example

- เราต้องการจำแนกประเภทของโรคเบ้าหวาน, โควิด และหัวใจ โดยพิจารณาจาก เพศ, อุณหภูมิร่างกาย, น้ำตาลในเลือด และ ความดันโลหิต

## Data

เพศ	อุณหภูมิร่างกาย (°C)	น้ำตาลในเลือด (mg/dl)	ความดันโลหิต (mmHg)	โรค
0	36.6	126	122	0
1	39.5	70	120	1
1	40.1	84	95	1
0	36.5	79	151	2

\*\*0=เบ้าหวาน, 1=โควิด, 2=หัวใจ

# Data Stating

- ข้อมูลตามแนวแอกว คือ Sample

เพศ	อุณหภูมิร่างกาย (°C)	น้ำตาลในเลือด (mg/dl)	ความดันโลหิต (mmHg)	โรค
0	36.6	126	122	0
1	39.5	70	120	1
1	40.1	84	95	1
0	36.5	79	151	2

# Data Stating

- ข้อมูลตามแนวหลัก คือ Feature and Target
  - Feature (ตัวแปรต้น) คือ ข้อมูลที่ส่งผลให้เกิด target
  - Target (ตัวแปรตาม) คือ ข้อมูลที่เราสนใจจะพยากรณ์

Feature

เพศ	อุณหภูมิร่างกาย (°C)	น้ำตาลในเลือด (mg/dl)	ความดันโลหิต (mmHg)
0	36.6	126	122
1	39.5	70	120
1	40.1	84	95
0	36.5	79	151

Target

โรค
0
1
1
2

# Data Stating

- Feature and Target
  - เราสามารถแยก และปรับให้เป็น matrix ได้ดังนี้

$$X = \begin{bmatrix} 0 & 36.5 & 126 & 122 \\ 1 & 39.5 & 70 & 120 \\ 1 & 40.1 & 84 & 259 \\ 0 & 36.5 & 79 & 249 \end{bmatrix}$$

$$y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \end{bmatrix}$$

# Data

**Data Stating**



**Data  
Requirement**

**More about  
Target**

# Data Requirement

- ข้อมูลต้องอยู่ในรูปแบบของตาราง
- ข้อมูลต้องเป็น numerical

เพศ	อุณหภูมิร่างกาย (°C)	น้ำตาลในเลือด (mg/dl)	ความดันโลหิต (mmHg)	โรค
0	36.6	126	122	0
1	39.5	70	120	1
1	40.1	84	95	1
0	36.5	79	151	2

# Data Requirement

- ตัวอย่างข้อมูลที่สามารถใช้งานได้เลย และยังไม่สามารถใช้งานได้

อายุ	เพศ	โรค
42	0	0
57	1	1
58	1	1
59	0	2

อายุ	เพศ	โรค
42	female	0
57	male	1
58	male	1
59	female	2



\*\*0=เบาหวาน, 1=โควิด, 2=หัวใจ

# Data Requirement

- เราสามารถแปลงได้โดยสามารถใช้ความรู้ในส่วนของ Data Preparation

อายุ	เพศ	โรค
42	female	0
57	male	1
58	male	1
59	female	2



อายุ	female	male	โรค
42	1	0	0
57	0	1	1
58	0	1	1
59	1	0	2

\*\*0=เบาหวาน, 1=โควิด, 2=หัวใจ

# Data Requirement



For more information



## Data Preparation

# Data

**Data Stating**



**Data  
Requirement**



**More about  
Target**



# More about Target

อายุ	female	male	โรค
42	1	0	เบาหวาน
57	0	1	โควิด
58	0	1	โควิด
59	1	0	หัวใจ



อายุ	female	male	โรค
42	1	0	0
57	0	1	1
58	0	1	1
59	1	2	2



อายุ	female	male	เบาหวาน	โควิด	หัวใจ
42	1	0	1	0	0
57	0	1	0	1	0
58	0	1	0	1	0
59	1	0	0	0	1



\*\*0=เบาหวาน, 1=โควิด, 2=หัวใจ

# More about Target

อายุ	female	male	โรค
42	1	0	เบาหวาน
57	0	1	โควิด
58	0	1	โควิด
59	1	0	หัวใจ



อายุ	female	male	โรค
42	1	0	0
57	0	1	1
58	0	1	1
59	1	2	2



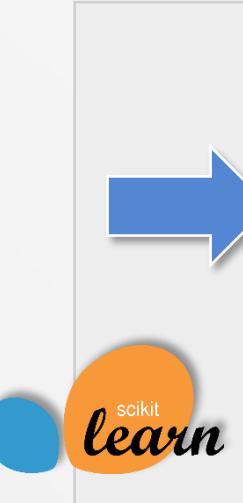
อายุ	female	male	เบาหวาน	โควิด	หัวใจ
42	1	0	1	0	0
57	0	1	0	1	0
58	0	1	0	1	0
59	1	0	0	0	1



\*\*0=เบาหวาน, 1=โควิด, 2=หัวใจ

# More about Target

อายุ	female	male	โรค
42	1	0	เบาหวาน
57	0	1	โควิด
58	0	1	โควิด
59	1	0	หัวใจ



อายุ	female	male	เบาหวาน	โควิด	หัวใจ
42	1	0	1	0	0
57	0	1	0	1	0
58	0	1	0	1	0
59	1	0	0	0	1

Prediction

โรค
โควิด
เบาหวาน
โควิด
หัวใจ

Prediction

เบาหวาน	โควิด	หัวใจ
0	1	0
1	0	0
0	1	0
0	0	1

Model



# Data

**Data Stating**



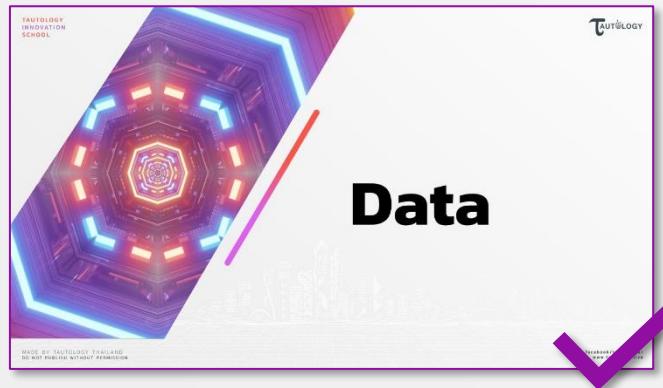
**Data  
Requirement**



**More about  
Target**



# Deep Learning for Multi-Class Classification





# Model

# Model

Assumption

Real Face of the  
Model

Cost Function and  
Cost Landscape

How to Create  
Model (Math)

How to Create  
Model (Code)

# Assumption

1. Nonlinear Relationship
2. Normality of Residuals
3. Homoscedasticity
4. No Missing Features
5. No Multicollinearity

# Assumption



For more information



Model Improvement  
In DL101

# Model

**Assumption**



Real Face of the  
Model

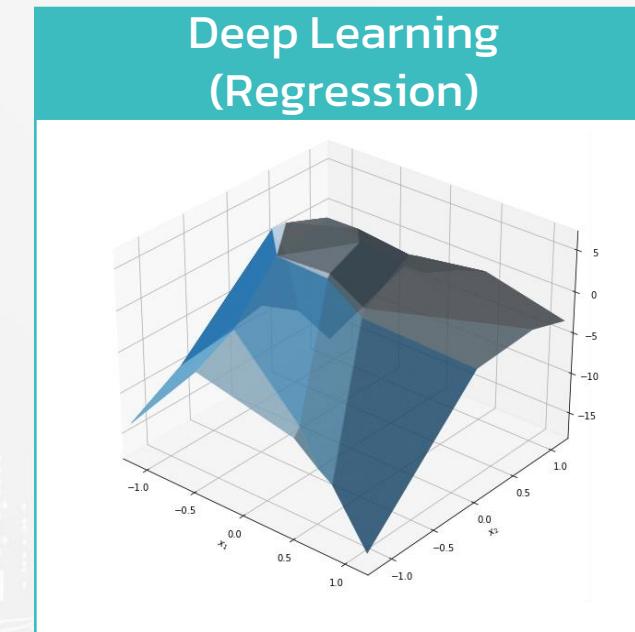
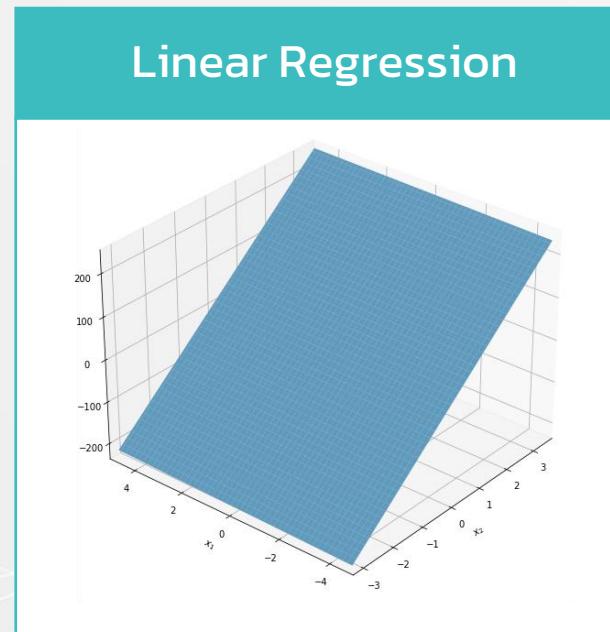
Cost Function and  
Cost Landscape

How to Create  
Model (Math)

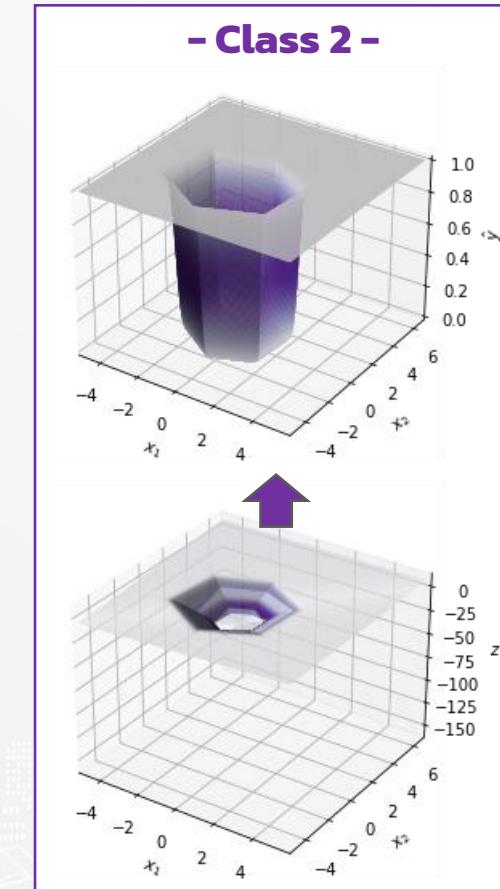
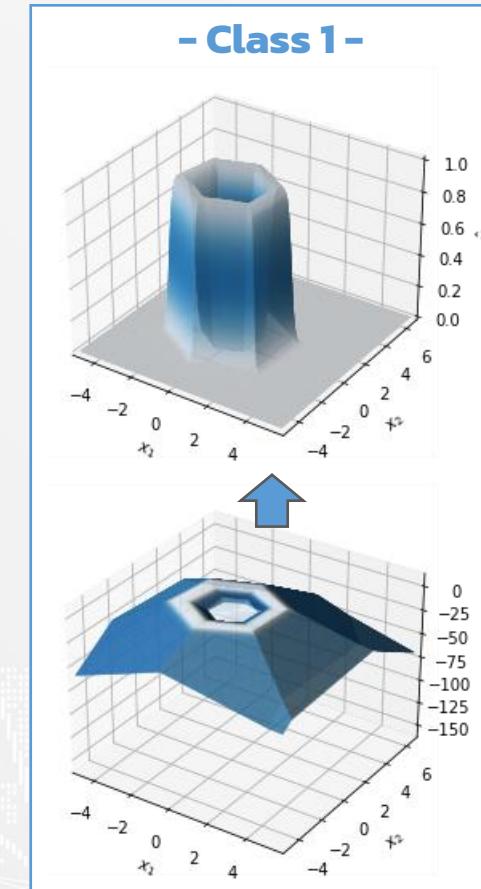
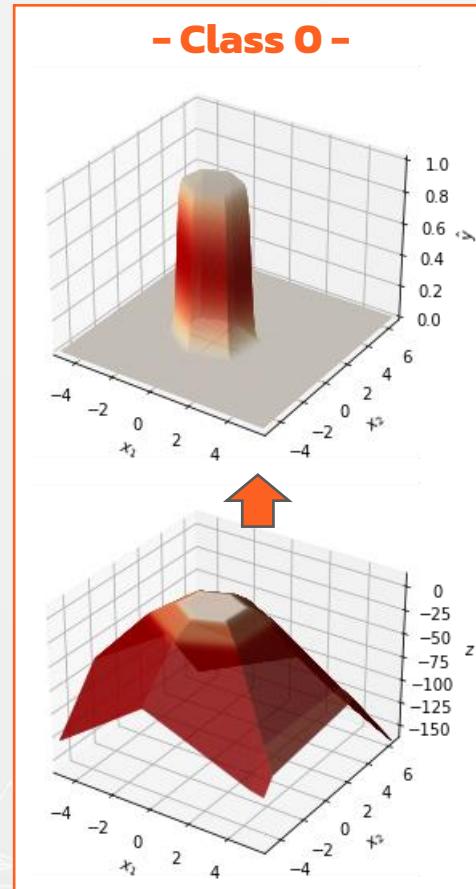
How to Create  
Model (Code)

# Real Face of the Model

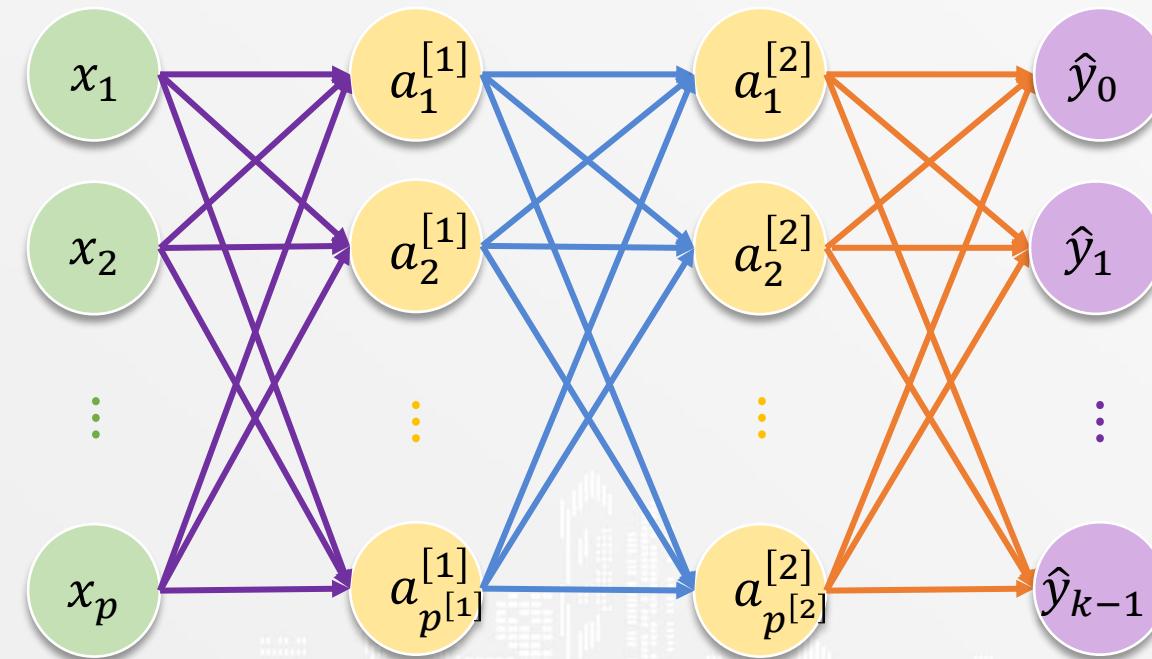
แนวคิดของ deep learning คือการนำ nonlinear function มาประกอบกันเพื่อประเมิน nonlinear function ที่ซับซ้อนได้



