

TAUTOLOGY  
INNOVATION  
SCHOOL

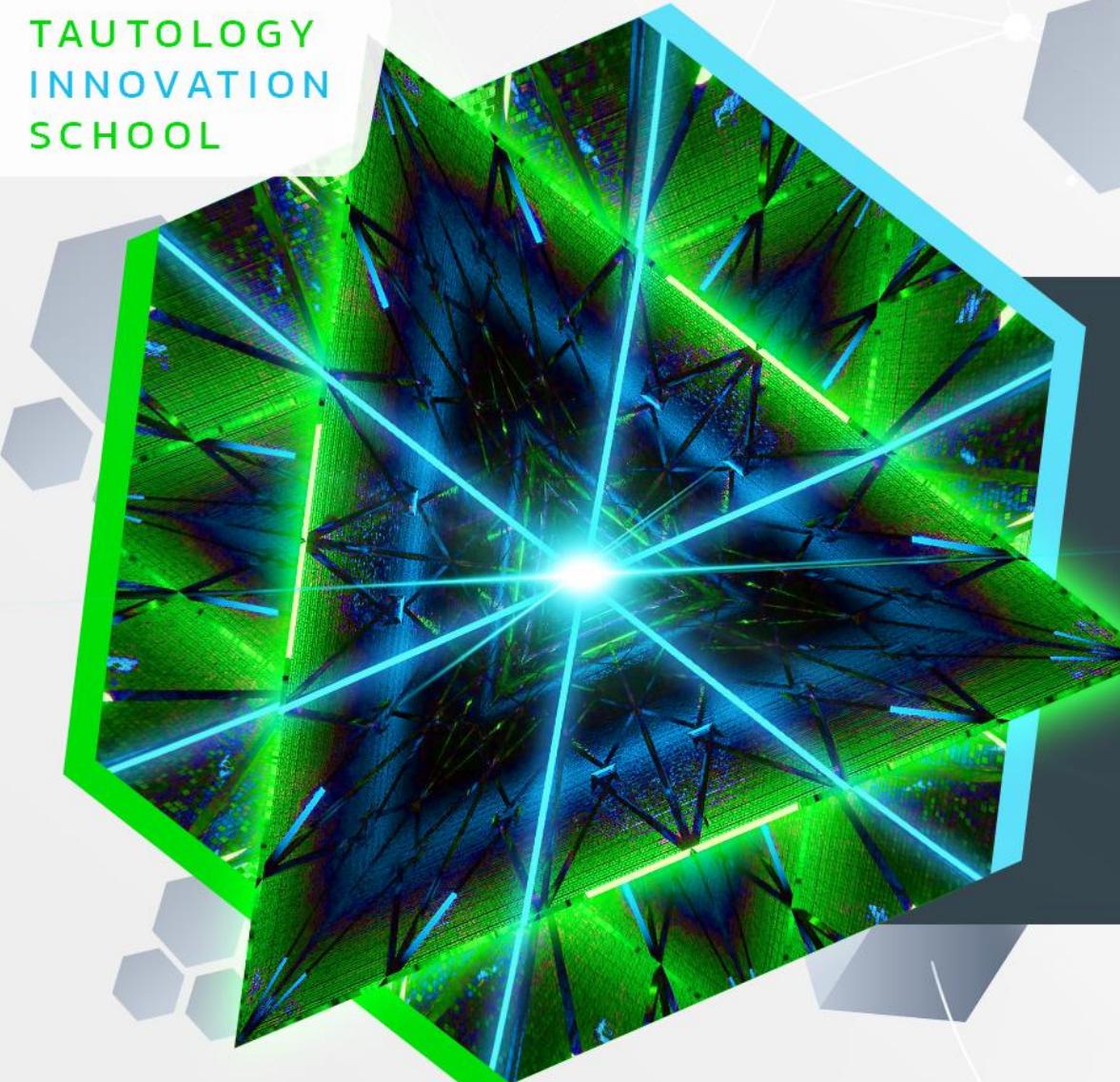


LINEAR REGRESSION

# DEEP101

DEEP LEARNING

BY TAUTOLOGY



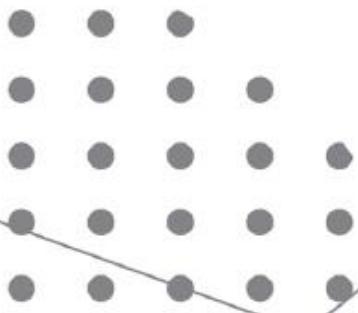
MADE BY TAUTOLOGY THAILAND  
DO NOT PUBLISH WITHOUT PERMISSION

facebook/tautologyai  
www.tautology.live

# KRIN CHINPRASATSAK

DATASCIENTIST  
INSTRUCTOR  
SPEAKER

Experienced data scientist, instructor and speaker,  
AI consultant for the public and private sectors,  
CEO & Co-founder of MADEBYAI and QuantMetric



# DL101 : Linear Regression



# Lecture

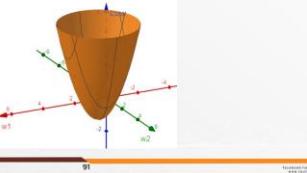
**Problem with Linearly Dependent**

**Normal Equation**

$$w = (X_b^T X_b)^{-1} X_b^T y$$

**How to create model**

กราฟนี่นาได้อ่อน弱?



2. Feature Selection



**Marketing**

	Linear Relationship (Scatter plot)	Normality of Residuals	Homoscedasticity
MARKETING	✓	✗	✓
INVESTMENT	SET50 EURUSD XAUUSD BTCUSD	✓ ✓ ✓ ✓	✓ ✓ ✗ ✗

✓ = ถูกต้องตาม assumption  
✗ = พวณ์จะเป็นรึไม่เป็นตาม assumption  
✗ = ไม่ถูกต้องตาม assumption

**Solution**

$$(X_b^T X_b)w = X_b^T y$$

+ reduce echelon  
+ สมการของร่อง  
+ Lagrange multipliers

minimize  $\sum_{j=0}^p w_j^2$   
subject to  $\sum_{i=1}^n (y_i - \hat{y}_i)^2 = c$

**How to create model**

1. Error ของแต่ละ sample จะไม่เท่ากัน

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

$$= (-0.3) + (0.6) + (0.3) + (0.4) + (-1)$$

$$= 0$$

การคำนวณต้องทำให้ error หักล้างกัน

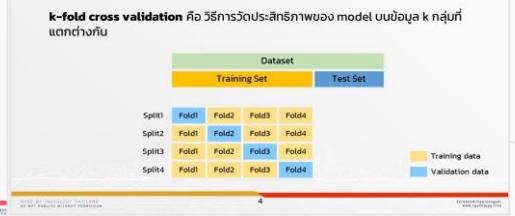
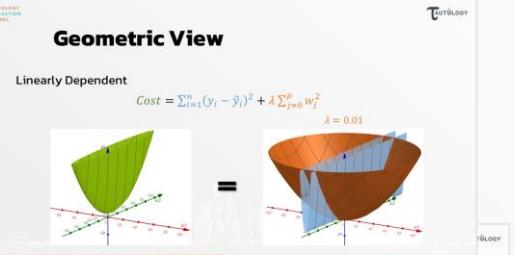
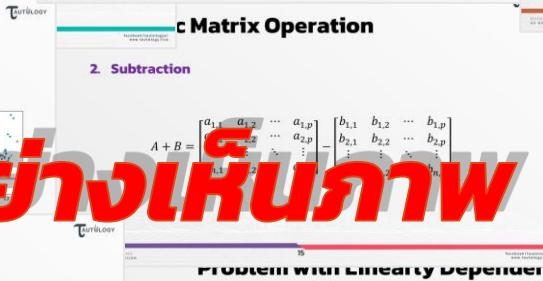
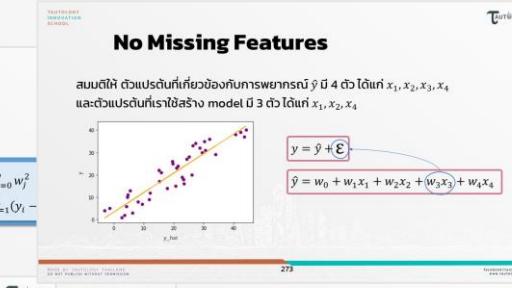
**Ridge Regression**

$$\text{minimize } \sum_{j=0}^p w_j^2$$

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

$$\text{minimize Cost } (\nabla \text{Cost} = 0)$$

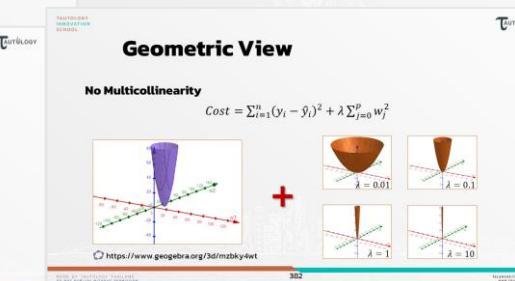
Cost =  $\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$



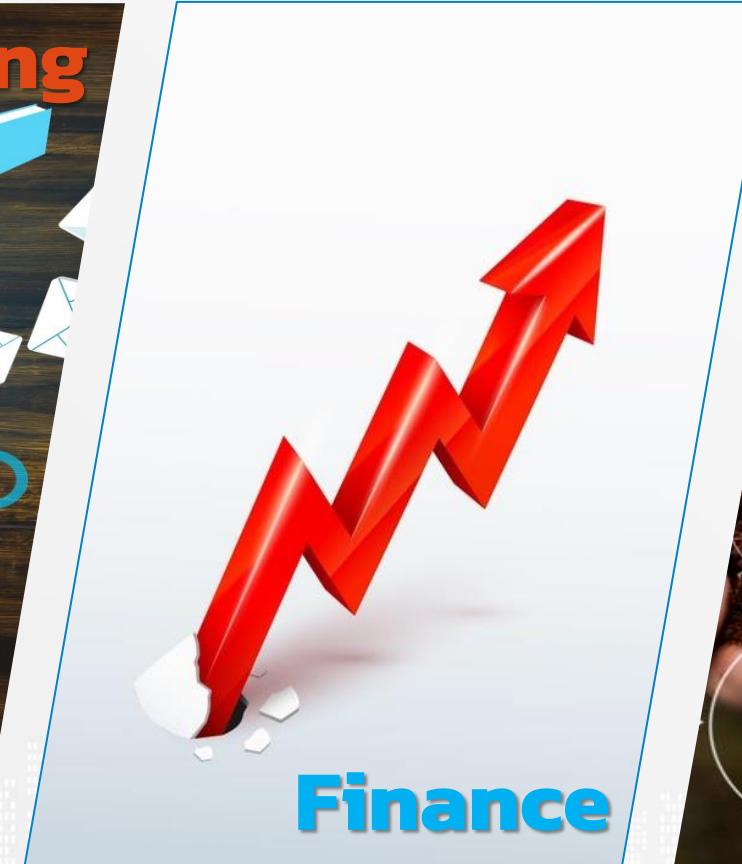
$$\sum_{j=0}^p w_j$$

$$\sum_{j=0}^p w_j^2$$

$$\sum_{j=0}^p |w_j|$$



# Workshop



# Workshop

**Insurance**



**Car Price**

**Bike  
Sharing**



# All Topics



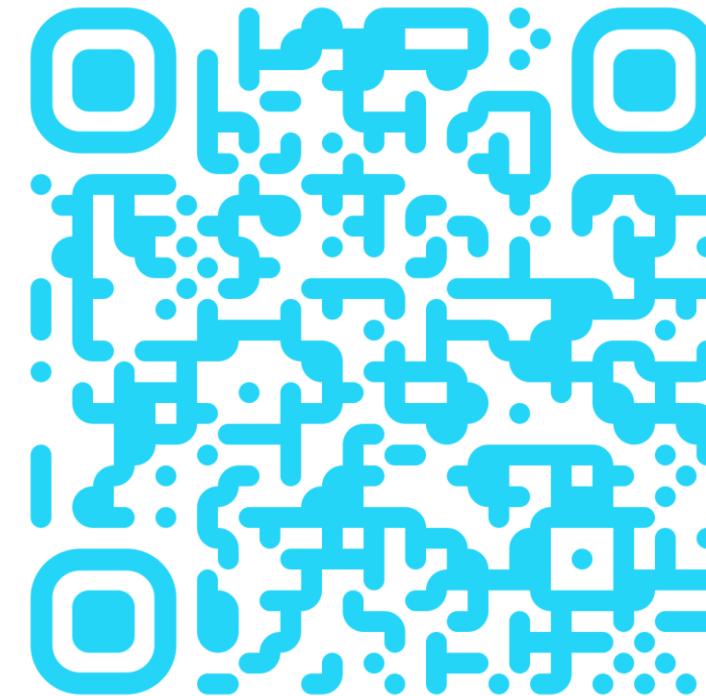
# DL101 : Linear Regression



# Course Journey



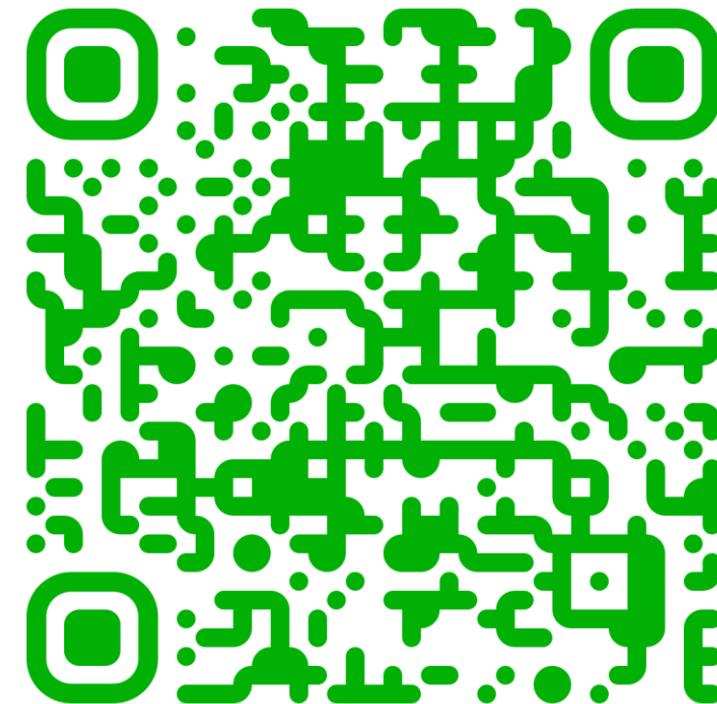
# Facebook Group



TAUTOLOGY Deep Learning

<https://www.facebook.com/groups/2723268457980889>

# Document for DL101



<https://github.com/TAUTOLOGY-EDUCATION/Deep-Learning-the-Series/tree/main/DL101>



# AI OVERVIEW

BY TAUTOLOGY

# AI Overview

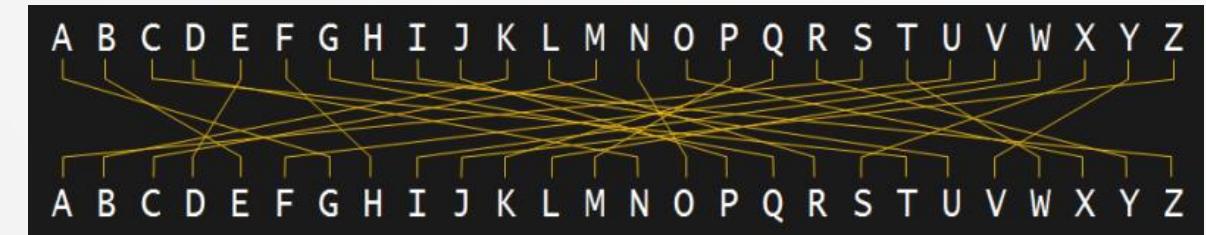




# AI and Machine Learning

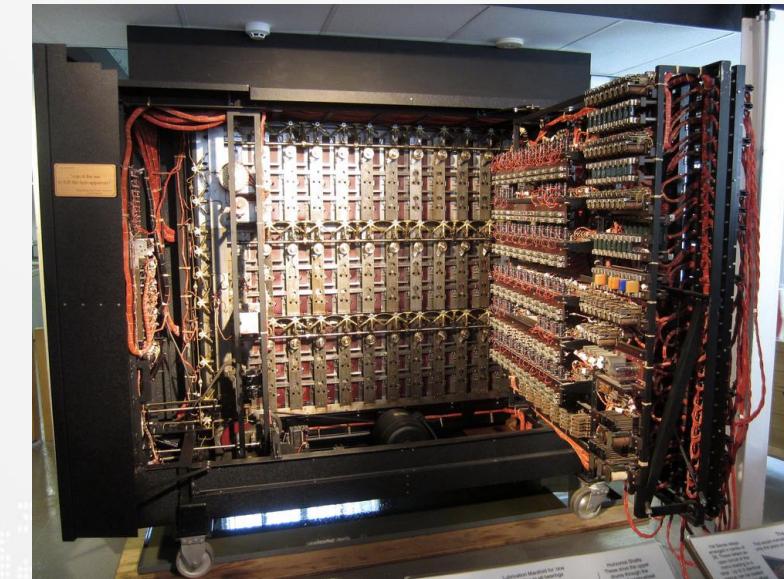
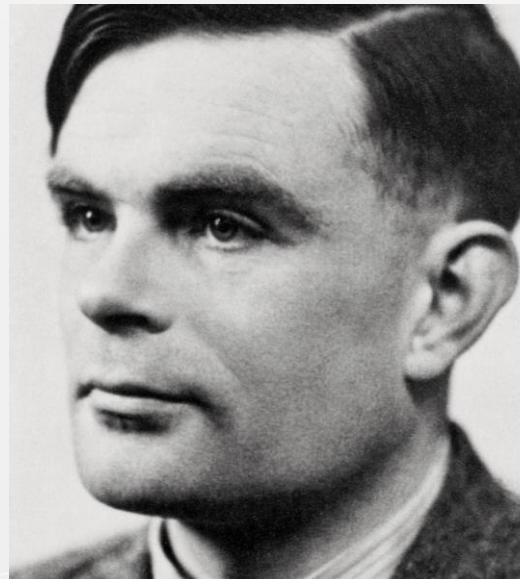
# History of AI

## Enigma Code



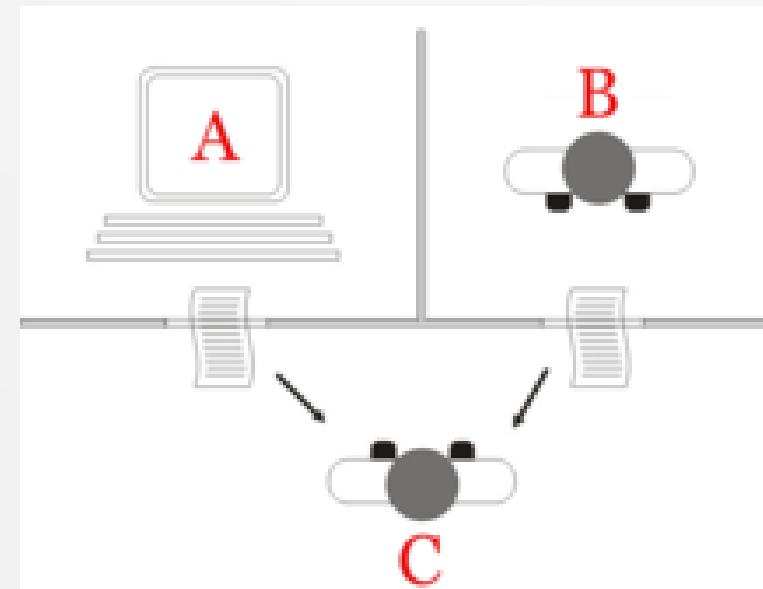
# History of AI

## The Bombe : Enigma code-breaking machine



# History of AI

## Turing test

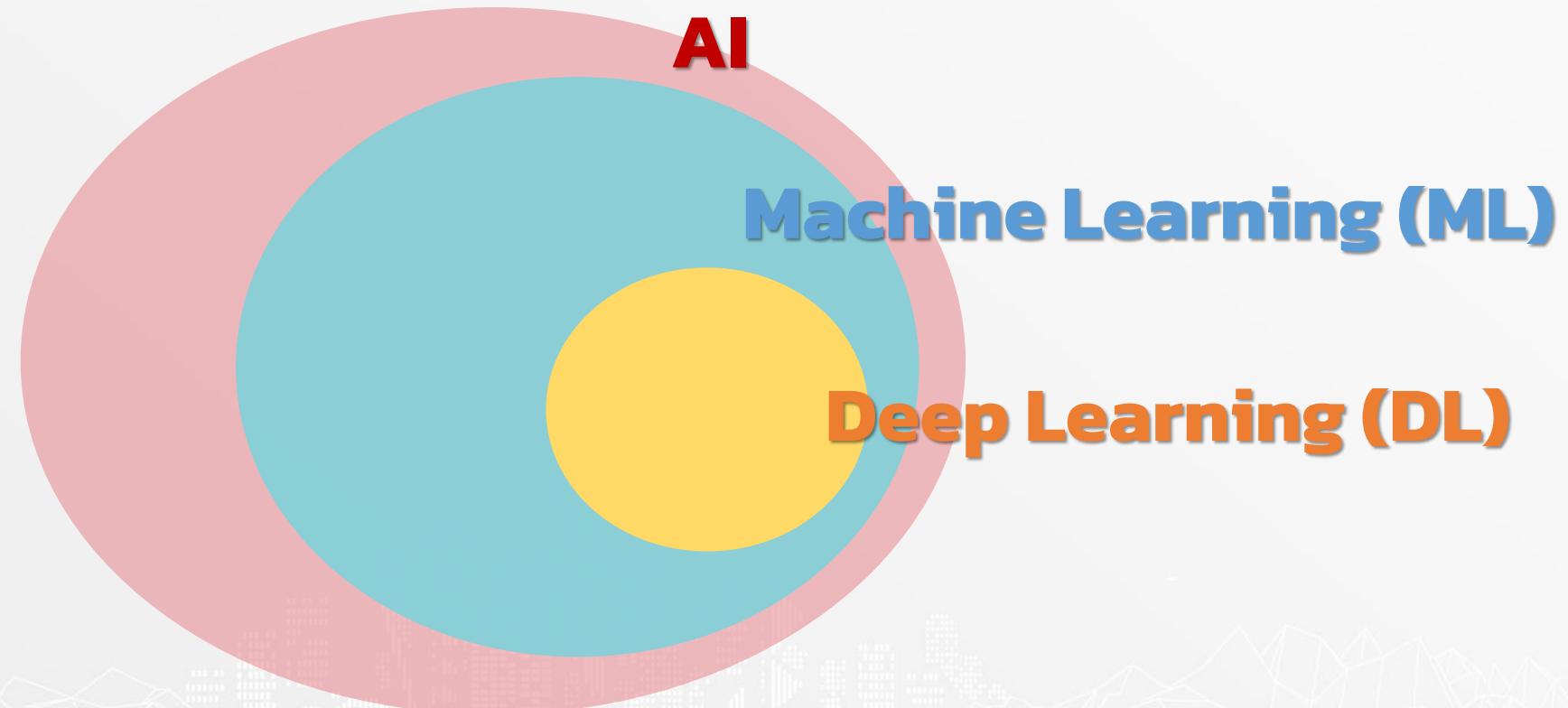


# What is AI?

"AI is a computer system able to perform tasks that ordinarily require human intelligence... Many of these artificial intelligence systems are **powered by machine learning**, some of them are **powered by deep learning** and some of them are **powered by very boring things like rules**."

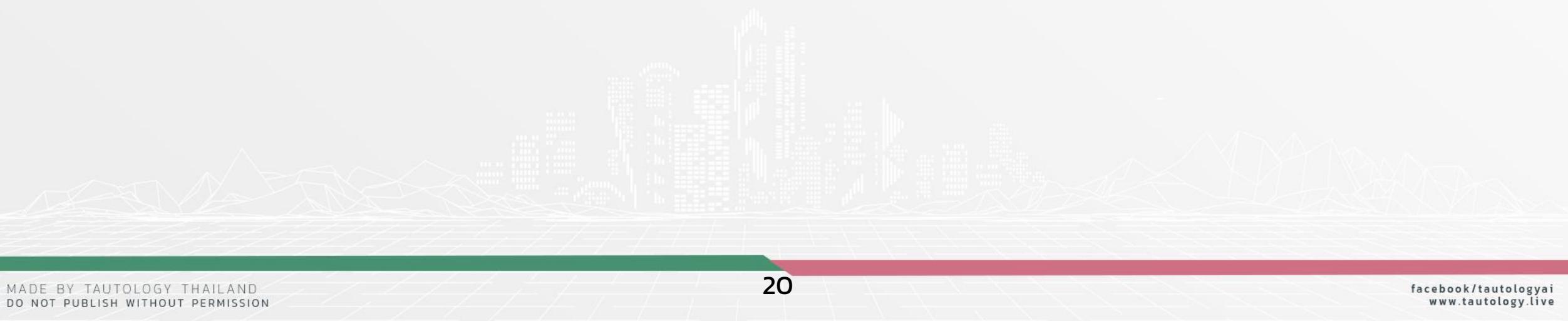
**Jeremy Achin, DataRobot CEO, speech at the Japan AI Experience in 2017**

# AI, ML and DL

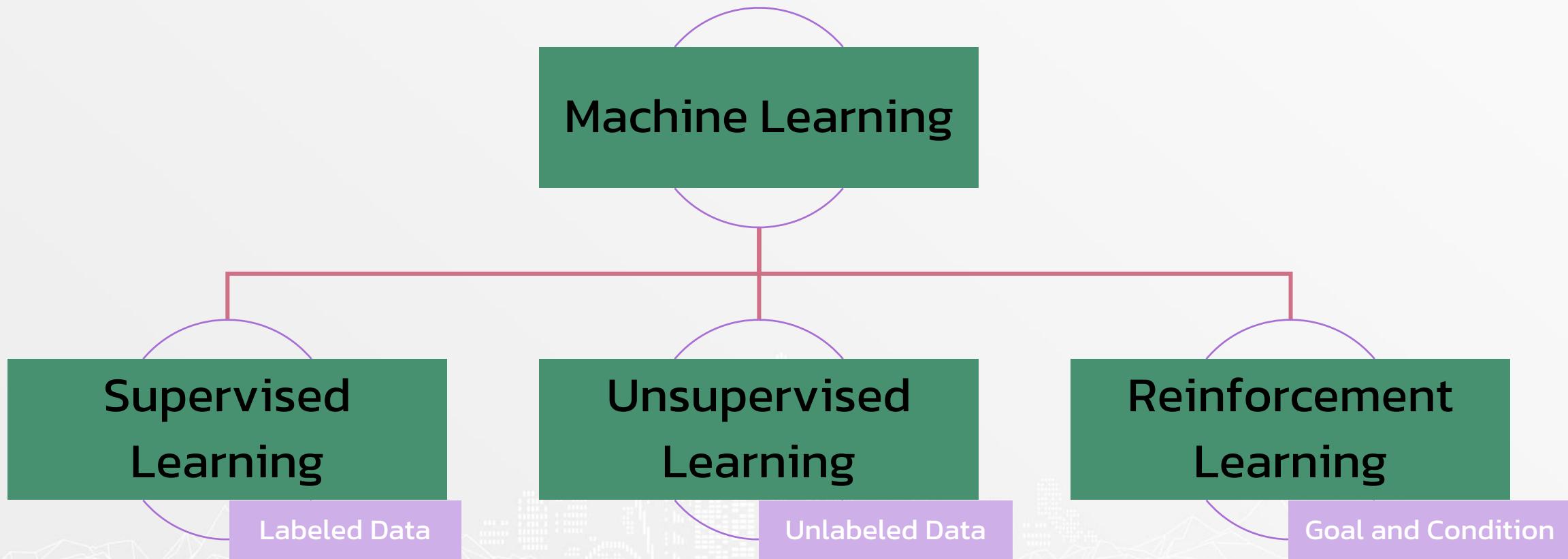


# Type of Machine Learning

1. Supervised Learning Algorithms
2. Unsupervised Learning Algorithms
3. Reinforcement Learning Algorithms



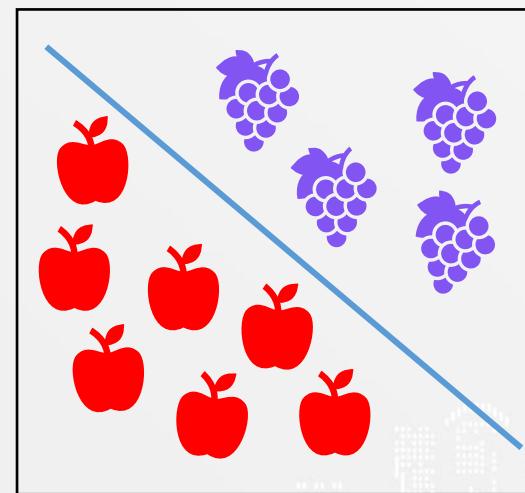
# Type of Machine Learning



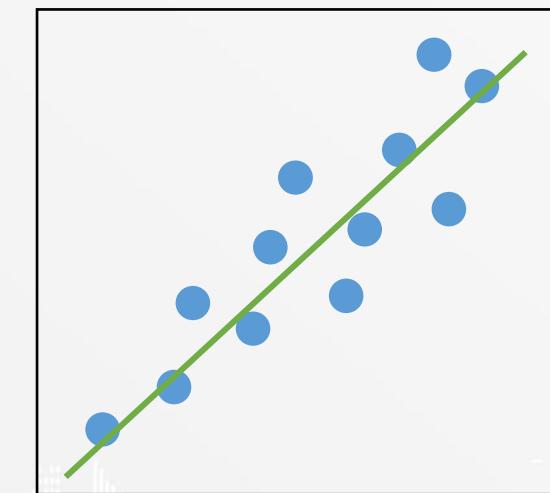
# Supervised Learning



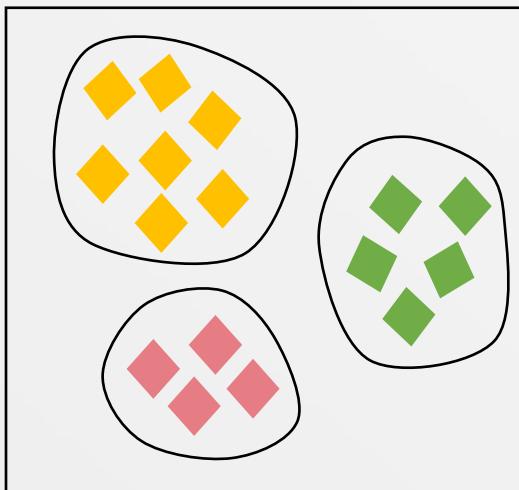
## Classification



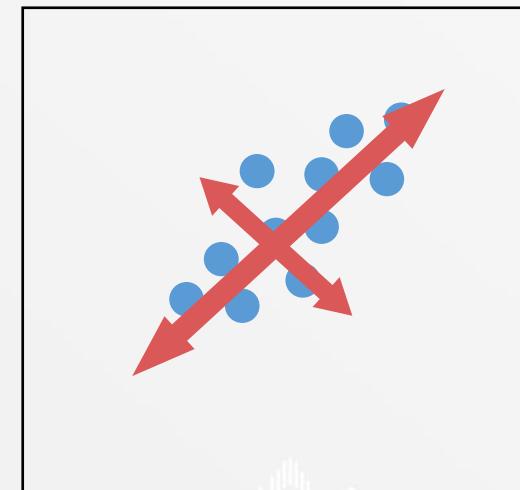
## Regression



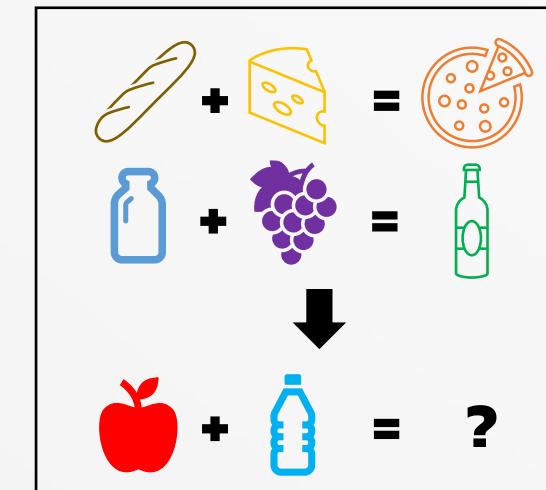
# Unsupervised Learning



**Clustering**

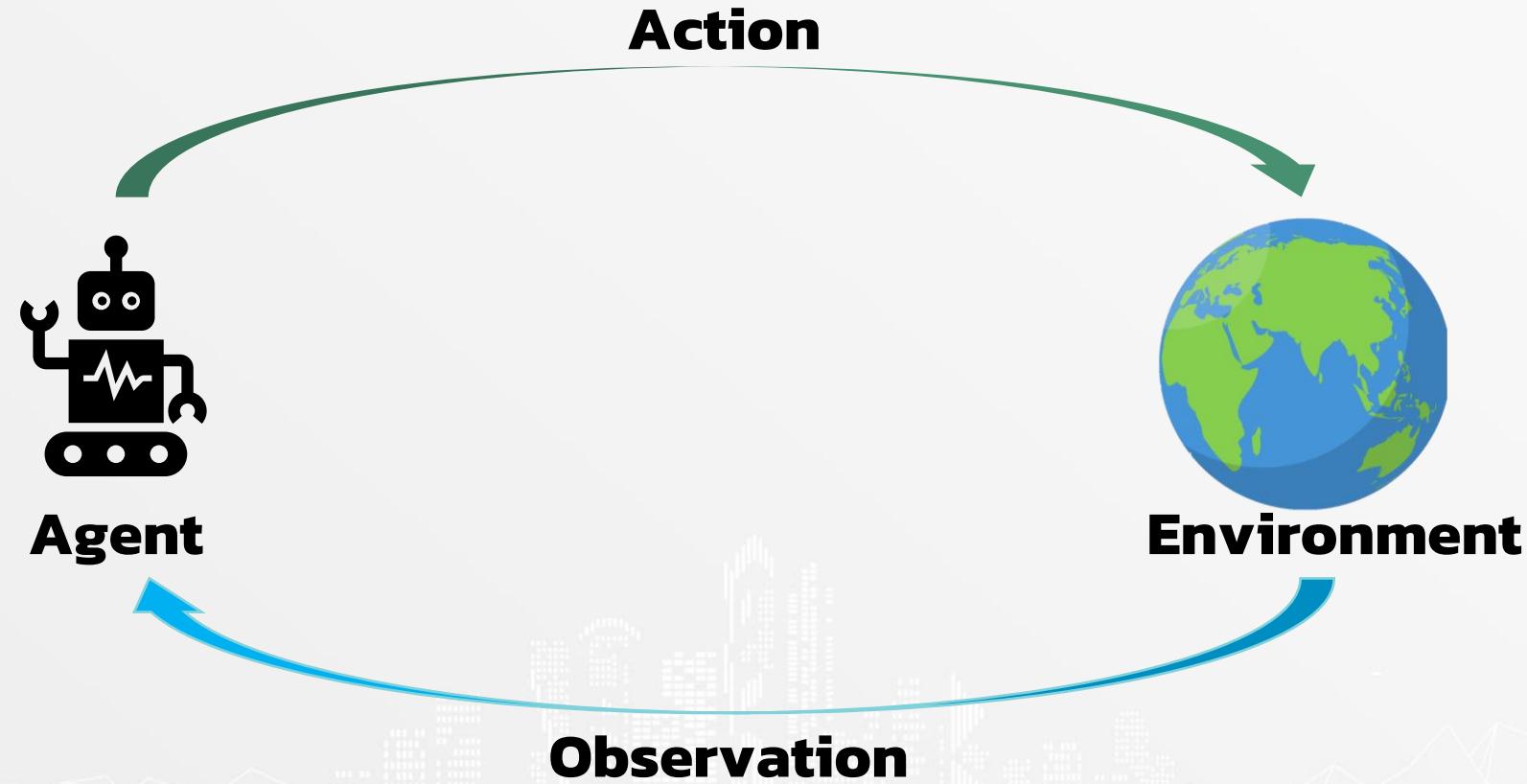


**Dimensionality  
Reduction**

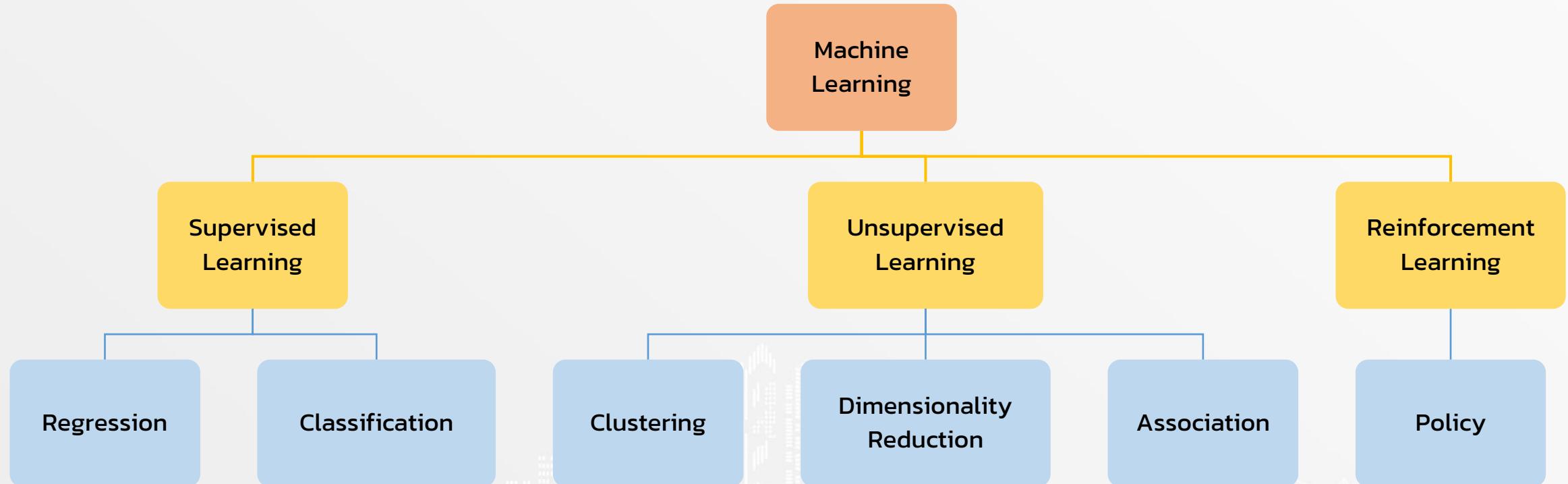


**Association**

# Reinforcement Learning



# Type of Machine Learning



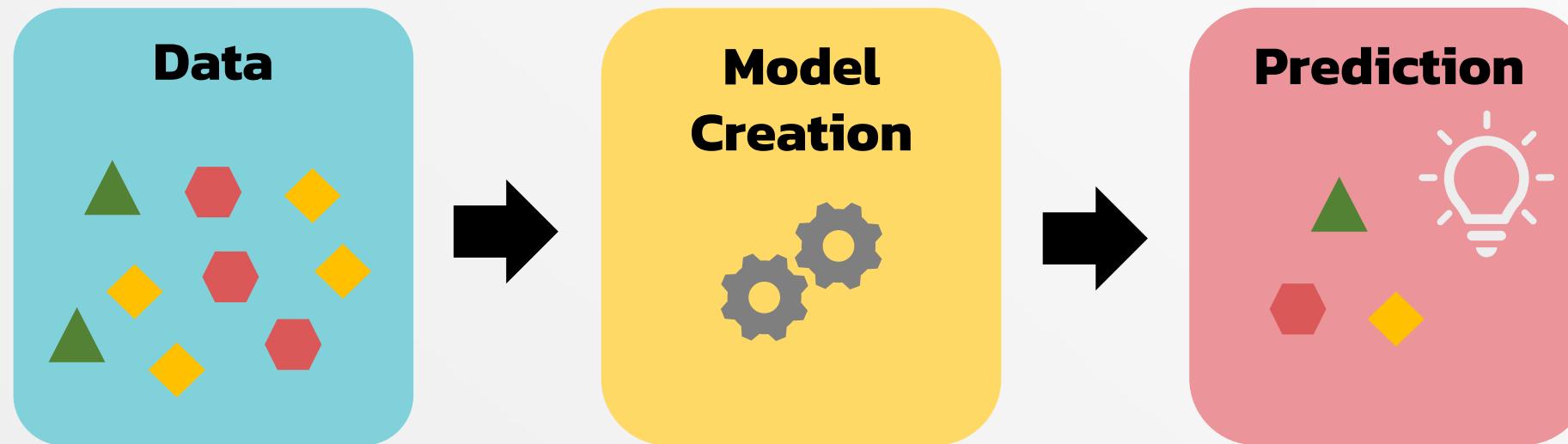
# AI Overview





# Supervised Learning

# Concept of Supervised Learning



# Concept of Supervised Learning

**Data   ⇒  Model   ⇒  Prediction**

# Regression and Classification



# Classification

น้ำหนัก (kg)	ความดัน (mmHg)	เป็นโรคเบาหวาน
65	130	ไม่เป็น
42	142	ไม่เป็น
56	171	เป็น
71	129	เป็น
59	135	ไม่เป็น

**60****127****?**

ตัวอย่างการพยากรณ์โรคเบาหวาน โดยใช้ตัวแปรต้น คือ น้ำหนัก และ ความดัน

# Classification

น้ำหนัก (kg)	ความดัน (mmHg)	เป็น โรคเบาหวาน
65	130	ไม่เป็น
42	142	ไม่เป็น
56	171	เป็น
71	129	เป็น
59	135	ไม่เป็น

**Data**



**Model**

น้ำหนัก (kg)	ความดัน (mmHg)	เป็น โรคเบาหวาน
60	127	?