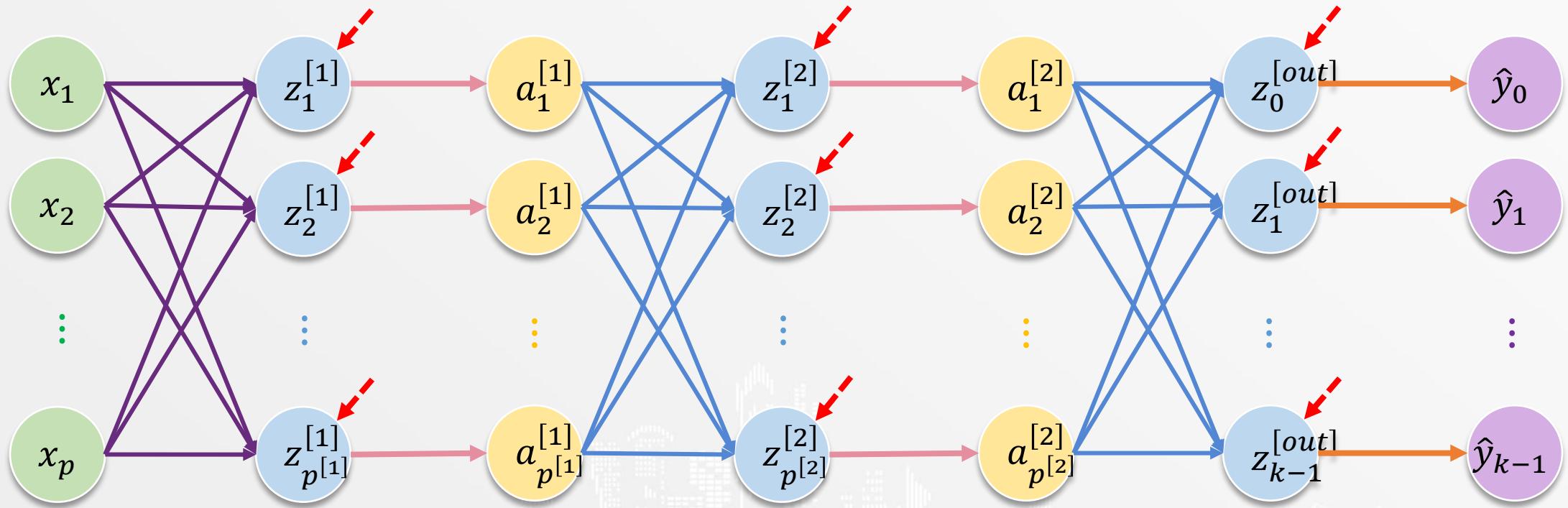
# Real Face of the Model



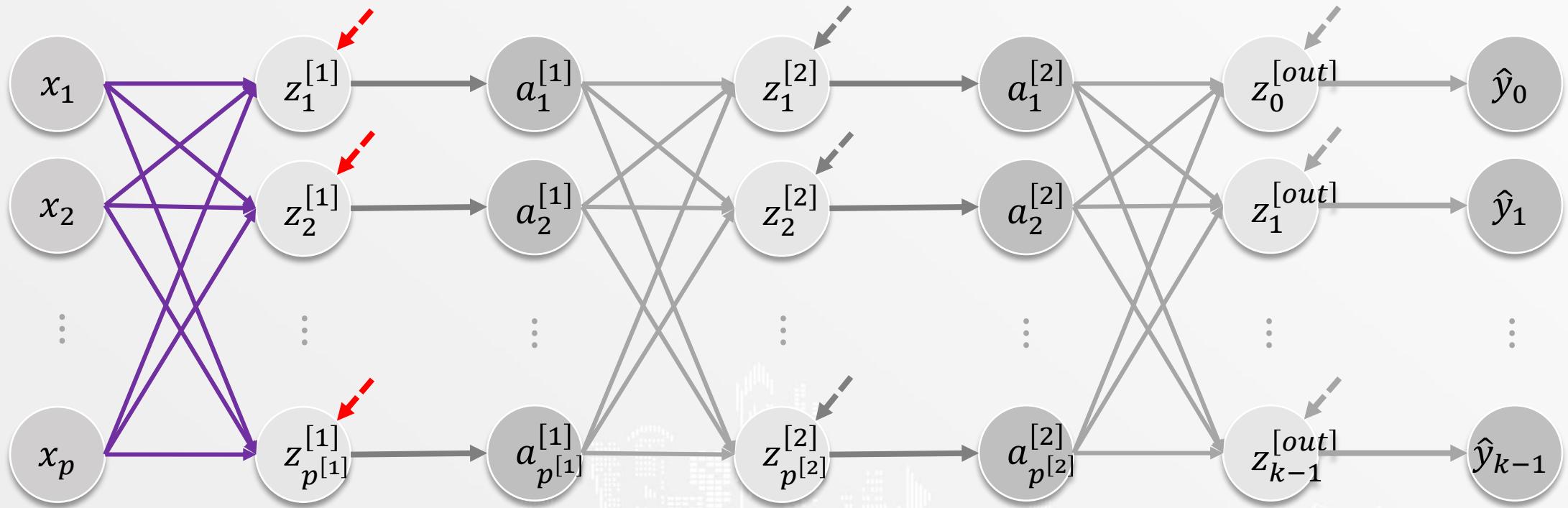
# Real Face of the Model



# Real Face of the Model



# Real Face of the Model

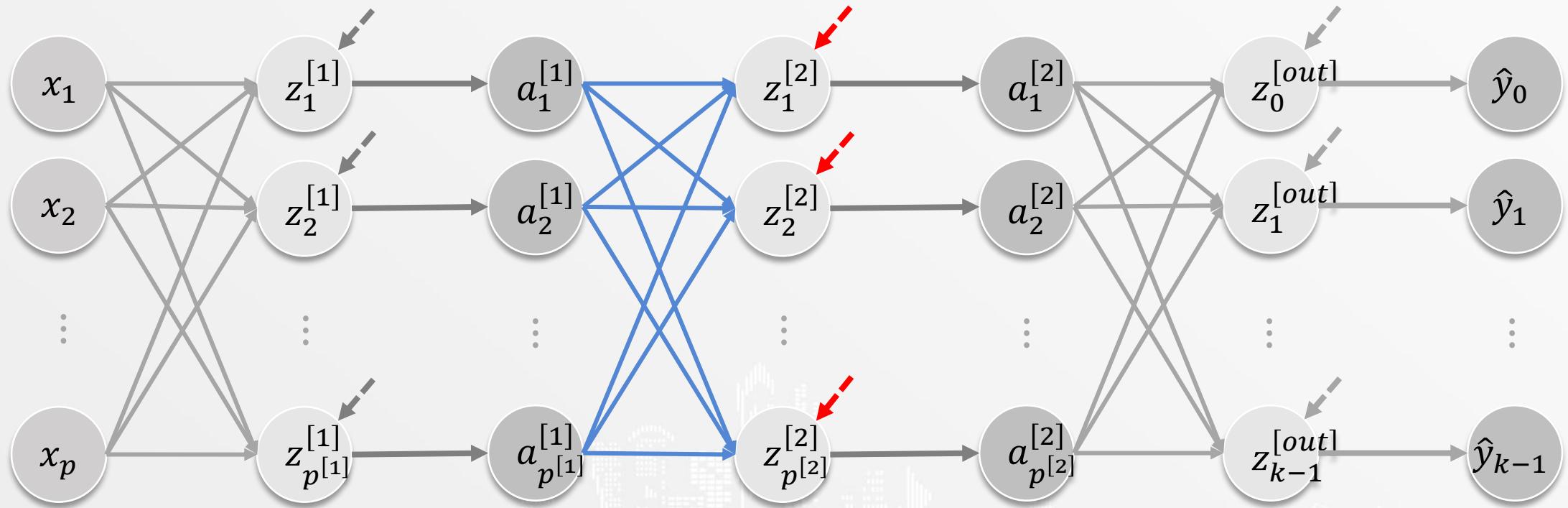


# Real Face of the Model

$$\mathbf{b}^{[1]} = \begin{bmatrix} b_1^{[1]} & b_2^{[1]} & \dots & b_{p^{[1]}}^{[1]} \end{bmatrix}$$

$$W^{[1]} = \begin{bmatrix} w_{1,1}^{[1]} & w_{1,2}^{[1]} & \dots & w_{1,p^{[1]}}^{[1]} \\ w_{2,1}^{[1]} & w_{2,2}^{[1]} & \dots & w_{2,p^{[1]}}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[1]} & w_{p,2}^{[1]} & \dots & w_{p,p^{[1]}}^{[1]} \end{bmatrix}$$

# Real Face of the Model

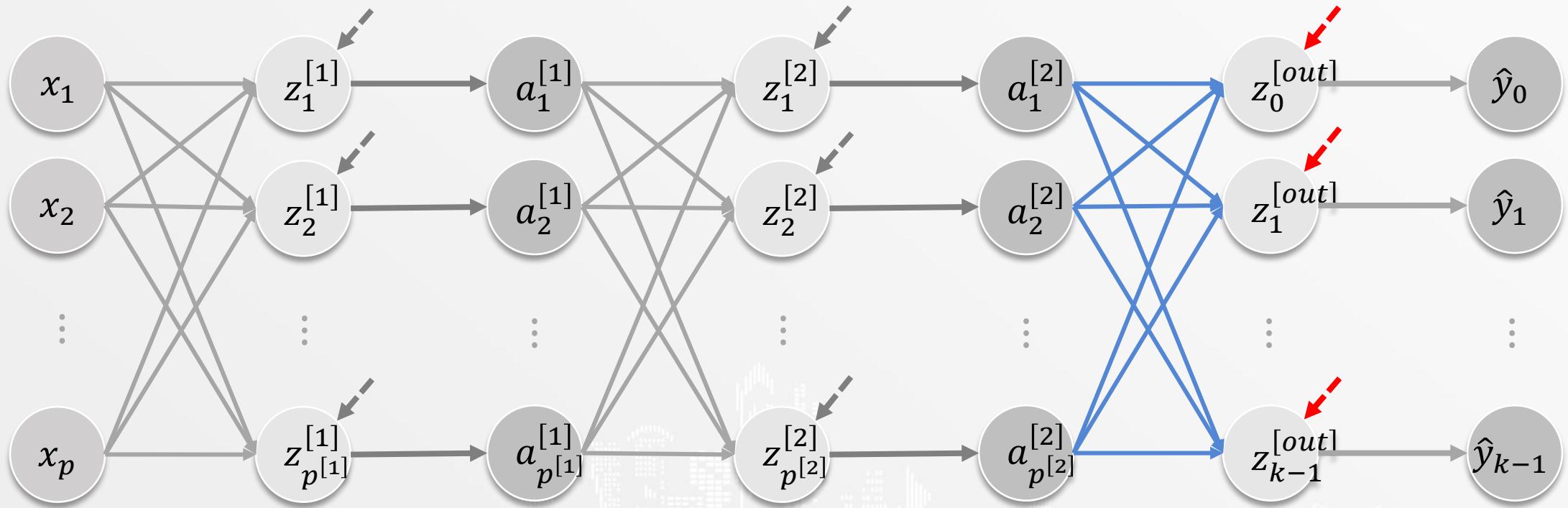


# Real Face of the Model

$$\mathbf{b}^{[2]} = \begin{bmatrix} b_1^{[2]} & b_2^{[2]} & \dots & b_{p^{[2]}}^{[2]} \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} w_{1,1}^{[2]} & w_{1,2}^{[2]} & \dots & w_{1,p^{[2]}}^{[2]} \\ w_{2,1}^{[2]} & w_{2,2}^{[2]} & \dots & w_{2,p^{[2]}}^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p,1}^{[2]} & w_{p,2}^{[2]} & \dots & w_{p,p^{[2]}}^{[2]} \end{bmatrix}$$

# Real Face of the Model



# Real Face of the Model

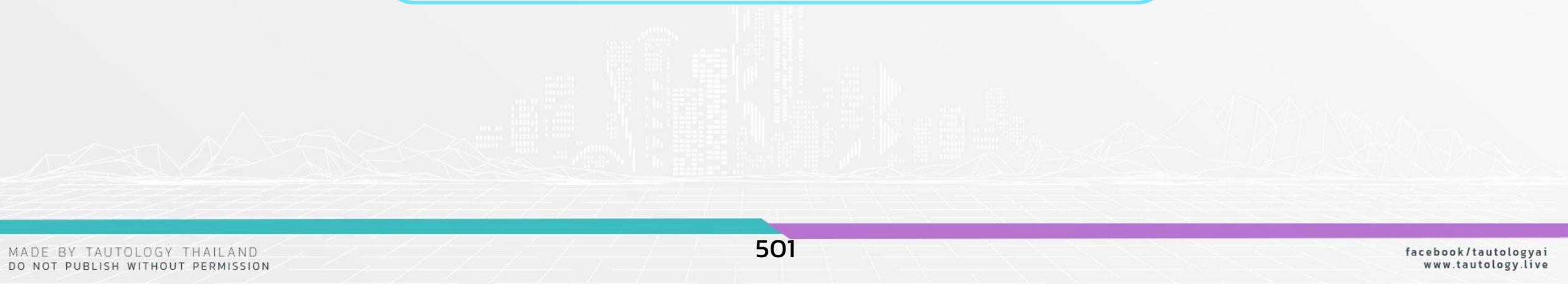
$$\mathbf{b}^{[out]} = [b_1^{[out]} \quad b_2^{[out]} \quad \dots \quad b_k^{[out]}]$$

$$W^{[out]} = \begin{bmatrix} w_{1,1}^{[out]} & w_{1,2}^{[out]} & \dots & w_{1,k}^{[out]} \\ w_{2,1}^{[out]} & w_{2,2}^{[out]} & \dots & w_{2,k}^{[out]} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p^{[2]},1}^{[out]} & w_{p^{[2]},2}^{[out]} & \dots & w_{p^{[2]},k}^{[out]} \end{bmatrix}$$

# Real Face of the Model

$\mathbf{b}^{[1]}, \mathbf{b}^{[2]}, \mathbf{b}^{[out]}$

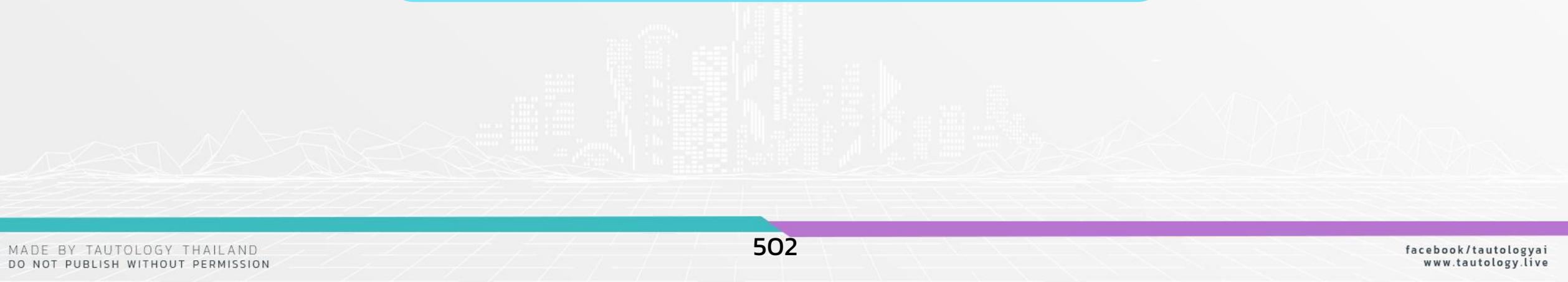
$W^{[1]}, W^{[2]}, W^{[out]}$



# Real Face of the Model

$\mathbf{b}^{[1]}, \mathbf{b}^{[2]}, \dots, \mathbf{b}^{[L]}, \mathbf{b}^{[out]}$

$W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$



# Real Face of the Model



“ เราต้องการหา  
 $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$   
ที่ทำให้ cost function ต่ำที่สุด ”

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



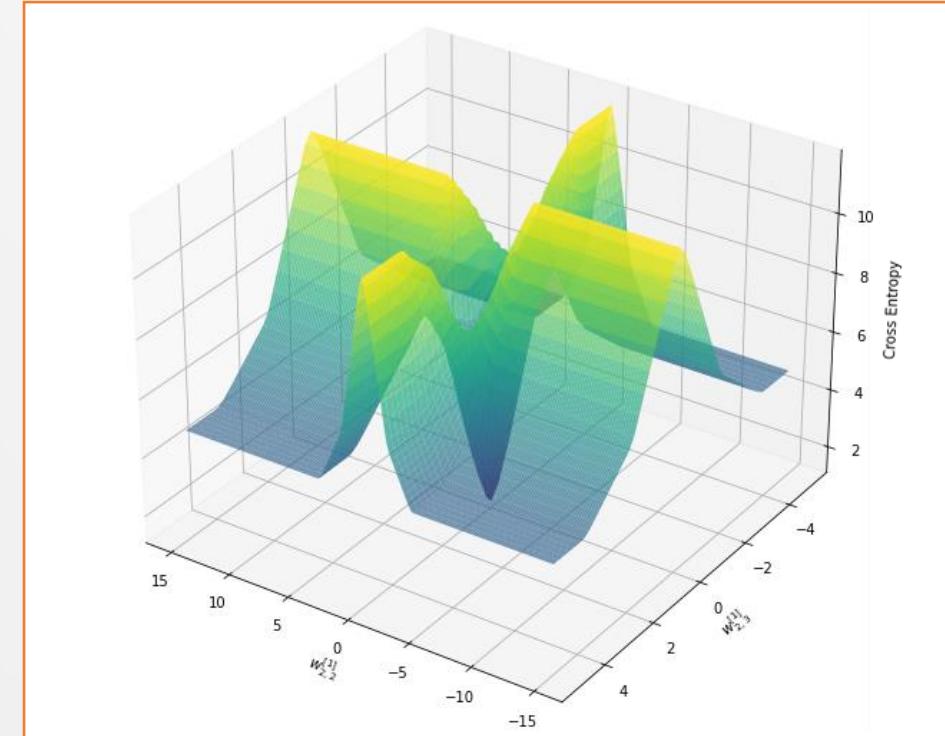
# Cost Function & Cost Landscape

**Cost function** กี่เราจะใช้ในการสร้าง model คือ

$$-\frac{1}{n} \sum_{i=1}^n \sum_{c=0}^{k-1} y_{i,c} \log(\hat{y}_{i,c})$$

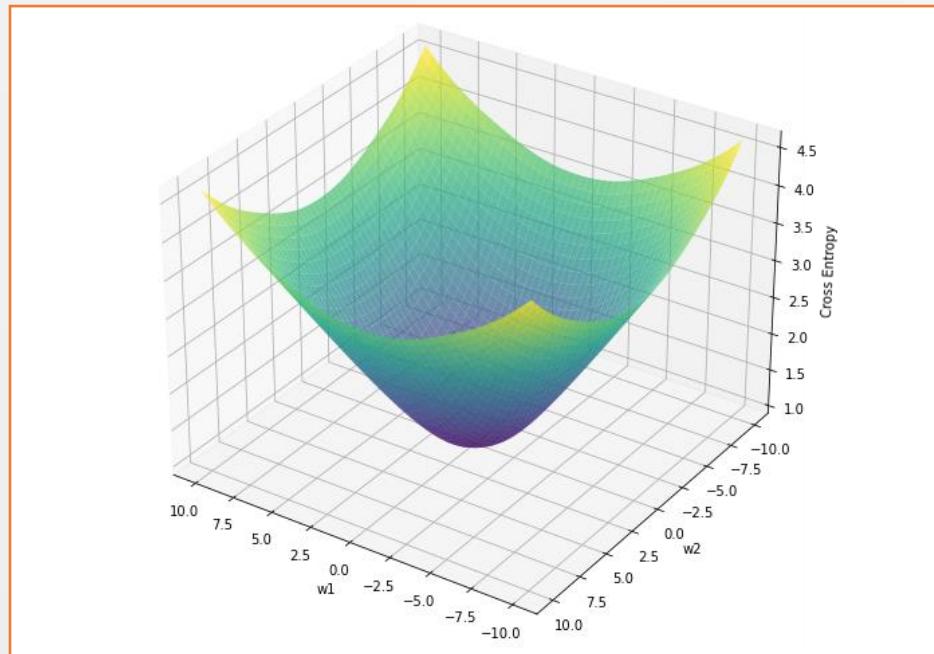
โดยสูตรข้างต้นมีชื่อว่า Cross Entropy

# Cost Function & Cost Landscape

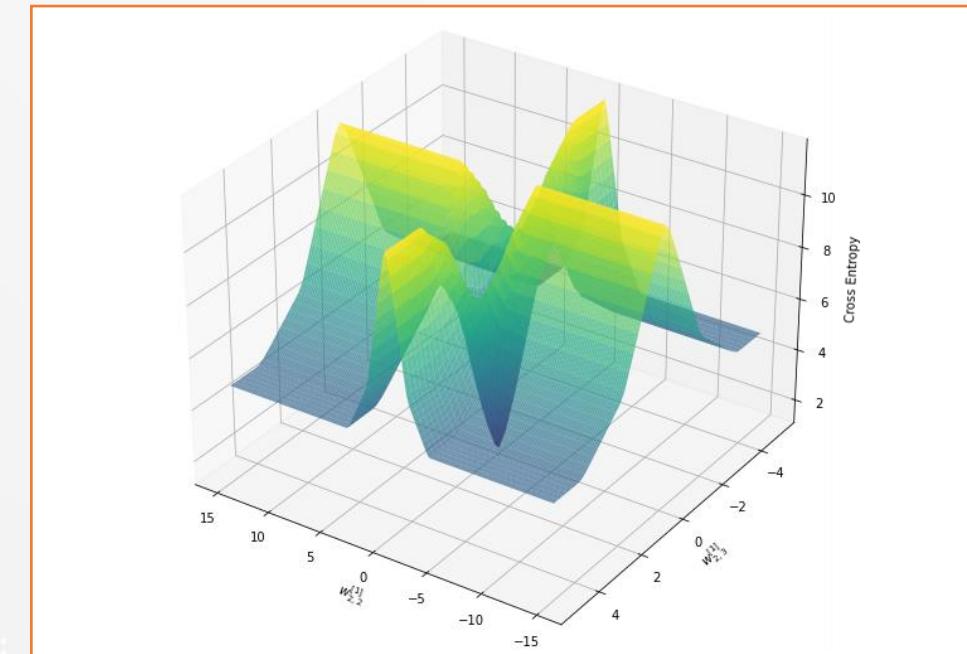


กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape



VS

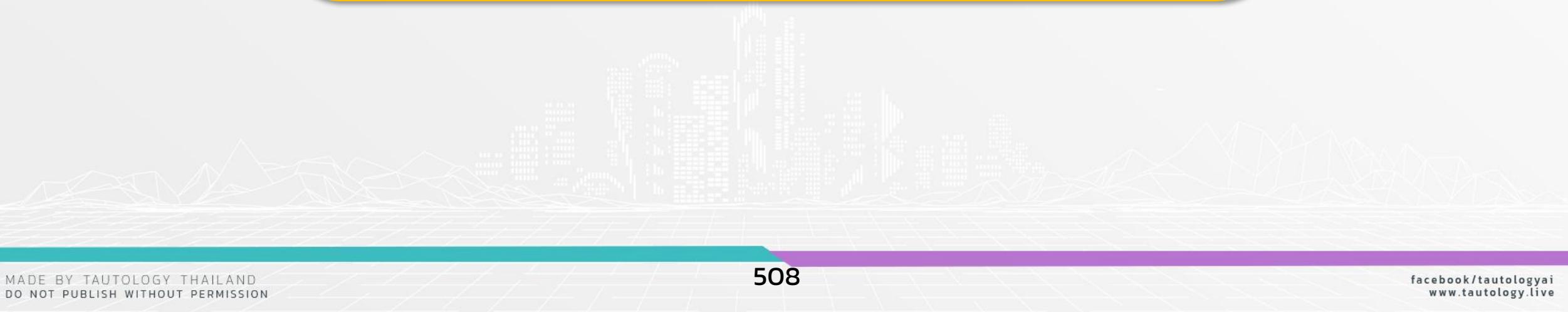


กราฟแสดง cost landscape ของ logistic regression โดยที่ cost function เป็น cross entropy

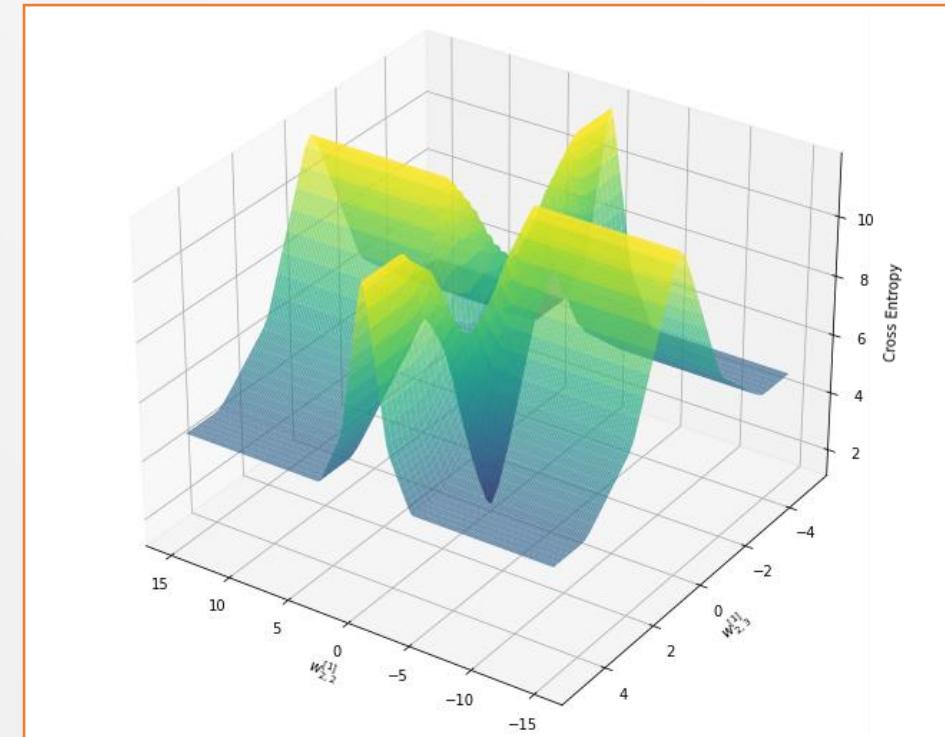
กราฟแสดง cost landscape ของ deep learning โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape

“**Cost landscape** ของ deep learning เป็น non-convex ซึ่งมีจุดต่ำสุดหลายตำแหน่ง”



# Cost Function & Cost Landscape



กราฟแสดง cost landscape ของ deep learning  
โดยที่ cost function เป็น cross entropy

# Cost Function & Cost Landscape



For more information



Improvement of  
Deep Learning

# Model

**Assumption**



**Real Face of the Model**



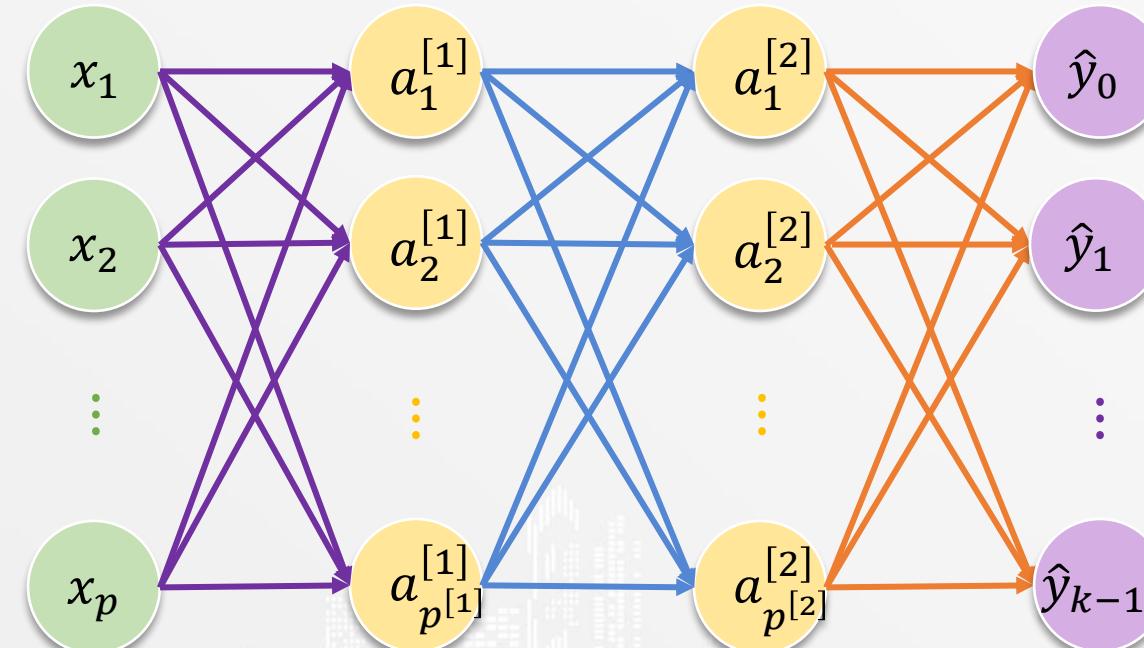
**Cost Function and Cost Landscape**



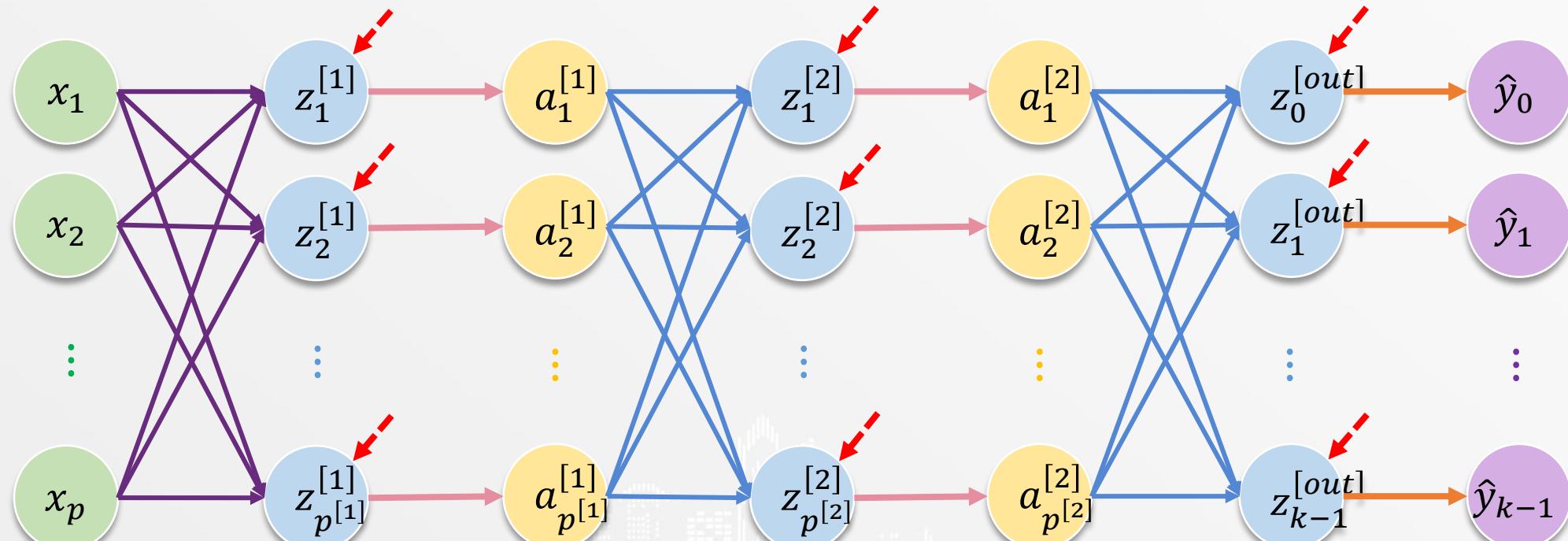
**How to Create Model (Math)**

**How to Create Model (Code)**

# How to Create Model (Math)



# How to Create Model (Math)



# How to Create Model (Math)



“ เราต้องการหา  
 $b^{[1]}, b^{[2]}, \dots, b^{[L]}, b^{[out]}$  และ $W^{[1]}, W^{[2]}, \dots, W^{[L]}, W^{[out]}$   
ที่ทำให้ cost function ต่ำที่สุด ”