**Prediction**

# Regression

พื้นที่บ้าน (ตร.ม.)	จำนวนชั้น	ราคา (ล้านบาท)
165	1	1.89
211	2	10.03
200	2	30.1
143	1	1.99
187	2	4.5
<b>142</b>	<b>1</b>	<b>?</b>

ตัวอย่างการพยากรณ์ราคาบ้าน โดยใช้ตัวแปรต้น คือ พื้นที่บ้าน และ จำนวนชั้นของบ้าน

# Regression

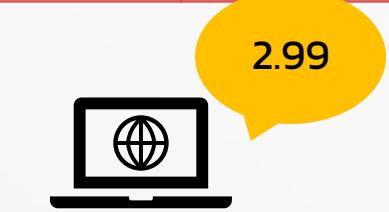
พื้นที่บ้าน (ตร.ม.)	จำนวนชั้น	ราคา (ล้านบาท)
165	1	1.89
211	2	10.03
200	2	30.1
143	1	1.99
187	2	4.5

Data



Model

พื้นที่บ้าน (ตร.ม.)	จำนวนชั้น	ราคา (ล้านบาท)
142	1	?



Prediction

# AI Overview



# DL101 : Linear Regression



TAUTOLOGY  
INNOVATION  
SCHOOL



INTRODUCTION

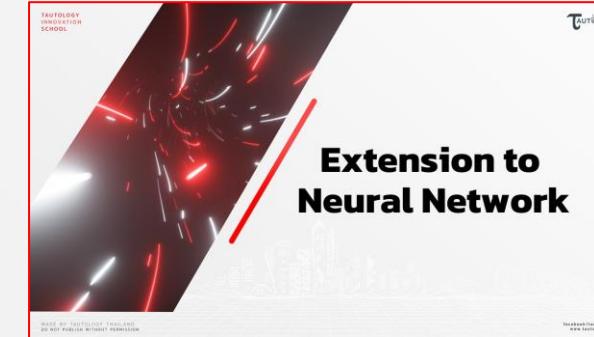


# INTRODUCTION

---

BY TAUTOLOGY

# Introduction

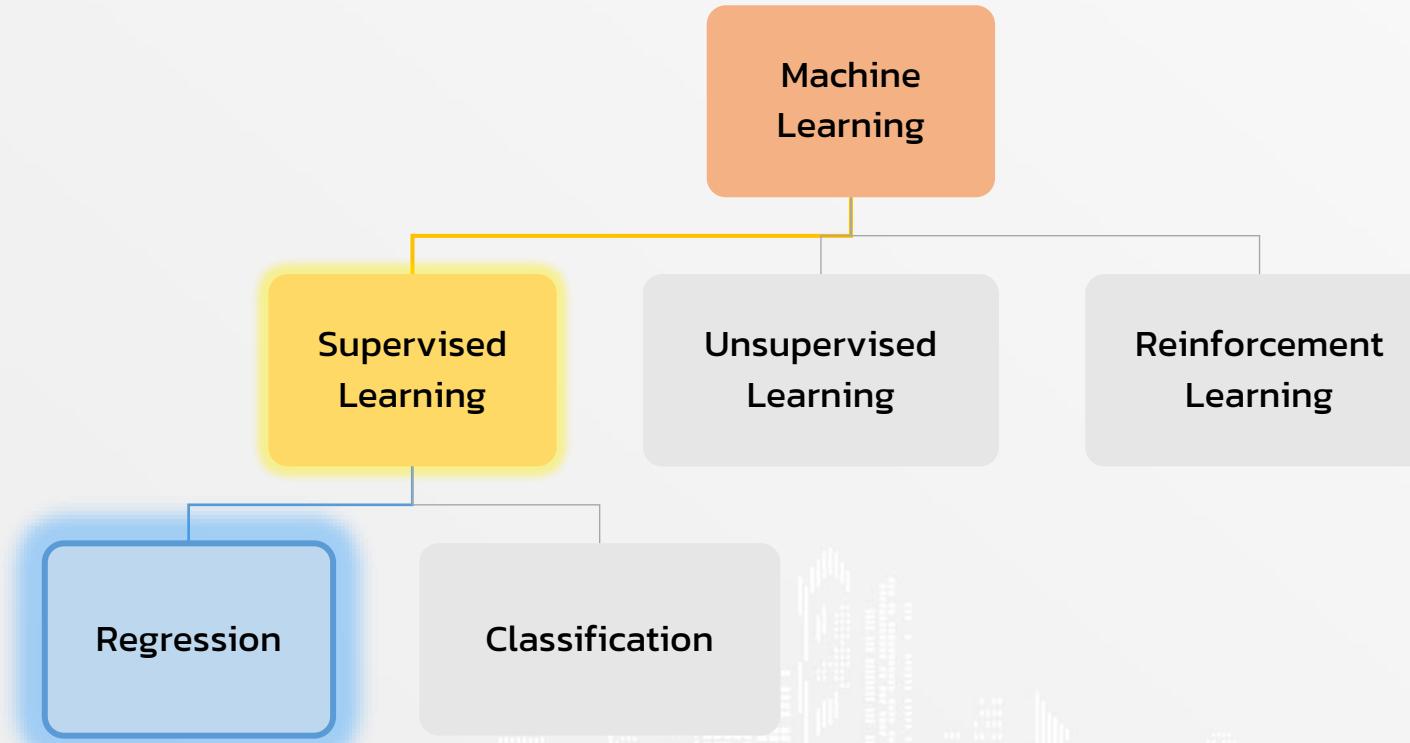


# What is Linear Regression?

# What is Linear Regression?

**Linear Regression** เป็นหนึ่งใน algorithm ประเภท **supervised learning** ที่ใช้สำหรับแก้ปัญหา regression โดยมีหลักการทำงานคือ การสร้างสมการเชิงเส้นที่ใช้ตัวแปรตั้นเพื่อพยากรณ์ตัวแปรตาม

# What is Linear Regression?



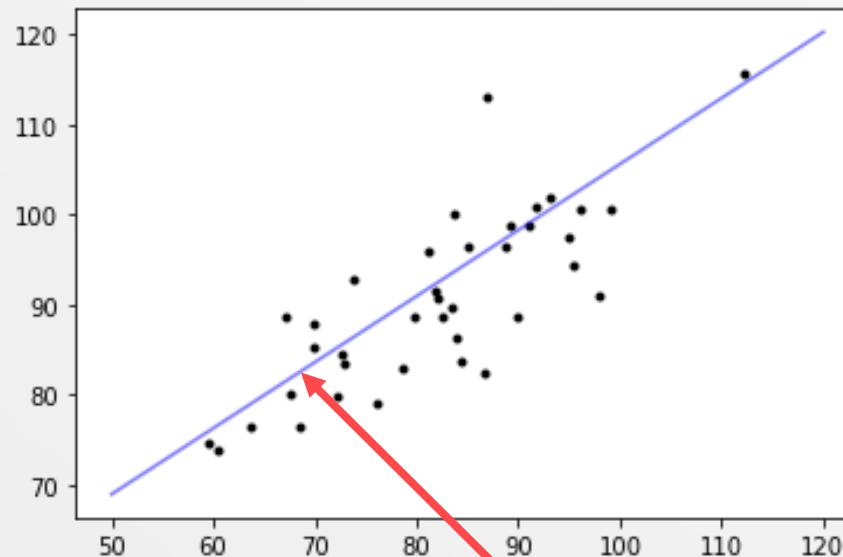
# What is Linear Regression?

สมการคงตัวของ Linear Regression

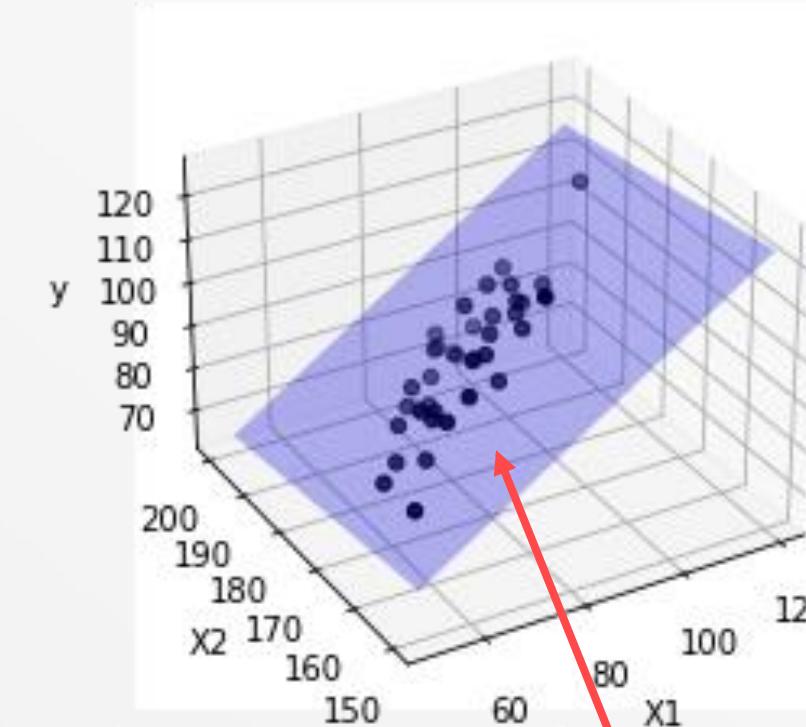
$$\hat{y} = w_0 + w_1x_1 + w_2x_2 + \cdots + w_px_p$$

- โดย
- $\hat{y}$  คือ ตัว預測 (predicted target)
  - $x_1, x_2, x_3, \dots, x_p$  คือ ตัว特征 (feature)
  - $w_0, w_1, w_2, \dots, w_p$  คือ 係數 (coefficient)

# What is Linear Regression?

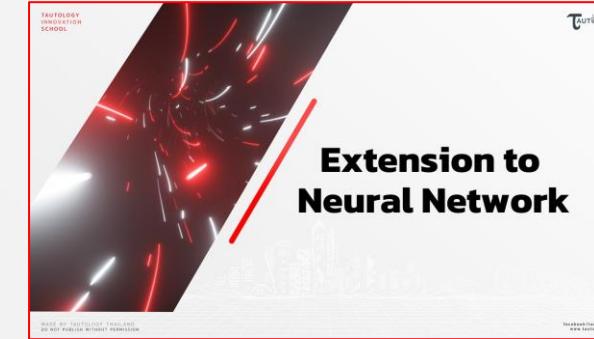


$$\hat{y} = w_0 + w_1 x_1$$
$$(\hat{y} = a + bx)$$



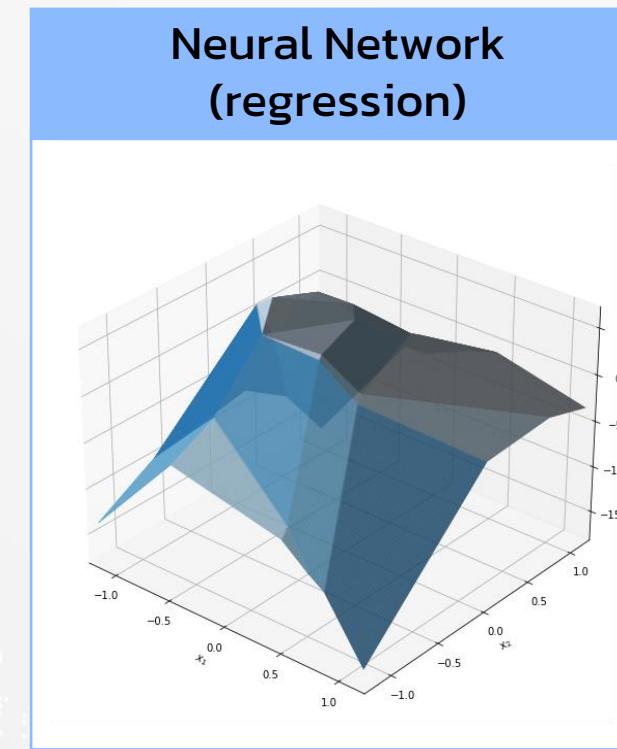
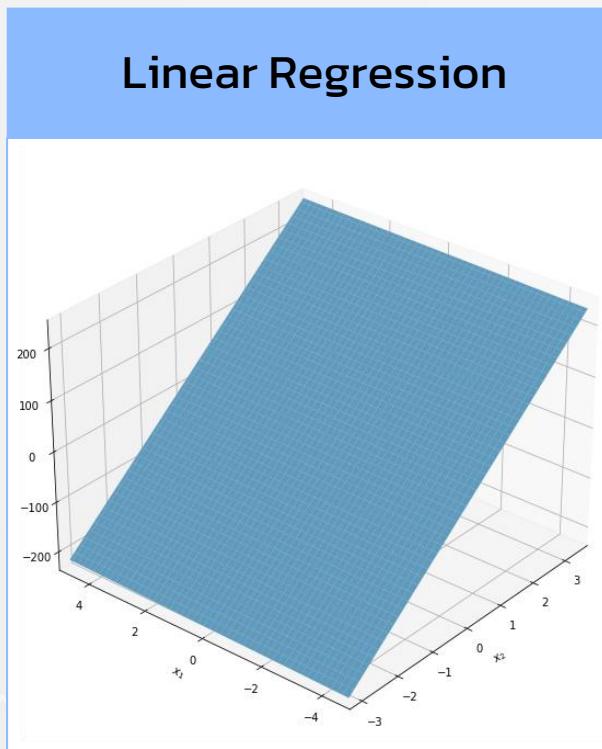
$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2$$

# Introduction

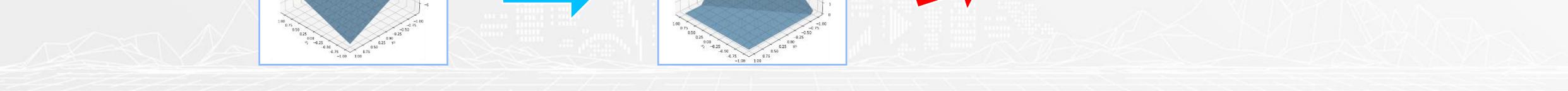
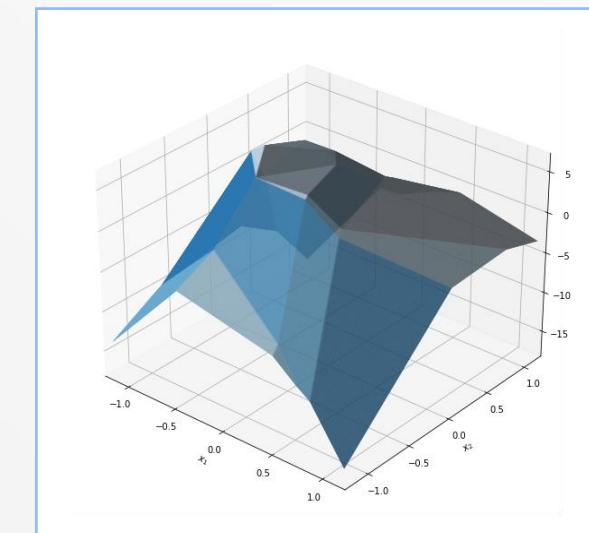
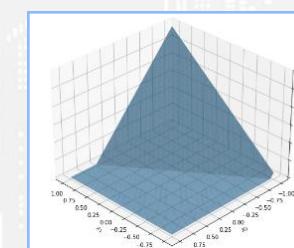
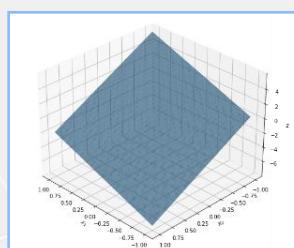
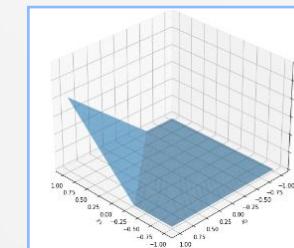
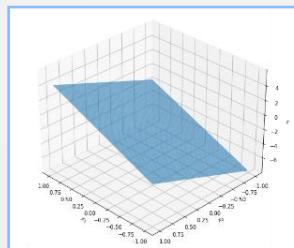
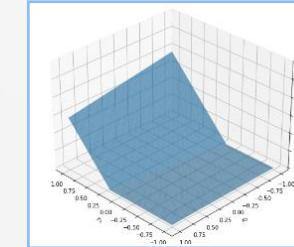
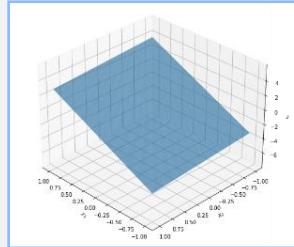


# Extension to Neural Network

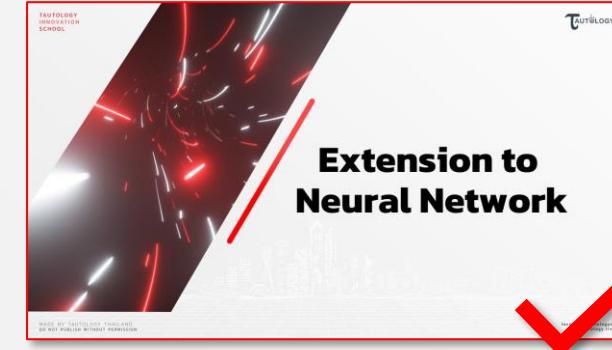
# Extension to Neural Network



# Extension to Neural Network

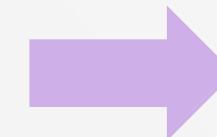


# Introduction



# Real World Application

# Real World Application



**การประเมินค่าตัวของนักฟุตบอลในตำแหน่งกองหน้า**  
โดยพิจารณาจาก อายุ ส่วนสูง การมีส่วนร่วมกับประตู จำนวนเกมที่ลงเล่น เป็นต้น

อ้างอิง : [2018, Yunus et al] Multiple Linear Regression Approach For Estimating the Market Value of Football Players in Forward Position

# Real World Application



## การคาดการณ์ค่าใช้ค่าไฟ ของฟาร์มโคนม

โดยพิจารณาจาก จำนวนโคนม  
จำนวนเครื่องรีดนมวัว จำนวนเครื่อง  
บัดพื้นไฟฟ้า เป็นต้น

อ้างอิง : [2018, Shine et al] Multiple linear regression modelling of  
on-farm direct water and electricity consumption on pasture  
based dairy farms

# Real World Application

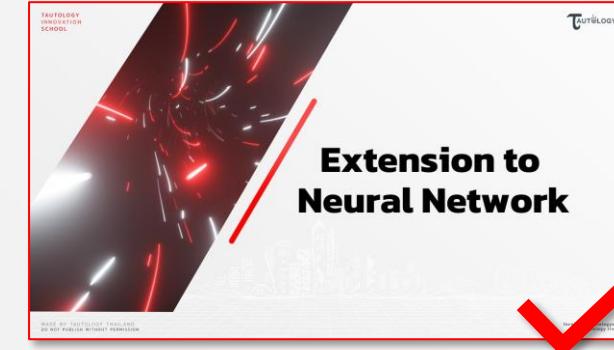


## พยากรณ์ปริมาณพืชที่จะ: ปลูกได้

โดยพิจารณาจาก สภาพดิน สภาพ  
อากาศ ปริมาณน้ำฝน อุณหภูมิ เป็น<sup>ต้น</sup>

อ้างอิง : [2017, Aditya Shastry, HA Sanjay and E. Bhanusree]  
Prediction of Crop Yield Using Regression Techniques

# Introduction



# DL101 : Linear Regression



TAUTOLOGY  
INNOVATION  
SCHOOL

MODEL CREATION



# MODEL CREATION

BY TAUTOLOGY

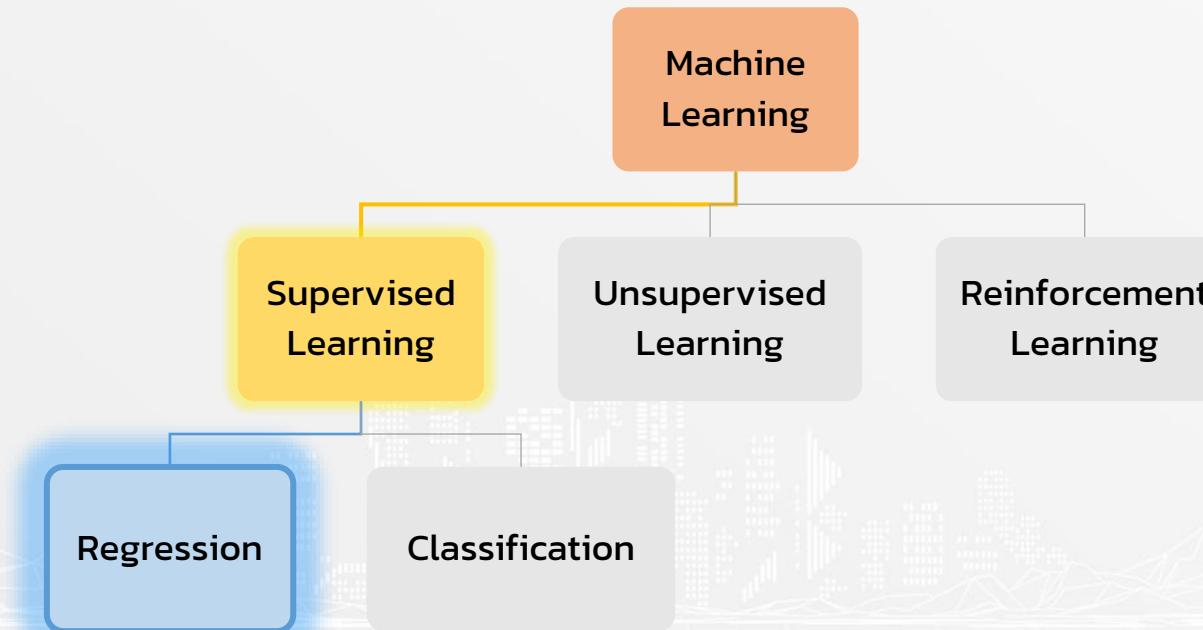
MADE BY TAUTOLOGY THAILAND  
DO NOT PUBLISH WITHOUT PERMISSION

facebook/tautologyai  
www.tautology.live

TAUTOLOGY

# Linear Regression

**Linear Regression** តូបអនុវត្តន៍ algorithm ឱ្យកេន្តិ៍ **supervised learning**



# Concept of Supervised Learning

**Data   ⇒  Model   ⇒  Prediction**

# Model Creation



# Data

# Data

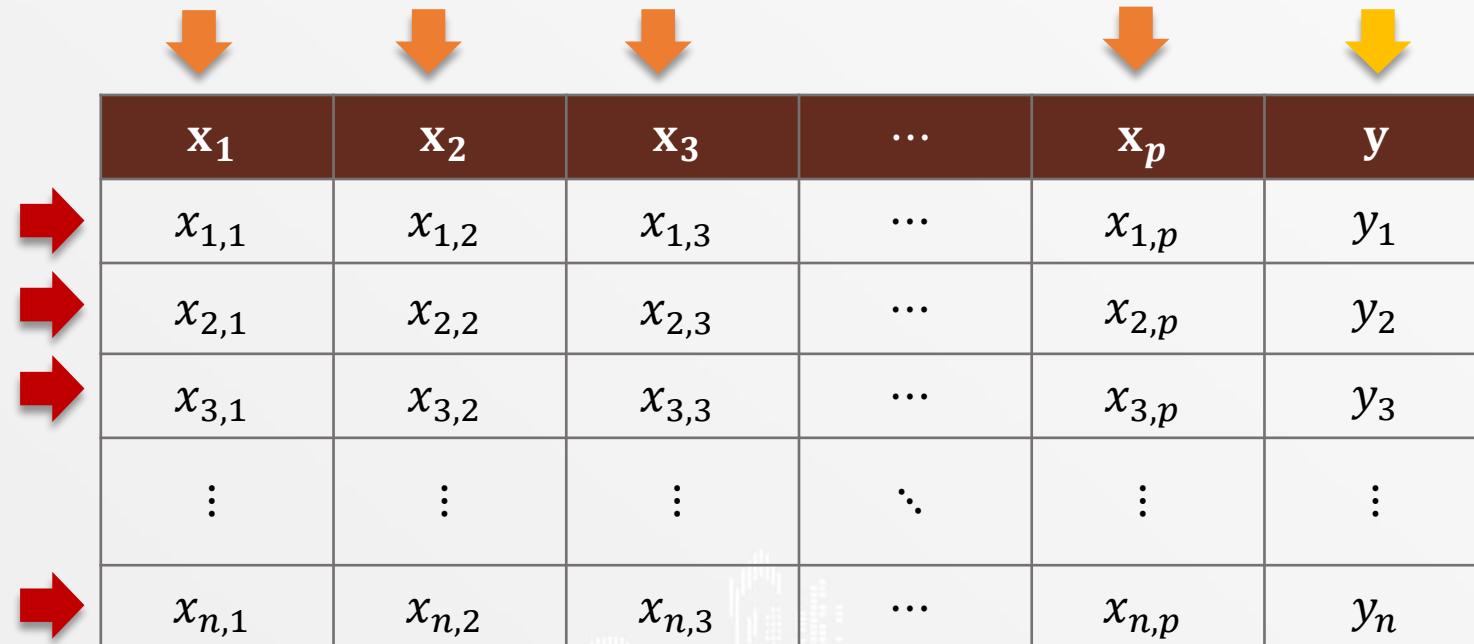
Data Stating

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



The diagram illustrates a data matrix with 6 rows and 6 columns. The columns are labeled  $x_1, x_2, x_3, \dots, x_p$  and the rows are labeled  $y$ . The matrix contains data points  $x_{i,j}$  for  $i = 1, 2, 3, \dots, n$  and  $j = 1, 2, 3, \dots, p$ . The matrix is shown as a grid of cells with alternating background colors. Orange arrows point downwards from the column labels to the matrix, and red arrows point to the left from the row labels to the matrix. The matrix is set against a background of abstract geometric shapes and patterns.

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

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

# Data Stating

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

- $x_{2,3}$  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, ราคาที่ดินต่อตารางวา

## Data

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

- ข้อมูลตามแนวแอกว คือ Sample



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 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.)	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 Requirement

- ตัวอย่างข้อมูลที่สามารถใช้งานได้เลย และยังไม่สามารถใช้งานได้

พื้นที่ของบ้าน (ตร.ว.)	ราคาที่ดิน (บาท/ตร.ว.)	ราคาขายบ้าน (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**



# Model Creation



# Model

# Model

Assumption

Real Face of the  
Model

Cost Function and  
Cost Landscape

How to Create  
Model (Math)

How to Create  
Model (Code)

Further Reading

# Assumption

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

# Assumption



For more information



## Model Improvement

# Model

**Assumption**



Real Face of the Model



Cost Function and Cost Landscape

How to Create Model (Math)



How to Create Model (Code)

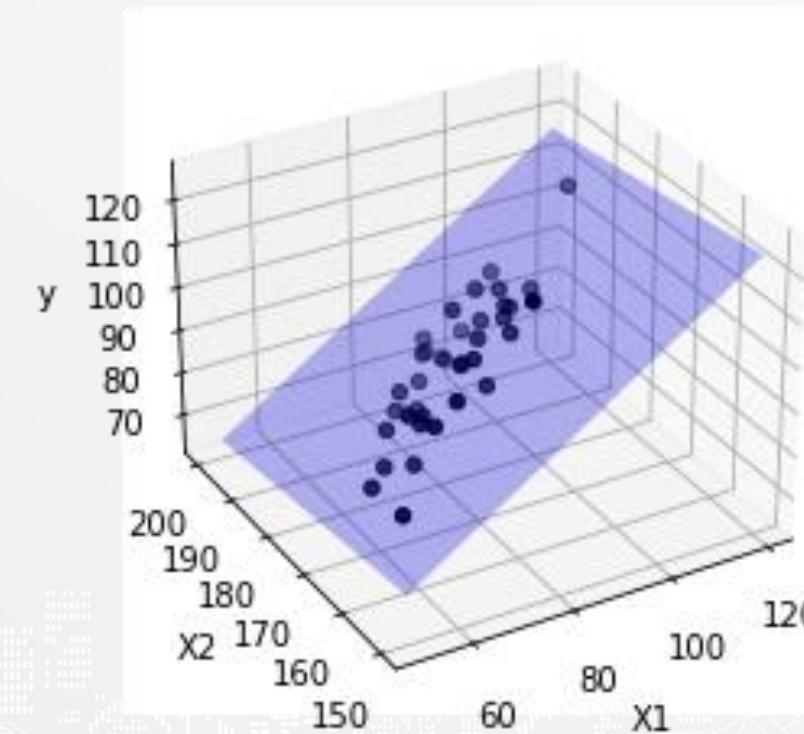


Further Reading



# Real Face of the Model

Linear regression คือ สมการเชิงเส้นที่ใช้ตัวแปรตัวนเพื่อพยากรณ์ตัวแปรตาม



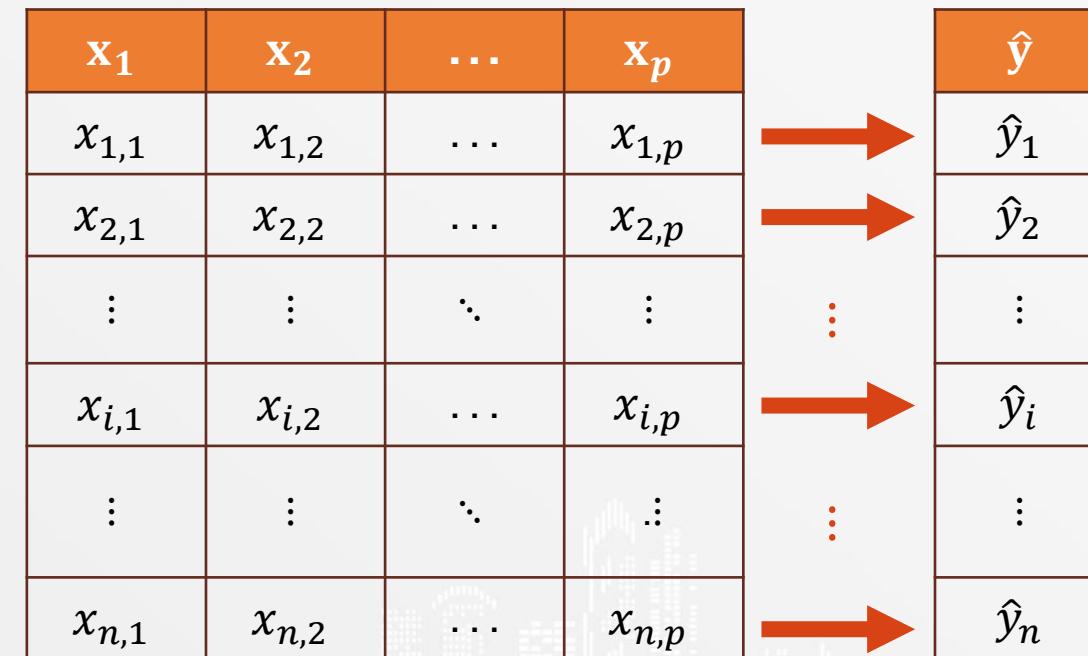
# Real Face of the Model

Linear regression คือ สมการเชิงเส้นที่ใช้ตัวแปรตัวนเพื่อพยากรณ์ตัวแปรตาม

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

- โดย
- ◆  $\hat{y}$  คือ ตัวแปรตาม (predicted target)
  - ◆  $x_1, x_2, \dots, x_p$  คือ ตัวแปรตัวน (feature)
  - ◆  $w_0, w_1, \dots, w_p$  คือ สัมประสิทธิ์ (coefficient)

# Real Face of the Model



# Real Face of the Model

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



$$\hat{y}_1 = w_0 + w_1 x_{1,1} + w_2 x_{1,2} + \cdots + w_p x_{1,p}$$

$$\hat{y}_2 = w_0 + w_1 x_{2,1} + w_2 x_{2,2} + \cdots + w_p x_{2,p}$$

⋮

$$\hat{y}_i = w_0 + w_1 x_{i,1} + w_2 x_{i,2} + \cdots + w_p x_{i,p}$$

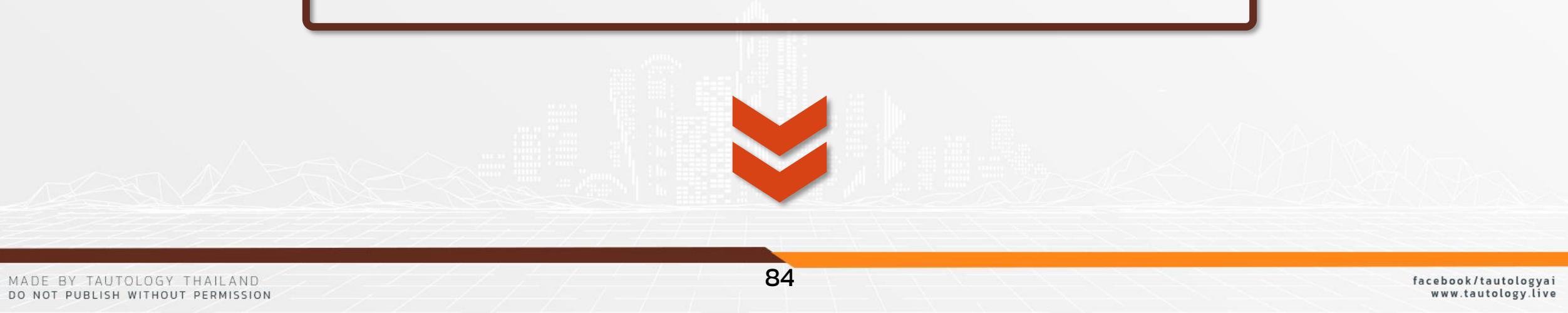
⋮

$$\hat{y}_n = w_0 + w_1 x_{n,1} + w_2 x_{n,2} + \cdots + w_p x_{n,p}$$

# Real Face of the Model

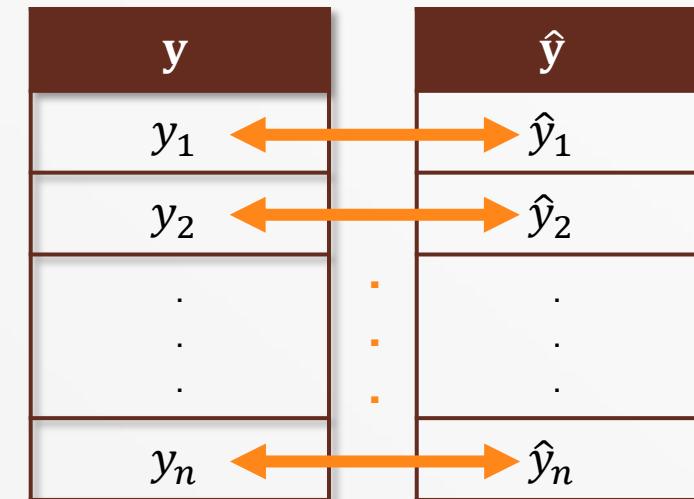
“ เป้าหมายของเราคือการหา  $w_0, w_1, w_2, \dots, w_p$  เพื่อสร้าง model ของ linear regression :

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \dots + w_p x_p ”$$



# Real Face of the Model

โดยที่  $w_0, w_1, w_2, \dots, w_p$  เหล่านี้ต้องทำให้ทำให้ผลรวมของ error ระหว่างค่าจริง ( $y_i$ ) กับค่าพยากรณ์ ( $\hat{y}_i$ ) น้อยที่สุด



# Real Face of the Model

“ เราต้องการหา  $w_0, w_1, w_2, \dots, w_p$  ที่ทำให้พารามิเตอร์ error ระหว่าง  $y_i$  กับ  $\hat{y}_i$  น้อยที่สุด ”

# Real Face of the Model

ผลรวมของ error ระหว่าง  $y_i$  กับ  $\hat{y}_i$  เรียกว่า “**Cost function**”



# Real Face of the Model



“ เราต้องการหา  $w_0, w_1, w_2, \dots, w_p$  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)**



**Further Reading**



# Cost Function and Cost Landscape

Cost function ที่เราจะใช้ในการสร้าง model คือ

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

โดยสูตรข้างต้นมีชื่อว่า Sum of Squared Errors หรือ SSE

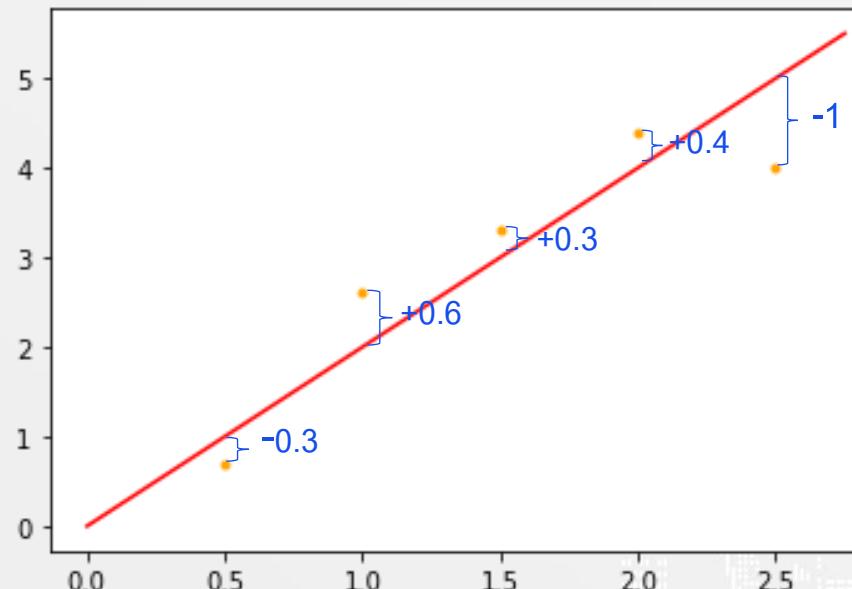
# Cost Function and Cost Landscape

## เหตุผลที่เลือกใช้ Sum of Squared Errors (SSE)

1. Error ของแต่ละ sample จะไม่หักล้างกัน
2. Cost function ที่นิยามแบบ SSE จะสามารถ diff ได้ และมีความต่อเนื่องทุกจุด
3. Cost function ที่นิยามแบบ SSE เป็น convex function และมีจุดต่ำสุดเพียง จุดเดียว

# Cost Function and Cost Landscape

## 1. Error ของแต่ละ sample จะไม่หักล้างกัน



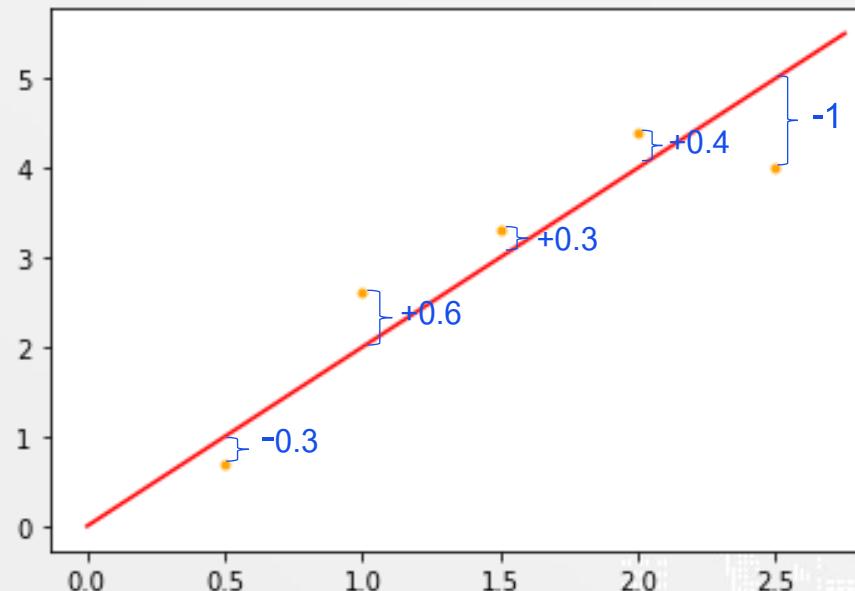
กราฟแสดงข้อมูลระหว่างค่าจริง  $y_i$  (จุด) และค่าพยากรณ์  $\hat{y}_i$  (เส้น)

$$\begin{aligned}SE &= \sum_{i=1}^n (y_i - \hat{y}_i) \\&= (-0.3) + (0.6) + (0.3) + (0.4) + (-1) \\&= 0\end{aligned}$$

 **การมีค่าติดลบทำให้ error หักล้างกัน**

# Cost Function and Cost Landscape

## 1. Error ของแต่ละ sample จะไม่หักล้างกัน



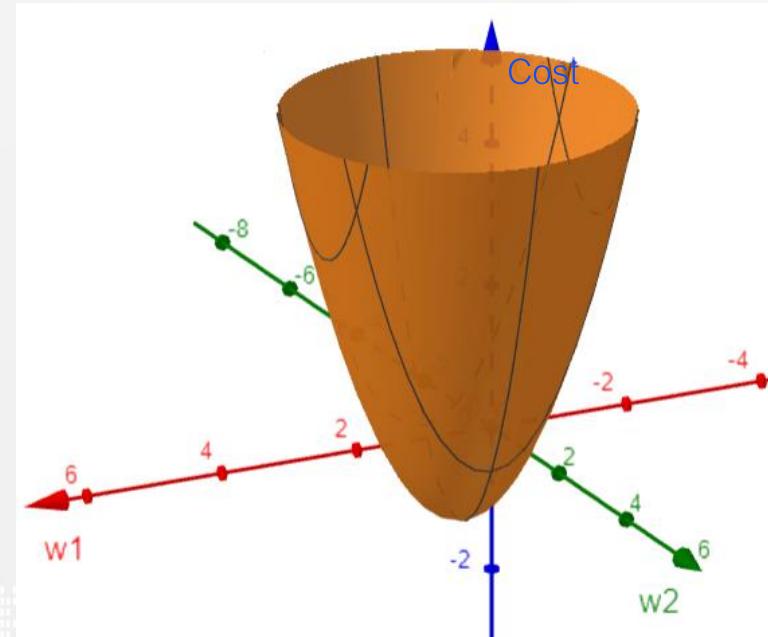
กราฟแสดงข้อมูลระหว่างค่าจริง  $y_i$  (จุด) และค่าพยากรณ์  $\hat{y}_i$  (เส้น)

$$\begin{aligned} SSE &= \sum_{i=1}^n (y_i - \hat{y}_i)^2 \\ &= (-0.3)^2 + (0.7)^2 + (0.2)^2 + (0.4)^2 + (-1)^2 \\ &= 0.09 + 0.49 + 0.04 + 0.16 + 1 \\ &= 1.78 \end{aligned}$$

การยกกำลังสองทำให้ error ไม่หักล้างกัน

# Cost Function and Cost Landscape

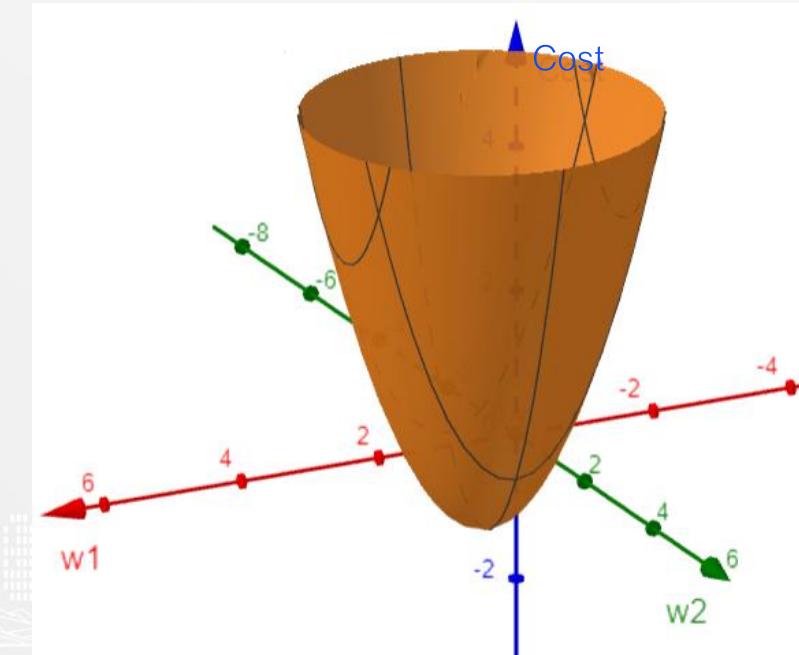
2. Cost function ที่นิยามแบบ SSE จะสามารถ diff ได้ และมีความต่อเนื่องทุกจุด



กราฟของ cost landscape โดยที่ cost function เป็น SSE

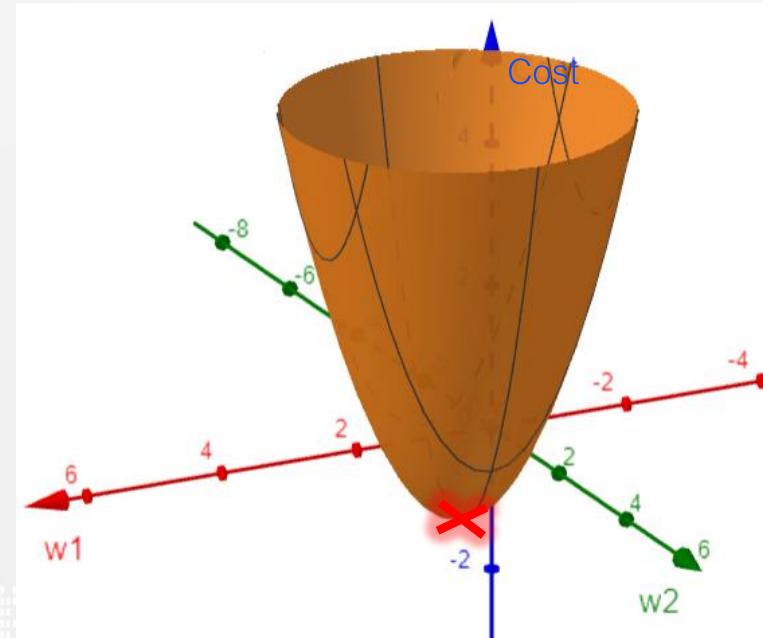
# Cost Function and Cost Landscape

การที่ function diff ได้ และต่อเนื่องทุกจุด ทำให้เราสามารถใช้ calculus ได้อย่างเต็มที่



# Cost Function and Cost Landscape

3. Cost function ที่นิยามแบบ SSE เป็น convex function และมีจุดต่ำสุดเพียงจุดเดียว

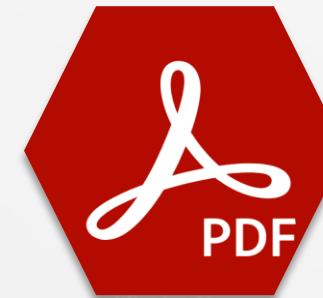


กราฟของ cost landscape โดยที่ cost function เป็น SSE

# Cost Function and Cost Landscape



Cost Landscape Plotting



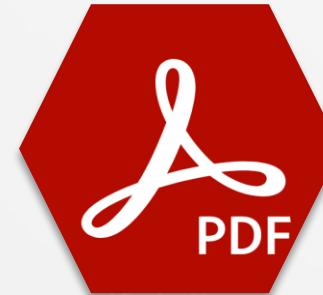
Open File

**CostLandscape\_Plotting.pdf**

# Cost Function and Cost Landscape



Cost Function



Open File

**Costfunction\_xxxx.pdf**

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



**Further Reading**



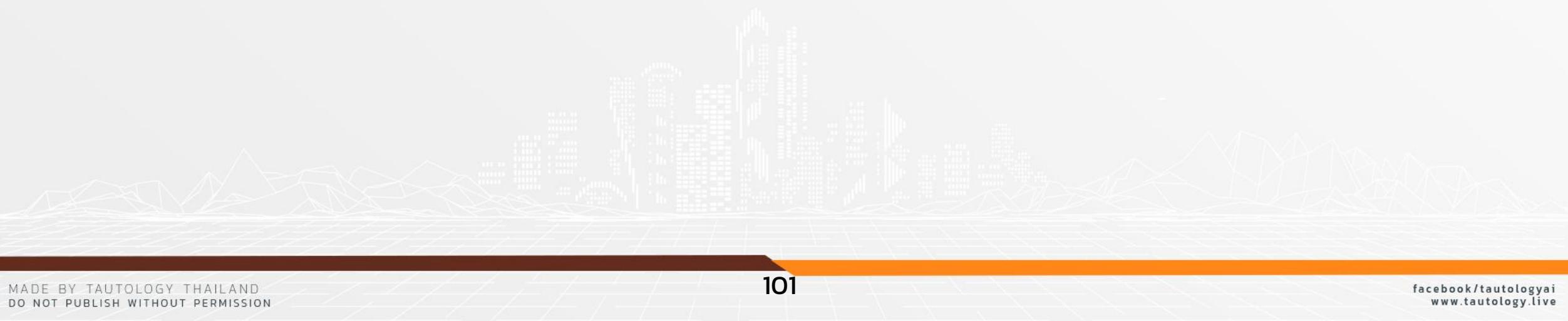
# How to Create Model (Math)



“ เราต้องการหา  $w_0, w_1, w_2, \dots, w_p$  ที่ทำให้ cost function น้อยที่สุด ”

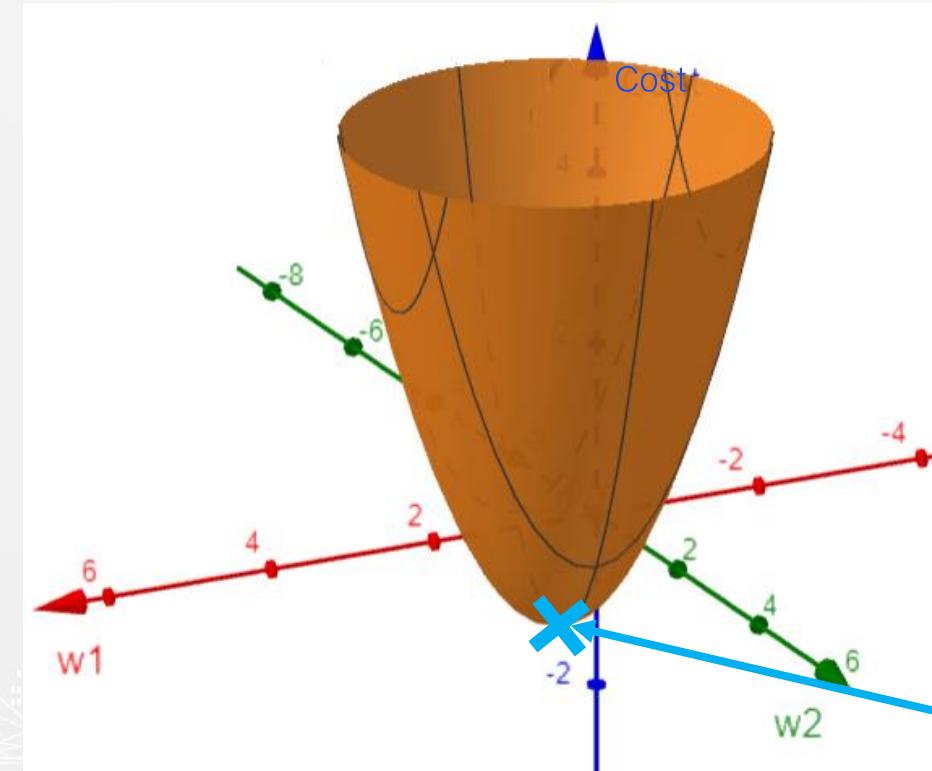
# How to Create Model (Math)

- Least Squares Method
- Calculation Example



# Least Squares Method

เป้าหมาย เราต้องการจุดต่ำสุดของ cost function ซึ่งสามารถใช้ calculus ในการหาได้



สามารถใช้ calculus  
เพื่อหาจุดต่ำสุดได้เลย

# Least Squares Method

วิธีการที่เราจะใช้ในการหาค่าตอบมีเช่นว่า

**“Least Squares Method”**

# Least Squares Method

จากคุณสมบัติของจุดต่ำสุดคือ

“ gradient ของ cost function เท่ากับ 0 ( $\nabla Cost = 0$ ) ”

ซึ่งหมายความว่า ความชันของ cost function ในมิติใด ๆ เท่ากับ 0

$$\frac{\partial Cost}{\partial w_d} = 0 ; d = 0, \dots, p$$

# Least Squares Method

การกี่ความชันของ  $cost$  ในมิติใด ๆ เท่ากับ 0 หรือ  $\frac{\partial Cost}{\partial w_d} = 0 ; d = 0, \dots, p$   
สามารถเขียนเป็นสมการได้ดังต่อไปนี้

$$\frac{\partial Cost}{\partial w_0} = 0$$

$$\frac{\partial Cost}{\partial w_1} = 0$$

⋮

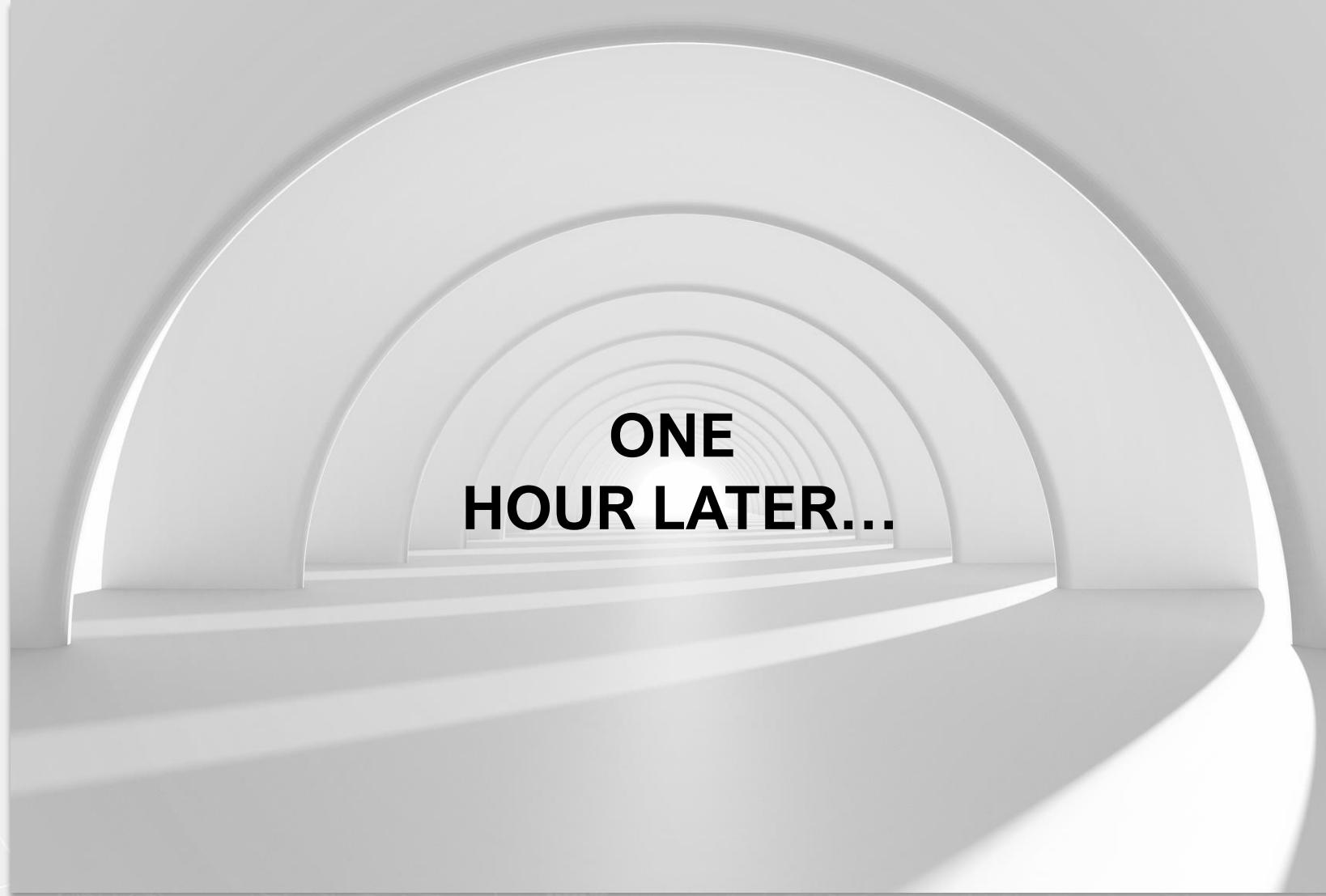
$$\frac{\partial Cost}{\partial w_p} = 0$$

# Least Squares Method

**ดังนี้**  $\nabla Cost = 0$  สามารถเขียนใหม่ได้ดังนี้

$$\nabla Cost = \begin{bmatrix} \frac{\partial Cost}{\partial w_0} \\ \frac{\partial Cost}{\partial w_1} \\ \vdots \\ \frac{\partial Cost}{\partial w_p} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

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



# Least Squares Method

**Normal Equation** คือ สมการที่ใช้ในการหา weight ของ model

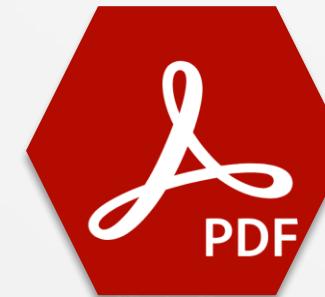
$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$

โดยที่  $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix}$ ,  $X_b = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{bmatrix}$ ,  $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

# Least Squares Method



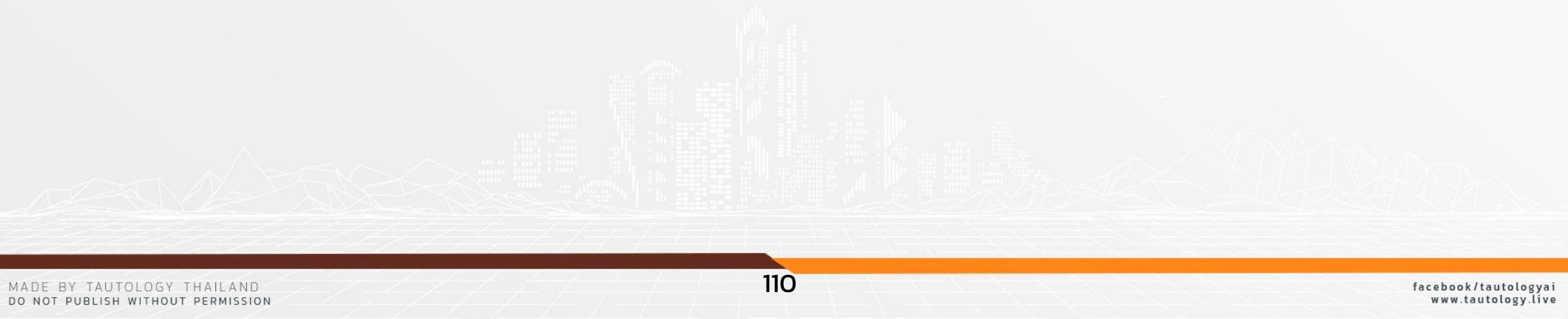
Derivation of Normal Equation



Open File  
**Derive\_NormalEq.pdf**

# How to Create Model (Math)

- Least Squares Method**
- Calculation Example



# Calculation Example

**ตัวอย่างการคำนวณ  $w$  ด้วย normal equation**

$x_1$	$x_2$	$y$
0	1	4
2	1	8
1	1	6
2	0	5

ตารางแสดง toy dataset

# Calculation Example

- จากข้อมูลใน dataset เราสามารถเขียน  $X, y$  และ  $X_b$  ได้ดังต่อไปนี้

$$X = \begin{bmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 1 \\ 2 & 0 \end{bmatrix}, \quad y = \begin{bmatrix} 4 \\ 8 \\ 6 \\ 5 \end{bmatrix} \quad \text{และ} \quad X_b = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \\ 1 & 2 & 0 \end{bmatrix}$$

# Calculation Example

- จากสูตร normal equation  $\mathbf{w} = (\mathbf{X}_b^T \mathbf{X}_b)^{-1} \mathbf{X}_b^T \mathbf{y}$  จะได้ว่า

$$\mathbf{w} = \left( \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 1 & 2 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \\ 1 & 2 & 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 1 & 2 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ 8 \\ 6 \\ 5 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 5 & 3 \\ 5 & 9 & 3 \\ 3 & 3 & 3 \end{bmatrix}^{-1} \begin{bmatrix} 23 \\ 32 \\ 18 \end{bmatrix}$$

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

# Calculation Example

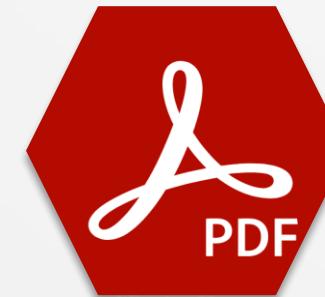
ดังนั้น เราจะได้ model ของ linear regression สำหรับข้อมูลชุดนี้คือ

$$\hat{y} = 1 + 2x_1 + 3x_2$$

# Calculation Example



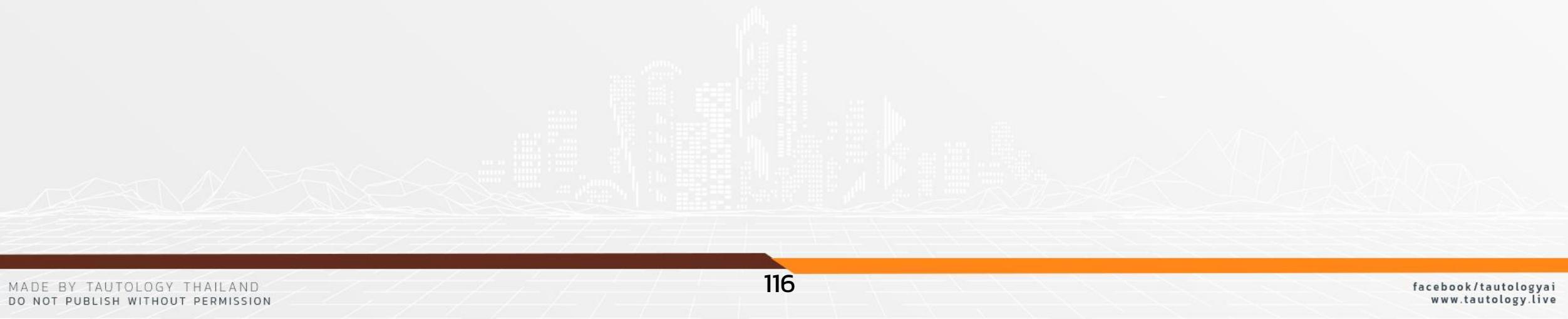
Exercise of Normal Equation



Open File  
**Exercise\_NormalEq.pdf**

# How to Create Model (Math)

- Least Squares Method**
- Calculation Example**



# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



**Further Reading**



# How to Create Model (Code)

ตัวอย่าง code สำหรับคำนวณ  $w$

$x_1$	$x_2$	$y$
0	1	4
2	1	8
1	1	6
2	0	5

ตารางแสดง toy dataset

# How to Create Model (Code)

- Code สำหรับสร้าง model จากข้อมูลของเราโดยที่

$$X = \begin{bmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 1 \\ 2 & 0 \end{bmatrix}, \quad y = \begin{bmatrix} 4 \\ 8 \\ 6 \\ 5 \end{bmatrix}$$

```
In [1]: 1 reg = LinearRegression()
          2 reg.fit(X, y)
```

Out[1]: LinearRegression()

# How to Create Model (Code)

- ค่า  $w_0$  จะเก็บไว้ใน attribute ชื่อ intercept\_

```
In [2]: 1 reg.intercept_
Out[2]: 1.
```

# How to Create Model (Code)

- ค่า  $w_1, \dots, w_p$  จะเก็บไว้ใน attribute ชื่อ `coef_`

In [3]: ► 1 reg.coef\_

Out[3]: array([2., 3.])

# How to Create Model (Code)

ดังนั้น เราจะได้  $w_0 = 1, w_1 = 2$  และ  $w_2 = 3$  ซึ่งสามารถเขียนเป็น model ของ linear regression สำหรับข้อมูลชุดนี้ได้ดังนี้

$$\hat{y} = 1 + 2x_1 + 3x_2$$

# How to Create Model (Code)

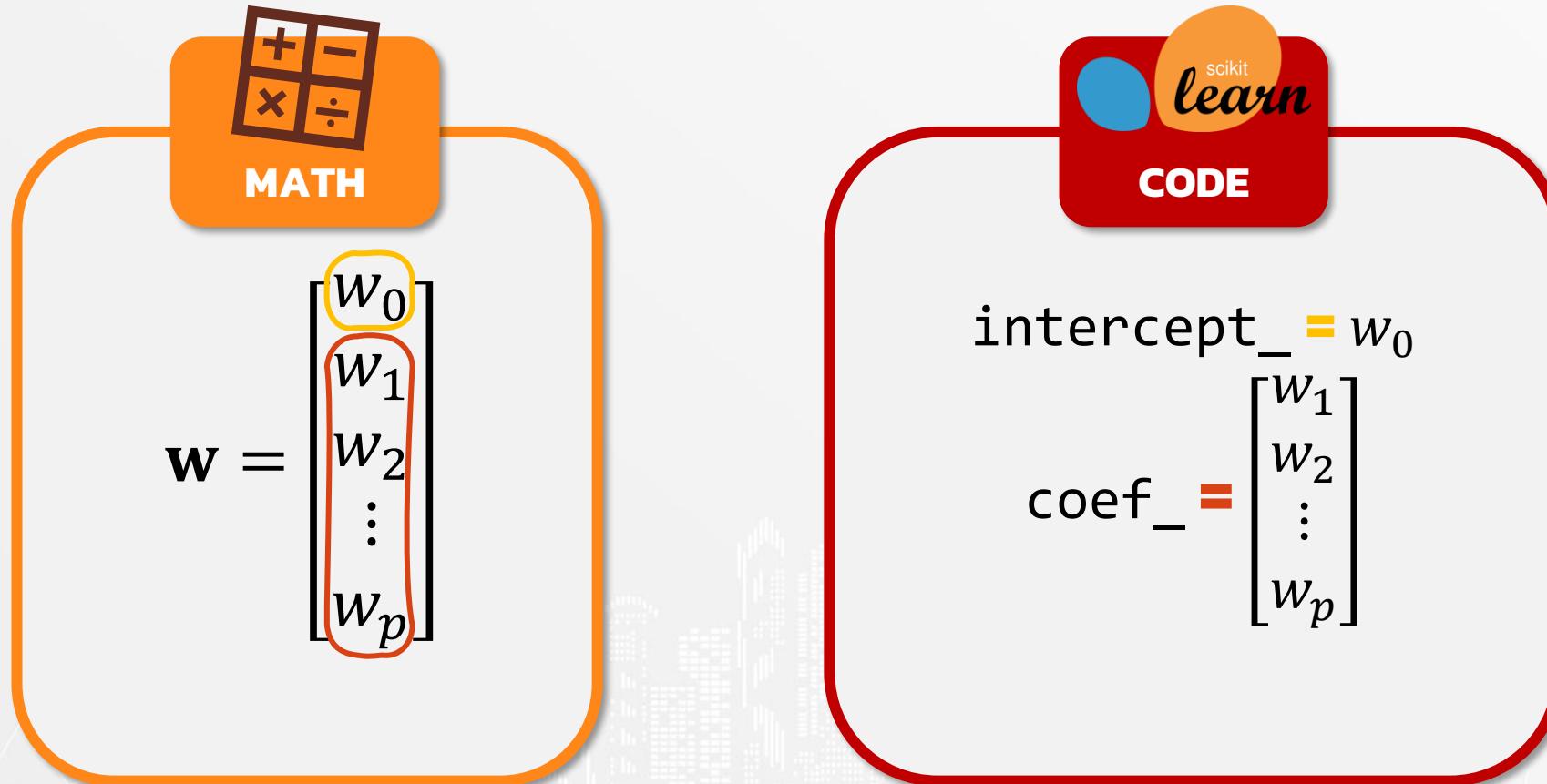


Code for this section

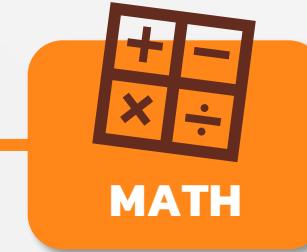


Open File  
**Model Creation.ipynb**

# How to Create Model (Code)



# How to Create Model (Code)



MATH

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$



CODE

```
In [1]: 1 reg = LinearRegression()  
2 reg.fit(X, y)
```

```
Out[1]: LinearRegression()
```

```
In [2]: 1 reg.intercept_
```

```
Out[2]: 1.
```

```
In [3]: 1 reg.coef_
```

```
Out[3]: array([2., 3.])
```

# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**

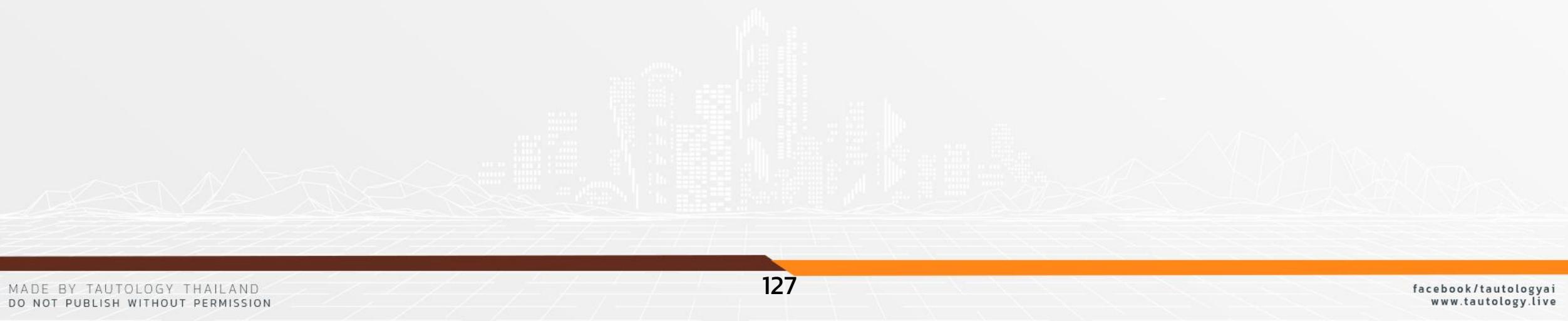


**Further Reading**

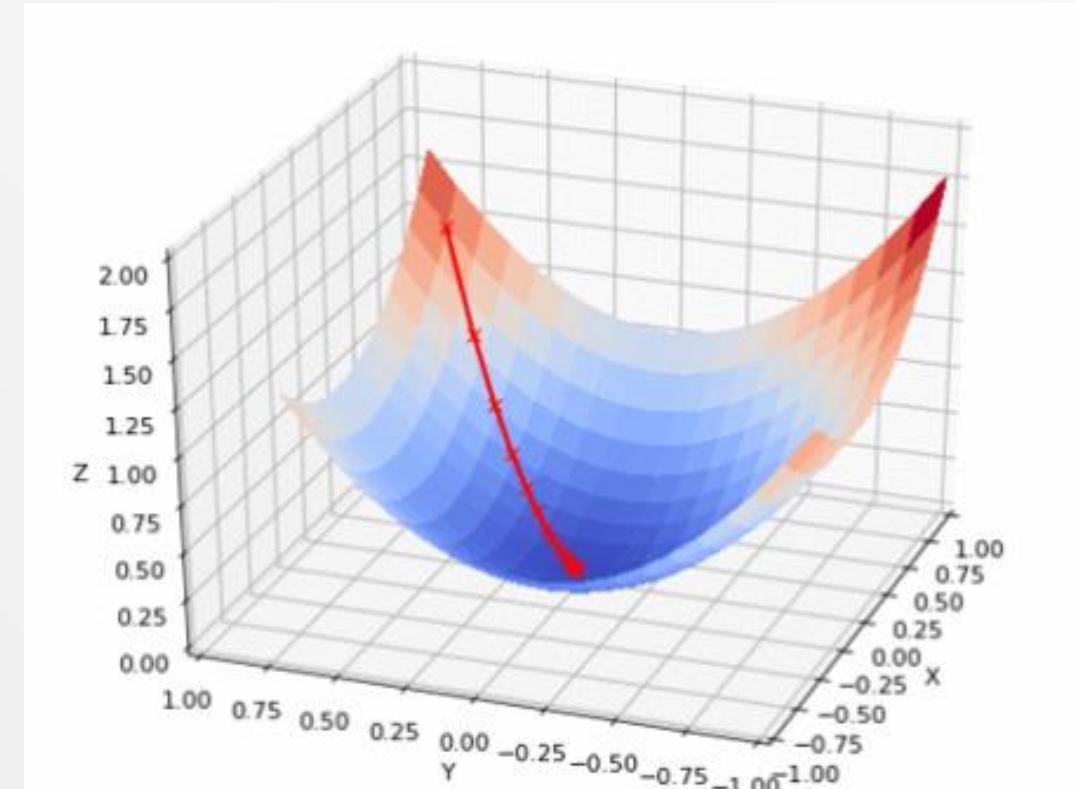


# Further Reading

- Gradient Descent
- Moore-Penrose pseudo inverse
- Feature importance with p-value



# Gradient Descent

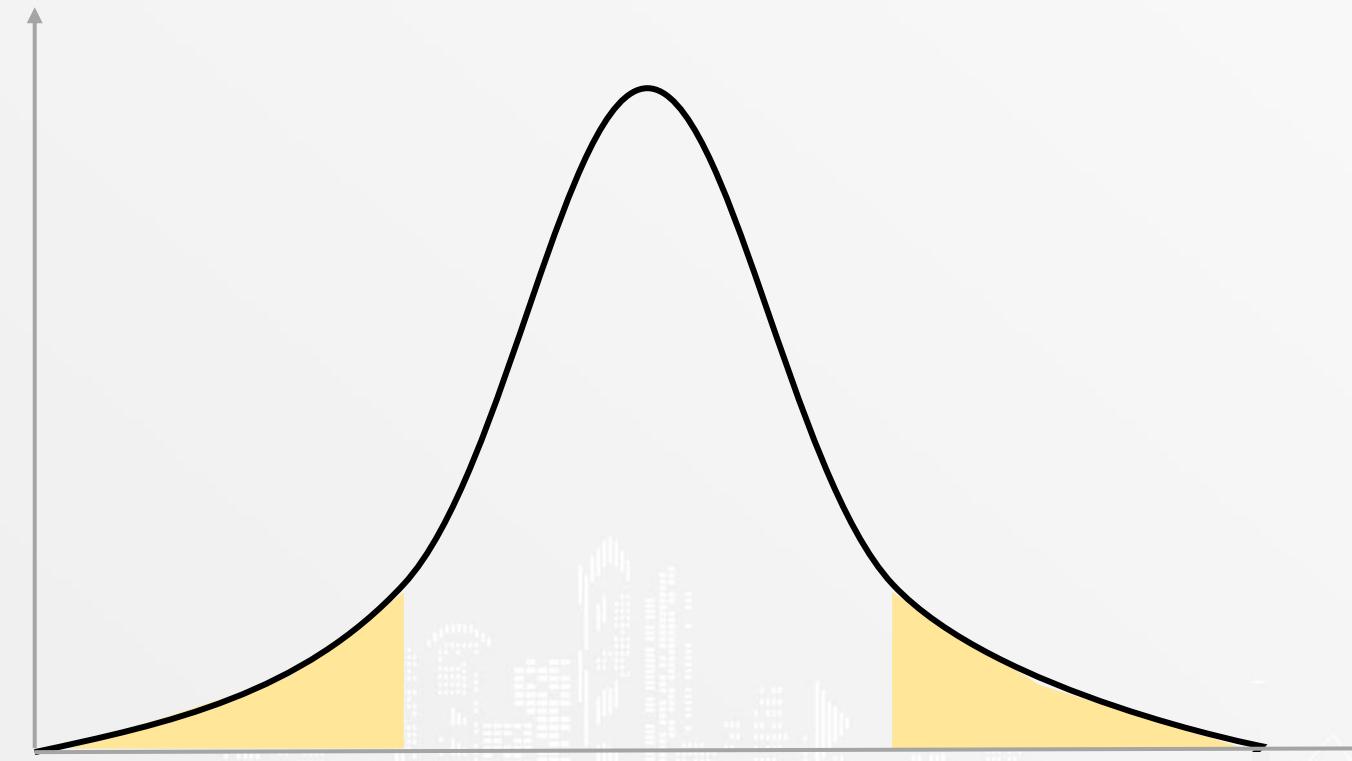


# Moore-Penrose pseudo inverse

“ Normal Equation ”