# How to Create Model (Math)

**Cost function** กี่เราจะใช้ในการสร้าง model คือ

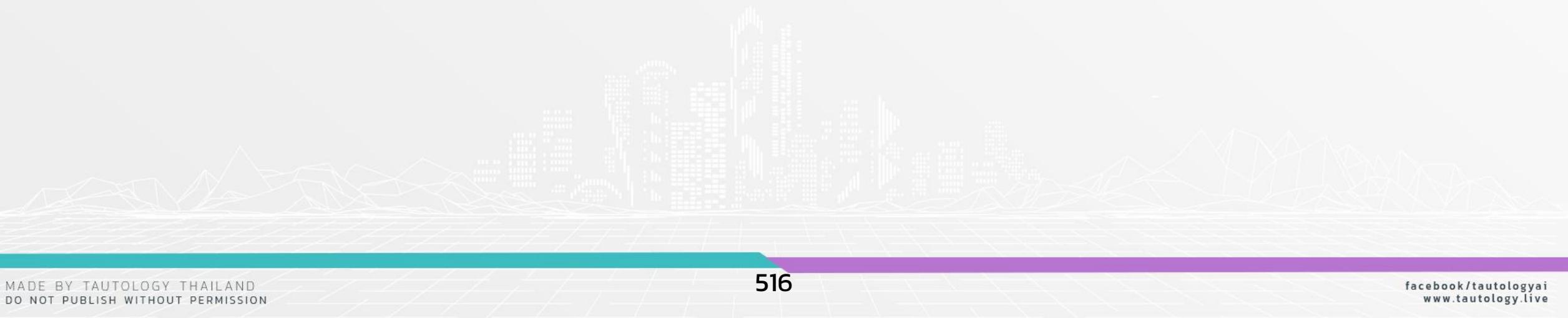
$$-\frac{1}{n} \sum_{i=1}^n \sum_{c=0}^{k-1} y_{i,c} \log(\hat{y}_{i,c})$$

โดยสูตรข้างต้นมีชื่อว่า Cross Entropy

# How to Create Model (Math)

เครื่องมือที่เราจะใช้ในการหาค่าตอบ คือ

**“Gradient Descent”**



# How to Create Model (Math)

## Equation of Gradient Descent

$$W = W - \alpha \nabla Cost$$

โดย ◇  $\alpha$  คือ ค่าที่ใช้ควบคุม step size ของ  $W$

# How to Create Model (Math)

$$\mathbf{b}^{[1]} = \mathbf{b}^{[1]} - \alpha \nabla Cost$$

$$\mathbf{b}^{[2]} = \mathbf{b}^{[2]} - \alpha \nabla Cost$$

⋮

$$\mathbf{b}^{[L]} = \mathbf{b}^{[L]} - \alpha \nabla Cost$$

$$\mathbf{b}^{[out]} = \mathbf{b}^{[out]} - \alpha \nabla Cost$$

# How to Create Model (Math)

$$W^{[1]} = W^{[1]} - \alpha \nabla Cost$$

$$W^{[2]} = W^{[2]} - \alpha \nabla Cost$$

⋮

$$W^{[L]} = W^{[L]} - \alpha \nabla Cost$$

$$W^{[out]} = W^{[out]} - \alpha \nabla Cost$$

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



# How to Create Model (Code)

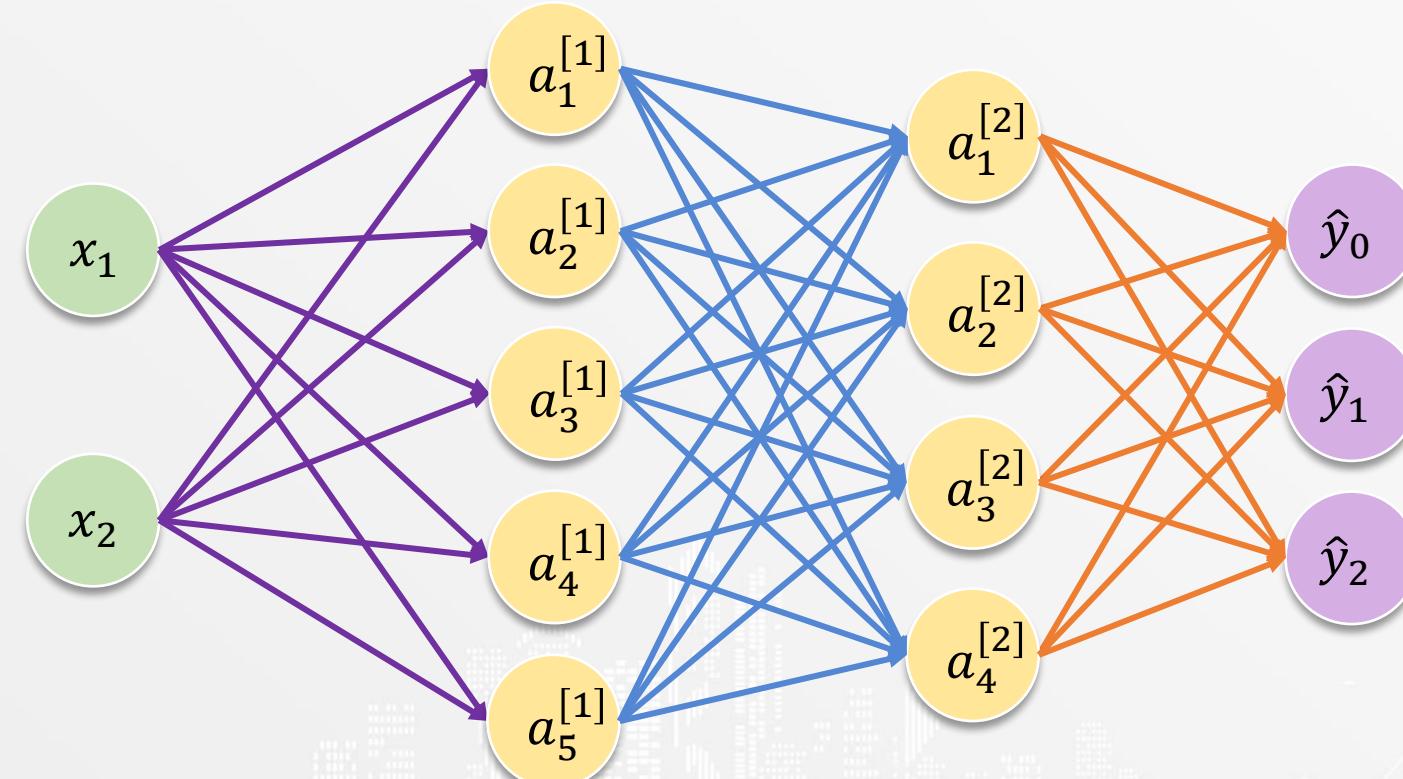
**ตัวอย่างการหา weight สำหรับ deep learning**

x <sub>1</sub>	x <sub>2</sub>	y
0	4	A
2	2	B
2	4	A
1	4	A
-1	3	A
0	2	A
3	1	B
-3	1	C
3	3	B

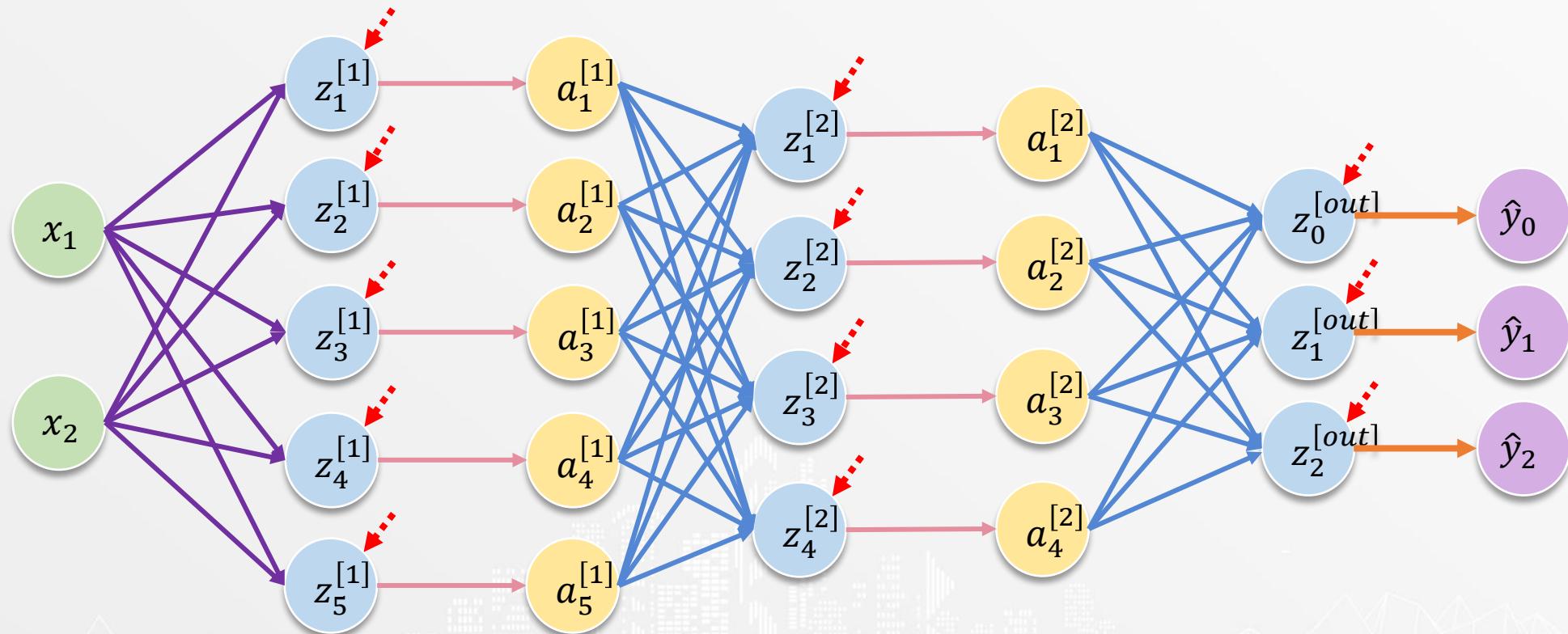
x <sub>1</sub>	x <sub>2</sub>	y
1	-2	C
1	-1	C
0	-1	C
3	2	B
-2	5	A
4	3	B
1	1	B
1	3	A
-1	-2	C

x <sub>1</sub>	x <sub>2</sub>	y
3	0	B
-1	-1	C
-2	-2	C
-1	-3	C
-1	4	A
0	-2	C
0	3	A
3	-1	B
4	-1	B

# How to Create Model (Code)



# How to Create Model (Code)



# How to Create Model (Code)

$$X = \begin{bmatrix} 0 & 4 \\ 2 & 2 \\ \vdots & \vdots \\ 0 & -2 \\ 0 & 3 \\ 3 & -1 \\ 4 & -1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} A \\ B \\ \vdots \\ C \\ A \\ B \\ B \end{bmatrix}$$

# How to Create Model (Code)

```
1 clf = MLPClassifier(  
2     hidden_layer_sizes=(5, 4),  
3     activation='relu',  
4     solver='sgd',  
5     alpha=0,  
6     learning_rate_init=1,  
7     max_iter=1000,  
8     momentum=0  
9 )  
10  
11 clf.fit(X, y)
```

# How to Create Model (Code)



Code for this section



Open File

**Multiclass Classification/ Model Creation (sklearn).ipynb**

# How to Create Model (Code)

$$\mathbf{b}^{[1]} = [0.936 \quad 0.391 \quad 0.403 \quad 0.639 \quad 0.774]$$

$$W^{[1]} = \begin{bmatrix} -0.803 & -0.146 & 0.816 & -0.835 & 1.412 \\ -1.7 & 1.283 & -0.367 & 0.832 & -0.348 \end{bmatrix}$$

# How to Create Model (Code)

$$\mathbf{b}^{[2]} = [-0.605 \quad 0.273 \quad 0.806 \quad -0.255]$$

$$W^{[2]} = \begin{bmatrix} 0.47 & 0.536 & 1.985 & 0.294 \\ 1.079 & -0.464 & -0.686 & -0.341 \\ -0.224 & 0.621 & 0.189 & 0.232 \\ 1.022 & 0.035 & 0.489 & 0.566 \\ -0.869 & 1.392 & -0.12 & -0.403 \end{bmatrix}$$

# How to Create Model (Code)

$$\mathbf{b}^{[out]} = [0.73 \quad 0.157 \quad -0.855]$$

$$W^{[out]} = \begin{bmatrix} 0.365 & -1.025 & -0.716 \\ -0.612 & 1.376 & 0.313 \\ -0.906 & -0.577 & 2.061 \\ 0.408 & 0.697 & 0.905 \end{bmatrix}$$

# How to Create Model (Code)

$$a_1^{[1]} = \text{ReLU}(0.936 - 0.803x_1 - 1.7x_2)$$

$$a_2^{[1]} = \text{ReLU}(0.391 - 0.146x_1 + 1.283x_2)$$

$$a_3^{[1]} = \text{ReLU}(0.403 + 0.816x_1 - 0.367x_2)$$

$$a_4^{[1]} = \text{ReLU}(0.639 - 0.835x_1 + 0.832x_2)$$

$$a_5^{[1]} = \text{ReLU}(0.774 + 1.412x_1 - 0.348x_2)$$

# How to Create Model (Code)

$$a_1^{[2]} = \text{ReLU}(-0.605 + 0.47a_1^{[1]} + 1.079a_2^{[1]} - 0.224a_3^{[1]} + 1.022a_4^{[1]} - 0.869a_5^{[1]})$$

$$a_2^{[2]} = \text{ReLU}(0.273 - 0.585a_1^{[1]} - 0.464a_2^{[1]} + 0.621a_3^{[1]} + 0.035a_4^{[1]} + 1.392a_5^{[1]})$$

$$a_3^{[2]} = \text{ReLU}(0.806 + 1.35a_1^{[1]} - 0.686a_2^{[1]} + 0.189a_3^{[1]} + 0.489a_4^{[1]} - 0.12a_5^{[1]})$$

$$a_4^{[2]} = \text{ReLU}(-0.255 - 0.294a_1^{[1]} - 0.314a_2^{[1]} + 0.232a_3^{[1]} + 0.566a_4^{[1]} - 0.403a_5^{[1]})$$

# How to Create Model (Code)

$$z_0^{[out]} = 0.73 + 0.365a_1^{[3]} - 0.612a_2^{[3]} - 0.906a_3^{[3]} + 0.408a_4^{[3]}$$

$$z_1^{[out]} = 0.157 - 1.025a_1^{[3]} + 1.376a_2^{[3]} - 0.577a_3^{[3]} - 0.697a_4^{[3]}$$

$$z_2^{[out]} = -0.855 - 0.716a_1^{[3]} + 0.313a_2^{[3]} + 2.061a_3^{[3]} - 0.905a_4^{[3]}$$

$$\hat{y}_m = \frac{e^{z_m^{[out]}}}{\sum_{c=0}^2 e^{z_c^{[out]}}}$$

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**

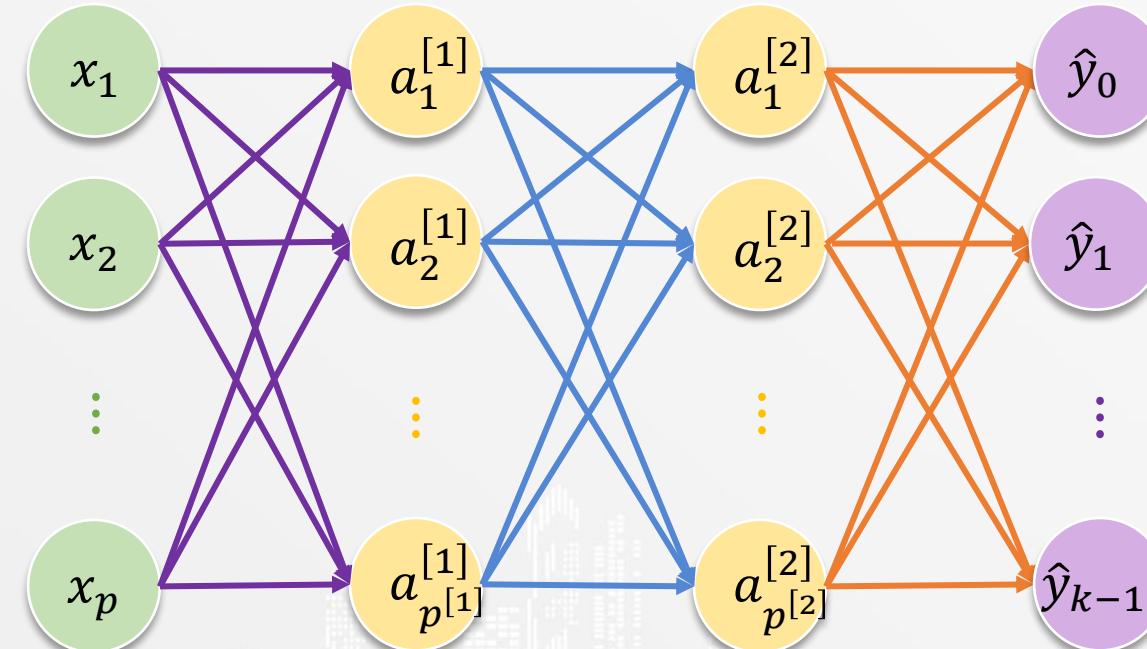


# Deep Learning for Multi-Class Classification

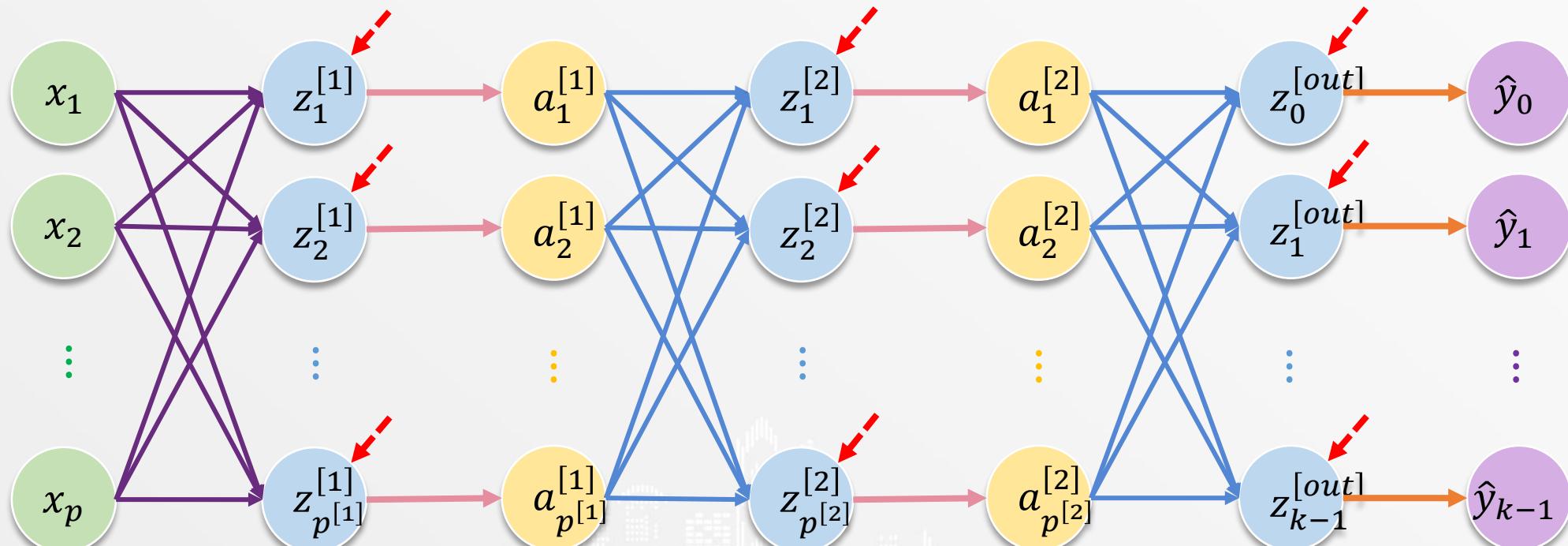


# Prediction

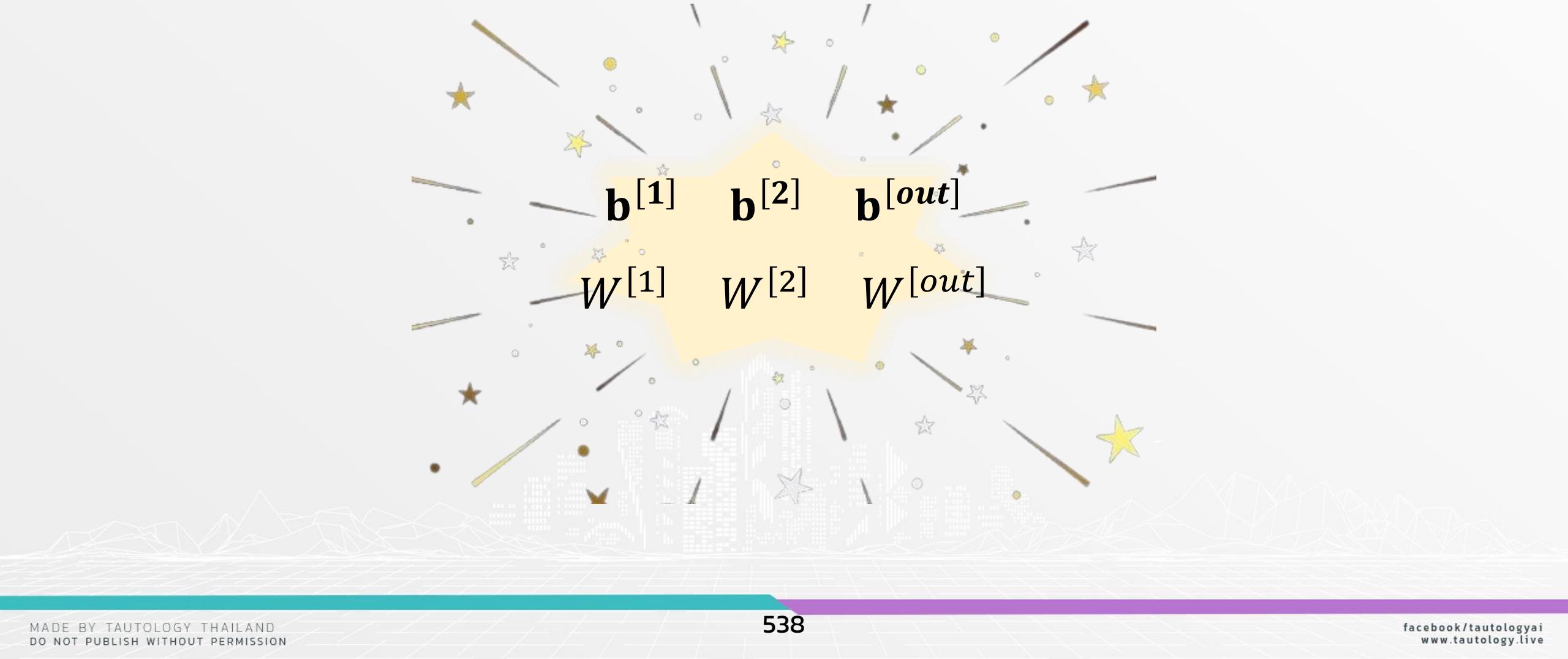
# Prediction



# Prediction



# Prediction

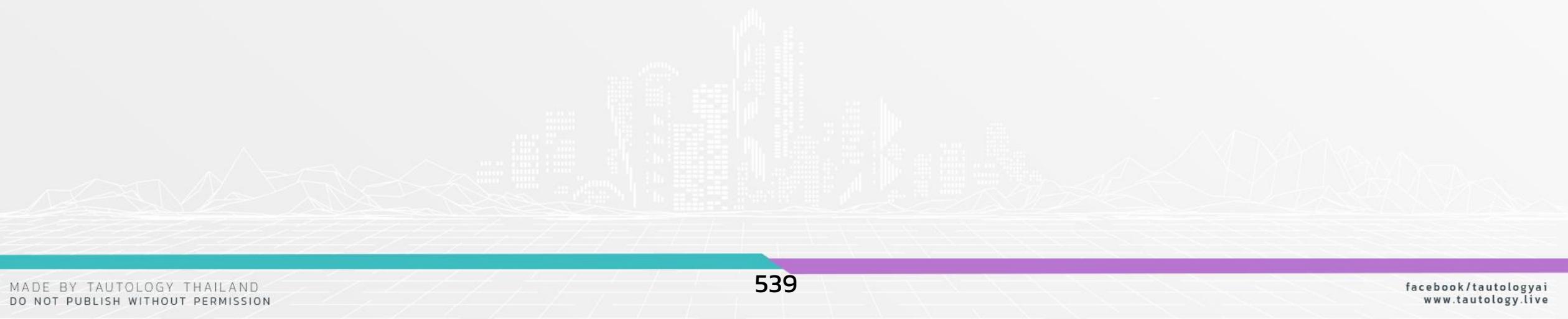


# Prediction

1-Sample

Multi-Sample

Code



# 1-Sample

ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
0	3



$\hat{y}$
?

# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$\mathbf{b}^{[1]} = [0.936 \quad 0.391 \quad 0.403 \quad 0.639 \quad 0.774]$$

$$W^{[1]} = \begin{bmatrix} -0.803 & -0.146 & 0.816 & -0.835 & 1.412 \\ -1.7 & 1.283 & -0.367 & 0.832 & -0.348 \end{bmatrix}$$

# 1-Sample

- สมมติว่า weight ของปัจจัยนี้กี่เราหมายได้คือ

$$\mathbf{b}^{[2]} = [-0.605 \quad 0.273 \quad 0.806 \quad -0.255]$$

$$W^{[2]} = \begin{bmatrix} 0.47 & 0.536 & 1.985 & 0.294 \\ 1.079 & -0.464 & -0.686 & -0.341 \\ -0.224 & 0.621 & 0.189 & 0.232 \\ 1.022 & 0.035 & 0.489 & 0.566 \\ -0.869 & 1.392 & -0.12 & -0.403 \end{bmatrix}$$

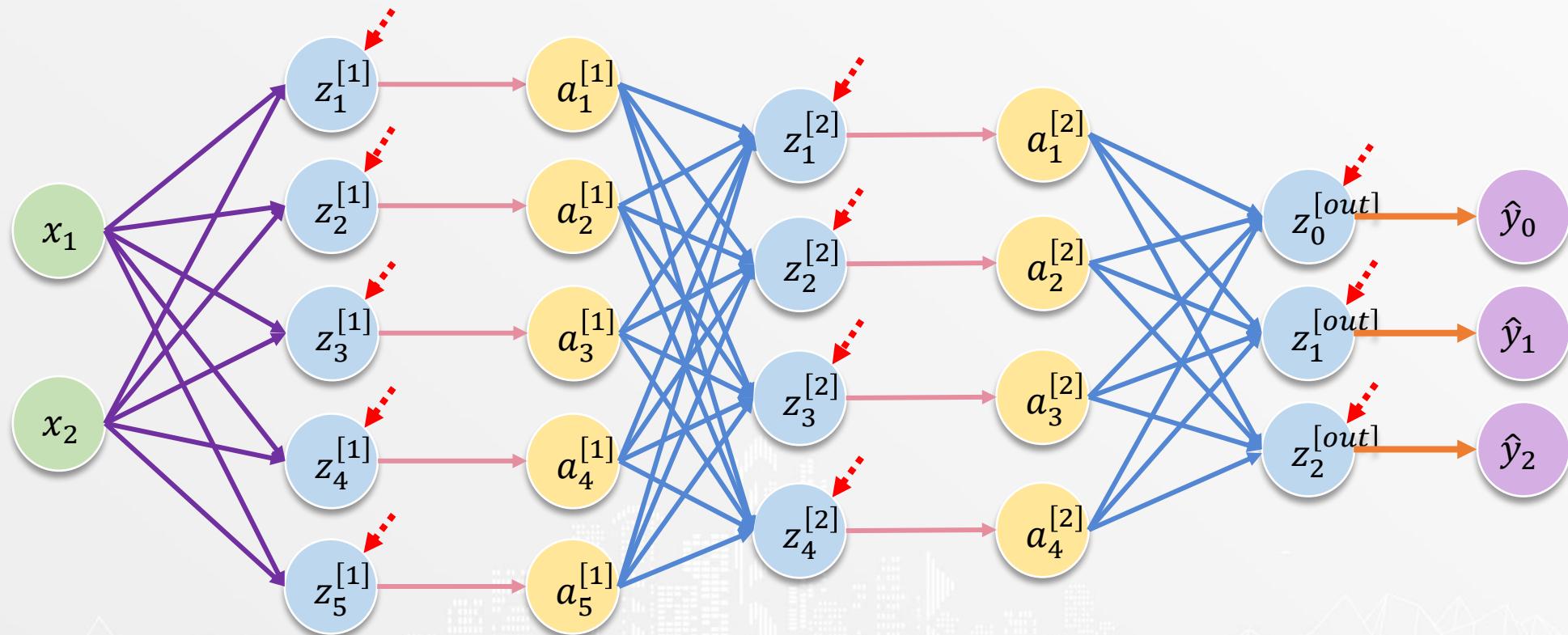
# 1-Sample

- สมมติว่า weight ของปัญหานี้กี่เราหมายได้คือ

$$\mathbf{b}^{[out]} = [0.73 \quad 0.157 \quad -0.855]$$

$$W^{[out]} = \begin{bmatrix} 0.365 & -1.025 & -0.716 \\ -0.612 & 1.376 & 0.313 \\ -0.906 & -0.577 & 2.061 \\ 0.408 & 0.697 & 0.905 \end{bmatrix}$$

# 1-Sample



# 1-Sample

$$a_1^{[1]} = \text{ReLU}(0.936 - 0.803x_1 - 1.7x_2) = 2.73$$

$$a_2^{[1]} = \text{ReLU}(0.391 - 0.146x_1 + 1.283x_2) = 0$$

$$a_3^{[1]} = \text{ReLU}(0.403 + 0.816x_1 - 0.367x_2) = 2.769$$

$$a_4^{[1]} = \text{ReLU}(0.639 - 0.835x_1 + 0.832x_2) = 0$$

$$a_5^{[1]} = \text{ReLU}(0.774 + 1.412x_1 - 0.348x_2) = 4.294$$

# 1-Sample

$$a_1^{[2]} = \text{ReLU}(-0.605 + 0.47a_1^{[1]} + 1.079a_2^{[1]} - 0.224a_3^{[1]} + 1.022a_4^{[1]} - 0.869a_5^{[1]}) = 0$$

$$a_2^{[2]} = \text{ReLU}(0.273 - 0.585a_1^{[1]} - 0.464a_2^{[1]} + 0.621a_3^{[1]} + 0.035a_4^{[1]} + 1.392a_5^{[1]}) = 9.433$$

$$a_3^{[2]} = \text{ReLU}(0.806 + 1.35a_1^{[1]} - 0.686a_2^{[1]} + 0.189a_3^{[1]} + 0.489a_4^{[1]} - 0.12a_5^{[1]}) = 6.233$$

$$a_4^{[2]} = \text{ReLU}(-0.255 - 0.294a_1^{[1]} - 0.314a_2^{[1]} + 0.232a_3^{[1]} + 0.566a_4^{[1]} - 0.403a_5^{[1]}) = 0$$