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$

# Feature importance with p-value



# Model

**Assumption**



**Real Face of the Model**



**Cost Function and Cost Landscape**



**How to Create Model (Math)**



**How to Create Model (Code)**



**Further Reading**



# Model Creation





# Prediction

# Prediction

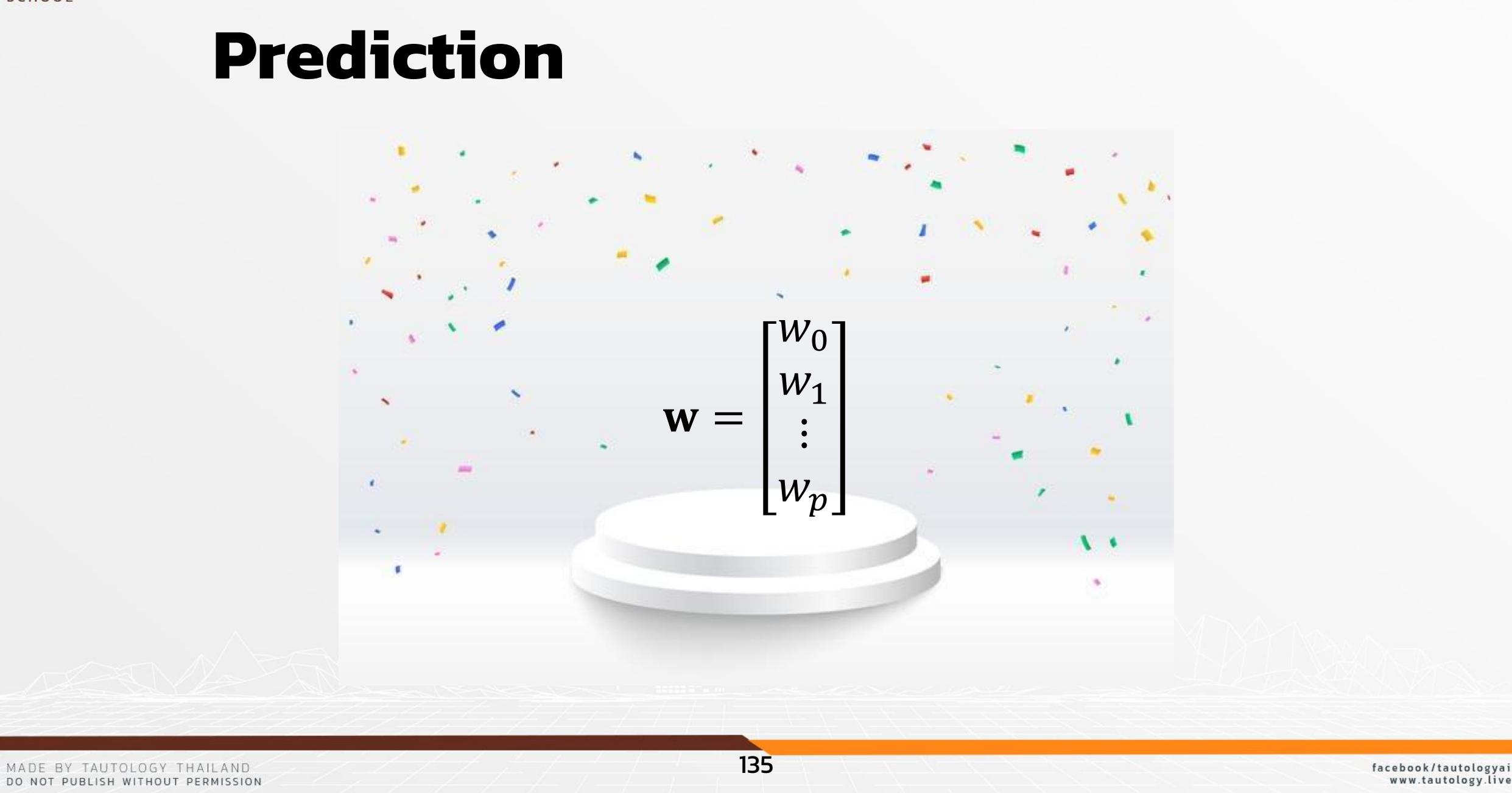
Linear regression คือ สมการเชิงเส้นที่ใช้ตัวแปรตัวนเพื่อพยากรณ์ตัวแปรตาม

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

- โดย
- ◆  $\hat{y}$  คือ ค่าพยากรณ์ของตัวแปรตาม (predicted target)
  - ◆  $x_1, x_2, x_3, \dots, x_p$  คือ ตัวแปรตัวน (feature)
  - ◆  $w_0, w_1, w_2, \dots, w_p$  คือ สัมประสิทธิ์ (coefficient)

# Prediction

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix}$$

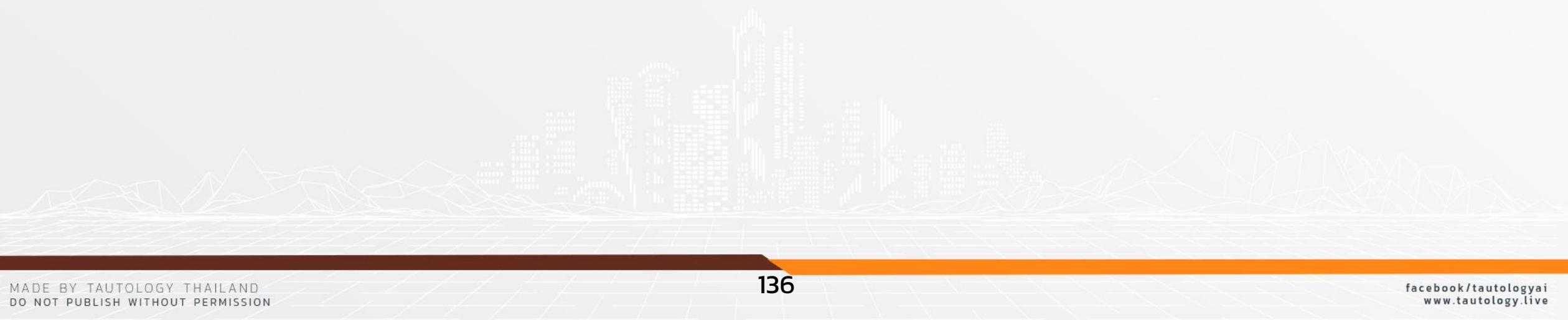


# Prediction

1-Sample

Multi-Sample

Code



# 1-Sample

ตัวอย่างการคำนวณ  $\hat{y}$

$x_1$	$x_2$
3	1



$\hat{y}$
?

# 1-Sample

สมมติว่า  $w$  ของปัญหานี้ที่เราหามาได้คือ

$$w = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

# 1-Sample

ซึ่งทำให้สามารถเขียนสมการ  $\hat{y}$  ได้ดังต่อไปนี้

$$\begin{aligned}\hat{y} &= 1 + 2x_1 + 3x_2 \\ &= 1 + 2(3) + 3(1) \\ &= 10\end{aligned}$$

# 1-Sample

ดังนั้น เราจะได้  $\hat{y}$  ของข้อมูลชุดนี้คือ

$x_1$	$x_2$
3	1



$\hat{y}$
10

# Prediction

**1-Sample**



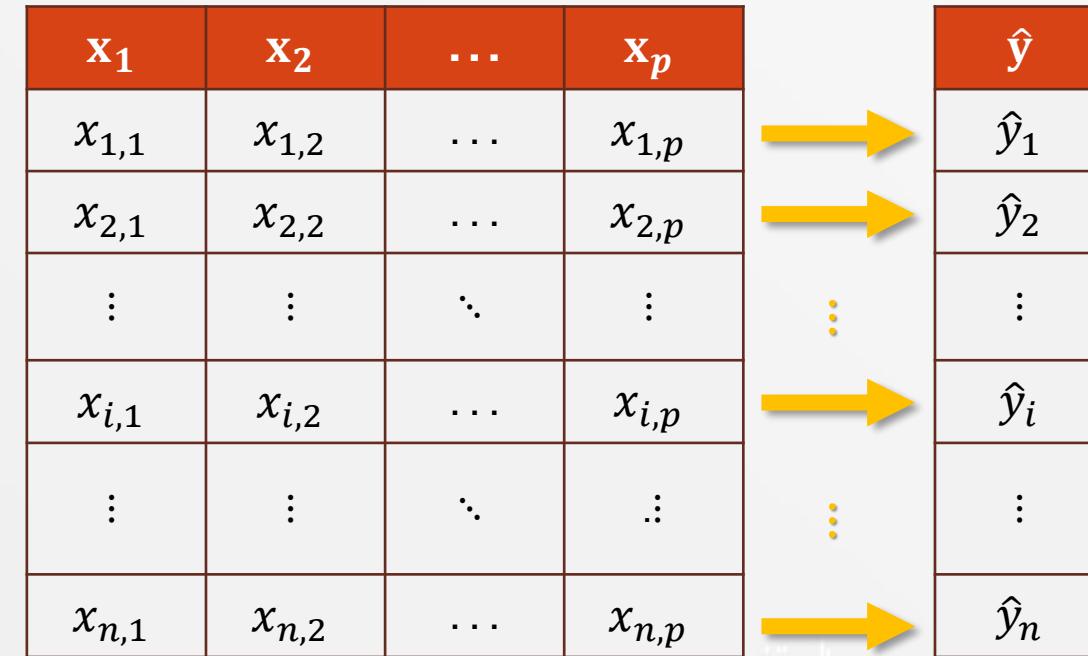
Multi-Sample



Code



# Multi-Sample



# Multi-Sample

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



$$\hat{y}_1 = w_0 + w_1 x_{1,1} + w_2 x_{1,2} + \cdots + w_p x_{1,p}$$

$$\hat{y}_2 = w_0 + w_1 x_{2,1} + w_2 x_{2,2} + \cdots + w_p x_{2,p}$$

⋮

$$\hat{y}_i = w_0 + w_1 x_{i,1} + w_2 x_{i,2} + \cdots + w_p x_{i,p}$$

⋮

$$\hat{y}_n = w_0 + w_1 x_{n,1} + w_2 x_{n,2} + \cdots + w_p x_{n,p}$$

# Multi-Sample

เพื่อให้สอดคล้องกับ format ของ data เราสามารถเขียนให้อยู่ในรูปของ matrix ได้ดังนี้

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_{1,1} + w_2 x_{1,2} + \cdots + w_p x_{1,p} \\ w_0 + w_1 x_{2,1} + w_2 x_{2,2} + \cdots + w_p x_{2,p} \\ \vdots \\ w_0 + w_1 x_{n,1} + w_2 x_{n,2} + \cdots + w_p x_{n,p} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_p \end{bmatrix}$$

# Multi-Sample

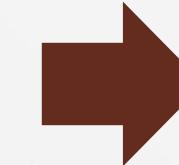
$$\hat{\mathbf{y}} = X_b \mathbf{w}$$

โดยที่  $\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix}$ ,  $X_b = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{bmatrix}$ ,  $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix}$

# Multi-Sample

ตัวอย่างการคำนวณ  $\hat{y}$

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



$\hat{y}$
?
?
?
?

# Multi-Sample

- สมมติว่า  $w$  ของปัจจุบันนี้ที่เรามาได้คือ

$$w = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

- และจากข้อมูลใน dataset เราสามารถเขียน  $X_b$  ได้ดังต่อไปนี้

$$X_b = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 3 & 1 \\ 1 & 3 & 0 \end{bmatrix}$$

# Multi-Sample

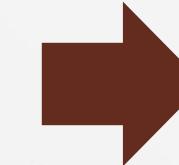
เราคำนวณค่า  $\hat{y}$  ได้จาก  $\hat{y} = X_b w$

$$\begin{aligned}\hat{y} &= \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 3 & 1 \\ 1 & 3 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \\ &= \begin{bmatrix} (1 \times 1) + (2 \times 1) + (3 \times 1) \\ (1 \times 1) + (2 \times 2) + (3 \times 0) \\ (1 \times 1) + (2 \times 3) + (3 \times 1) \\ (1 \times 1) + (2 \times 3) + (3 \times 0) \end{bmatrix} \\ &= \begin{bmatrix} 6 \\ 5 \\ 10 \\ 7 \end{bmatrix}\end{aligned}$$

# Multi-Sample

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

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



$\hat{y}$
6
5
10
7

# Prediction

**1-Sample**



**Multi-Sample**



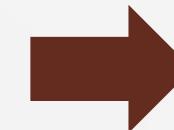
**Code**



# Code

ตัวอย่าง code สำหรับหา  $\hat{y}$

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



$\hat{y}$
?
?
?
?

ตารางแสดง toy dataset

# Code

- Code สำหรับหา ŷ จากข้อมูลของเรา โดยที่  $X = \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ 3 & 1 \\ 3 & 0 \end{bmatrix}$

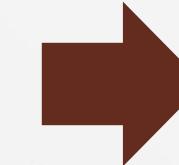
```
In [4]: ► 1 reg.predict(X)
```

```
Out[4]: array([ 6.,  5., 10.,  7.])
```

# Code

ดังนั้น เราจะได้  $\hat{y}$  สำหรับข้อมูลชุดนี้คือ

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



$\hat{y}$
6
5
10
7

# Code

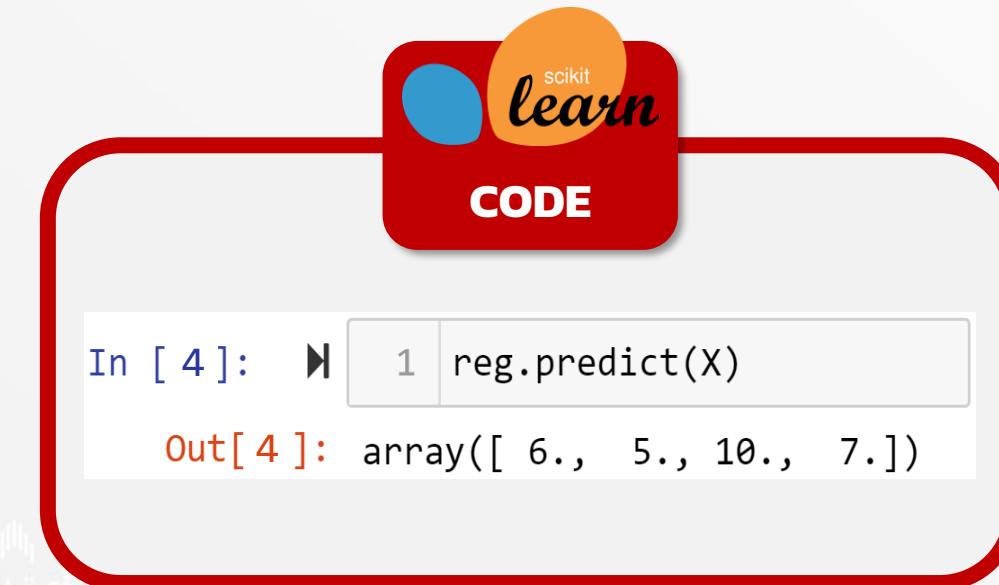
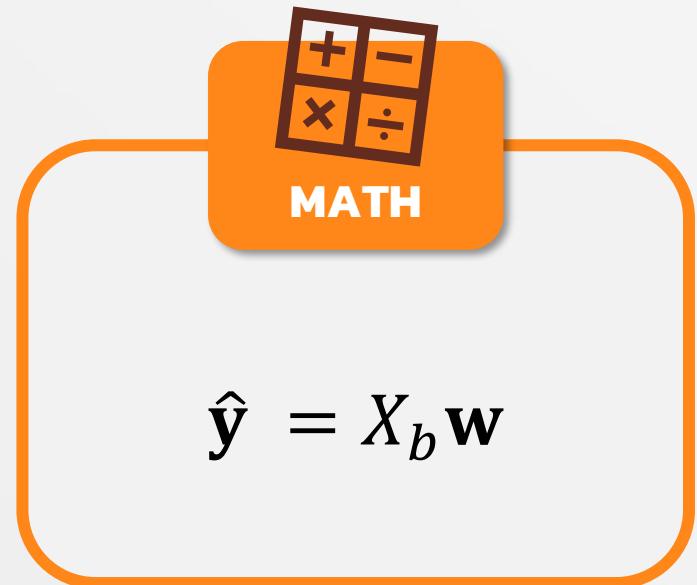


Code for this section



Open File  
**Model Creation.ipynb**

# Code



# Prediction

**1-Sample**



**Multi-Sample**



**Code**



# Model Creation

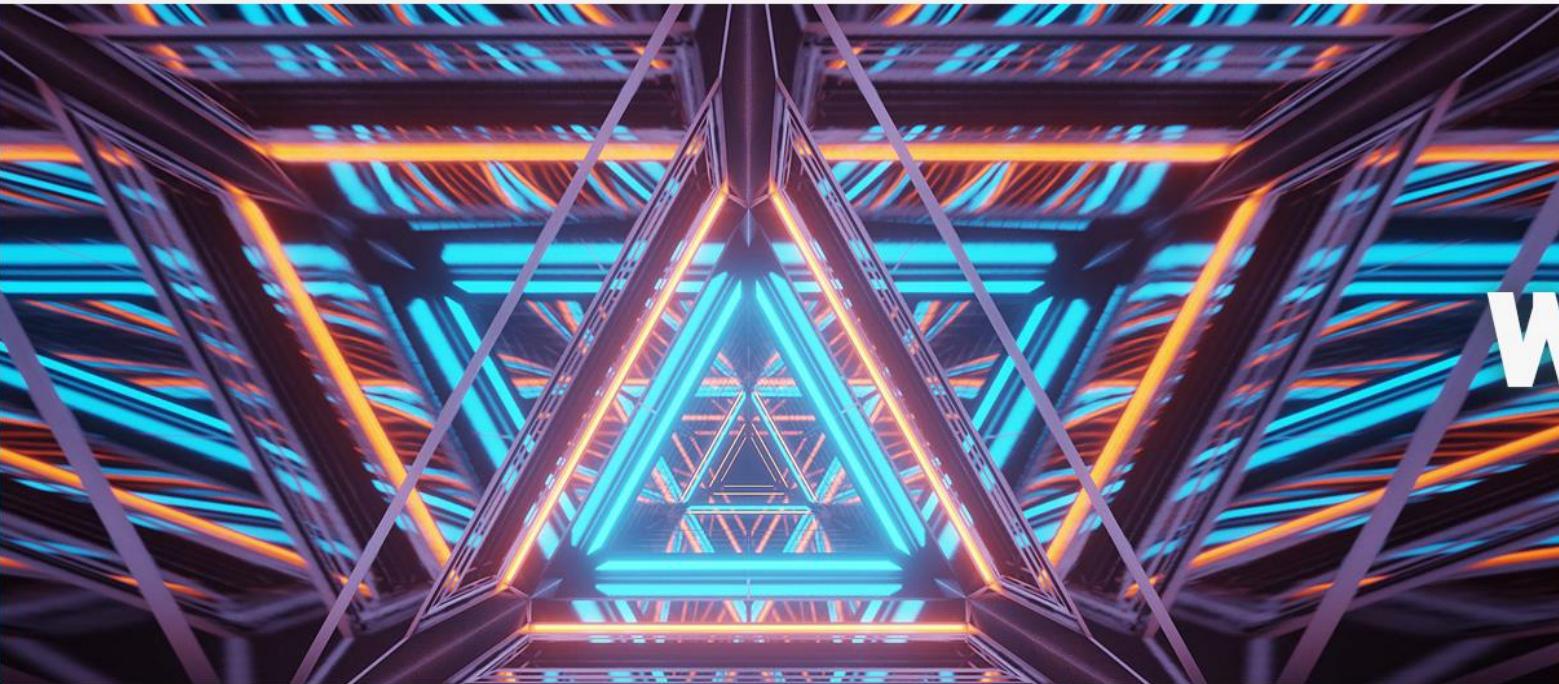


# DL101 : Linear Regression



TAUTOLOGY  
INNOVATION  
SCHOOL

BASIC WORKSHOP



# BASIC WORKSHOP

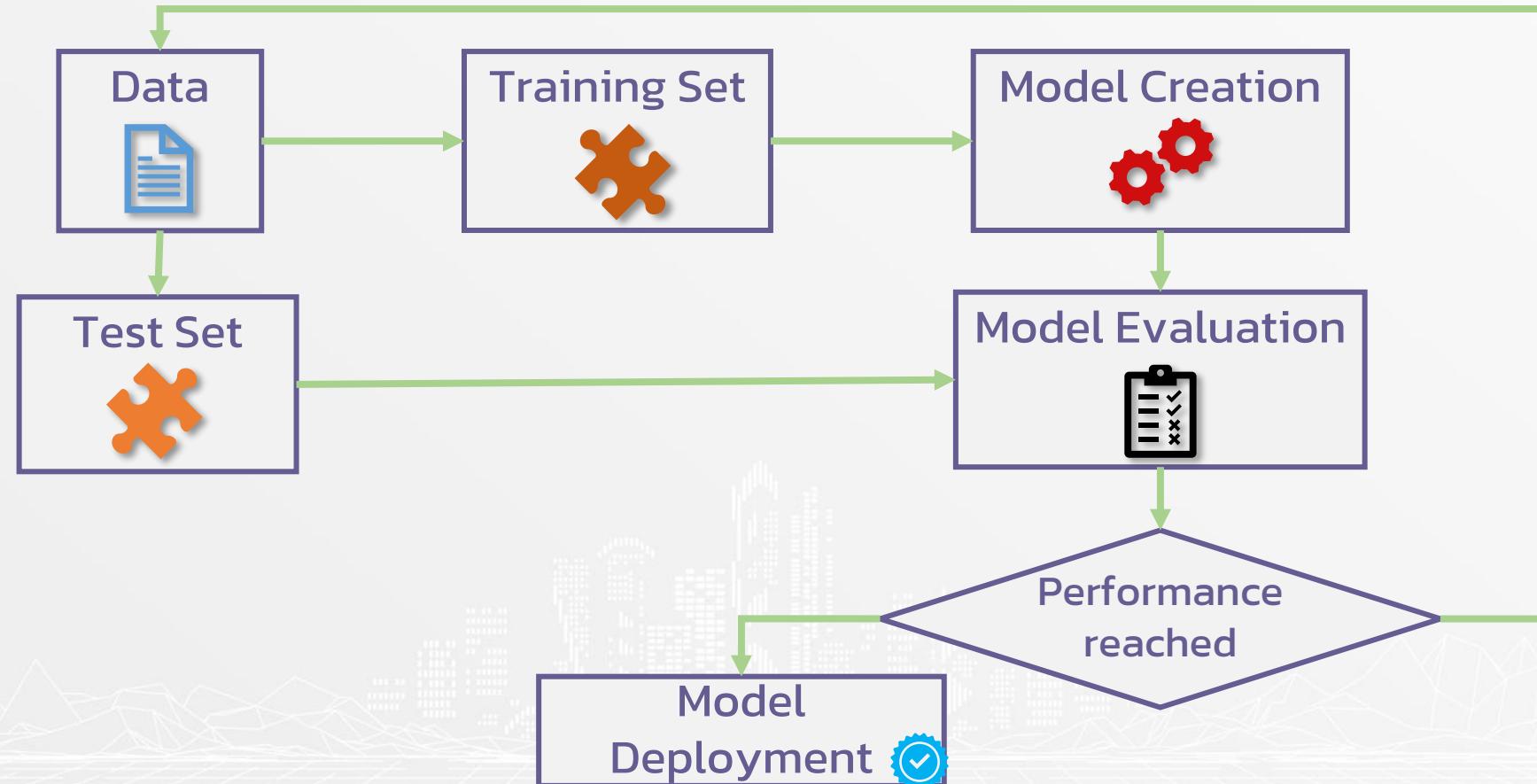
BY TAUTOLOGY

MADE BY TAUTOLOGY THAILAND  
DO NOT PUBLISH WITHOUT PERMISSION

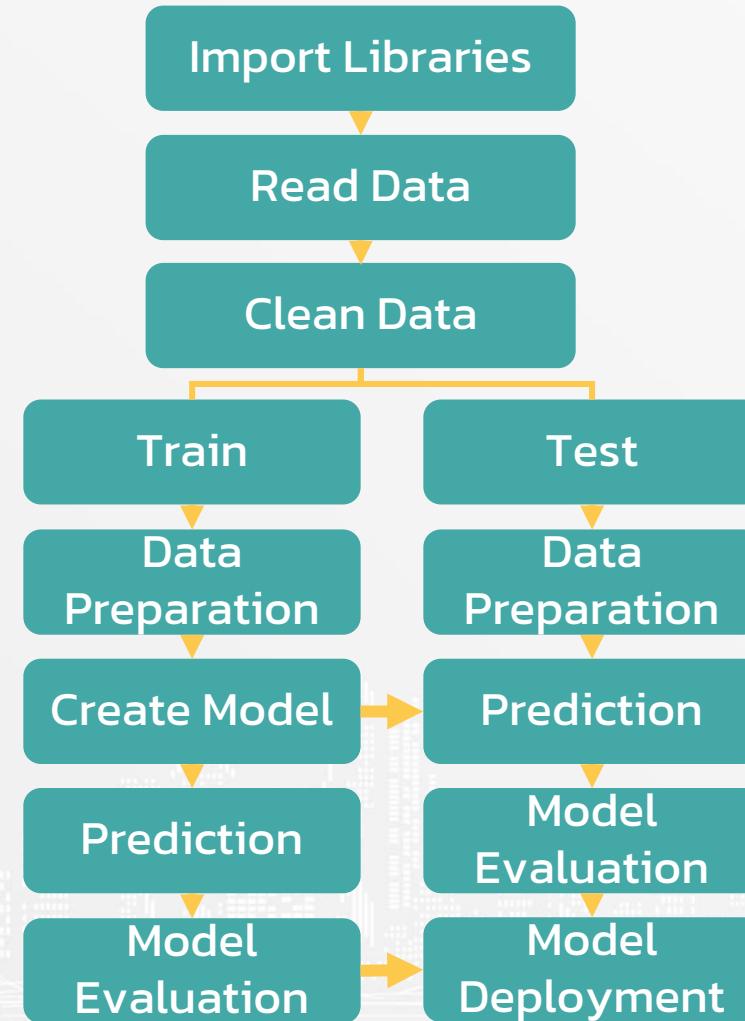
facebook/tautologyai  
www.tautology.live

TAUTOLOGY

# Supervised Learning Workflow



# Code Pipeline



Import Libraries

Read Data

Clean Data

Train

Test

Data Preparation

Data Preparation

Create Model

Prediction

Prediction

Model Evaluation

Model Evaluation

Model Deployment

# Import Libraries

1



2



3



4



# Code

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 from sklearn.model_selection import train_test_split
6 from sklearn.preprocessing import OrdinalEncoder, OneHotEncoder
7 from sklearn.linear_model import LinearRegression
8 from sklearn.metrics import (r2_score,
9                             mean_squared_error,
10                            mean_absolute_error,
11                            mean_absolute_percentage_error)
12
13 import warnings
14 warnings.filterwarnings('ignore')
15
16 np.random.seed(12345)
```

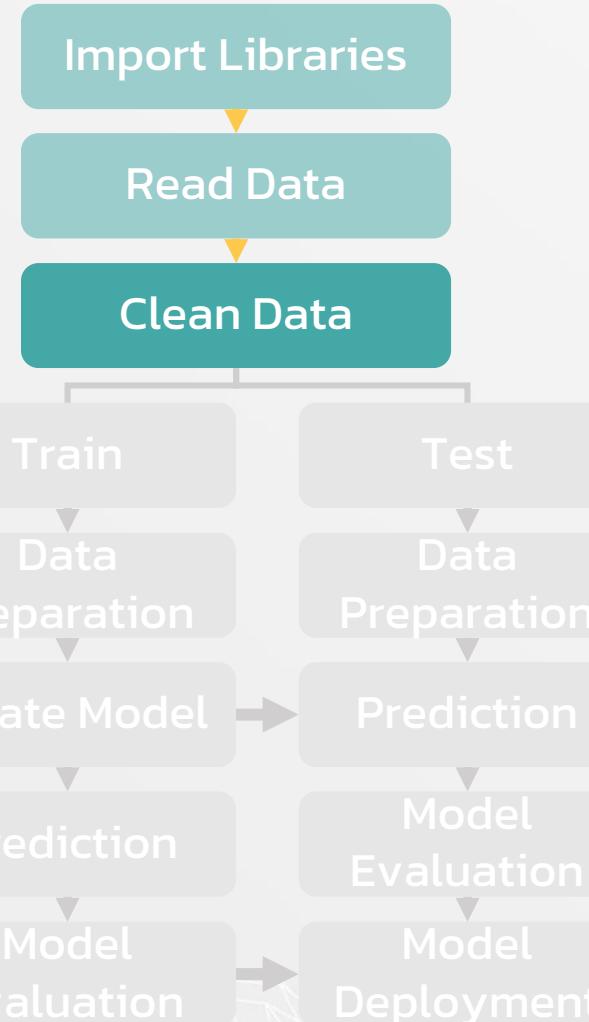


# Read Data

	age	experience	gpa	degree	position	salary
0	30.0	7.0	3.94	bachelor	engineer	32500.0
1	26.0	2.0	2.86	bachelor	NaN	22500.0
2	27.0	0.0	3.13	doctorate	secretary	37000.0
3	32.0	NaN	3.10	bachelor	engineer	24500.0
4	24.0	1.0	3.81	bachelor	accountant	23500.0
5	35.0	7.0	3.93	doctorate	secretary	43500.0
6	23.0	1.0	3.78	master	accountant	30500.0
7	32.0	8.0	3.04	bachelor	accountant	31500.0
8	27.0	2.0	3.52	bachelor	secretary	18500.0
	:	:	:	:	:	:

# Code

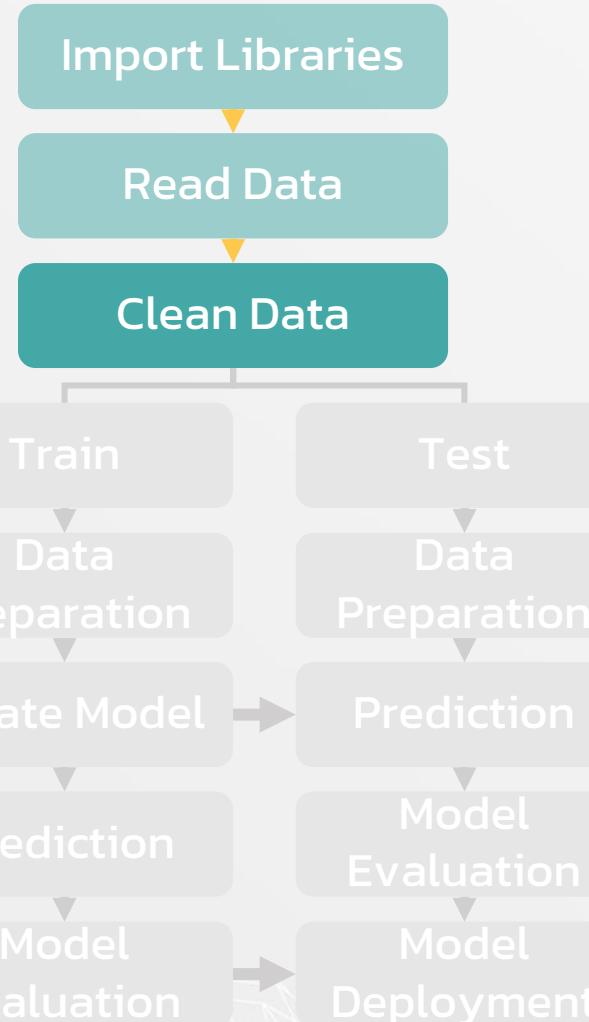
```
1 data = pd.read_csv('salary_dataset.csv')
```



# Clean Data

1. Handle Missing Values
2. Handle Outliers





# Clean Data

## 1. Handle Missing Values

## 2. Handle Outliers

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   age          89 non-null    float64
 1   experience  89 non-null    float64
 2   gpa          90 non-null    float64
 3   degree       90 non-null    object  
 4   position     89 non-null    object  
 5   salary       89 non-null    float64
dtypes: float64(4), object(2)
memory usage: 4.3+ KB
```

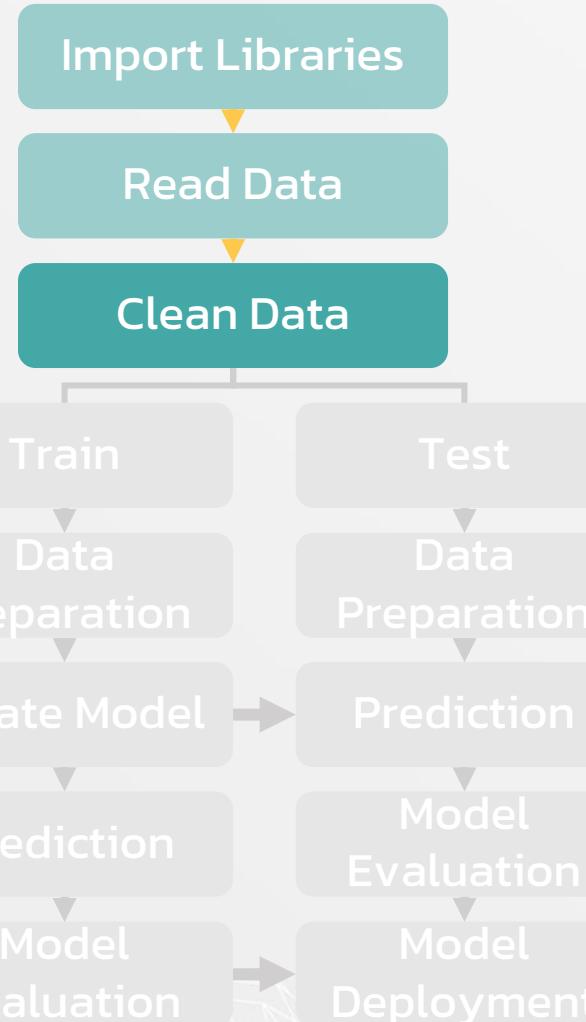
# Code

- Check Missing Values

```
1 data.info()
```

- Remove Missing Values

```
1 data.dropna(axis=0, inplace=True)
```



# Clean Data

1. Handle Missing Values
2. Handle Outliers

	age	experience	gpa	salary
count	86.000000	86.000000	86.000000	86.000000
mean	28.023256	3.848837	3.278605	31348.837209
std	4.408486	3.702201	0.528937	9255.227384
min	21.000000	0.000000	2.540000	13000.000000
25%	24.000000	1.000000	2.820000	24500.000000
50%	28.000000	3.000000	3.260000	30500.000000
75%	32.000000	7.000000	3.640000	37375.000000
max	35.000000	13.000000	5.880000	54000.000000

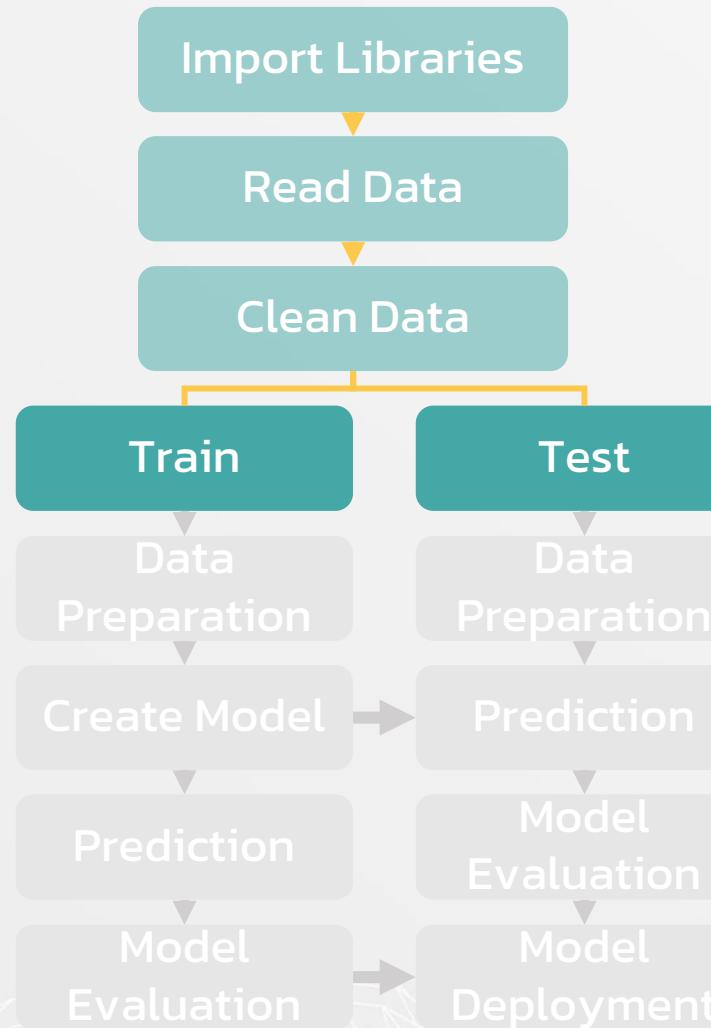
# Code

- Check Outliers

```
1 data.describe()
```

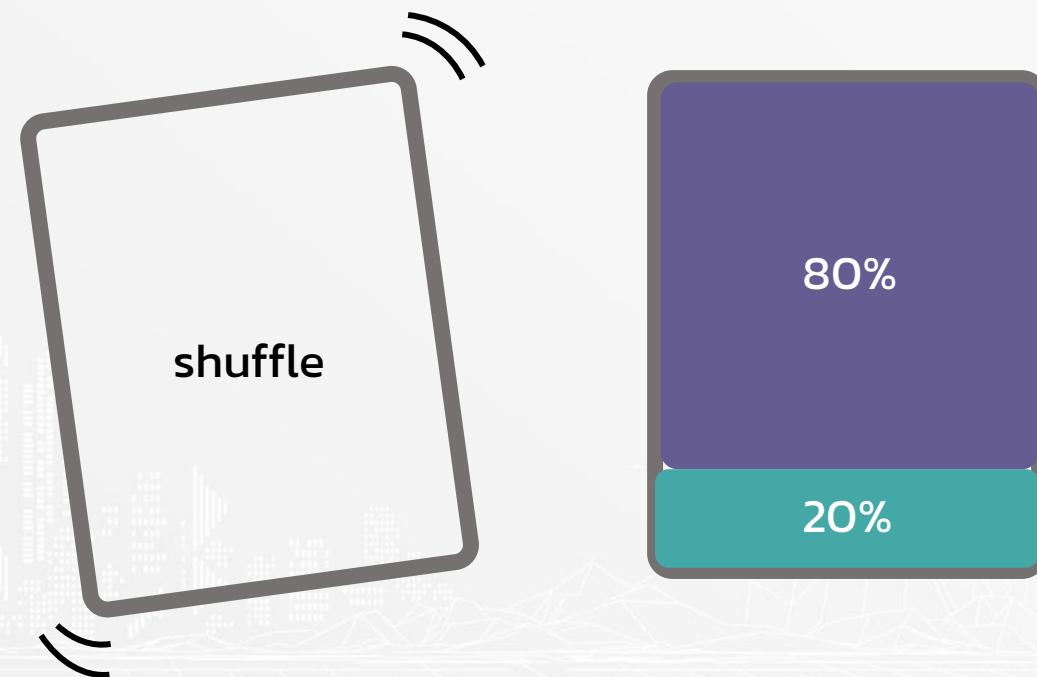
- Remove Outliers

```
1 _filter = data['gpa'] <= 4.00
2 data = data[_filter]
```



## Train/Test

แบ่งข้อมูลออกเป็น 2 ชุด คือ training set และ test set ด้วยอัตราส่วน 80:20 ตามลำดับ

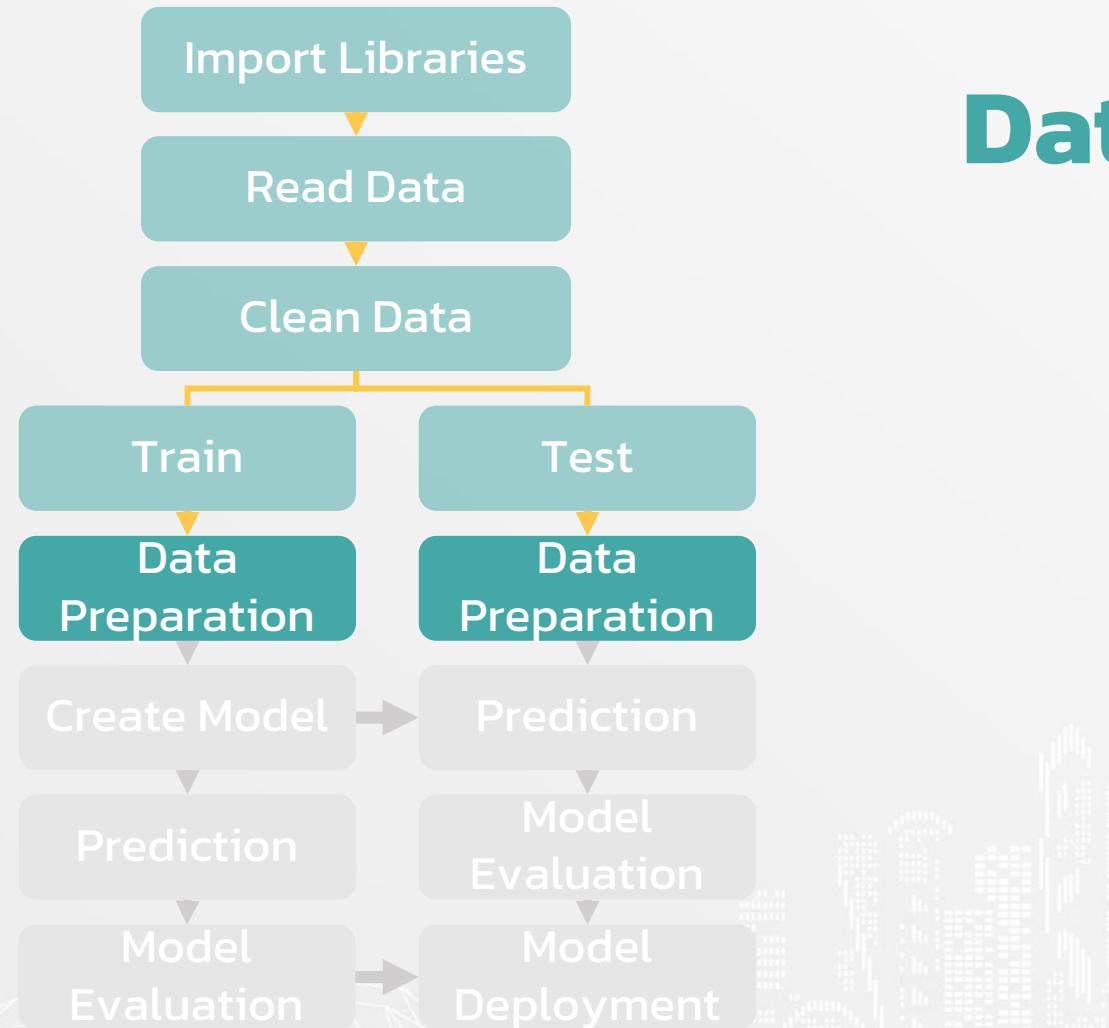


# Code

```
1 target_name = 'salary'  
2 feature_name = list(data.columns.drop(target_name))
```

```
1 X = data[feature_name]  
2 y = data[target_name]
```

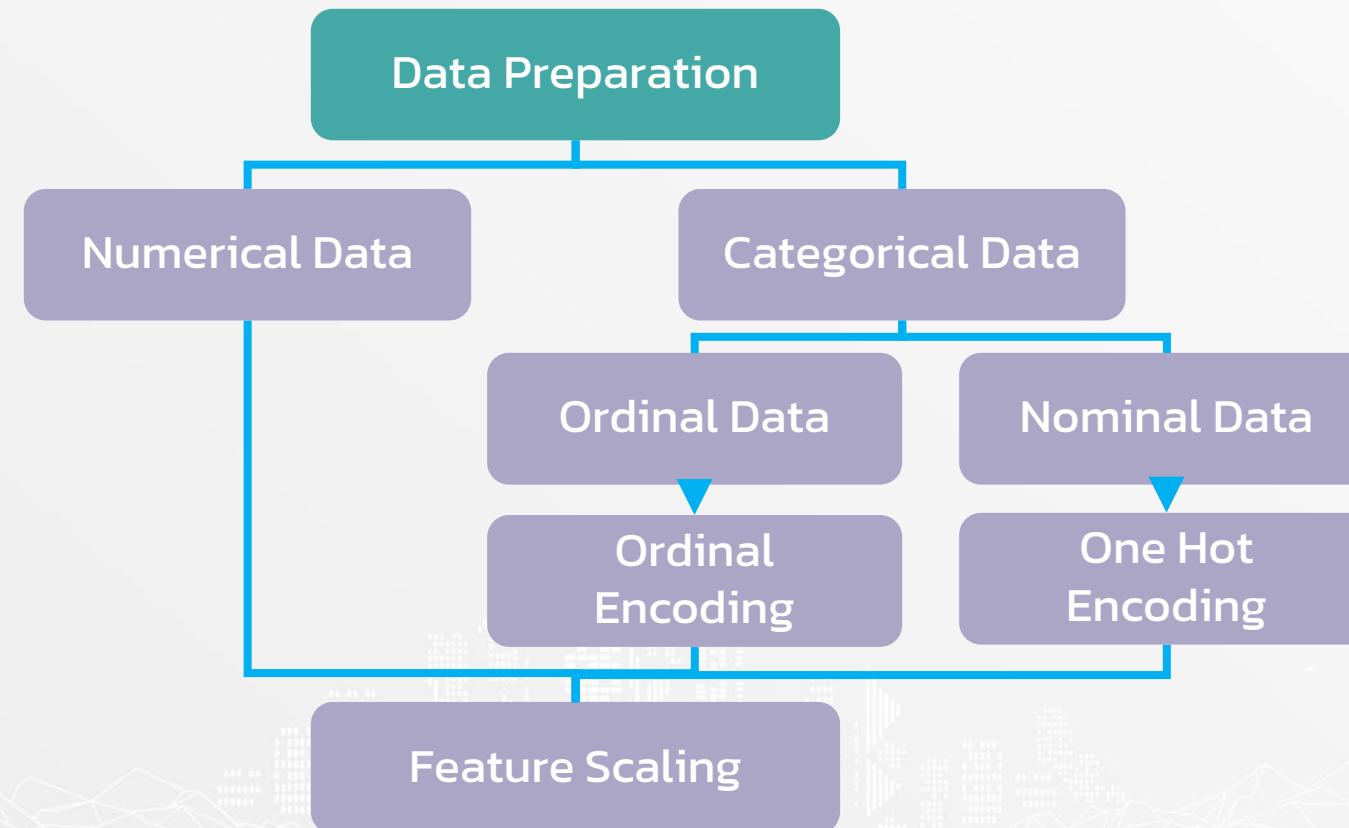
```
1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, shuffle=True)
```



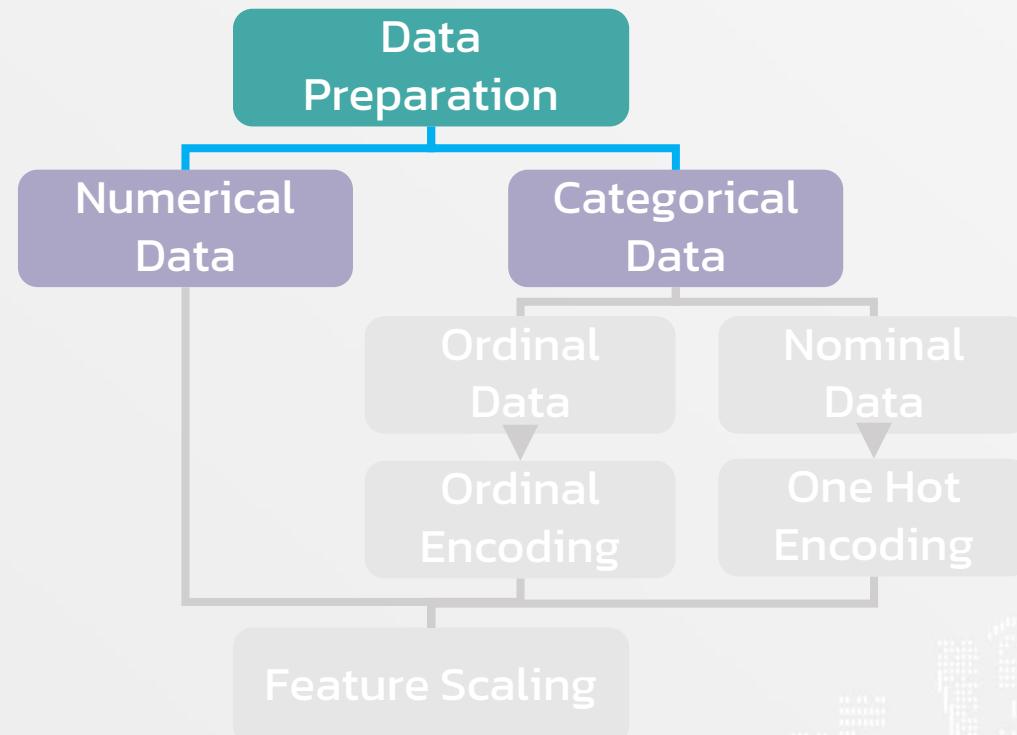
# Data Preparation



# Data Preparation



# Data Preparation



## Type of Features

พิจารณาและจำแนก feature ที่มีลักษณะข้อมูลแบบ numerical data และ categorical data ออกจากกัน

Numerical  
Data

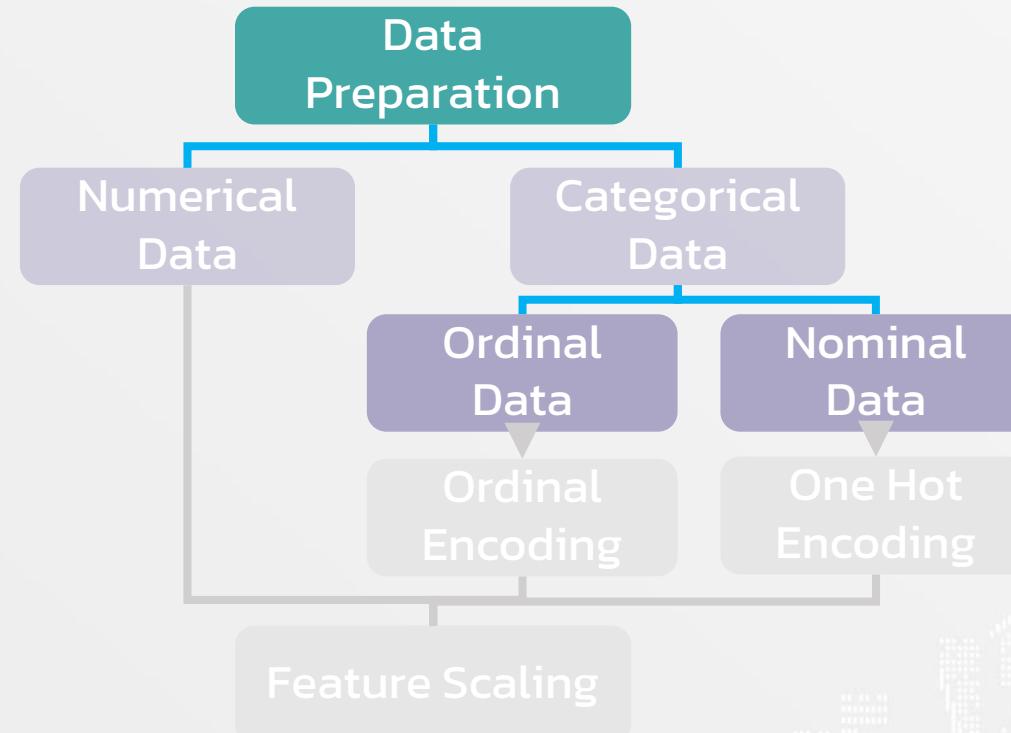
Categorical  
Data



# Code

```
1 numerical_feature = ['age', 'experience', 'gpa']  
2 categorical_feature = ['degree', 'position']
```

# Data Preparation



## Type of Categorical Features

พิจารณาและจำแนก feature ที่มีลักษณะข้อมูลแบบ ordinal data และ nominal data ออกจากกัน

```
degree : ['bachelor' 'doctorate' 'master']  
position : ['accountant' 'engineer' 'secretary']
```

# Code

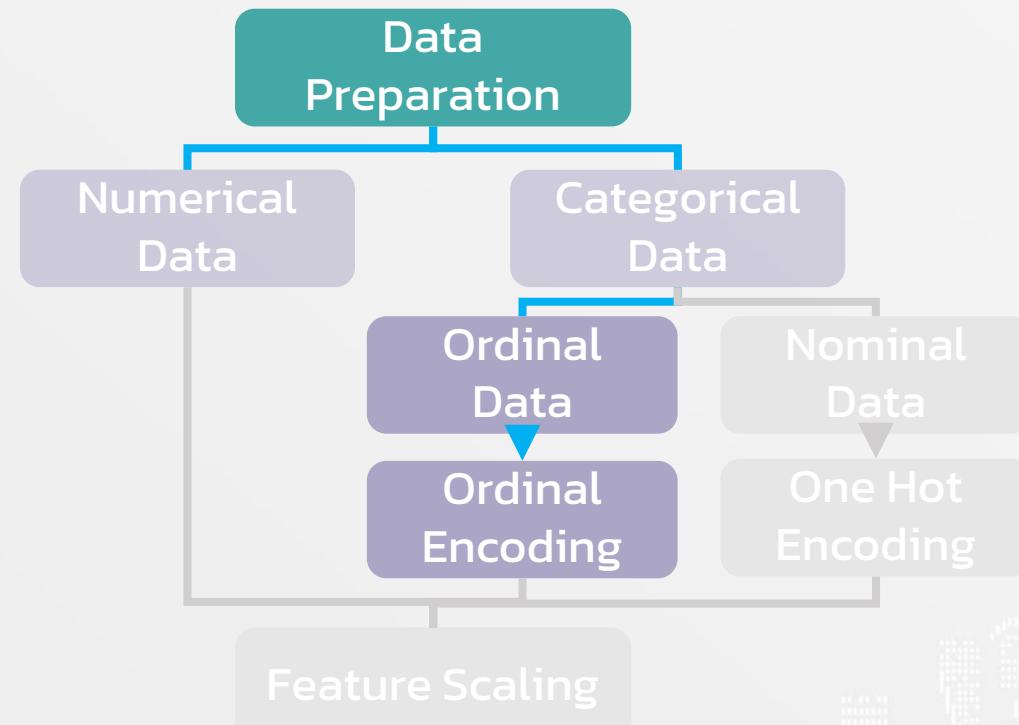
- Consider each Feature of Categorical Features

```
1 for feature in categorical_feature:  
2     print(feature, ':', np.unique(X_train[feature]))
```

- Classify into Ordinal Feature and Nominal Feature

```
1 ordinal_feature = ['degree']  
2 nominal_feature = ['position']
```

# Data Preparation



## Ordinal Encoding

การทำ ordinal encoding จะต้องทำแบบเดียวกันกับใน training set และ test set

degree
0 bachelor
1 master
2 bachelor
3 bachelor
4 bachelor

degree
0 0.0
1 1.0
2 0.0
3 0.0
4 0.0

# Code

- Ordinal Data

```
1 ordinal_feature = ['degree']
```

- Ordinal Encoding

```
1 categories = [
2     np.array(['bachelor', 'master', 'doctorate'], dtype=object)
3 ]
```

# Code

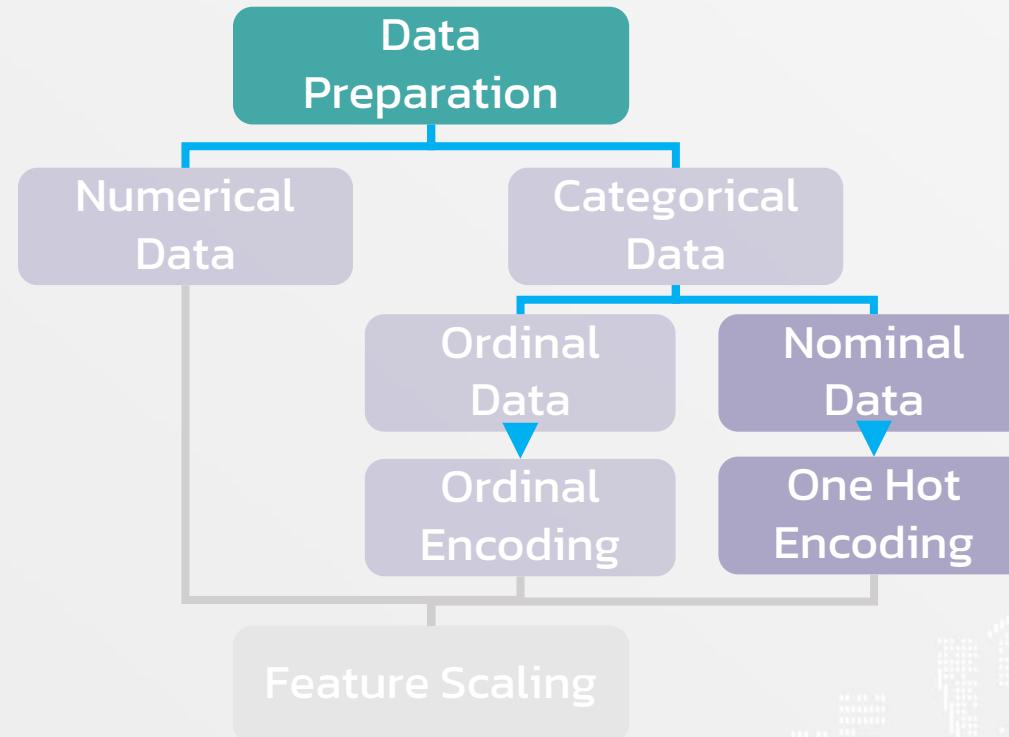
- Ordinal Encoding for **training set**

```
1 ordinal_encoder = OrdinalEncoder(categories=categories)
2 X_train[ordinal_feature] = ordinal_encoder.fit_transform(X_train[ordinal_feature])
```

- Ordinal Encoding for **test set**

```
1 X_test[ordinal_feature] = ordinal_encoder.transform(X_test[ordinal_feature])
```

# Data Preparation



## One Hot Encoding

การกำ one hot encoding จะต้องกำแบบเดียวกัน กันใน training set และ test set

position
0 secretary
1 secretary
2 engineer
3 engineer
4 secretary



	position_accountant	position_engineer	position_secretary
0	0.0	0.0	1.0
1	0.0	0.0	1.0
2	0.0	1.0	0.0
3	0.0	1.0	0.0
4	0.0	0.0	1.0

# Code

- Nominal Data

```
1 nominal_feature = ['position']
```

- One Hot Encoding

```
1 one_hot_encoder = OneHotEncoder(sparse=False, handle_unknown='ignore')
2 one_hot_encoder.fit(X_train[nominal_feature])
```

```
1 one_hot_feature = []
2 for i, feature in enumerate(nominal_feature):
3     for cate in one_hot_encoder.categories_[i]:
4         one_hot_feature_name = str(feature) + '_' + str(cate)
5         one_hot_feature.append(one_hot_feature_name)
```

# Code

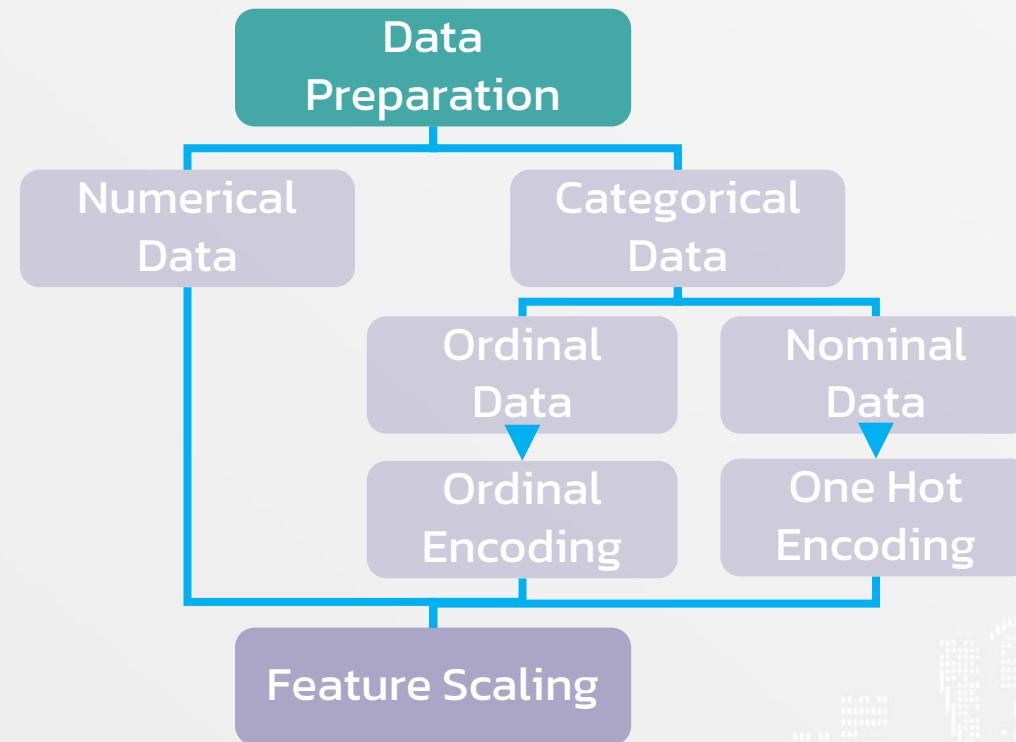
- One Hot Encoding for **training set**

```
1 X_train[one_hot_feature] = one_hot_encoder.transform(X_train[nominal_feature])
2 X_train.drop(nominal_feature, axis=1, inplace=True)
```

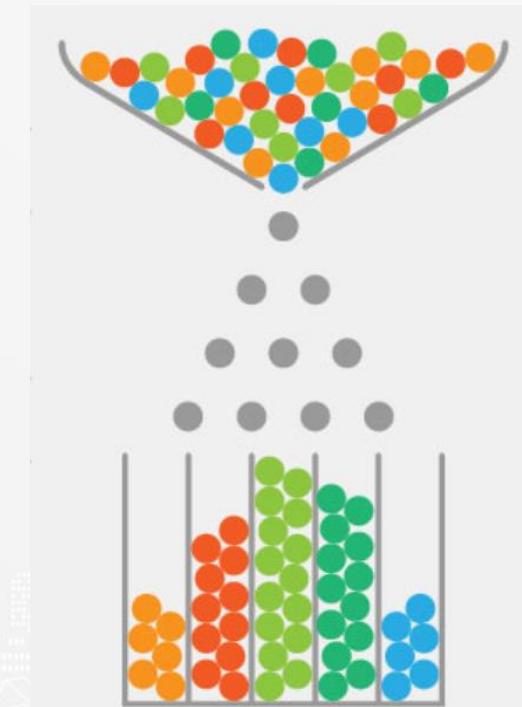
- One Hot Encoding for **test set**

```
1 X_test[one_hot_feature] = one_hot_encoder.transform(X_test[nominal_feature])
2 X_test.drop(nominal_feature, axis=1, inplace=True)
```

# Data Preparation



## Feature Scaling



# สำหรับ Normal Equation

✗ ไม่จำเป็นต้องคำ ✗



# Create Model

Setting Parameter

Train Model

Model's Weight & Bias

# Code – Setting Parameter

```
1 | reg = LinearRegression()
```



# Create Model

-  **Setting Parameter**
-  **Train Model**
-  **Model's Weight & Bias**

# Code – Train Model

```
1 | reg.fit(X_train, y_train)
```



# Create Model

- Setting Parameter**
- Train Model**
- Model's Weight & Bias**

# Code – Model's Weight & Bias

- **Bias ( $w_0$ )**

```
1 reg.intercept_
```

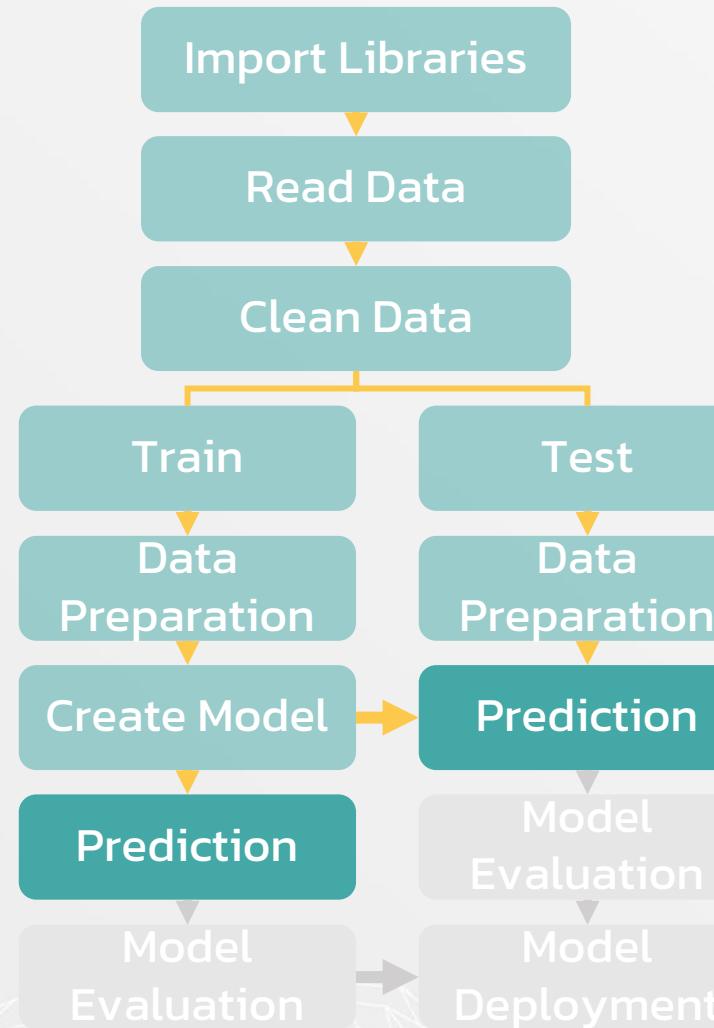
- **Weight ( $w_1, \dots, w_p$ )**

```
1 reg.coef_
```

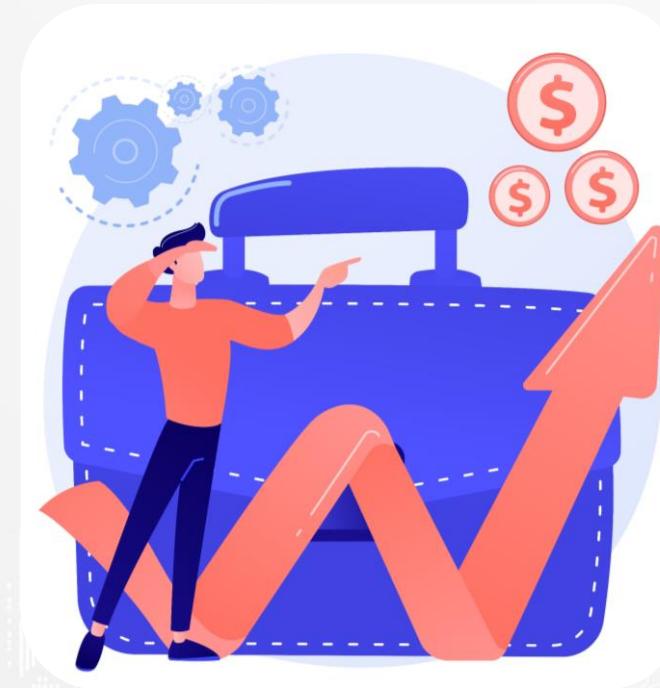


# Create Model

- Setting Parameter**
- Train Model**
- Model's Weight & Bias**



# Prediction



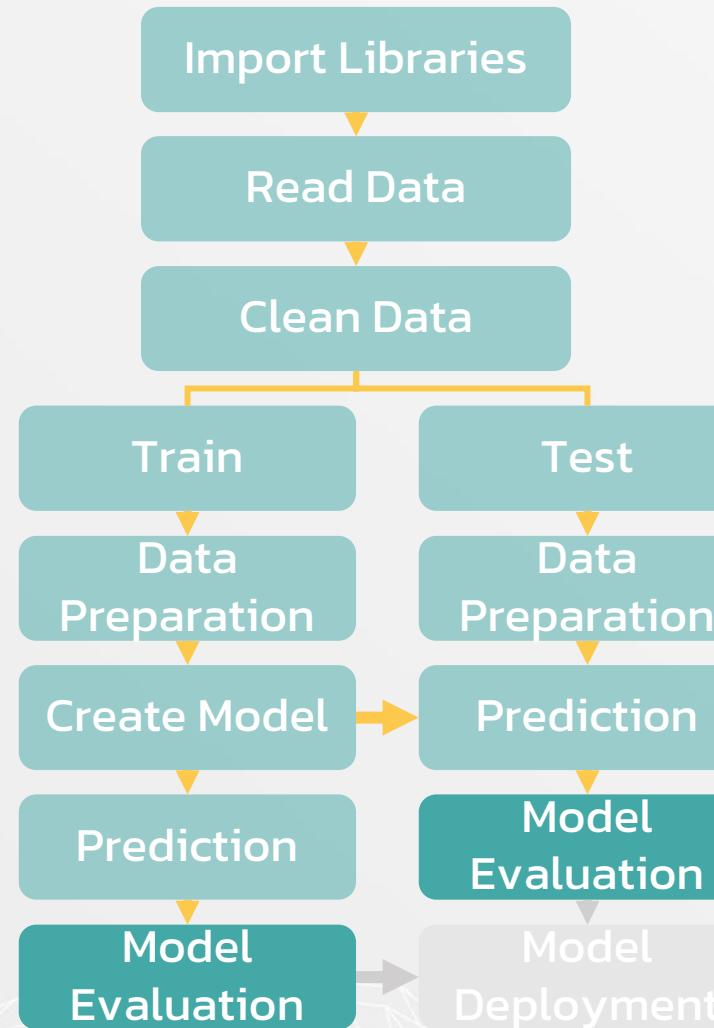
# Code

- Prediction for **training set**

```
1 y_pred_train = reg.predict(X_train)
```

- Prediction for **test set**

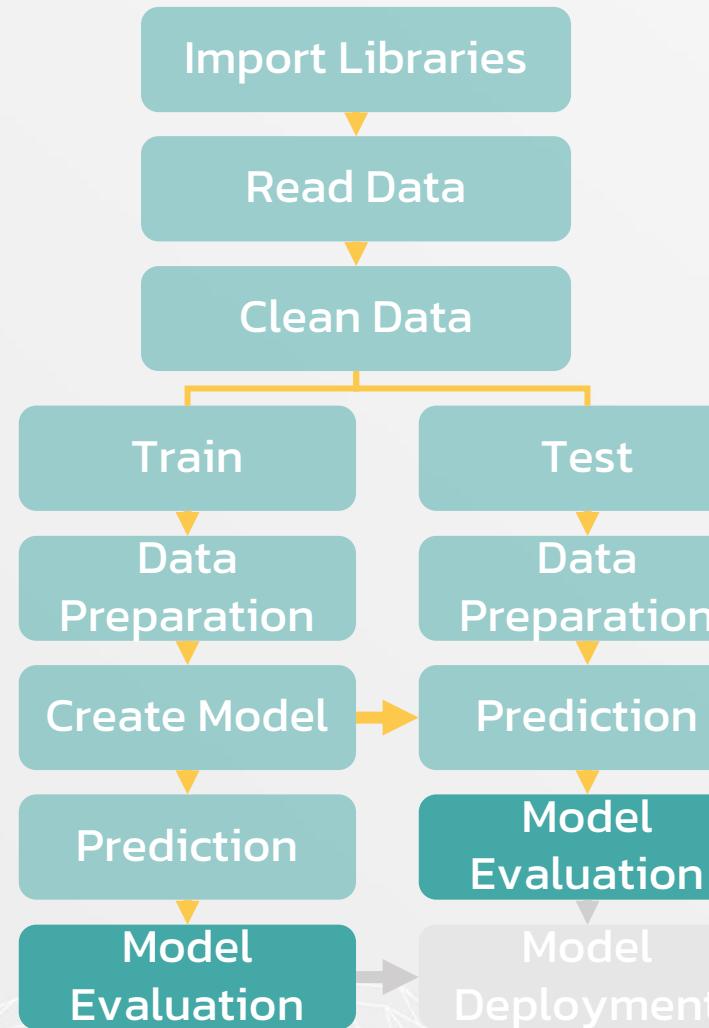
```
1 y_pred_test = reg.predict(X_test)
```



# Model Evaluation

1. Scoring ( $R^2, MSE, MAE, MAPE$ )
2. Scatter Plot between Predicted & Actual Values





# Model Evaluation

1. **Scoring ( $R^2$ ,  $MSE$ ,  $MAE$ ,  $MAPE$ )**
2. Scatter Plot between Predicted & Actual Values

- Scoring for **training set**

r2_score =	0.8834907214681481
mean_squared_error =	9954664.630857592
mean_absolute_error =	2661.546753531732
mean_absolute_percentage_error =	0.09401655935291867

- Scoring for **test set**

r2_score =	0.8921423690778735
mean_squared_error =	9282847.245418865
mean_absolute_error =	2569.1394359308074
mean_absolute_percentage_error =	0.09654987694303836

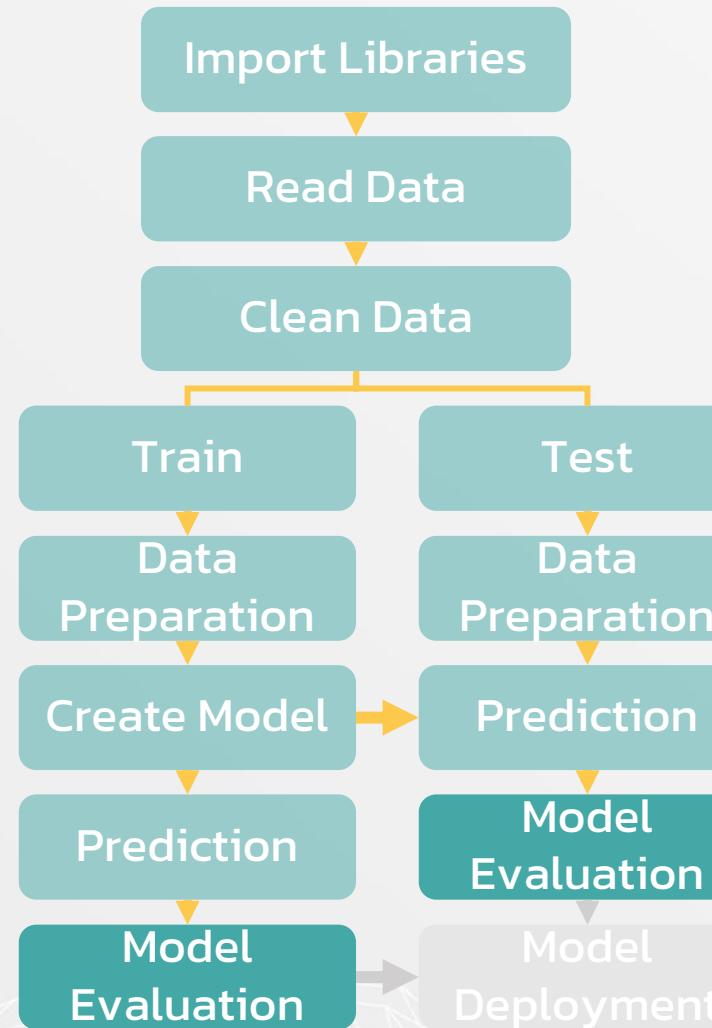
# Code

- Scoring for **training set**

```
1 print('r2_score =\t\t\t', r2_score(y_train, y_pred_train))
2 print('mean_squared_error =\t\t', mean_squared_error(y_train, y_pred_train))
3 print('mean_absolute_error =\t\t', mean_absolute_error(y_train, y_pred_train))
4 print('mean_absolute_percentage_error =', mean_absolute_percentage_error(y_train, y_pred_train))
```

- Scoring for **test set**

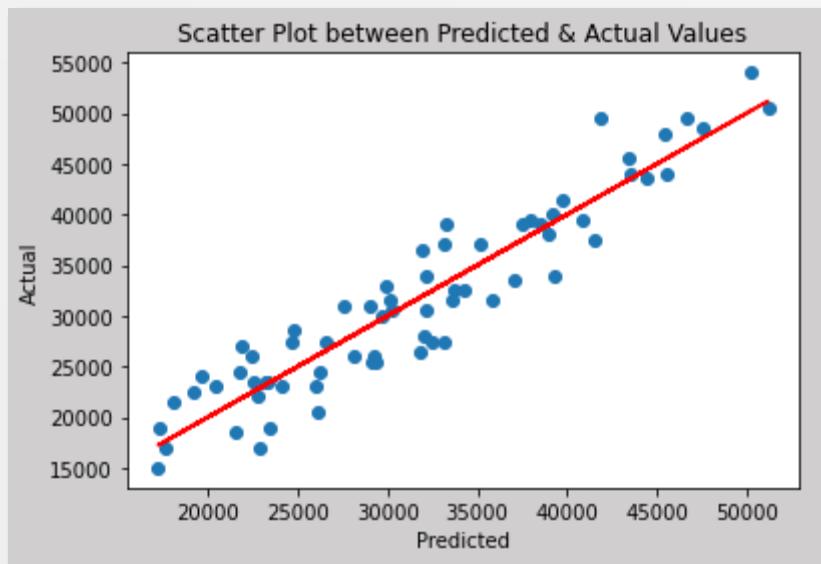
```
1 print('r2_score =\t\t\t', r2_score(y_test, y_pred_test))
2 print('mean_squared_error =\t\t', mean_squared_error(y_test, y_pred_test))
3 print('mean_absolute_error =\t\t', mean_absolute_error(y_test, y_pred_test))
4 print('mean_absolute_percentage_error =', mean_absolute_percentage_error(y_test, y_pred_test))
```