# 1-Sample

$$z_0^{[out]} = 0.73 + 0.365a_1^{[3]} - 0.612a_2^{[3]} - 0.906a_3^{[3]} + 0.408a_4^{[3]} = -10.69$$

$$z_1^{[out]} = 0.157 - 1.025a_1^{[3]} + 1.376a_2^{[3]} - 0.577a_3^{[3]} - 0.697a_4^{[3]} = 9.54$$

$$z_2^{[out]} = -0.855 - 0.716a_1^{[3]} + 0.313a_2^{[3]} + 2.061a_3^{[3]} - 0.905a_4^{[3]} = 14.944$$

# 1-Sample

$$\hat{\mathbf{y}} = \begin{bmatrix} e^{z_0^{[out]}} \\ \sum_{c=0}^2 e^{z_i^{[out]}} \end{bmatrix} \begin{bmatrix} e^{z_1^{[out]}} \\ \sum_{c=0}^2 e^{z_i^{[out]}} \end{bmatrix} \begin{bmatrix} e^{z_2^{[out]}} \\ \sum_{c=0}^2 e^{z_i^{[out]}} \end{bmatrix} = [0 \quad 0 \quad 1] \Rightarrow \textcircled{C}$$

# 1-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
0	3



$\hat{y}$
$c$

# Prediction

1-Sample



Multi-Sample



Code



# Multi-Sample

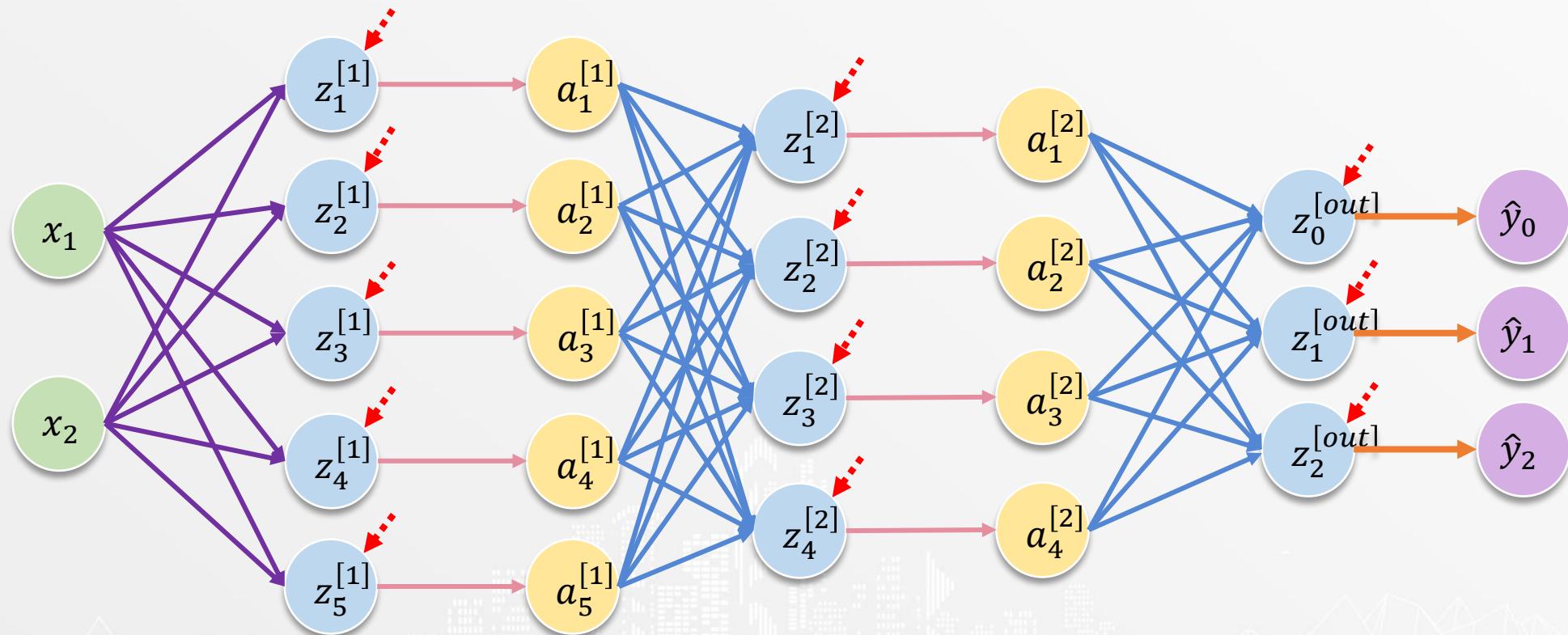
ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
2	-2
-2	4
2	1
1	5



$\hat{y}$
?
?
?
?

# Multi-Sample



# Multi-Sample

$$X \rightarrow Z^{[1]} \rightarrow A^{[1]} \rightarrow Z^{[2]} \rightarrow A^{[2]} \rightarrow Z^{[out]} \rightarrow \hat{Y}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + \mathbf{b}^{[1]}$$

$$\begin{aligned} Z^{[1]} &= \begin{bmatrix} 2 & -2 \\ -2 & 4 \\ 2 & 1 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} -0.803 & -0.146 & 0.816 & -0.835 & 1.412 \\ -1.7 & 1.283 & -0.367 & 0.832 & -0.348 \end{bmatrix} + [0.936 \quad 0.391 \quad 0.403 \quad 0.639 \quad 0.774] \\ &= \begin{bmatrix} 1.794 & -2.858 & 2.367 & -3.335 & 3.521 \\ -5.194 & 5.424 & -3.102 & 5 & -4.216 \\ -3.306 & 0.991 & 1.265 & -0.838 & 2.477 \\ -9.303 & 6.269 & -1.021 & 3.326 & -0.327 \end{bmatrix} + \begin{bmatrix} 0.936 & 0.391 & 0.403 & 0.639 & 0.774 \\ 0.936 & 0.391 & 0.403 & 0.639 & 0.774 \\ 0.936 & 0.391 & 0.403 & 0.639 & 0.774 \\ 0.936 & 0.391 & 0.403 & 0.639 & 0.774 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[1]} = XW^{[1]} + \mathbf{b}^{[1]}$$

$$Z^{[1]} = \begin{bmatrix} 2.73 & -2.467 & 2.77 & -2.696 & 4.295 \\ -4.258 & 4.258 & -2.699 & 5.639 & -3.442 \\ -2.37 & 1.382 & 1.668 & -0.199 & 3.251 \\ -8.367 & 6.66 & -0.618 & 3.965 & 0.447 \end{bmatrix}$$

# Multi-Sample

$$A^{[1]} = \text{ReLU}(Z^{[1]})$$

$$A^{[1]} = \text{ReLU} \begin{pmatrix} 2.73 & -2.467 & 2.77 & -2.696 & 4.295 \\ -4.258 & 4.258 & -2.699 & 5.639 & -3.442 \\ -2.37 & 1.382 & 1.668 & -0.199 & 3.251 \\ -8.367 & 6.66 & -0.618 & 3.965 & 0.447 \end{pmatrix}$$

$$= \begin{bmatrix} 2.73 & 0 & 2.77 & 0 & 4.295 \\ 0 & 4.258 & 0 & 5.639 & 0 \\ 0 & 1.382 & 1.668 & 0 & 3.251 \\ 0 & 6.66 & 0 & 3.965 & 0.447 \end{bmatrix}$$

# Multi-Sample

$$Z^{[2]} = A^{[2]}W^{[2]} + \mathbf{b}^{[2]}$$

$$\begin{aligned} Z^{[2]} &= \begin{bmatrix} 2.73 & 0 & 2.77 & 0 & 4.295 \\ 0 & 4.258 & 0 & 5.639 & 0 \\ 0 & 1.382 & 1.668 & 0 & 3.251 \\ 0 & 6.66 & 0 & 3.965 & 0.447 \end{bmatrix} \begin{bmatrix} 0.47 & 0.536 & 1.985 & 0.294 \\ 1.079 & -0.464 & -0.686 & -0.341 \\ -0.224 & 0.621 & 0.189 & 0.232 \\ 1.022 & 0.035 & 0.489 & 0.566 \\ -0.869 & 1.392 & -0.12 & -0.403 \end{bmatrix} \\ &\quad + [-0.605 \ 0.273 \ 0.806 \ -0.255] \\ &= \begin{bmatrix} -3.07 & 9.163 & 5.427 & -0.286 \\ 12.04 & -2.499 & -1.232 & 1.205 \\ -1.708 & 4.921 & -1.024 & -1.395 \\ 10.852 & -2.327 & -2.684 & -0.21 \end{bmatrix} + \begin{bmatrix} -0.605 & 0.273 & 0.806 & -0.255 \\ -0.605 & 0.273 & 0.806 & -0.255 \\ -0.605 & 0.273 & 0.806 & -0.255 \\ -0.605 & 0.273 & 0.806 & -0.255 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[2]} = A^{[2]}W^{[2]} + \mathbf{b}^{[2]}$$

$$Z^{[2]} = \begin{bmatrix} -3.675 & 9.436 & 6.233 & -0.541 \\ 11.435 & -2.226 & -0.426 & 0.95 \\ -2.313 & 5.194 & -0.218 & -1.65 \\ 10.247 & -2.054 & -1.878 & -0.465 \end{bmatrix}$$

# Multi-Sample

$$A^{[2]} = \text{ReLU}(Z^{[2]})$$

$$A^{[2]} = \text{ReLU} \left( \begin{bmatrix} -3.675 & 9.436 & 6.233 & -0.541 \\ 11.435 & -2.226 & -0.426 & 0.95 \\ -2.313 & 5.194 & -0.218 & -1.65 \\ 10.247 & -2.054 & -1.878 & -0.465 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 0 & 9.436 & 6.233 & 0 \\ 11.435 & 0 & 0 & 0.95 \\ 0 & 5.194 & 0 & 0 \\ 10.247 & 0 & 0 & 0 \end{bmatrix}$$

# Multi-Sample

$$Z^{[out]} = A^{[2]}W^{[out]} + \mathbf{b}^{[out]}$$

$$\begin{aligned} Z^{[out]} &= \begin{bmatrix} 0 & 9.436 & 6.233 & 0 \\ 11.435 & 0 & 0 & 0.95 \\ 0 & 5.194 & 0 & 0 \\ 10.247 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.365 & -1.025 & -0.716 \\ -0.612 & 1.376 & 0.313 \\ -0.906 & -0.577 & 2.061 \\ 0.408 & 0.697 & 0.905 \end{bmatrix} + [0.73 \quad 0.157 \quad -0.855] \\ &= \begin{bmatrix} -11.418 & 9.394 & 15.802 \\ 4.561 & -11.059 & -7.331 \\ -3.178 & 7.149 & 1.628 \\ 3.74 & -10.503 & -7.339 \end{bmatrix} + \begin{bmatrix} 0.73 & 0.157 & -0.855 \\ 0.73 & 0.157 & -0.855 \\ 0.73 & 0.157 & -0.855 \\ 0.73 & 0.157 & -0.855 \end{bmatrix} \end{aligned}$$

# Multi-Sample

$$Z^{[out]} = A^{[2]}W^{[out]} + \mathbf{b}^{[out]}$$

$$Z^{[out]} = \begin{bmatrix} -10.688 & 9.551 & 14.947 \\ 5.291 & -10.902 & -8.186 \\ -2.448 & 7.306 & 0.773 \\ 4.47 & -10.346 & -8.194 \end{bmatrix}$$

# Multi-Sample

$$\hat{Y} = \begin{bmatrix} \frac{e^{z_{1,1}^{[out]}}}{\sum_{c=0}^2 e^{z_{1,c}^{[out]}}} & \frac{e^{z_{1,2}^{[out]}}}{\sum_{c=0}^2 e^{z_{1,c}^{[out]}}} & \frac{e^{z_{1,3}^{[out]}}}{\sum_{c=0}^2 e^{z_{1,c}^{[out]}}} \\ \frac{e^{z_{2,1}^{[out]}}}{\sum_{c=0}^2 e^{z_{2,c}^{[out]}}} & \frac{e^{z_{2,2}^{[out]}}}{\sum_{c=0}^2 e^{z_{2,c}^{[out]}}} & \frac{e^{z_{2,3}^{[out]}}}{\sum_{c=0}^2 e^{z_{2,c}^{[out]}}} \\ \frac{e^{z_{3,1}^{[out]}}}{\sum_{c=0}^2 e^{z_{3,c}^{[out]}}} & \frac{e^{z_{3,2}^{[out]}}}{\sum_{c=0}^2 e^{z_{3,c}^{[out]}}} & \frac{e^{z_{3,3}^{[out]}}}{\sum_{c=0}^2 e^{z_{3,c}^{[out]}}} \\ \frac{e^{z_{4,1}^{[out]}}}{\sum_{c=0}^2 e^{z_{4,c}^{[out]}}} & \frac{e^{z_{4,2}^{[out]}}}{\sum_{c=0}^2 e^{z_{4,c}^{[out]}}} & \frac{e^{z_{4,3}^{[out]}}}{\sum_{c=0}^2 e^{z_{4,c}^{[out]}}} \end{bmatrix} = \begin{bmatrix} A & B & C \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} C \\ A \\ B \\ A \end{bmatrix}$$

# Multi-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

$x_1$	$x_2$
2	-2
-2	4
2	1
1	5



$\hat{y}$
C
A
B
A

# Prediction

1-Sample



Multi-Sample



Code



# Code

```
1 clf.predict(X)
```

```
array(['C', 'A', 'B', 'A'], dtype='<U1')
```

# Code



Code for this section



Open File

**Multiclass Classification/ Model Creation (sklearn).ipynb**

# Prediction

1-Sample



Multi-Sample



Code



# Deep Learning for Multi-Class Classification



# Deep Learning for Classification

**Deep Learning for  
Binary Classification**



**Deep Learning for  
Multi-Class Classification**



# Neural Network & Deep Learning

**Neural Network**



**Deep Learning**



**Deep Learning for  
Regression**



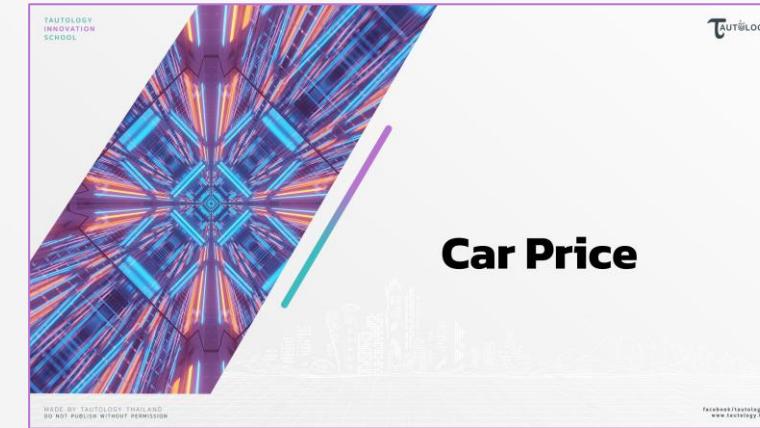
**Deep Learning for  
Classification**



**Workshop**



# Workshop



# AI in Skin Cancer

# Abstract

สร้างmodelเพื่อวินิจฉัยผู้ป่วยโรคมะเร็งผิวหนัง โดยพิจารณาจากภาพถ่ายผิวหนังของผู้ป่วย



# Why this project important?



- สามารถสร้างระบบสำหรับตรวจโรคมะเร็งผิวหนังที่ทำงานได้ตลอด 24 ชั่วโมง
- สามารถนำไปต่อยอดกับการวินิจฉัยโรคอื่น ๆ
- สามารถใช้เป็นพื้นฐานสำหรับการแพทย์ทางไกล

# Who this project is for?

- ✿ ผู้บริหารโรงพยาบาล
- ✿ บุคลากรทางการแพทย์
- ✿ นักวิเคราะห์ข้อมูล



# Skin Cancer Dataset



<https://www.kaggle.com/datasets/hasnainjaved/melanoma-skin-cancer-dataset-of-10000-images>