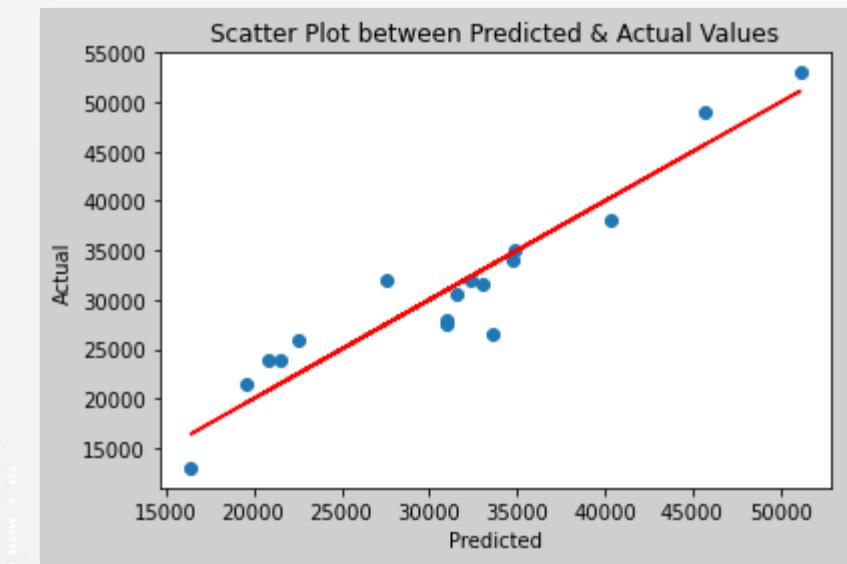
# Model Evaluation

1. Scoring ( $R^2, MSE, MAE, MAPE$ )
2. **Scatter Plot between Predicted & Actual Values**

- Scatter Plot for **training set**



- Scatter Plot for **test set**



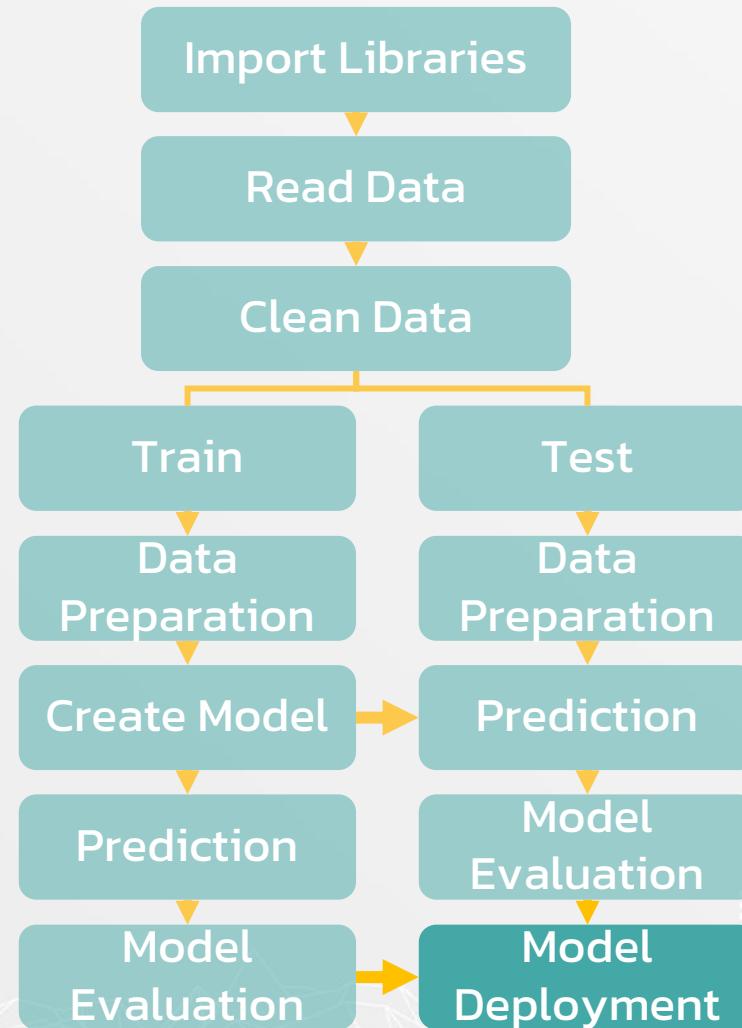
# Code

- Scatter Plot for **training set**

```
1 plt.scatter(y_pred_train, y_train)
2
3 plt.plot(y_pred_train, y_pred_train, color='red')
4
5 plt.title('Scatter Plot between Predicted & Actual Values')
6 plt.xlabel('Predicted')
7 plt.ylabel('Actual')
```

- Scatter Plot for **test set**

```
1 plt.scatter(y_pred_test, y_test)
2
3 plt.plot(y_pred_test, y_pred_test, color='red')
4
5 plt.title('Scatter Plot between Predicted & Actual Values')
6 plt.xlabel('Predicted')
7 plt.ylabel('Actual')
```

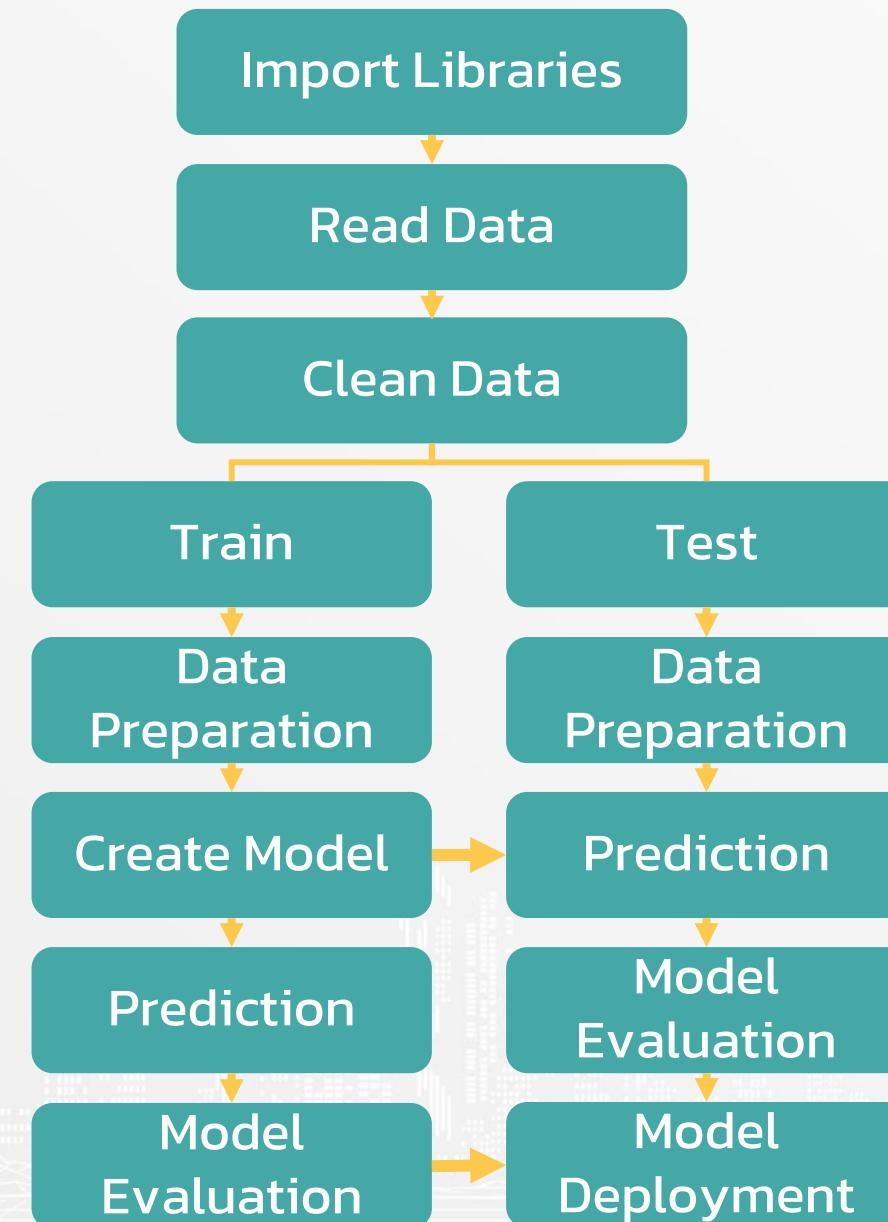


# Model Deployment



# Code

```
1 import pickle
2
3 pickle.dump((reg,
4                 ordinal_encoder,
5                 one_hot_encoder,
6                 feature_name,
7                 numerical_feature,
8                 ordinal_feature,
9                 nominal_feature),
10                open('salary_model.pickle', 'wb'))
```





# AI in Marketing

# Abstract

สร้าง model เพื่อพยากรณ์รายได้ที่ได้รับจากการยิง ads โดย feature กี่คำมาใช้ คือ ข้อมูลต่าง ๆ ที่เก็บได้จากการยิง ads เช่น

- จำนวนเงินที่ใช้ในการยิง ads
- จำนวนครั้งที่มีคนคลิก ads



# Why this project important?



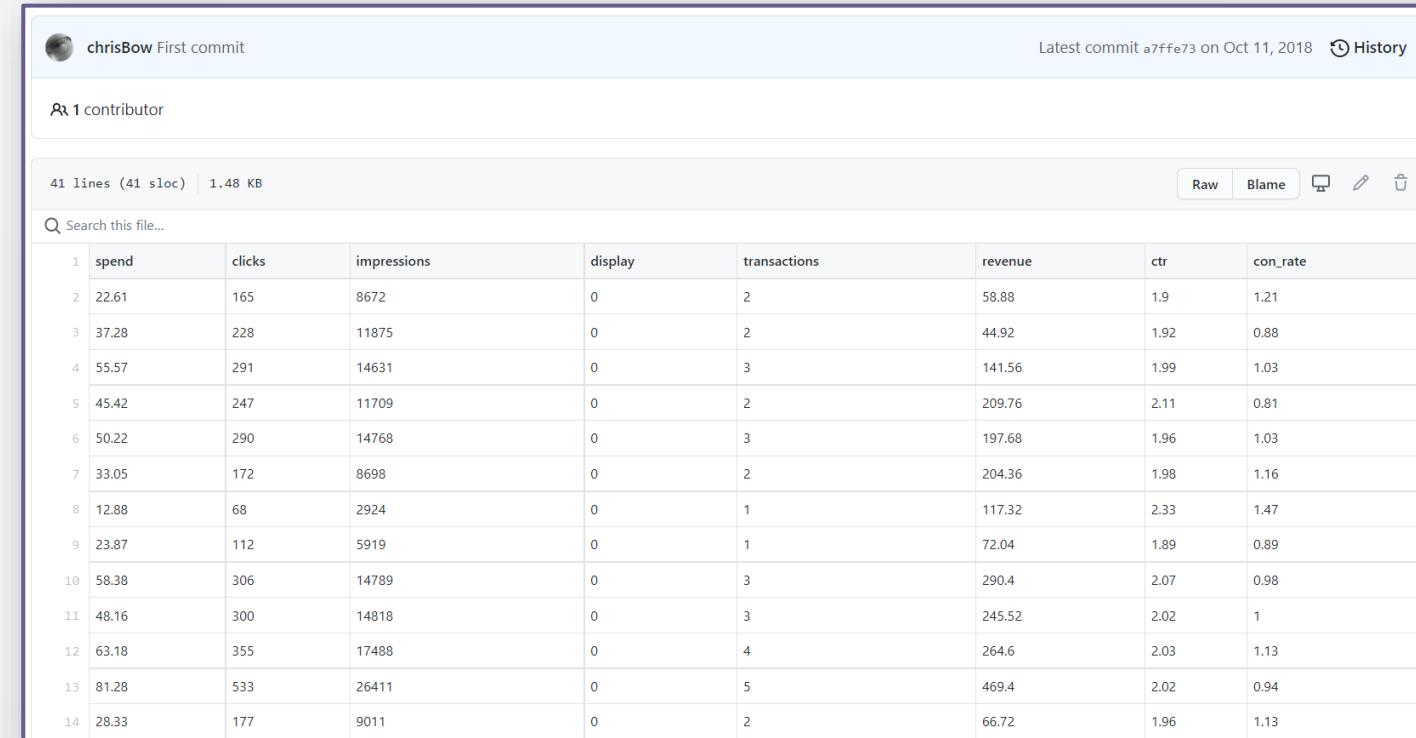
- สามารถวางแผนรายได้ของบริษัท จาก performance ของ ads
- สามารถต่อยอดกับการทำการตลาดบน platform อื่น ๆ

# Who this project is for?

- ผู้ขายสินค้าออนไลน์
- ผู้ดูแลเพจ
- Marketing consultant
- นักวิเคราะห์ข้อมูล



# Ads Dataset



chrisBow First commit

Latest commit a7ffe73 on Oct 11, 2018 History

1 contributor

41 lines (41 sloc) | 1.48 KB

Raw Blame

Search this file...

1	spend	clicks	impressions	display	transactions	revenue	ctr	con_rate
2	22.61	165	8672	0	2	58.88	1.9	1.21
3	37.28	228	11875	0	2	44.92	1.92	0.88
4	55.57	291	14631	0	3	141.56	1.99	1.03
5	45.42	247	11709	0	2	209.76	2.11	0.81
6	50.22	290	14768	0	3	197.68	1.96	1.03
7	33.05	172	8698	0	2	204.36	1.98	1.16
8	12.88	68	2924	0	1	117.32	2.33	1.47
9	23.87	112	5919	0	1	72.04	1.89	0.89
10	58.38	306	14789	0	3	290.4	2.07	0.98
11	48.16	300	14818	0	3	245.52	2.02	1
12	63.18	355	17488	0	4	264.6	2.03	1.13
13	81.28	533	26411	0	5	469.4	2.02	0.94
14	28.33	177	9011	0	2	66.72	1.96	1.13

<https://github.com/chrisBow/marketing-regression-part-one>

# Ads Dataset

## Feature

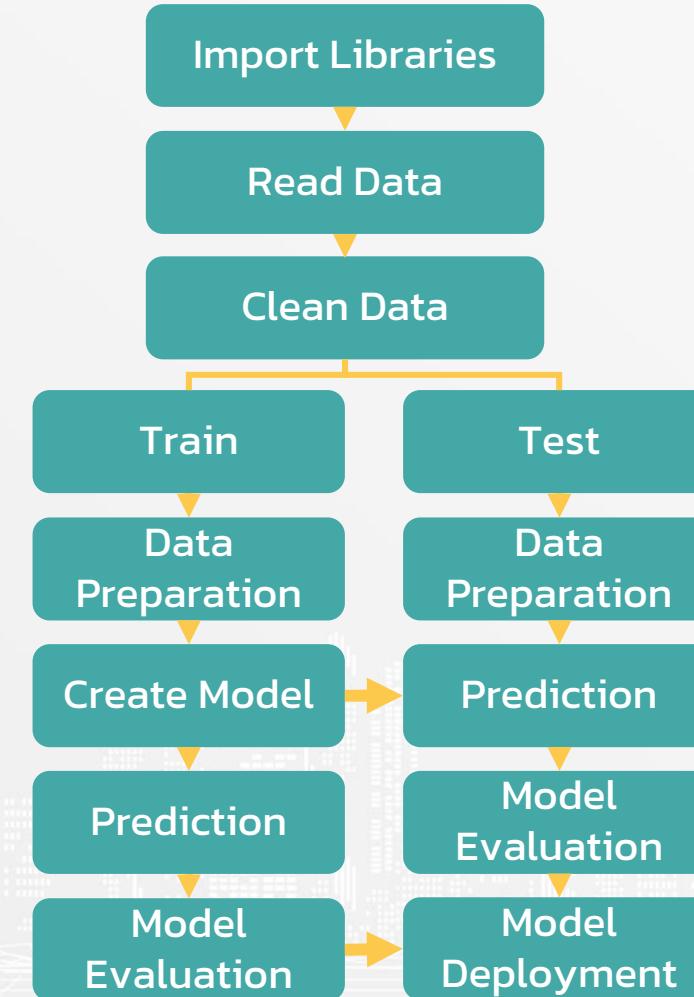
- **spend** : จำนวนเงินที่ใช้ในการยิง ads
- **clicks** : จำนวนครั้งที่มีการคลิก ads
- **impressions** : จำนวนครั้งที่ ads ปรากฏบน website
- **display** : มี web address ปรากฏบน ads หรือไม่
- **transactions** : จำนวนครั้งที่มีการทำธุรกรรม
- **ctr** : จำนวนครั้งที่คลิกโฆษณาต่อจำนวนครั้งที่โฆษณาปรากฏ (clickthrough rate)
- **con\_rate** : จำนวนสินค้าที่ถูกซื้อต่อจำนวนคนที่เข้ามาดูสินค้า (conversion rate)

## Target

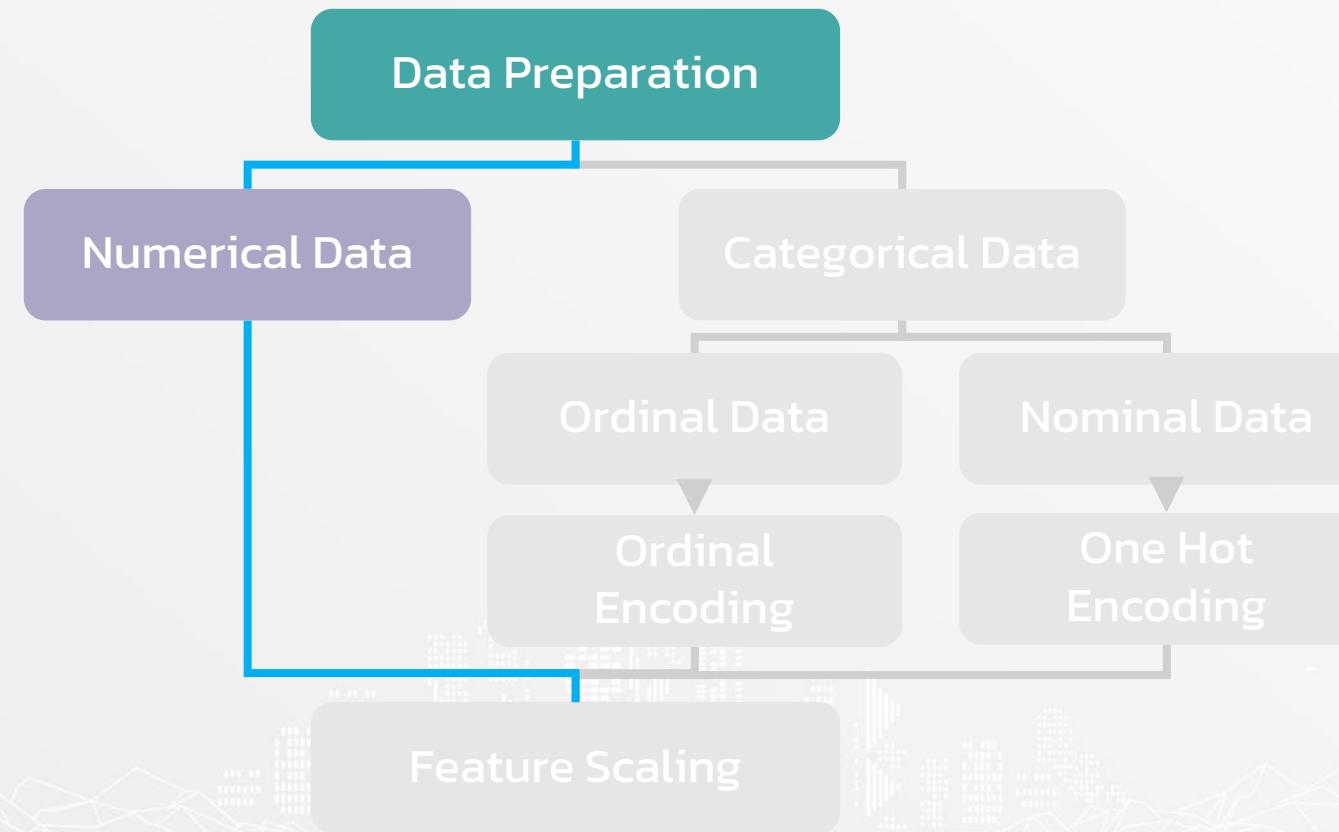
- **revenue** : รายได้ที่ได้รับจากการยิง ads



# What we learn from this project?



# Data Preparation





## 01. MARKETING



# AI in Investment

# Abstract

สร้าง model เพื่อพยากรณ์ราคากลางสุดรายวันของ SET50 โดย feature ที่นำมาใช้ คือ technical indicator ต่าง ๆ เช่น

- standard deviation
- RSI



# Why this project important?



- สามารถสร้างระบบการซื้อขายที่ปราศจาก  
อารมณ์ของมนุษย์
- สามารถเป็นพื้นฐานสำหรับสร้างระบบการซื้อขาย  
ที่ robust มากขึ้น
- สามารถต่อรองกับการเก็บกำไรบนสินทรัพย์ชนิด  
อื่น ๆ

# Who is this project for?

- นักลงทุน
- ผู้ดูแลกองทุน
- นักวิจัยเชิงปริมาณ
- นักวิเคราะห์ข้อมูล



# SET50 Dataset

ข้อมูลของ SET50 สามารถเก็บได้ที่เว็บไซต์ <https://www.investing.com>

SET 50 Historical Data

Time Frame: Daily Download Data 09/06/2011 - 10/07/2021

Date	Price	Open	High	Low	Vol.	Change %
Sep 06, 2011	734.49	724.53	734.49	722.61	836.64K	0.66%
Sep 07, 2011	744.18	736.88	744.25	735.60	915.18K	1.32%
Sep 08, 2011	747.06	740.70	747.06	738.38	844.77K	0.39%
Sep 09, 2011	739.73	747.53	748.28	738.83	1.09M	-0.98%
Sep 12, 2011	723.30	728.89	730.86	720.25	925.92K	-2.22%
Sep 13, 2011	716.87	726.64	728.53	713.70	685.74K	-0.89%
Sep 14, 2011	711.34	720.18	720.69	701.79	845.61K	-0.77%
Sep 15, 2011	721.34	714.74	721.62	708.28	1.19M	1.41%
Sep 16, 2011	718.64	726.70	728.32	716.88	663.14K	-0.37%
Sep 19, 2011	706.87	712.53	714.67	706.87	621.48K	-1.64%
Sep 20, 2011	714.70	706.12	714.70	700.93	643.67K	1.11%
Sep 21, 2011	717.08	711.76	718.85	710.24	650.54K	0.33%

# SET50 Dataset

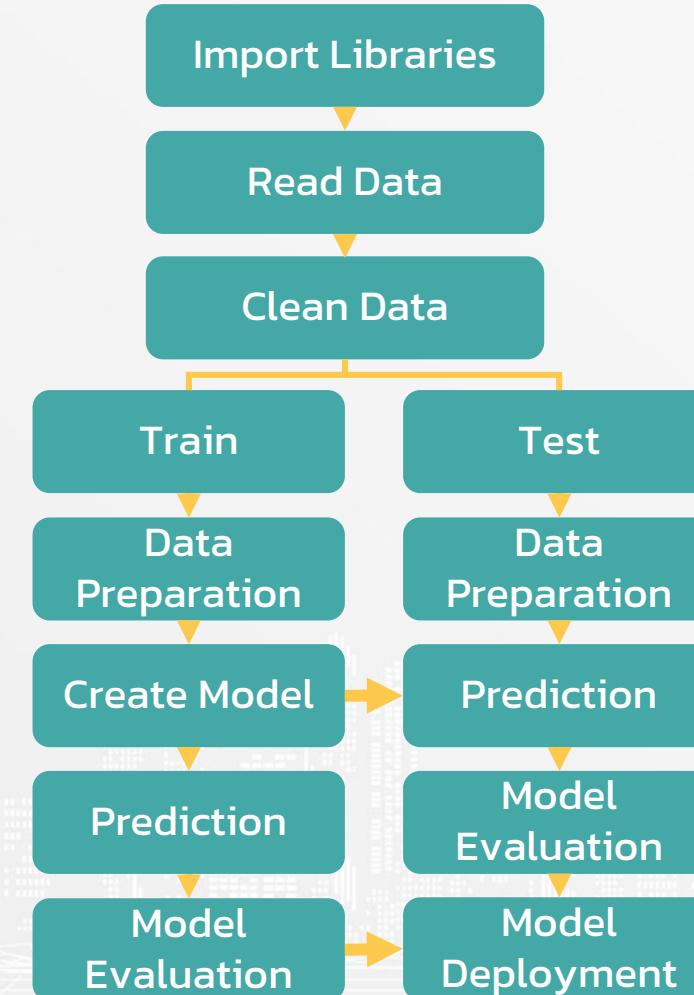
## Feature

- Open\_0 : ราคาเปิดของวันนั้น ๆ
- STD5\_Open\_0 : ส่วนเบี่ยงเบนมาตรฐานของราคาเปิด 5 วันล่าสุด
- RSI14\_Open\_0 : ค่าที่บอกรายการแกว่งของราคาเปิด 14 วันล่าสุด
- ATR14\_0 : ค่าที่บอกรายการผันผวนของราคา 14 วันล่าสุด

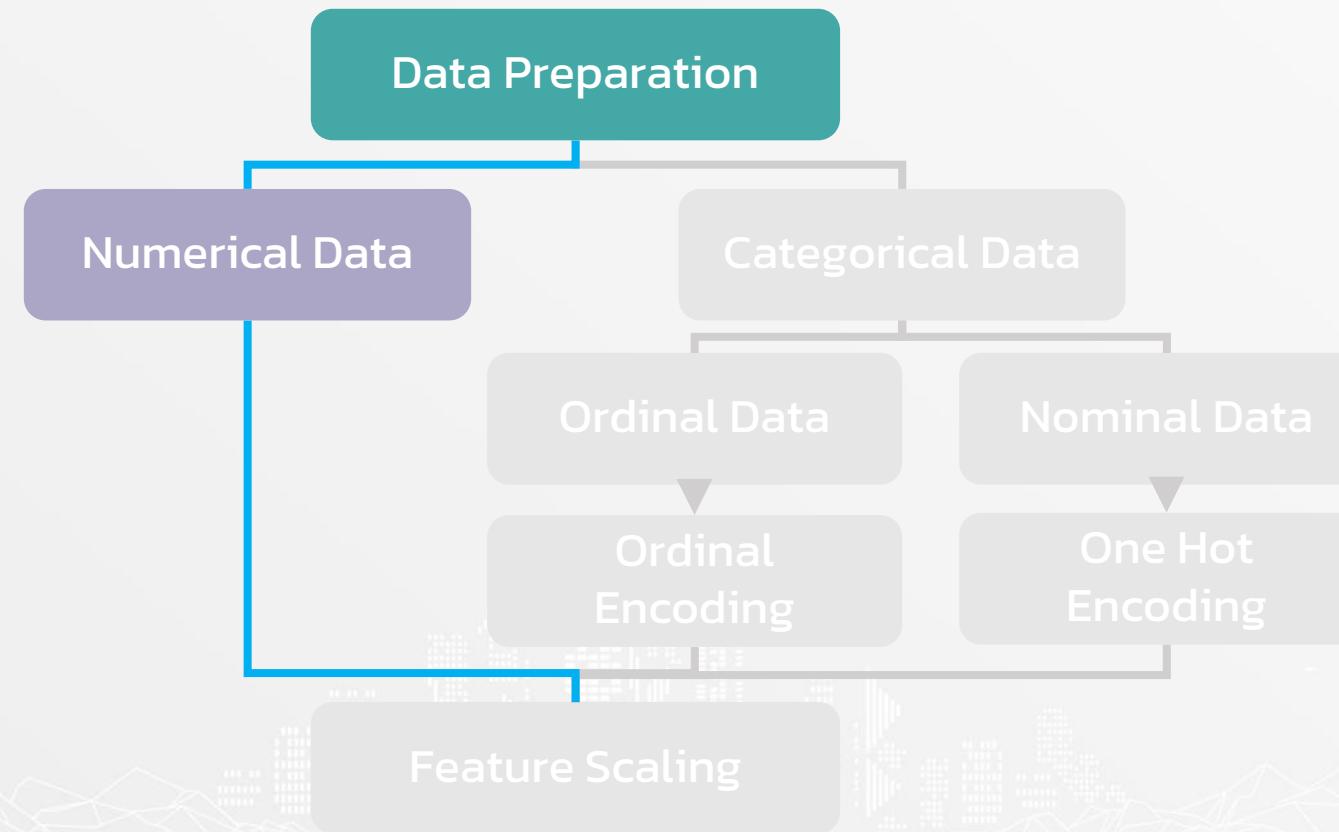
## Target

- High\_0 : ราคาสูงสุดของวันนั้น ๆ

# What we learn from this project?

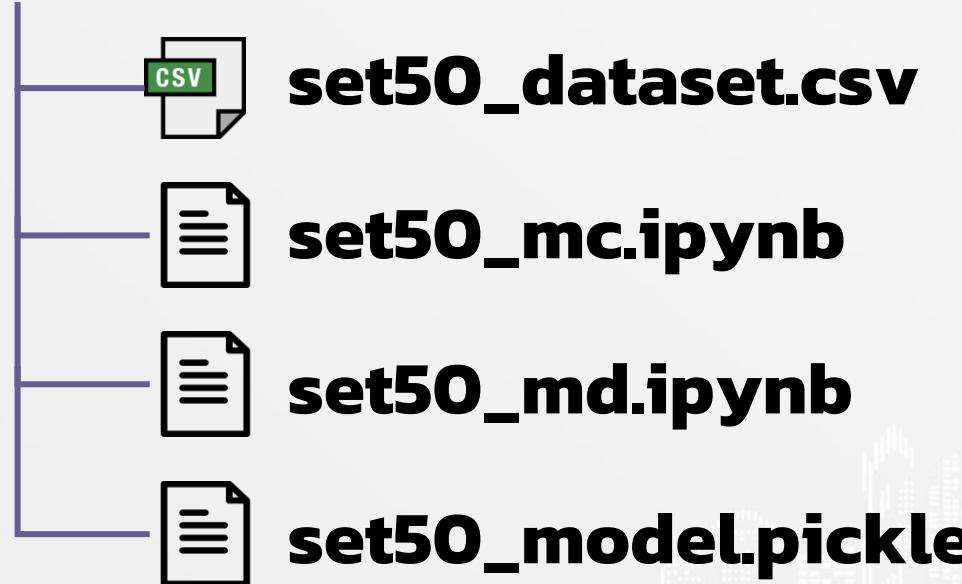


# Data Preparation





## 02. INVESTMENT/**set50**





## 02. INVESTMENT/btcusd

-  **btcusd\_dataset.csv**
-  **btcusd\_mc.ipynb**
-  **btcusd\_md.ipynb**
-  **btcusd\_model.pickle**

## 02. INVESTMENT/eurusd

-  **eurusd\_dataset.csv**
-  **eurusd\_mc.ipynb**
-  **eurusd\_md.ipynb**
-  **eurusd\_model.pickle**

## 02. INVESTMENT/xauusd

-  **xauusd\_dataset.csv**
-  **xauusd\_mc.ipynb**
-  **xauusd\_md.ipynb**
-  **xauusd\_model.pickle**

\*\*ข้อมูลสามารถเก็บได้ที่เว็บไซต์ <https://www.investing.com>

# Smart Farm

# Abstract

สร้าง model เพื่อพยากรณ์ปริมาณน้ำก็ต้นข้าวต้องการ โดย feature กี่นำมาใช้คือข้อมูลของปัจจัยต่าง ๆ กี่ส่งผลต่อการใช้น้ำของข้าว เช่น

- สภาพดิน
- สภาพอากาศ



# Why this project important?



- สามารถควบคุม วางแผนปริมาณน้ำที่จะใช้ได้ และประหยัดต้นทุน
- สามารถนำความรู้ไปต่อยอดเพื่อสร้าง smart farm
- สามารถต่อยอดกับการพยากรณ์ปริมาณน้ำที่พืชต้องการ ฯ ต้องการ

# Who this project is for?

- เกษตรกรที่สนใจ AI กับการเกษตร
- ผู้ควบคุม/วางแผนการผลิต
- นักวิเคราะห์ข้อมูล



# Rice Dataset



Dataset

## CROP WATER REQUIREMENT

Based on Condition of sowing



Prateek Kumar • updated a month ago (Version 3)

<https://www.kaggle.com/prateekkumar/crop-water-requirement>

# Rice Dataset

## Feature

- SOIL TYPE
- REGION
- TEMPERATURE
- WEATHER CONDITION

## Target

- WATER REQUIREMENT

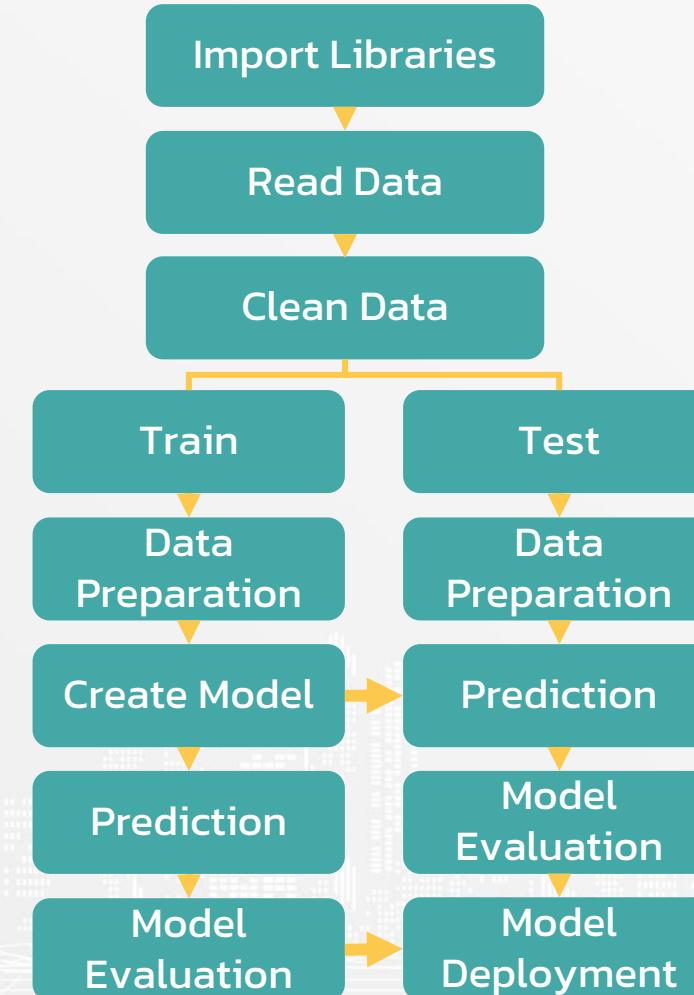


# Rice Dataset

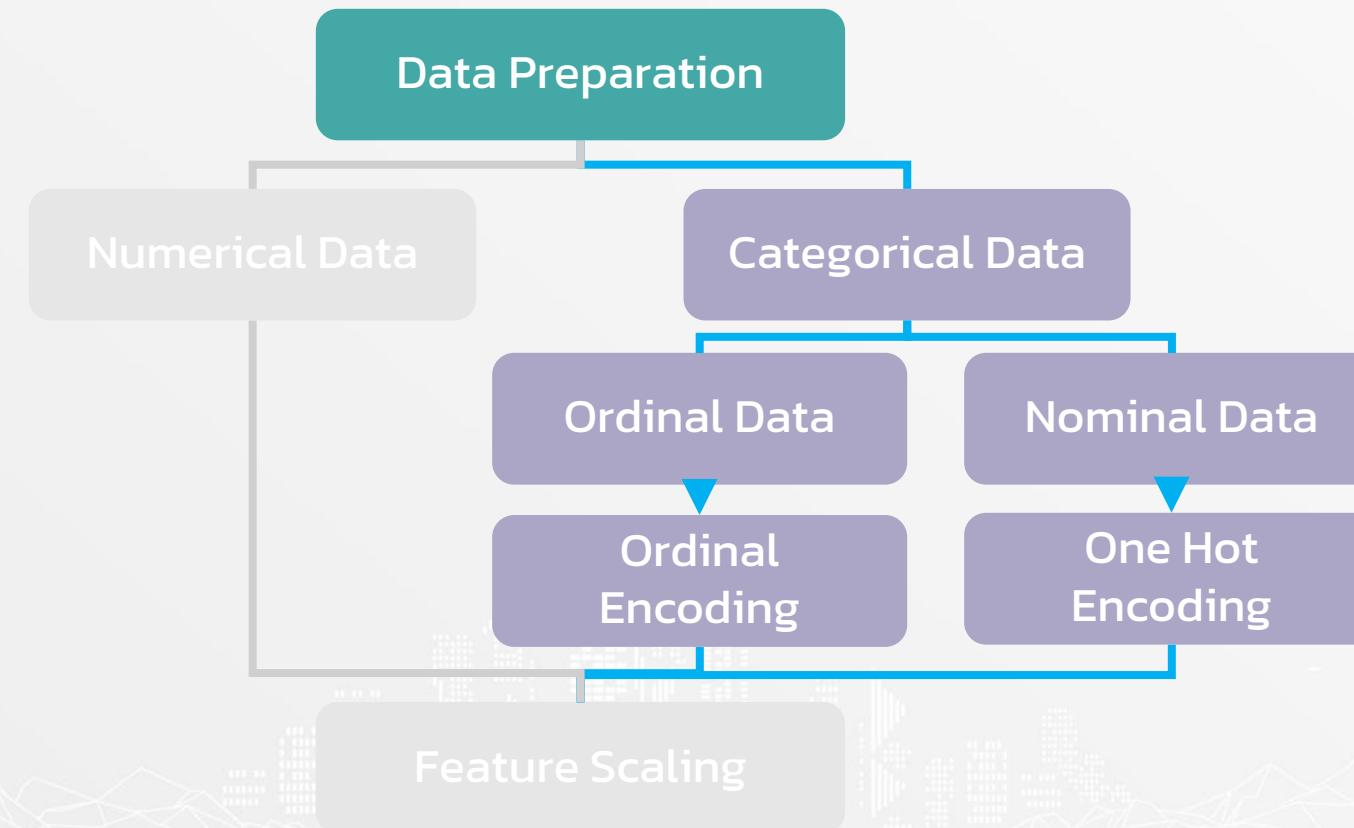
ปริมาณน้ำที่พืชต้องการ ถูกเก็บจากเครื่องมือที่มีชื่อว่า “Lysimeter Tank”



# What we learn from this project?

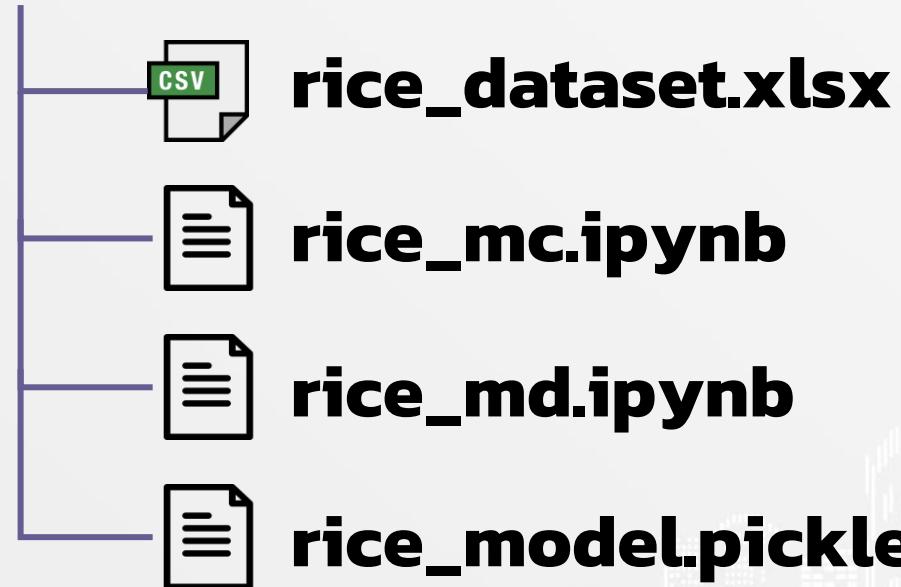


# Data Preparation





## 03. SMART FARM/rice





## 02. INVESTMENT/banana

- banana\_dataset.xlsx**
- banana\_mc.ipynb**
- banana\_md.ipynb**
- banana\_model.pickle**



## 02. INVESTMENT/melon

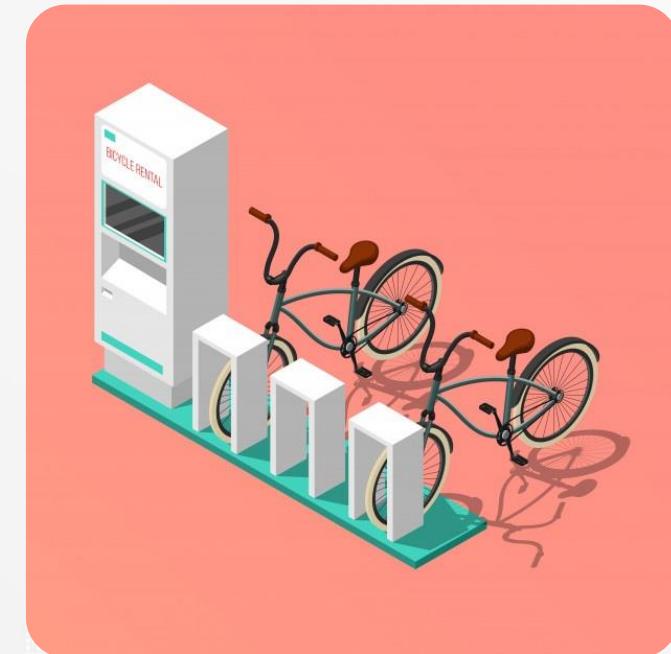
- melon\_dataset.xlsx**
- melon\_mc.ipynb**
- melon\_md.ipynb**
- melon\_model.pickle**

# AI in Business

# Abstract

สร้าง model เพื่อพยากรณ์จำนวนลูกค้าที่จะมาใช้บริการเช่ายืมจักรยาน โดย feature ที่นำมาใช้ คือ ข้อมูลเหตุการณ์ที่เกิดขึ้นในแต่ละวัน เช่น

- ฤดูกาล
- สภาพอากาศ
- เป็นวันทำงานหรือวันหยุด



# Why this project important?



- สามารถนำความรู้ที่ได้จากการสร้าง model ไปประยุกต์ใช้กับธุรกิจประเภทอื่น ๆ ที่มีลักษณะคล้ายกัน
- สามารถ stock สินค้าที่แต่ละสาขาได้อย่างเหมาะสมกับสภาวะปัจจุบัน

# Who this project is for?

- เจ้าของธุรกิจที่มีลักษณะใกล้เคียงกับ bike sharing เช่น ธุรกิจซัพพลาย ชาร์ต หรือเช่ารถ
- Business consultant
- ผู้จัดการ stock สินค้า
- นักวิเคราะห์ข้อมูล



# Bike Sharing Dataset



<https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset>

# Bike Sharing Dataset

## Feature

- season : ฤดูกาล
- yr : ปี
- mnth : เดือน
- holiday : เป็น holiday หรือไม่ใช่ holiday
- weekday : วันของแต่ละสัปดาห์
- workingday : เป็น working day หรือไม่ใช่ working day
- weathersit : เป็นวันที่สภาพอากาศเป็นอย่างไร เช่น สภาพอากาศสดใส, ฝนตกหนัก, มีหมอก

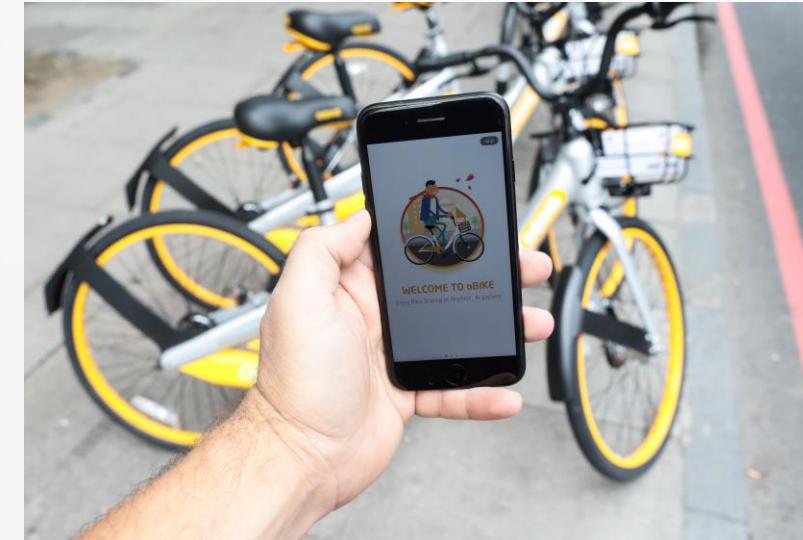
# Bike Sharing Dataset

## Feature

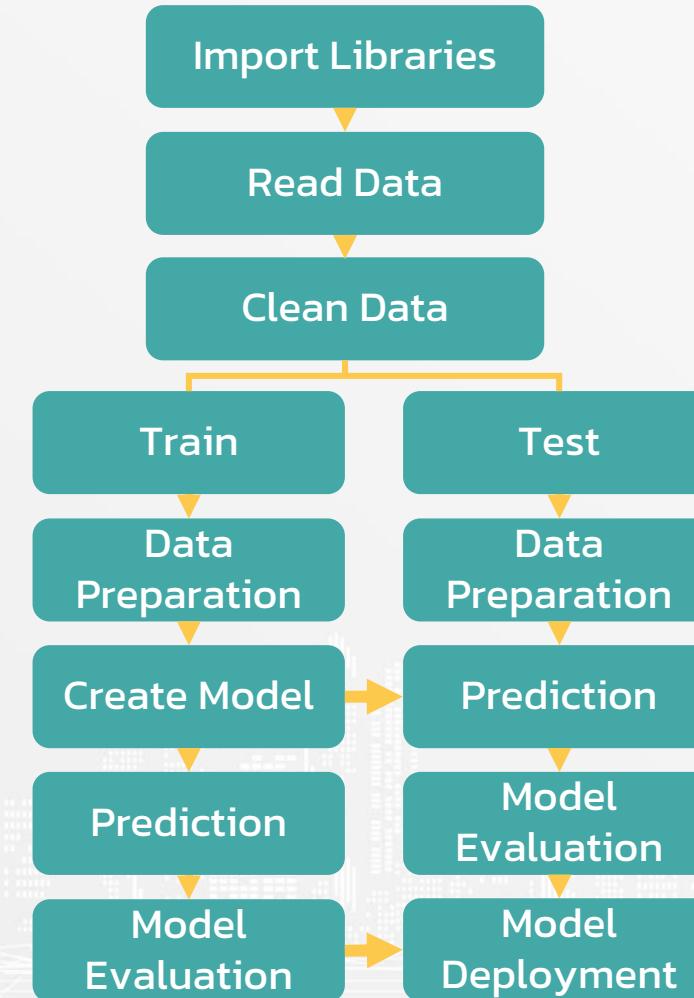
- temp : อุณหภูมิ
- hum : ความชื้น
- windspeed : เป็นวันที่มีความเร็วลมเท่าไร

## Target

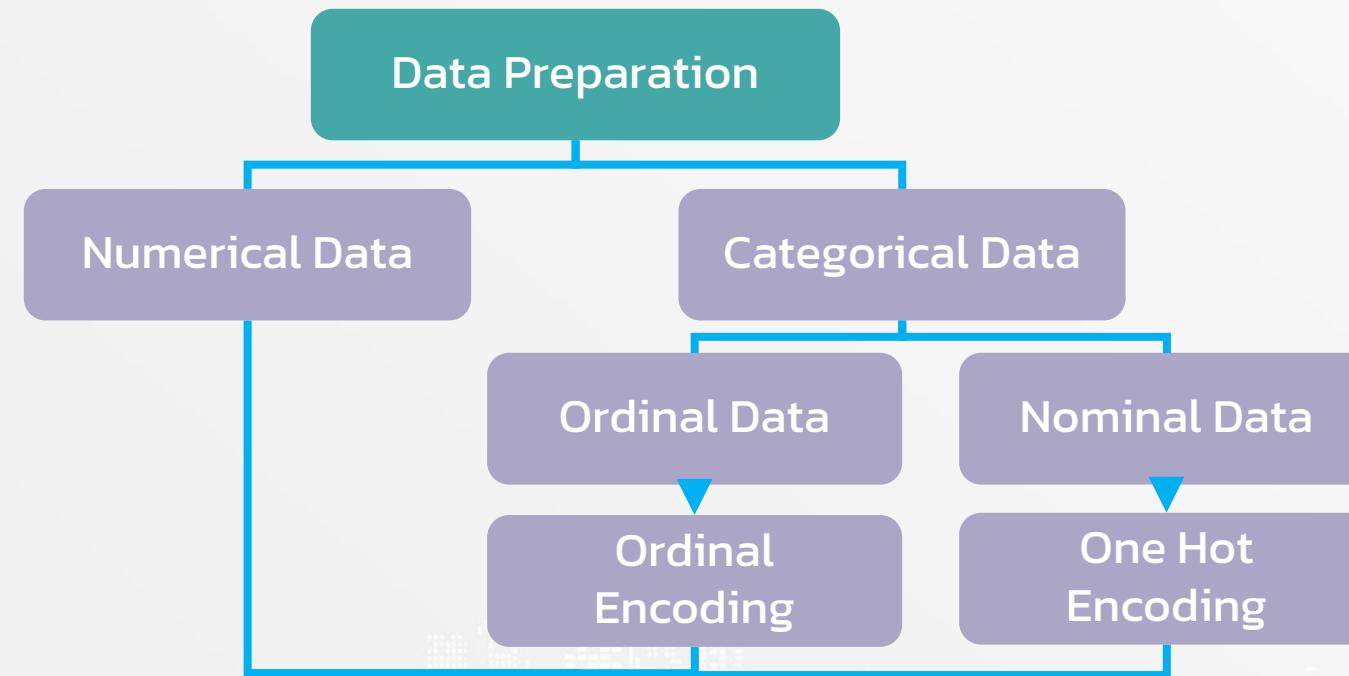
- count : จำนวนการเช่าจักรยานในวันนั้น ๆ



# What we learn from this project?



# Data Preparation





## 04. BIKE RENTAL

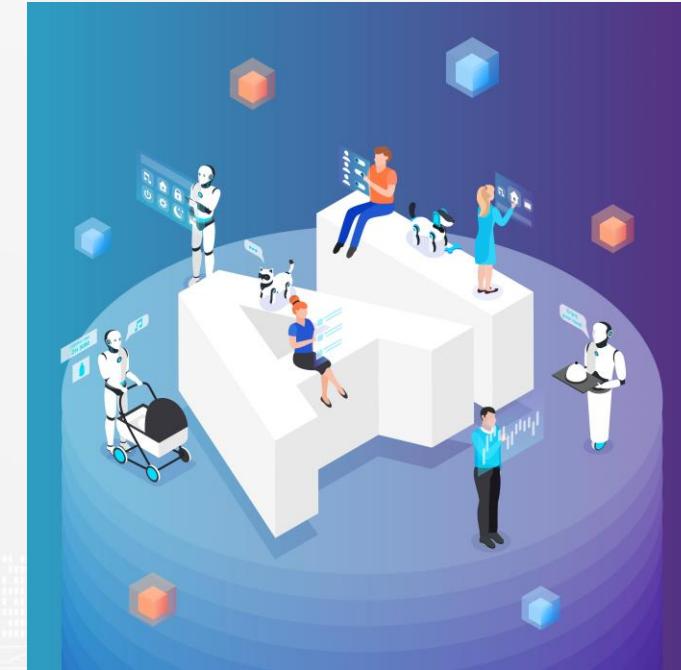


# AI in Insurance

# Abstract

สร้าง model เพื่อประเมินค่าประกันภัยส่วนบุคคล โดย feature ที่นำมาใช้ คือ ประวัติ กว่าไป และประวัติด้านสุขภาพ ของผู้ทำประกัน เช่น

- เพศ
- อายุ
- BMI



# Why this project important?



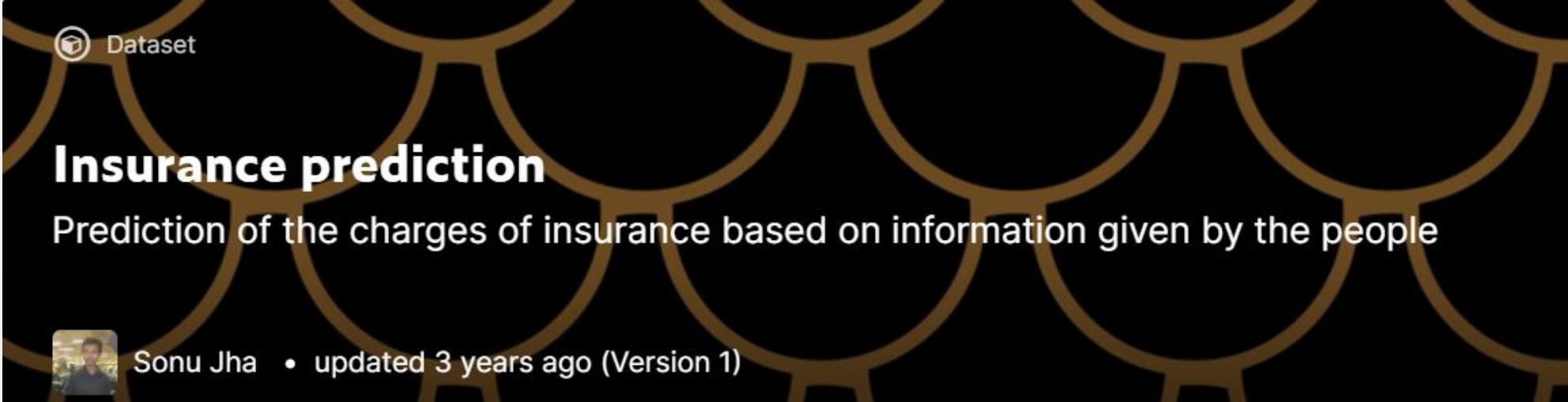
- สามารถวางแผนตั้งราคาเบี้ยประกันภัยกีเเนะะสมกี่สุด
- สามารถเพิ่มขีดความสามารถในการแบ่งขันกับบริษัทคู่แข่ง

# Who this project is for?

- นักคณิตศาสตร์ประถนภัย
- ผู้วางแผนกำประถนภัย
- นักวิเคราะห์ข้อมูล



# Insurance Dataset



Dataset

## Insurance prediction

Prediction of the charges of insurance based on information given by the people

 Sonu Jha • updated 3 years ago (Version 1)

<https://www.kaggle.com/sonujha090/insurance-prediction>

# Insurance Dataset

## Feature

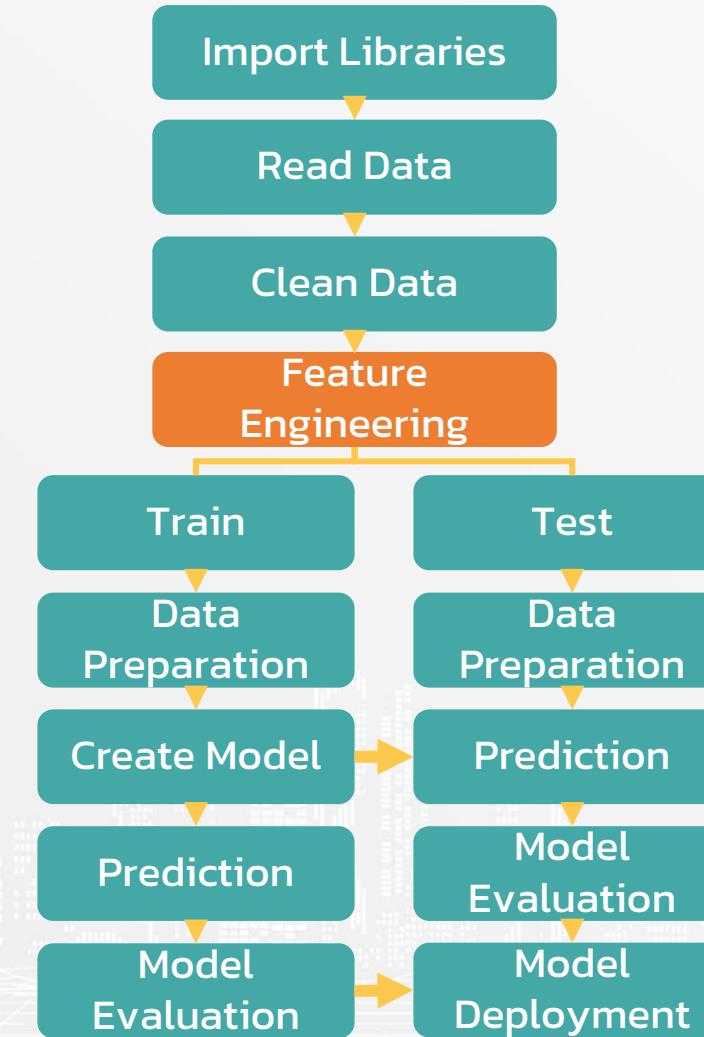
- age : อายุ
- sex : เพศ
- BMI : ค่า BMI
- children : จำนวนบุตร
- region : พื้นที่ที่พักอาศัย

## Target

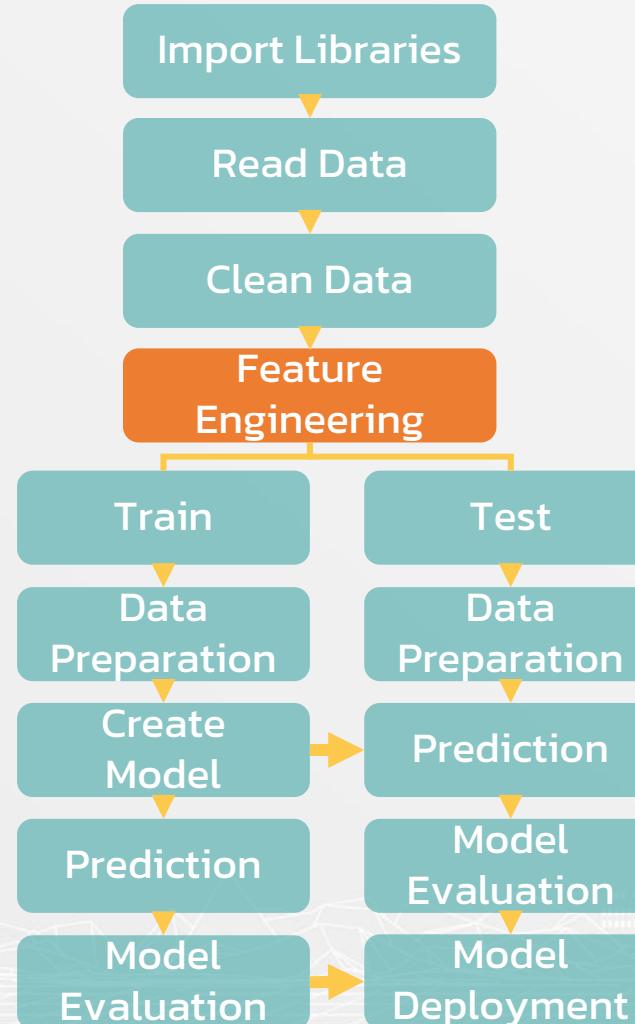
- charges : ค่าประกันที่จ่ายให้กับบริษัทประกันภัย



# What we learn from this project?



# What we learn from this project?



## Feature Engineering

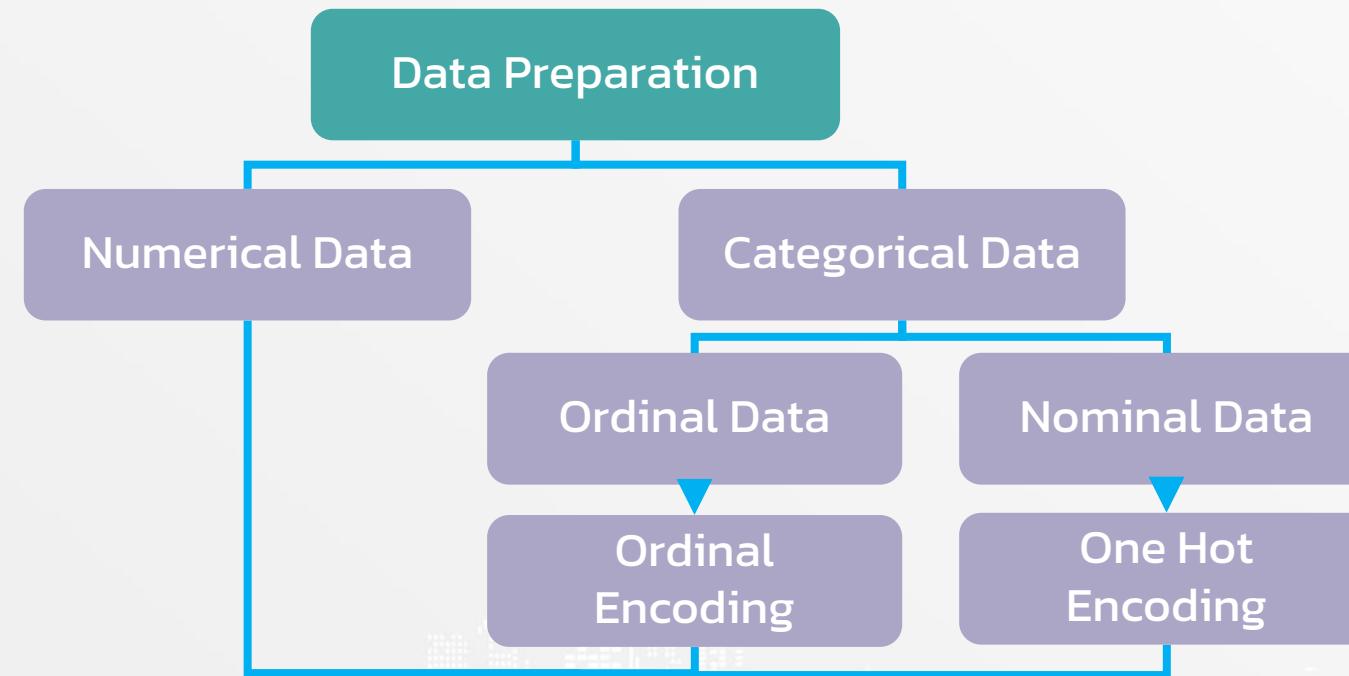
สร้าง feature ใหม่ เพื่อเก็บข้อมูลระดับภาวะน้ำหนักตัวโดยใช้ค่า BMI ในการแบ่ง

	age	sex	bmi	children	region	charges	bmiclass
0	19	0	27.900	0	3	16884.92400	3
1	62	0	26.290	0	2	27808.72510	3
2	27	1	42.130	0	2	39611.75770	4
3	30	1	35.300	0	3	36837.46700	4
4	34	0	31.920	1	0	37701.87680	4

# Code

```
1 data['bmiclass'] = (
2     (data['bmi'] < 18.5) * 1
3     + ((data['bmi'] >= 18.5) & (data['bmi'] < 23)) * 2
4     + ((data['bmi'] >= 23) & (data['bmi'] < 30)) * 3
5     + (data['bmi'] >= 30) * 4
6 )
```

# Data Preparation





## 05. INSURANCE/SMOKER

- Insurance\_smoker\_dataset.csv**
- Insurance\_smoker\_mc.ipynb**
- Insurance\_smoker\_md.ipynb**
- Insurance\_smoker\_model.pickle**



## 05. INSURANCE/NON SMOKER

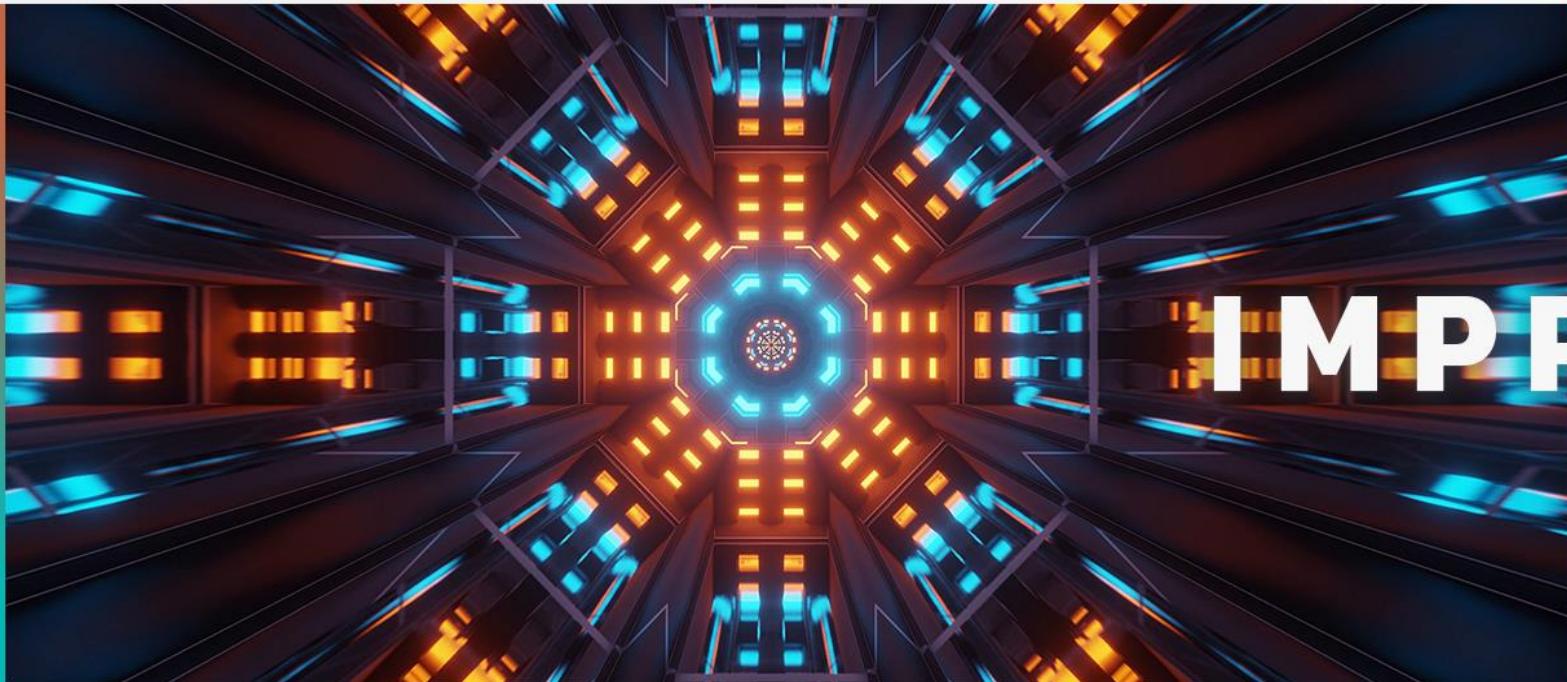
- Insurance\_non\_smoker\_dataset.csv**
- Insurance\_non\_smoker\_mc.ipynb**
- Insurance\_non\_smoker\_md.ipynb**
- Insurance\_non\_smoker\_model.pickle**

# DL101 : Linear Regression



TAUTOLOGY  
INNOVATION  
SCHOOL

MODEL IMPROVEMENT



IMPROVEMENT

MODEL  
IMPROVEMENT

BY TAUTOLOGY

MADE BY TAUTOLOGY THAILAND  
DO NOT PUBLISH WITHOUT PERMISSION

facebook/tautologyai  
www.tautology.live

TAUTOLOGY

# Model Improvement





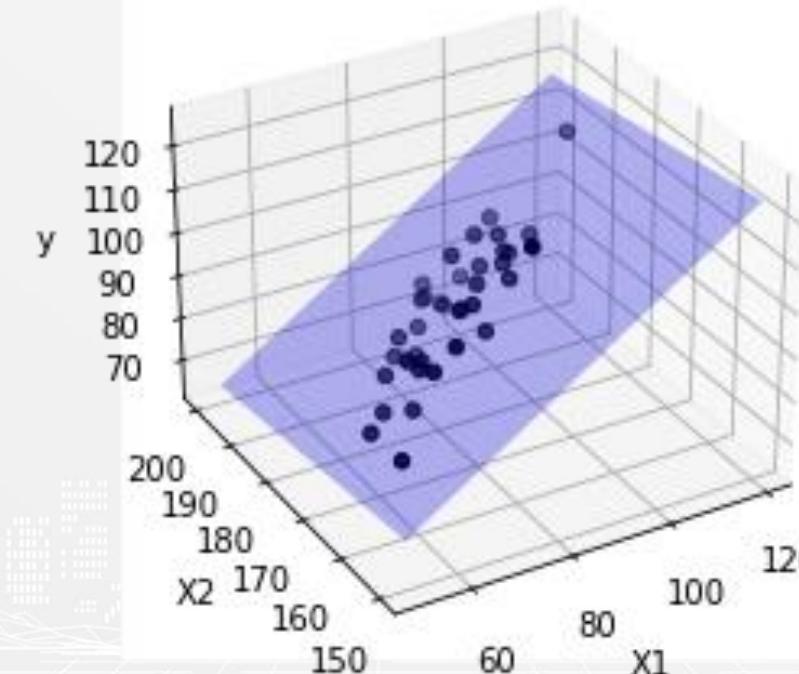
# Assumption

# Assumption

- Linear Relationship
- Normality of Residuals
- Homoscedasticity
- No Missing Features
- No Multicollinearity

# Linear Relationship

Linear Relationship : ความสัมพันธ์ระหว่างตัวแปรต้น และตัวแปรตาม ต้องเป็นความสัมพันธ์แบบเชิงเส้นต่อ กัน



# Linear Relationship

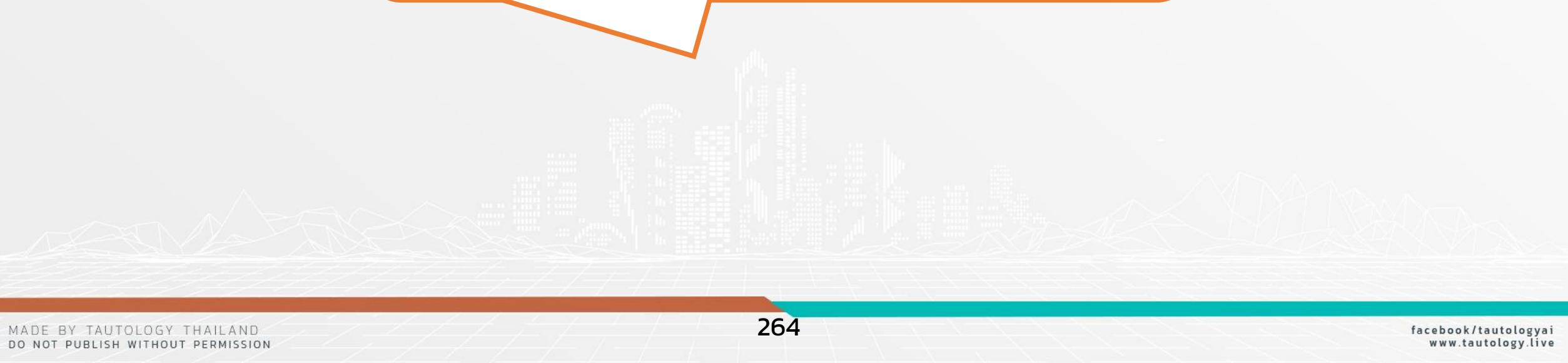
เหตุผลที่เป็นเช่นนี้ เพราะ model อยู่ในรูป

$$\hat{y} = w_0 + w_1x_1 + w_2x_2 + \cdots + w_px_p$$

- โดย
- ◆  $\hat{y}$  คือ ค่าพยากรณ์ของตัวARGET (predicted target)
  - ◆  $x_1, x_2, \dots, x_p$  คือ ตัวแปรตัว (feature)
  - ◆  $w_0, w_1, w_2, \dots, w_p$  คือ สัมประสิทธิ์ (coefficient)

# Linear Relationship

**Q :** เราจะรู้ได้อย่างไรว่าตัวแปรต้น และตัวแปรตาม มีความสัมพันธ์เชิงเส้นต่อกันหรือไม่?



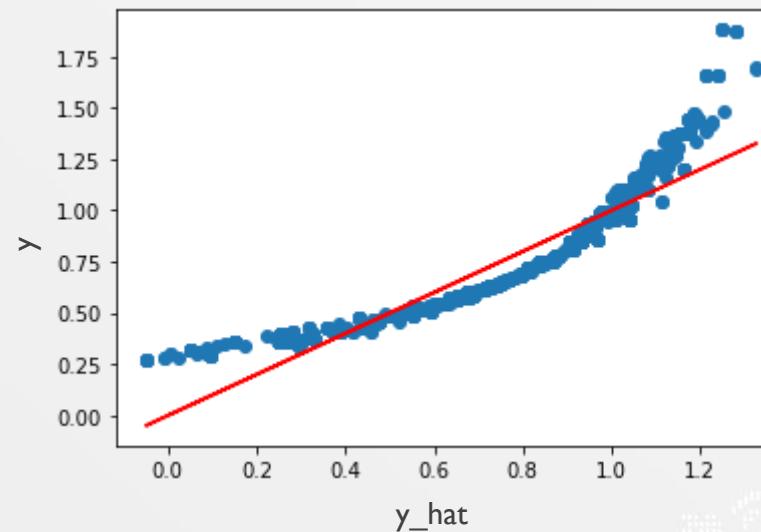
# Linear Relationship

**A :** ตรวจสอบความสัมพันธ์ระหว่างตัวแปรต้นและตัวแปรตามได้ด้วย

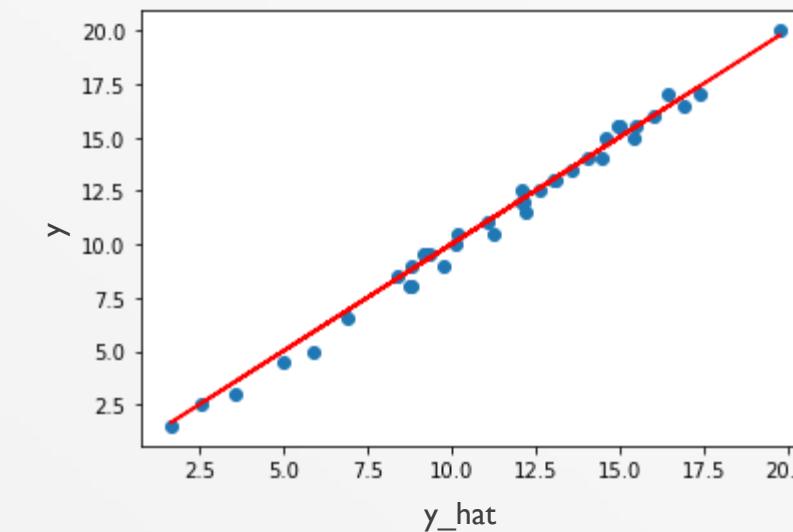
- i) Scatter plot ระหว่าง  $\hat{y}$  กับ  $y$
- ii) Assumption ข้อที่ 2
- iii) Assumption ข้อที่ 3

# Linear Relationship

scatter plot សេវាឯំងក់ ក្នុង  $\hat{y}$



Nonlinear Relationship



Linear Relationship

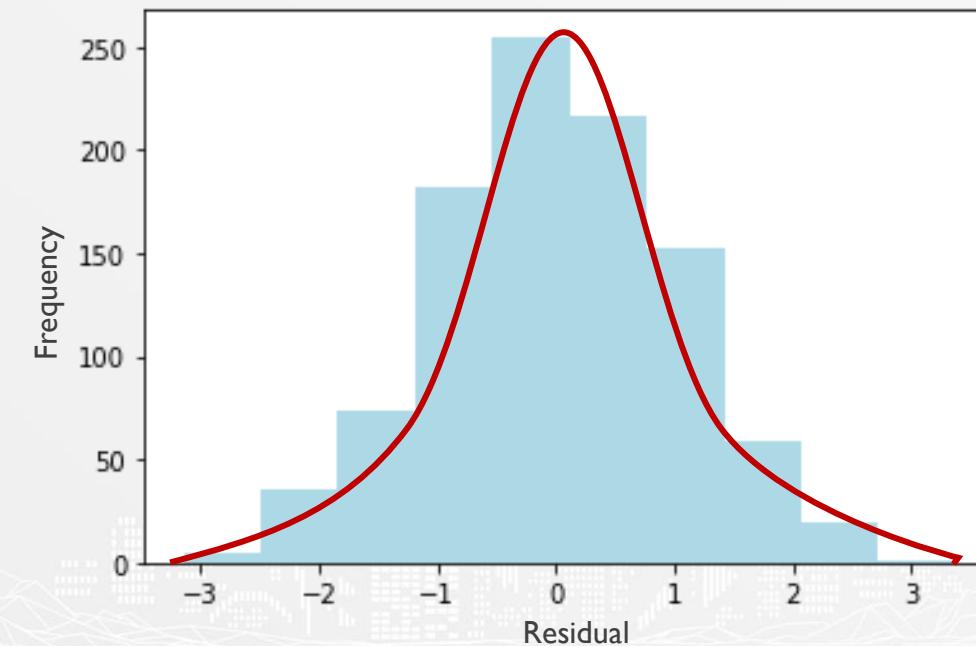
# Assumption

## **Linear Relationship**

- Normality of Residuals
- Homoscedasticity
- No Missing Features
- No Multicollinearity

# Normality of Residuals

Normality of residuals : residuals ต้องมีการกระจายตัวแบบ normal distribution ที่มี mean เท่ากับ 0 และ variance เท่ากับ  $\sigma^2$  ( $\varepsilon \sim N(0, \sigma^2)$ )



# Normality of Residuals

Residuals คือ ผลต่างระหว่างค่าจริง ( $y$ ) กับค่าพยากรณ์ ( $\hat{y}$ )