# Skin Cancer Dataset

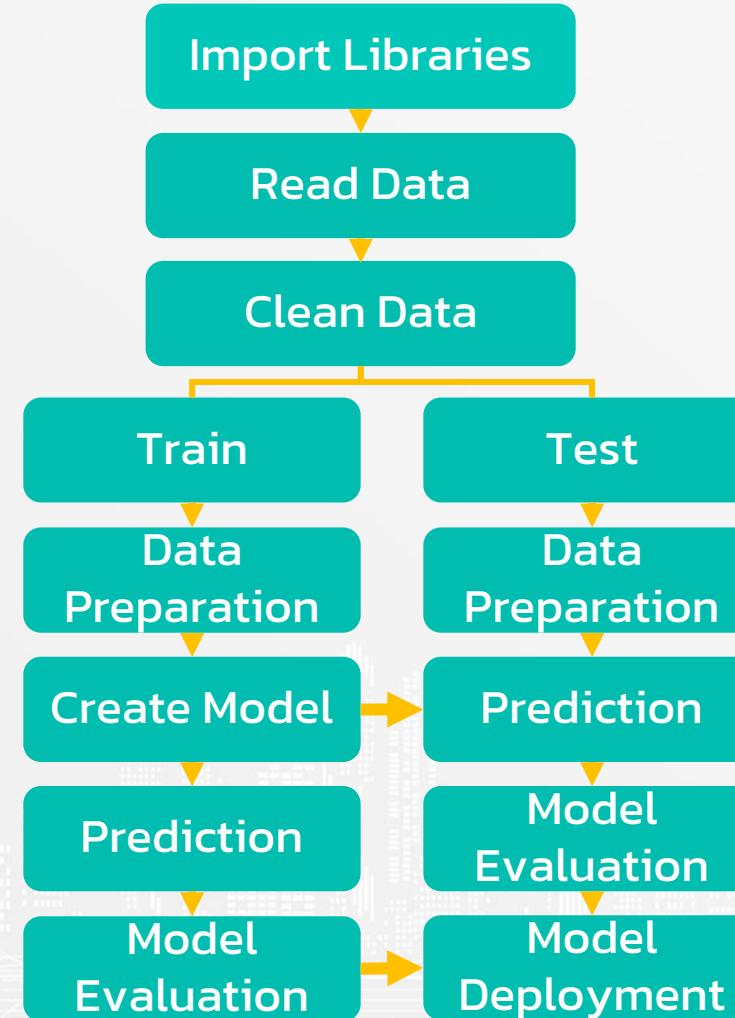
## Feature



## Target

- target : การเป็นโรคมะเร็งผิวหนัง (benign, malignant)

# What we learn from this project?

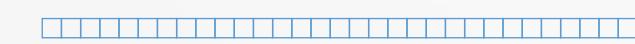
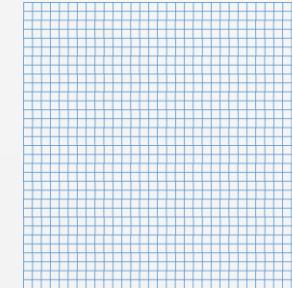
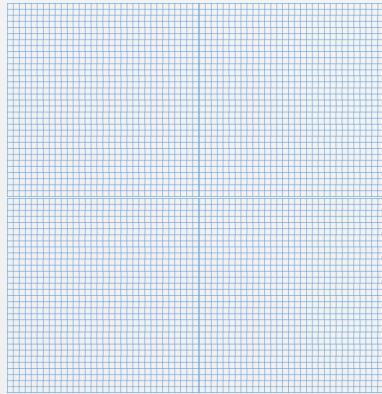


# Image to CSV

```
1 classes = ['benign', 'malignant']

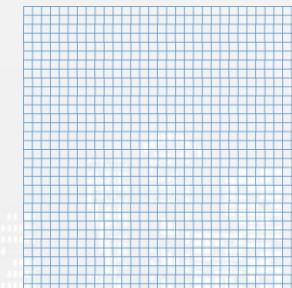
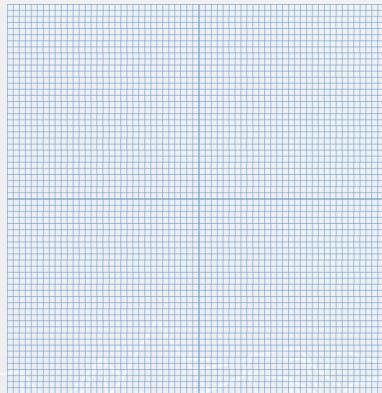
1 width = 64
2 height = 64
3
4 X = np.empty([0, width*height*3])
5 y = np.empty([0, 1])
6
7 for _class in tqdm(classes):
8     img_path = glob('dataset/' + _class + '*')
9     for path in tqdm(img_path):
10         img = Image.open(path)
11         img = img.resize([width, height])
12         img = np.array(img)
13         if img.shape[2] == 4:
14             img = cv2.cvtColor(img, cv2.COLOR_BGRA2BGR)
15         img = img.reshape(1, -1)
16         X = np.vstack([X, img])
17         y = np.vstack([y, _class])
```

# Image to CSV



benign

64x64



malignant

64x64

# Image to CSV

$x_1$	$x_2$	$x_3$	...	$x_{12287}$	$y$
198.0	159.0	160.0	...	117.0	benign
121.0	115.0	117.0	...	124.0	benign
186.0	136.0	125.0	...	110.0	benign
:	:	:	:	:	:
44.0	36.0	24.0	...	12.0	malignant

 $X$  $y$

# Image to CSV

```
1 columns = [f'pixel_{i}' for i in range(width*height*3)]
2
3 data = pd.DataFrame(X, columns=columns)
4 data['label'] = y
5
6 data.to_csv('skin_cancer_dataset.csv', index=False)
```

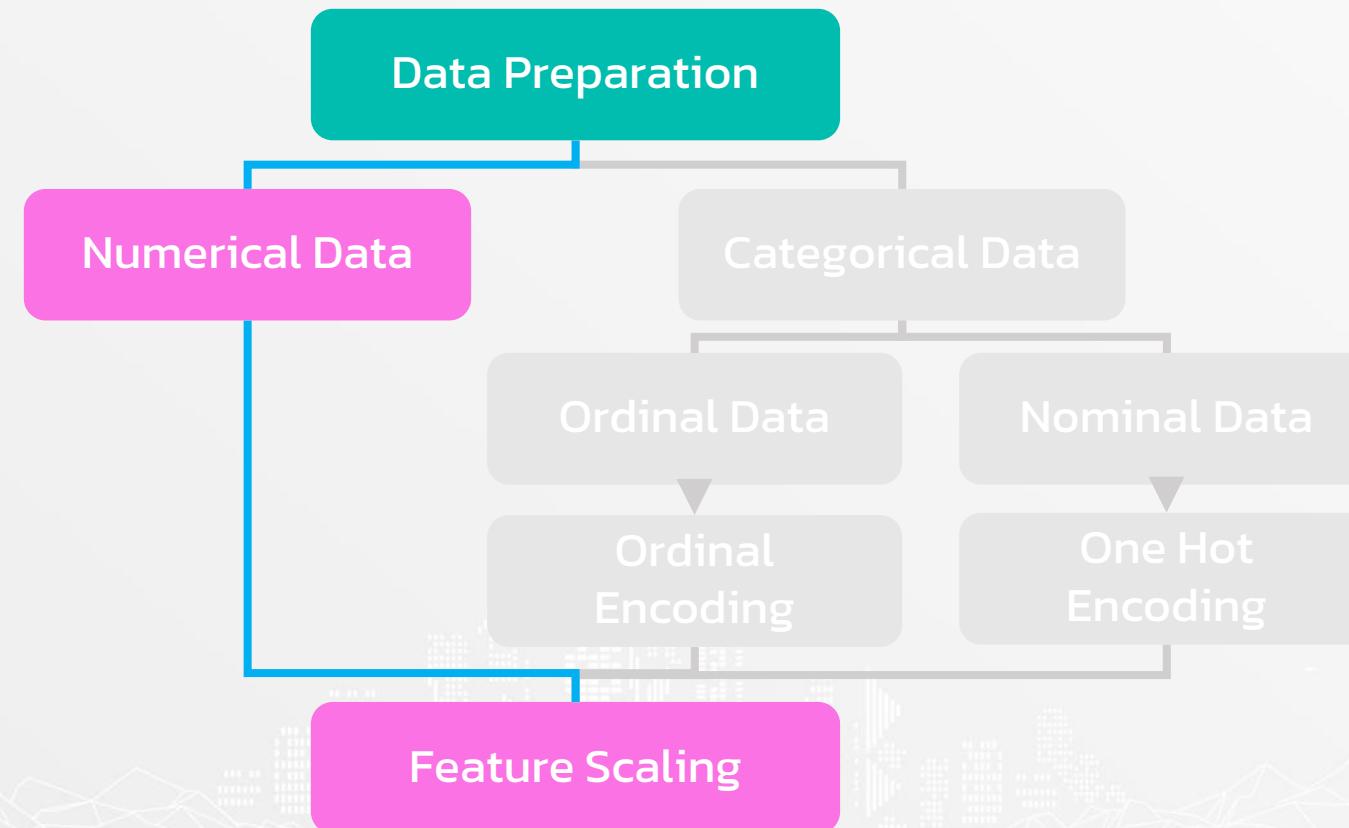
# Read Data

```
1 data = pd.read_csv('../image_to_csv/skin_cancer_dataset.csv')
2
3 data
```

	pixel_0	pixel_1	pixel_2	pixel_3	pixel_4	pixel_5	pixel_6	pixel_7	pixel_8	pixel_9	...	pixel_12286	pixel_12287	label
0	198.0	159.0	160.0	201.0	162.0	163.0	209.0	170.0	171.0	211.0	...	122.0	117.0	benign
1	121.0	115.0	117.0	134.0	128.0	130.0	136.0	131.0	133.0	145.0	...	104.0	124.0	benign
2	186.0	136.0	125.0	186.0	137.0	121.0	189.0	142.0	118.0	194.0	...	116.0	110.0	benign
3	178.0	132.0	136.0	180.0	140.0	137.0	185.0	140.0	131.0	180.0	...	94.0	89.0	benign
4	193.0	156.0	167.0	195.0	155.0	166.0	199.0	156.0	168.0	204.0	...	156.0	167.0	benign
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9600	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	malignant
9601	171.0	164.0	171.0	174.0	167.0	174.0	178.0	171.0	178.0	178.0	...	170.0	180.0	malignant
9602	148.0	131.0	124.0	149.0	131.0	123.0	146.0	125.0	114.0	153.0	...	57.0	54.0	malignant
9603	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	...	0.0	0.0	malignant
9604	44.0	36.0	24.0	56.0	45.0	33.0	69.0	56.0	42.0	81.0	...	14.0	12.0	malignant

9605 rows × 12289 columns

# Data Preparation



# File



## 03. SKIN CANCER



# File



## 03. SKIN CANCER/sklearn



**hepatitis\_c\_model.pickle**



**hepatitis\_c\_sklearn\_mc.ipynb**

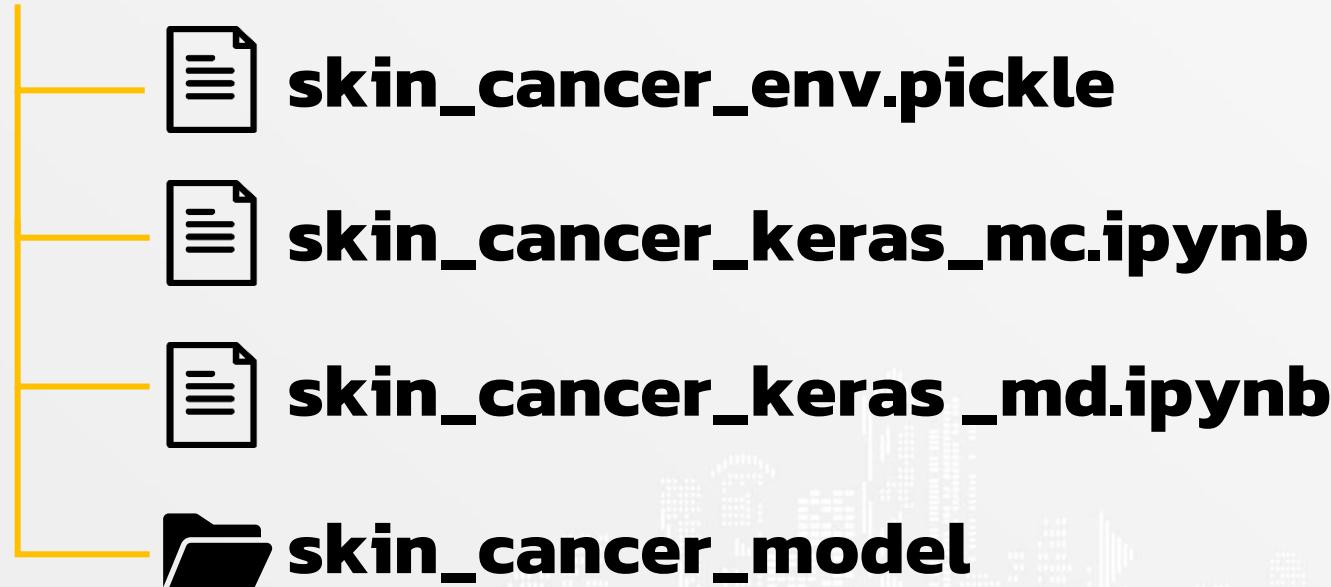


**hepatitis\_c\_sklearn\_md.ipynb**

# File



## 03. SKIN CANCER/keras



# File



## 03. SKIN CANCER/pytorch



**skin\_cancer\_env.pickle**

**skin\_cancer\_pytorch\_mc.ipynb**

**skin\_cancer\_pytorch\_md.ipynb**

**skin\_cancer\_model**

# Workshop



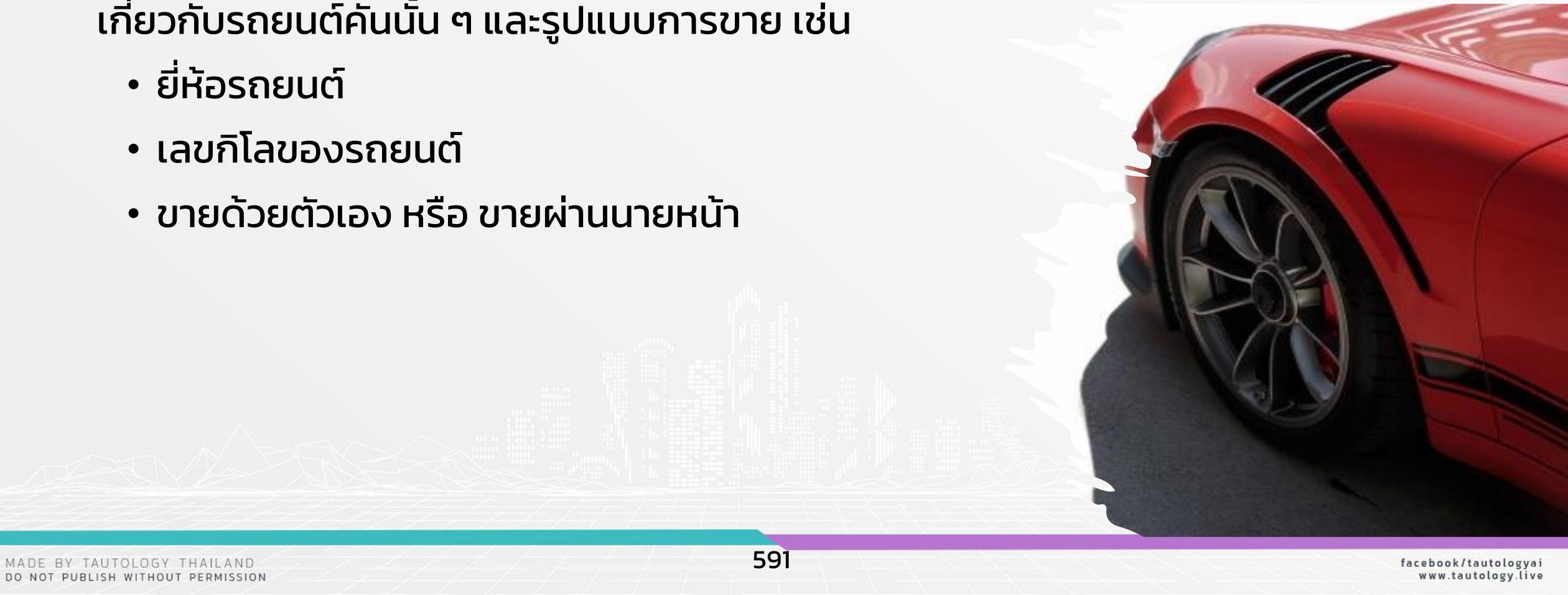


# Car Price

# Abstract

สร้าง model เพื่อประเมินราคาของรถยนต์มือสอง โดย feature กี่นำมาใช้ คือ ข้อมูลเกี่ยวกับรถยนต์คันนั้น ๆ และรูปแบบการขาย เช่น

- ยี่ห้อรถยนต์
- เลขกิโลของรถยนต์
- ขายด้วยตัวเอง หรือ ขายผ่านนายหน้า



# Why this project Important?

- สามารถประเมินราคารถยนต์ได้อย่างสมเหตุสมผลที่สุด
- สามารถนำความรู้ที่ได้จากการสร้าง model ไปประยุกต์ใช้กับธุรกิจประเภทอื่น ๆ ที่มีลักษณะคล้ายกัน

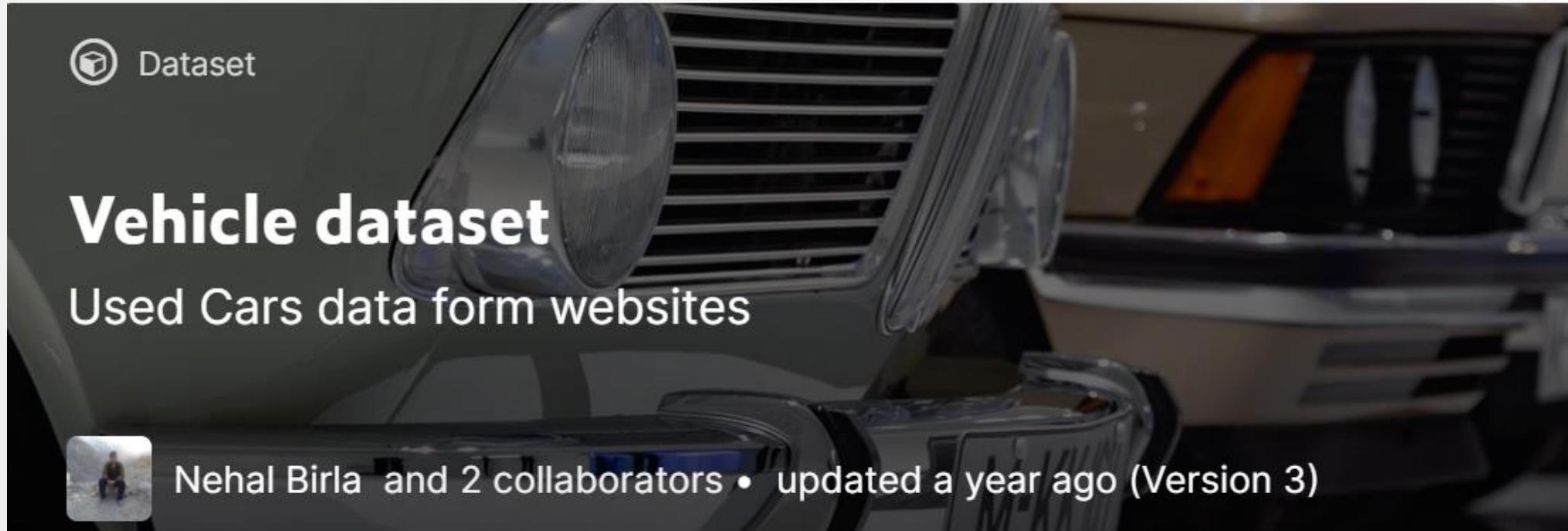


# Who this project is for?

- เจ้าของธุรกิจเต็มก์รถ
- นักประเมินราคารถยนต์
- ผู้เกี่ยวข้องกับธุรกิจที่ต้องประเมินราคา เช่น ธุรกิจโรงรับจำนำ ร้านซื้อ/ขายเครื่องดันตรี ธนาคาร
- นักวิเคราะห์ข้อมูล



# Car Price Dataset



<https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho>

# Car Price Dataset

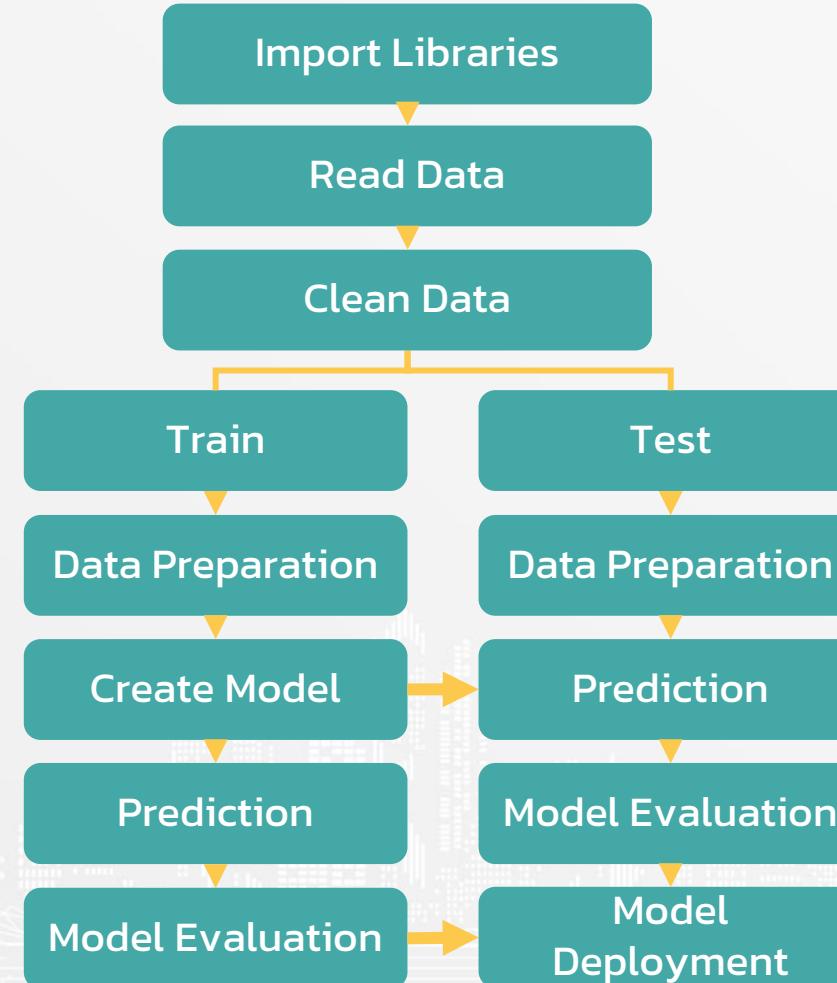
## Feature

- name : ยี่ห้อ และรุ่นของรถยนต์
- year : ปีที่รถยนต์ถูกซื้อ
- km\_driven : เลขกิโลของรถยนต์
- fuel : ประเภทของน้ำมันที่รถยนต์ใช้
- seller\_type : ขายด้วยตัวเอง หรือ ขายผ่านนายหน้า
- transmission : ระบบเกียร์
- owner : เป็นรถยนต์มือหนึ่ง มือสอง หรือ อื่น ๆ

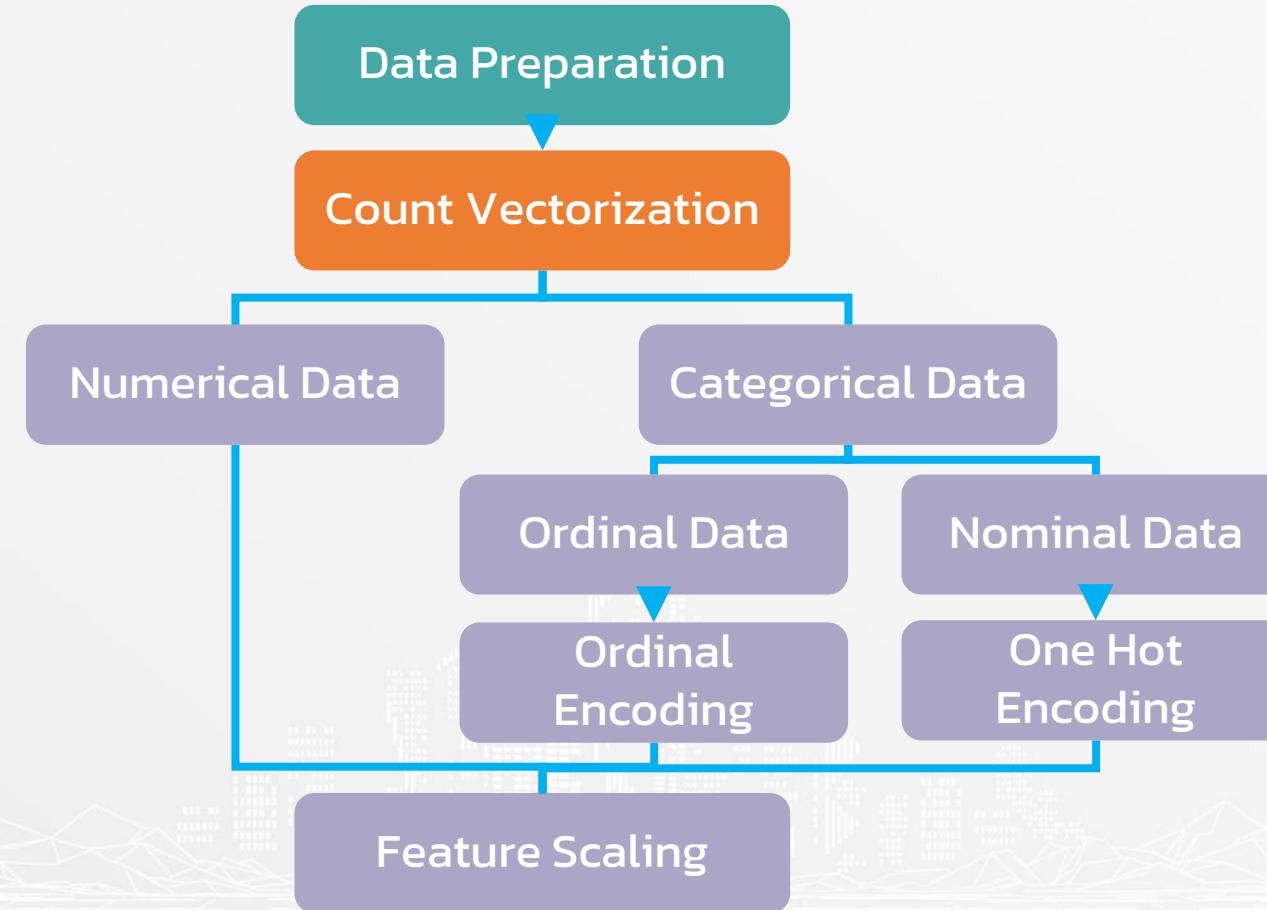
## Target

- selling\_price : ราคารถยนต์ที่ขายได้

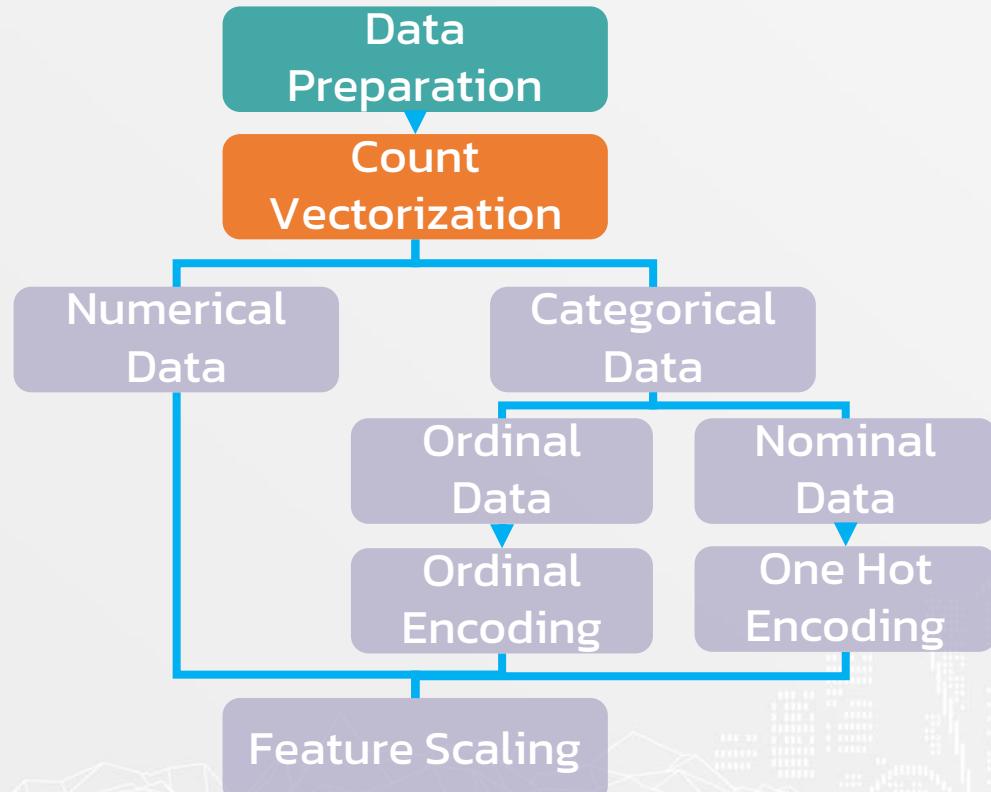
# What we learn from this project?



# Data Preparation



# What we learn from this project?



## Count vectorization

สร้าง feature ใหม่ โดยการหา unique word จากข้อความทั้งหมดใน dataset จากนั้นให้พิจารณาว่าแต่ละข้อความประกอบด้วย unique word อะไรบ้าง และจำนวนกี่ครั้ง

	'apple'	'green'	'is'	'kiwi'	'orange'	'red'
'Apple is red'	1	0	1	0	0	1
'Kiwi is green'	0	1	1	1	0	0
'Orange is orange'	0	0	1	0	2	0

# What we learn from this project?

	con_05	con_0i	con_10	con_100	...	con_xza	con_yaris	...
Mahindra Xylo D4	0	0	0	0	...	0	0	...
Maruti Swift Dzire VDI	0	0	0	0	...	0	0	...
Mahindra KUV 100 mFALCON G80 K8 5str	0	0	0	1	..	0	0	..
:	:	:	:	:	:	:	:	:

# Code

- Count vectorization for **training set**

```
1 corpus_train = X_train['name'].tolist()
2 vectorizer = CountVectorizer()
3 vectorizer.fit(corpus_train)
4 cnt_vec_train = vectorizer.transform(corpus_train).toarray()
```

```
1 cnt_vec_feature_name = ['con_' + feature for feature in vectorizer.get_feature_names()]
```

```
1 X_train[cnt_vec_feature_name] = cnt_vec_train
2 X_train.drop('name', axis=1, inplace=True)
```

# Code

- Count vectorization for **test set**

```
1 corpus_test = X_test['name'].tolist()  
2 cnt_vec_test = vectorizer.transform(corpus_test).toarray()
```

```
1 X_test[cnt_vec_feature_name] = cnt_vec_test  
2 X_test.drop('name', axis=1, inplace=True)
```



## 01. CAR PRICE



**Normal Equation**

**Ridge Regression**

**Lasso Regression**

**Elastic net**



## 01. CAR PRICE/**Normal Equation**





## 01. CAR PRICE/Ridge Regression





## 01. CAR PRICE/Lasso Regression





## 01. CAR PRICE/Elastic net



# Workshop



# Neural Network & Deep Learning

**Neural Network**



**Deep Learning**



**Deep Learning for  
Regression**



**Deep Learning for  
Classification**



**Workshop**