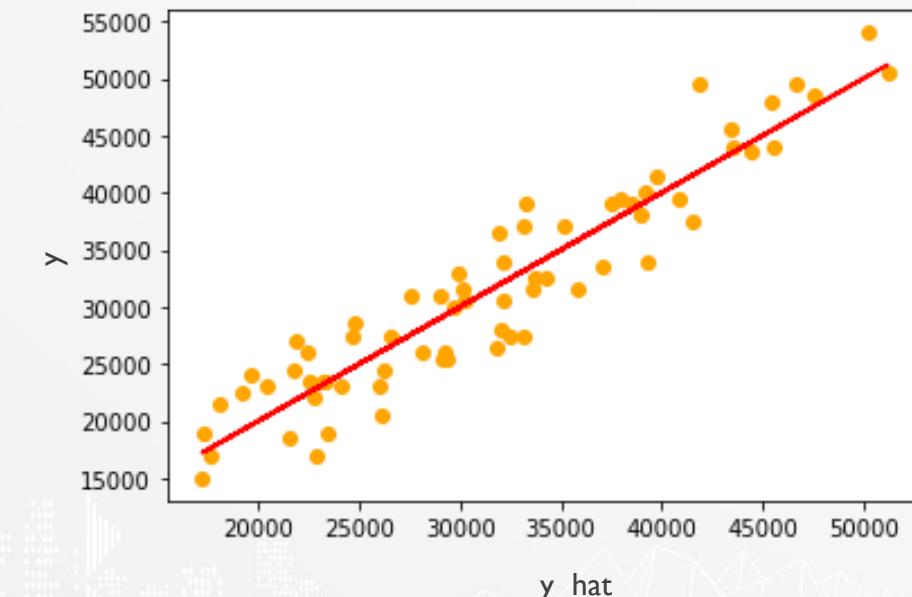
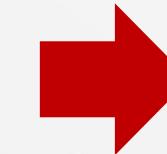
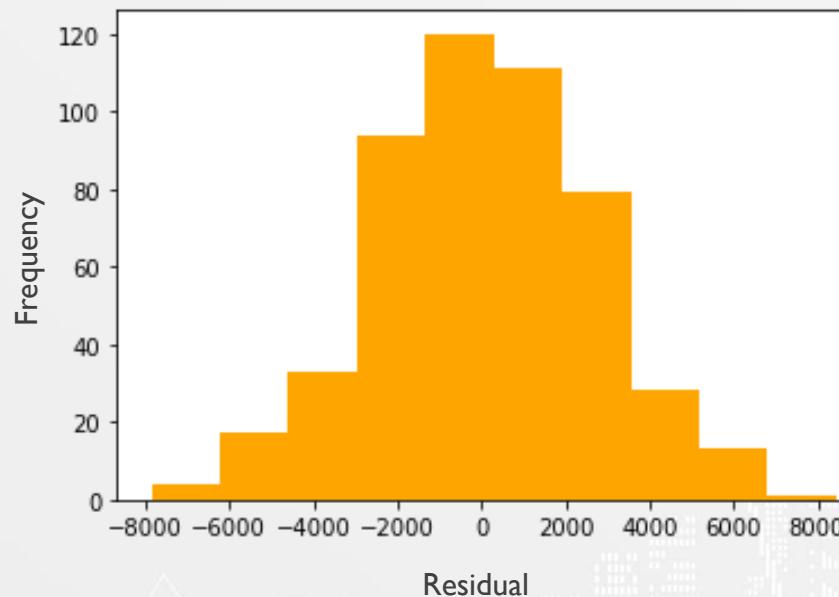
$$\boldsymbol{\varepsilon} = \mathbf{y} - \hat{\mathbf{y}}$$

$$\begin{array}{c|c|c} \boldsymbol{\varepsilon} & \mathbf{y} & \hat{\mathbf{y}} \\ \hline \varepsilon_1 & y_1 & \hat{y}_1 \\ \hline \varepsilon_2 & y_2 & \hat{y}_2 \\ \hline \vdots & \vdots & \vdots \\ \hline \varepsilon_3 & y_n & \hat{y}_n \end{array}$$

= -

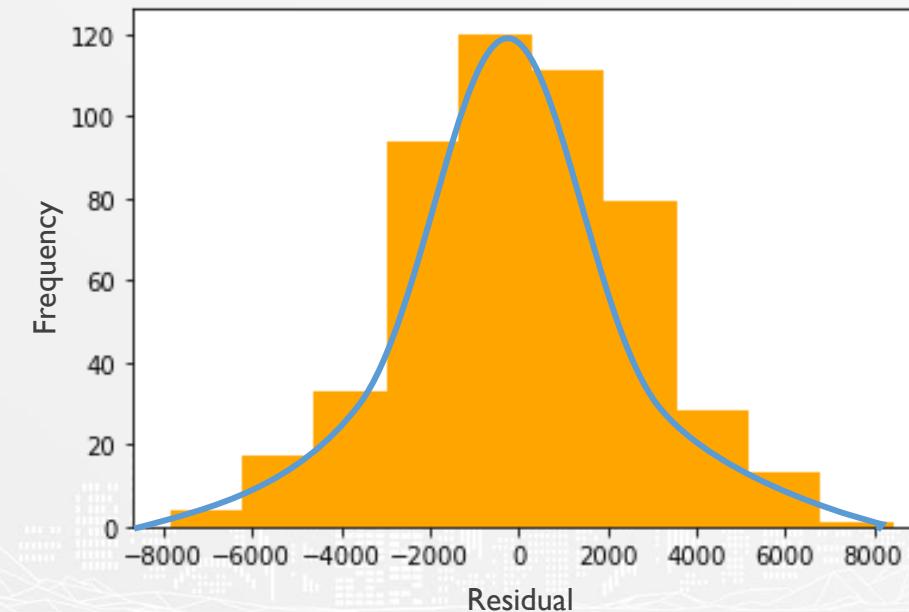
# Normality of Residuals

เหตุผลที่ residuals มี mean เท่ากับ 0 เพราะ  $\hat{y}$  ที่ได้จาก model ต้องเป็นค่ากลางของ  $y$



# Normality of Residuals

เหตุผลที่ residuals มีการกระจายตัวแบบ normal distribution เพราะ เป็นการยืนยันว่า w ที่เราได้จากการ minimize SSE เป็น w ที่ดีที่สุด

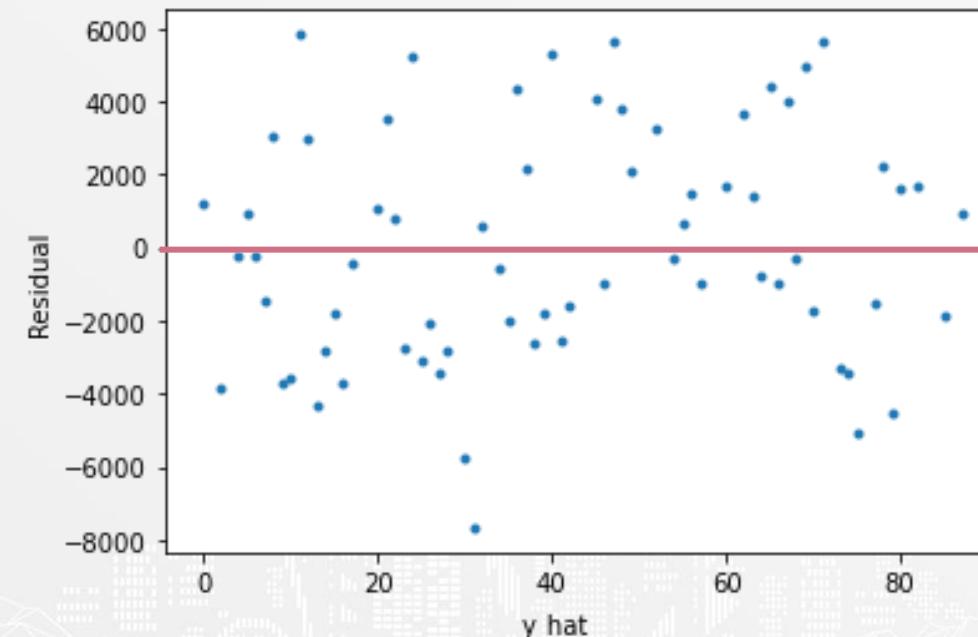


# Assumption

- Linear Relationship**
- Normality of Residuals**
- Homoscedasticity
- No Missing Features
- No Multicollinearity

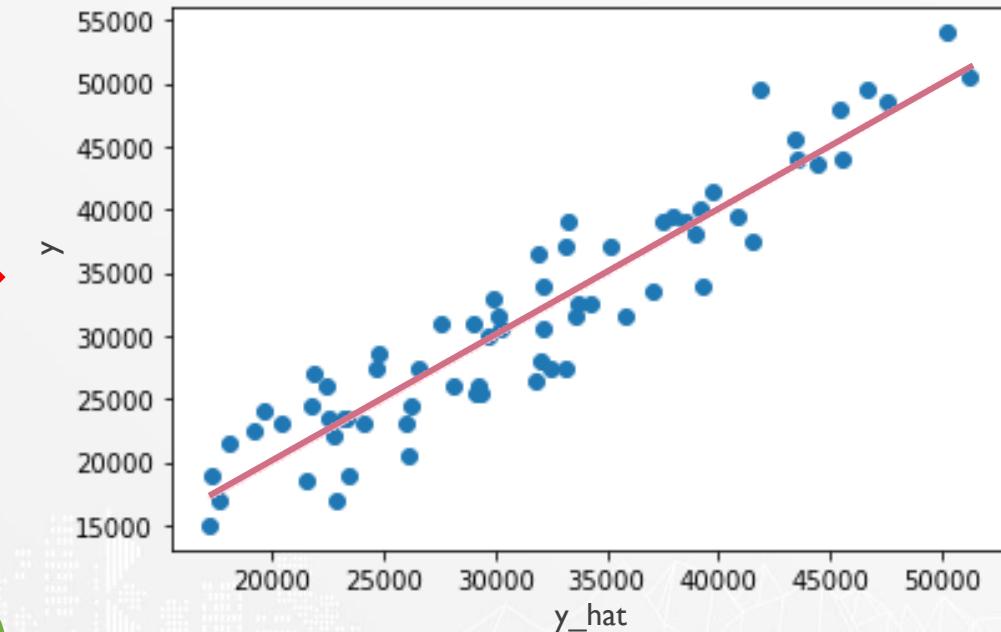
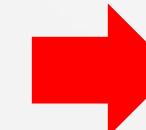
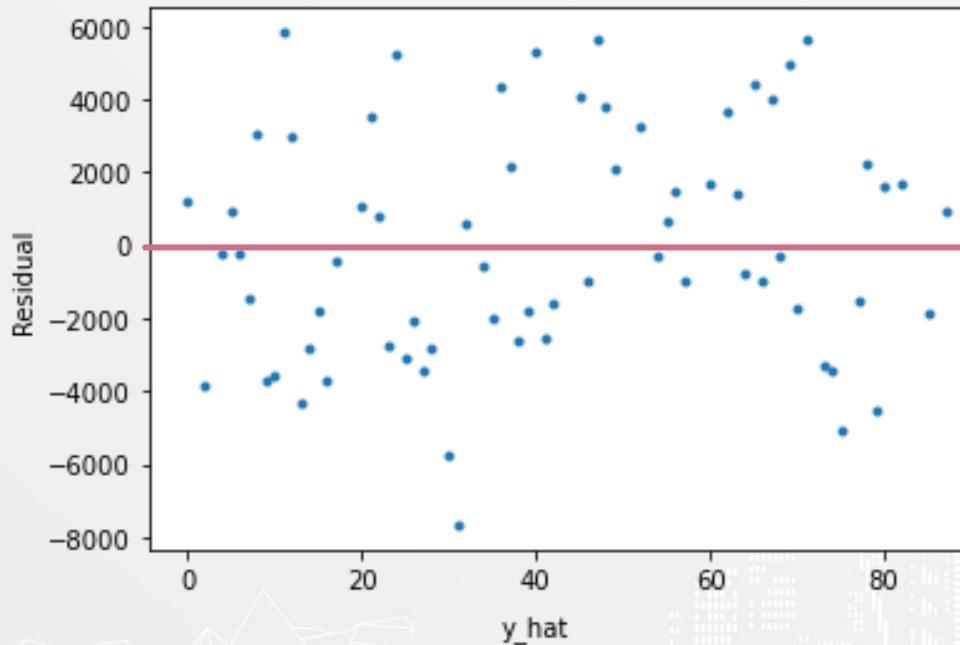
# Homoscedasticity

Homoscedasticity : variance ของ residuals เป็นค่าคงที่ต่ออดค่า  $\hat{y}$



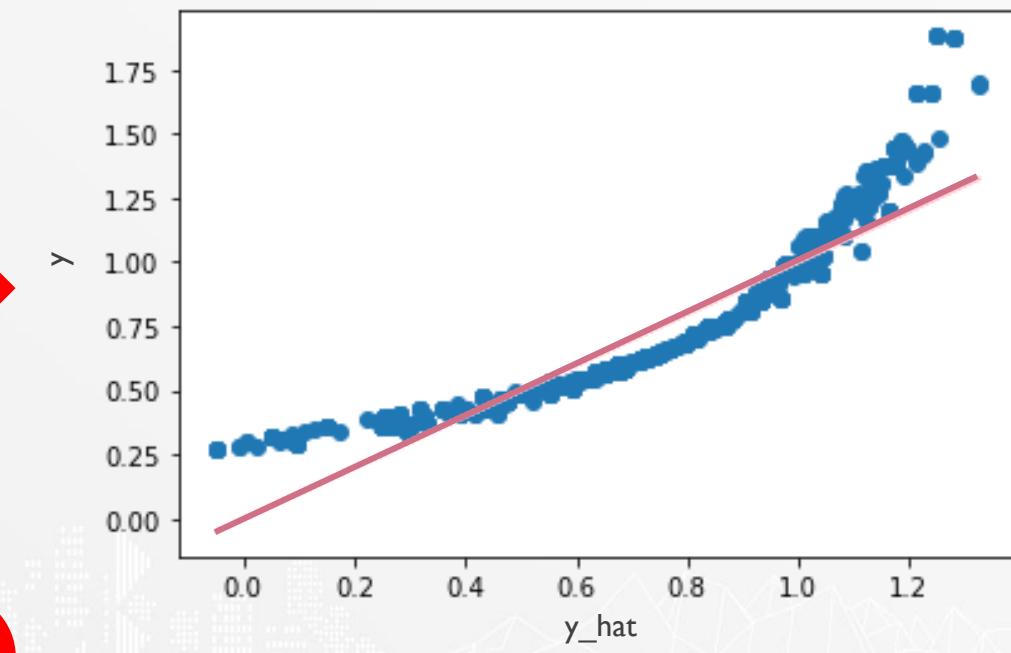
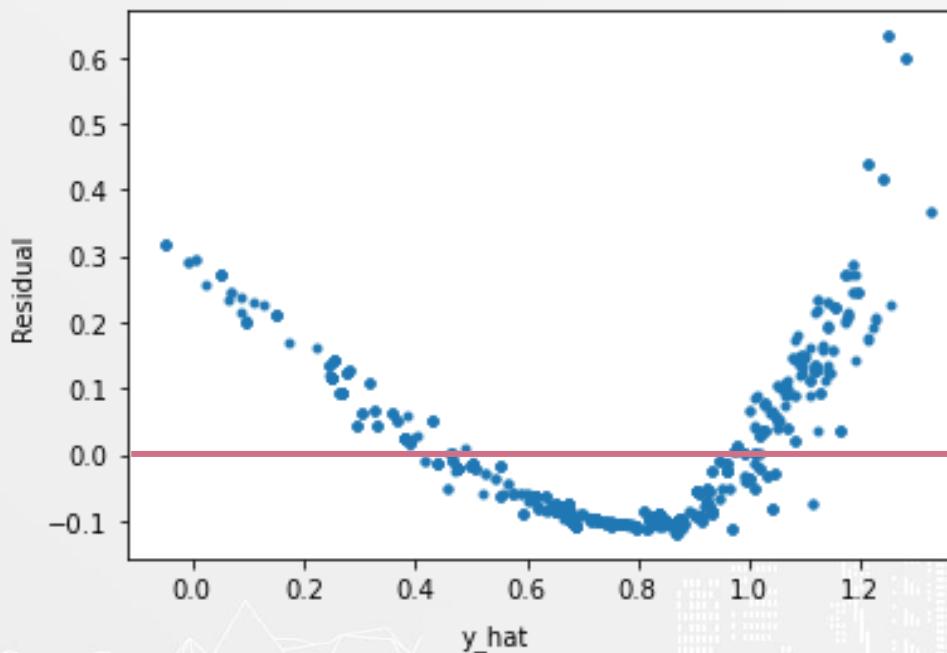
# Homoscedasticity

เหตุผลที่เป็นเช่นนี้ เพราะ residuals ต้องกระจายตัวแบบเดิมถึงแม้  $\hat{y}$  จะมีค่าเพิ่มขึ้น



# Homoscedasticity

เหตุผลที่เป็นเช่นนี้ เพราะ residuals ต้องกระจายตัวแบบเดิมถึงแม้  $\hat{y}$  จะมีค่าเพิ่มขึ้น



# Assumption

- Linear Relationship**
- Normality of Residuals**
- Homoscedasticity**
- No Missing Features**
- No Multicollinearity**

# Assumption



## CHECK ASSUMPTION



**01. MARKETING**



**02. INVESTMENT**



**03. SMART FARM**



**04. BIKE RENTAL**



**05. INSURANCE**

# Assumption

		Linear Relationship (Scatter plot)	Normality of Residuals	Homoscedasticity
MARKETING		✓	✗	✓
INVESTMENT	SET50	✓	○	✓
	EURUSD	✓	○	✓
	XAUUSD	✓	○	○
	BTCUSD	✓	○	✗

✓ = ถูกต้องตาม assumption

○ = พยายามรับได้ว่า  
เป็นไปตาม assumption

✗ = ไม่ถูกต้องตาม assumption

# Assumption

		Linear Relationship (Scatter plot)	Normality of Residuals	Homoscedasticity
SMART FARM	RICE	○	○	✗
	BANANA	✗	○	✗
	MELON	✗	○	✗
BIKE RENTAL		✓	○	○
	SMOKER	✓	✓	✓
INSURANCE	NON SMOKER	✗	✗	✗

✓ = ถูกต้องตาม assumption

○ = พยายามรับได้ว่า  
เป็นไปตาม assumption

✗ = ไม่ถูกต้องตาม assumption

# Assumption

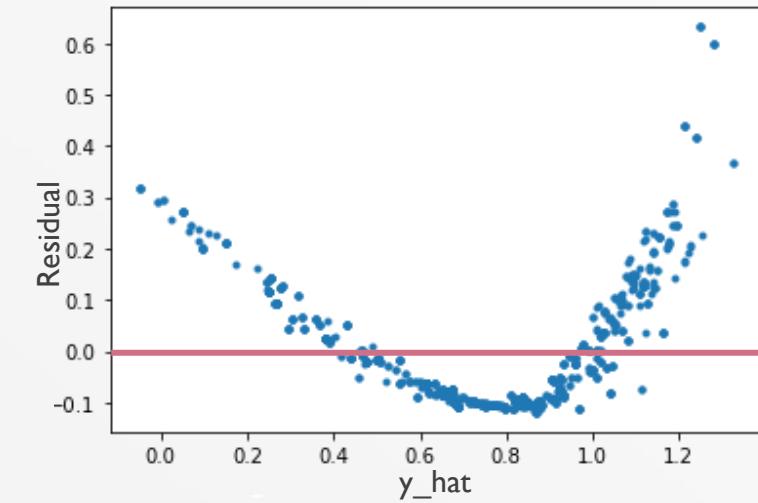
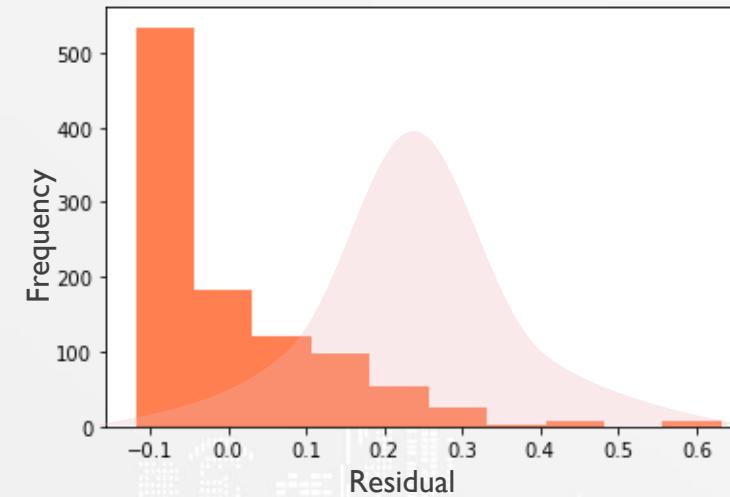
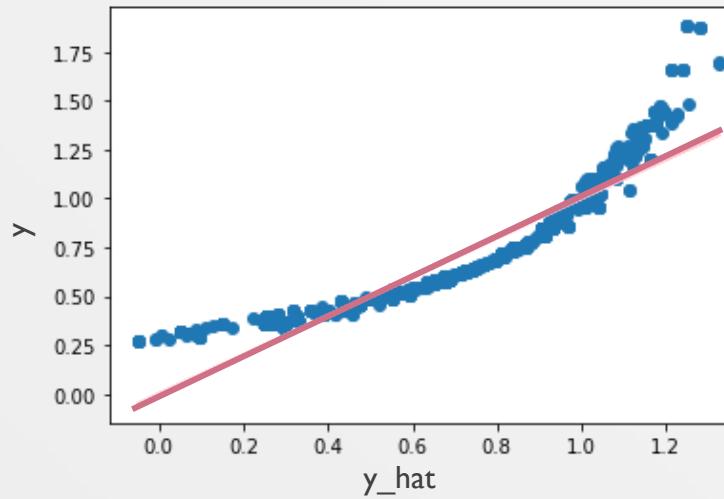
## Note

**Q: ถ้าก็ง 3 ข้อนี้ไม่เป็นจริง ต้องทำอย่างไร?**

- Scatter plot ระหว่าง  $\hat{y}$  กับ  $y$
- Assumption ข้อที่ 2
- Assumption ข้อที่ 3

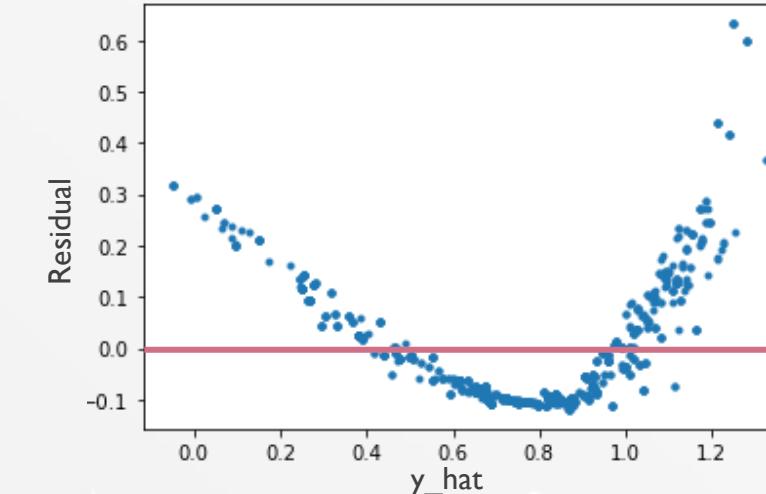
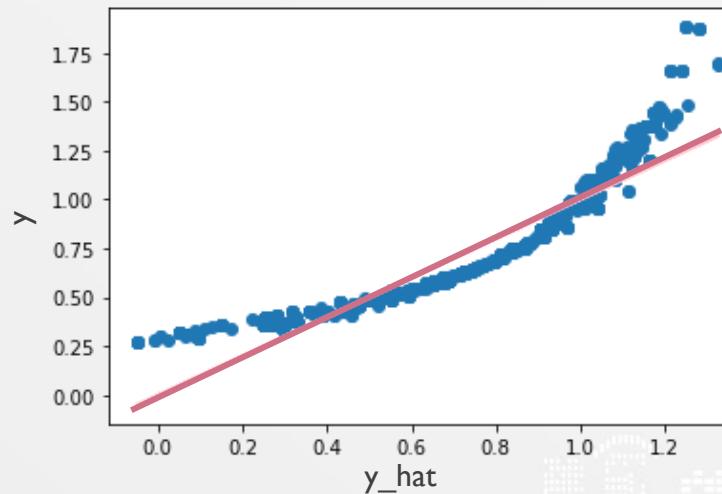
# Assumption

## Note



# Assumption

## Note



# Assumption

## Note

**Nonlinear  
Transformation**

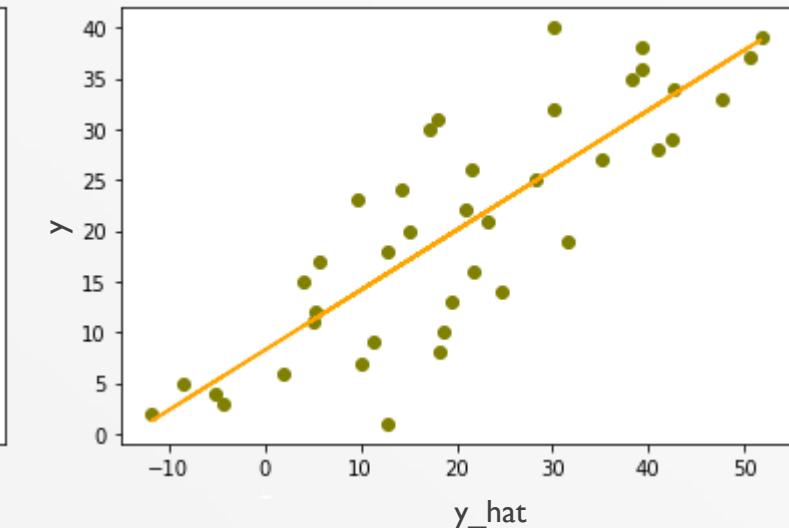
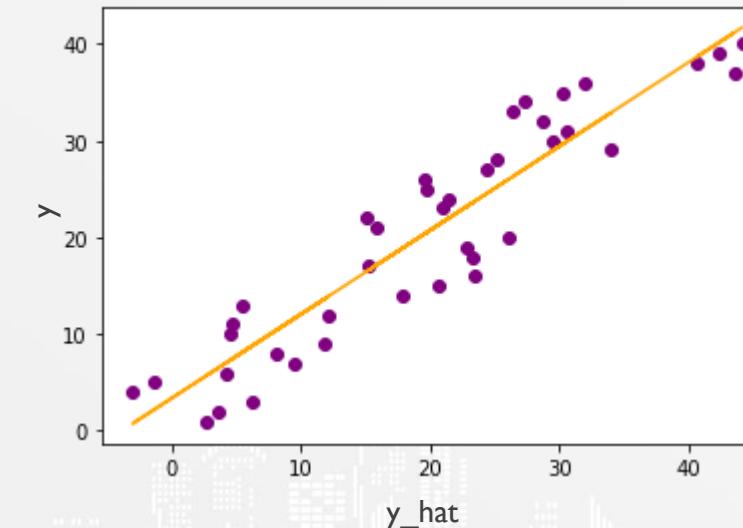
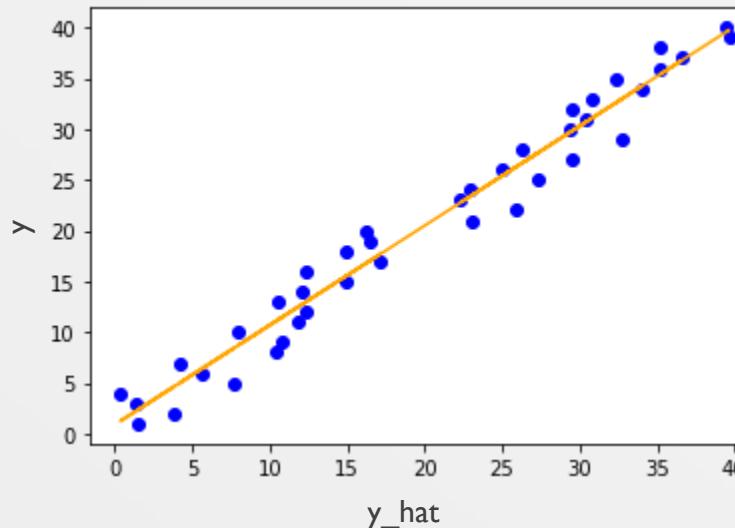
**Consider Other  
Models such as  
Neural Network**

Further reading :

<https://people.revoledu.com/kardi/tutorial/Regression/nonlinear/NonLinearTransformation.htm>

# No Missing Features

No missing features : มีตัวแปรต้นที่ใช้ในการพยากรณ์  $\hat{y}$  อย่างครบถ้วน



# No Missing Features

เหตุที่เป็นเช่นนี้ หรือว่าได้ด้วยสมการดังต่อไปนี้

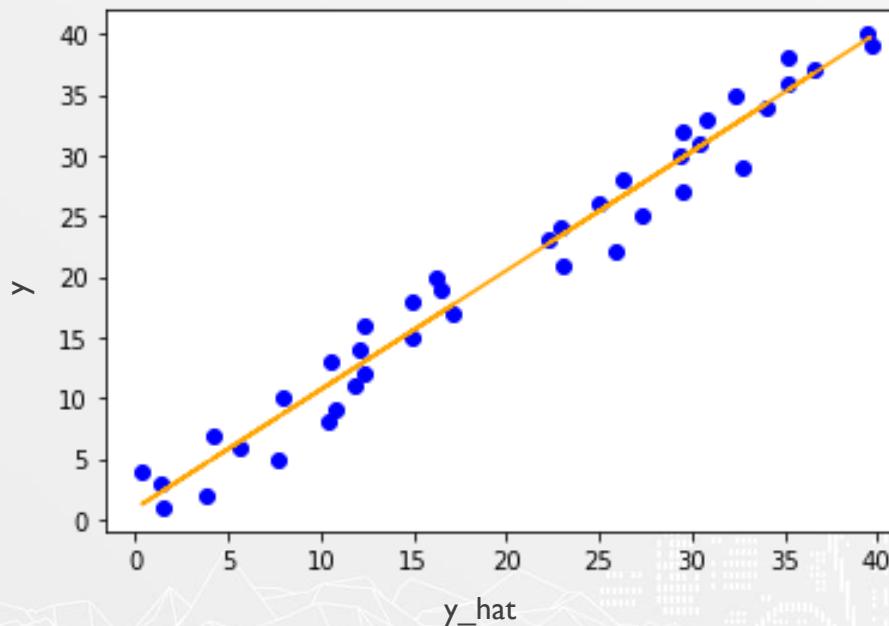
$$y = \hat{y} + \varepsilon$$

และ

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

# No Missing Features

สมมติให้ ตัวแปรต้นที่เกี่ยวข้องกับการพยากรณ์  $\hat{y}$  มี 4 ตัว ได้แก่  $x_1, x_2, x_3, x_4$   
และตัวแปรต้นที่เราใช้สร้าง model มี 4 ตัว ได้แก่  $x_1, x_2, x_3, x_4$

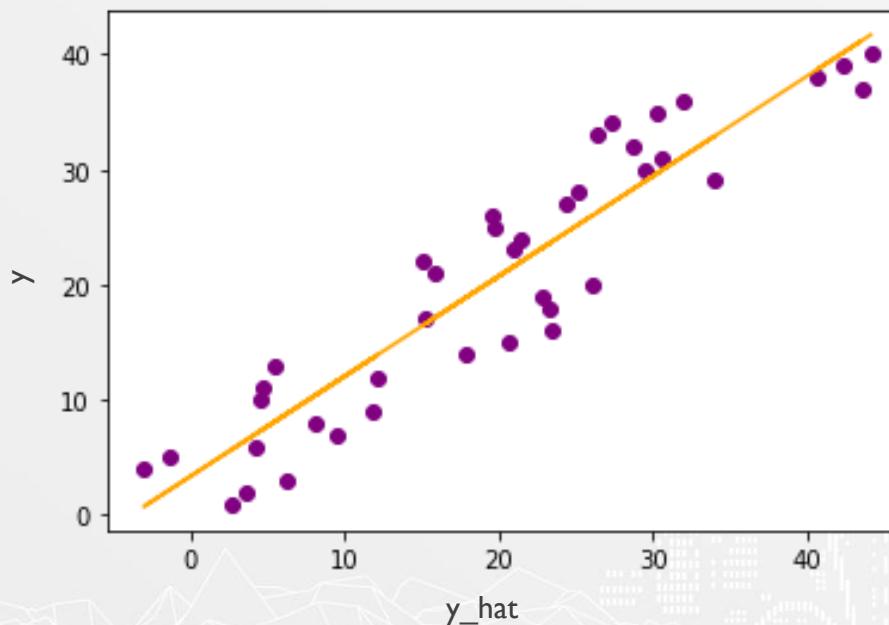


$$y = \hat{y} + \varepsilon$$

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$$

# No Missing Features

สมมติให้ ตัวแปรต้นที่เกี่ยวข้องกับการพยากรณ์  $\hat{y}$  มี 4 ตัว ได้แก่  $x_1, x_2, x_3, x_4$  และตัวแปรต้นที่เราใช้สร้าง model มี 3 ตัว ได้แก่  $x_1, x_2, x_4$

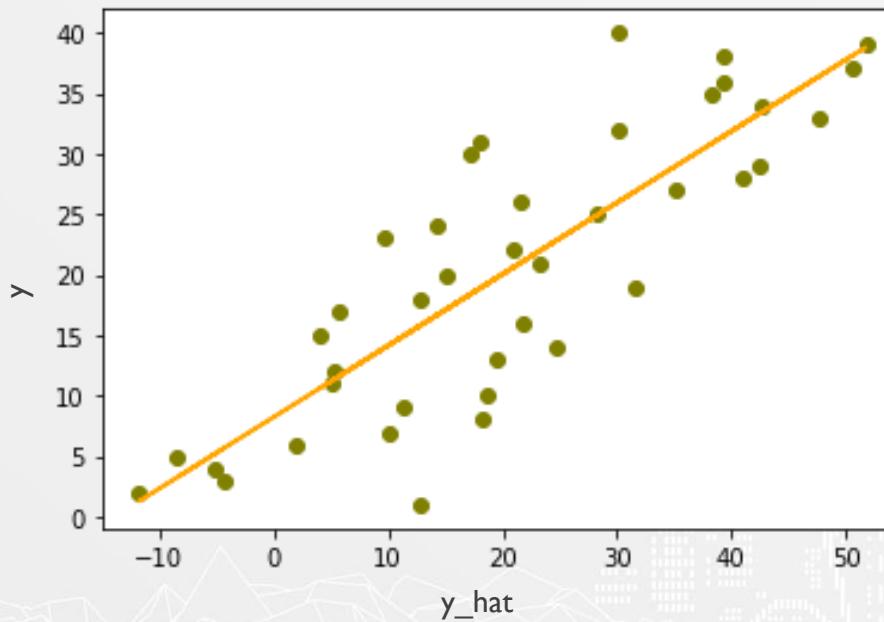


$$y = \hat{y} + \boldsymbol{\varepsilon}$$

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$$

# No Missing Features

สมมติให้ ตัวแปรต้นที่เกี่ยวข้องกับการพยากรณ์  $\hat{y}$  มี 4 ตัว ได้แก่  $x_1, x_2, x_3, x_4$  และตัวแปรต้นที่เราใช้สร้าง model มี 2 ตัว ได้แก่  $x_1, x_2$



$$y = \hat{y} + \epsilon$$

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$$

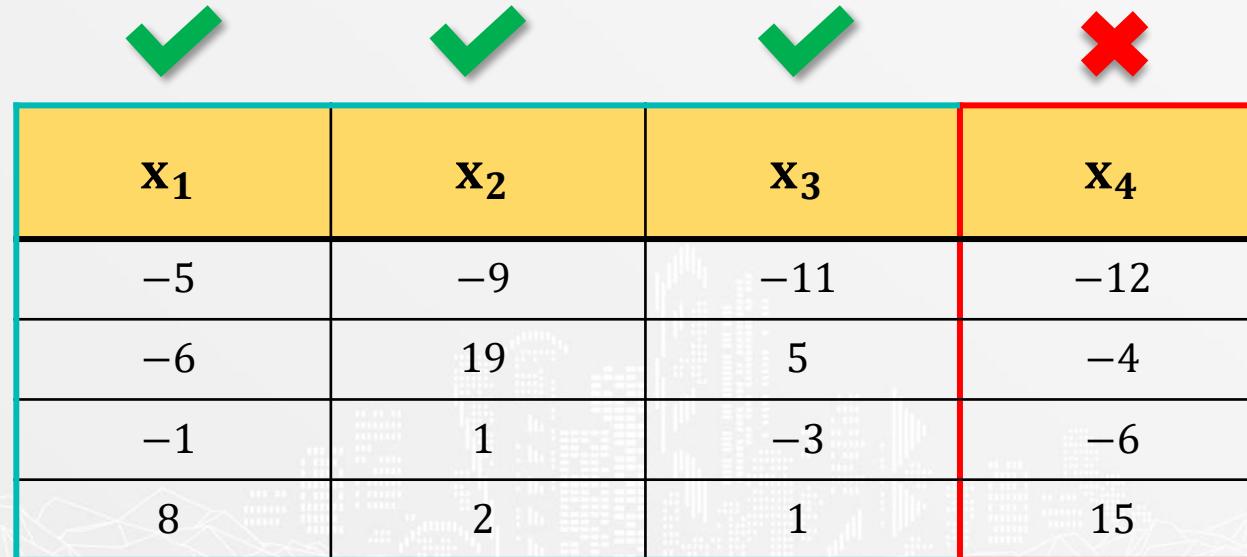
# Assumption

- Linear Relationship**
- Normality of Residuals**
- Homoscedasticity**
- No Missing Features**
- No Multicollinearity**

# No Multicollinearity

No Multicollinearity : ตัวแปรตัวตัวต้องไม่มีความสัมพันธ์เชิงเส้นต่อกัน (ไม่สามารถใช้ตัวแปรตัวเพื่อพยากรณ์ตัวแปรตัวอื่นได้)

## ตัวอย่าง (1)



$x_1$	$x_2$	$x_3$	$x_4$
-5	-9	-11	-12
-6	19	5	-4
-1	1	-3	-6
8	2	1	15

# No Multicollinearity

ตัวอย่าง (1)

<b><math>x_1</math></b>	<b><math>x_2</math></b>	<b><math>x_3</math></b>	<b><math>x_4</math></b>
-5	-9	-11	-12
-6	19	5	-4
-1	1	-3	-6
8	2	1	15

# No Multicollinearity

ตัวอย่าง (1)

$$\begin{bmatrix} \mathbf{x}_4 \\ -12 \\ -4 \\ -6 \\ 15 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_4 \\ -12 \\ -4 \\ -6 \\ 15 \end{bmatrix} = 2 \begin{bmatrix} \mathbf{x}_1 \\ -5 \\ -6 \\ -1 \\ 8 \end{bmatrix} - \begin{bmatrix} \mathbf{x}_2 \\ -9 \\ 19 \\ 1 \\ 2 \end{bmatrix} + \begin{bmatrix} \mathbf{x}_3 \\ -11 \\ 5 \\ -3 \\ 1 \end{bmatrix}$$

# No Multicollinearity

ตัวอย่าง (1)

$$\begin{matrix} x_4 \\ -12 \\ -4 \\ -6 \\ 15 \end{matrix} = \begin{matrix} \hat{x}_4 \\ -12 \\ -4 \\ -6 \\ 15 \end{matrix}$$

**“Perfect Multicollinearity”  
or  
“Linearly Dependent”**

# No Multicollinearity

ตัวอย่าง (1)

<b><math>x_1</math></b>	<b><math>x_2</math></b>	<b><math>x_3</math></b>	<b><math>x_4</math></b>
-5	-9	-11	-12
-6	19	5	-4
-1	1	-3	-6
8	2	1	15

# No Multicollinearity

ตัวอย่าง (1)

$x_1$	$x_2$	$x_3$
-5	-9	-11
-6	19	5
-1	1	-3
8	2	1

# No Multicollinearity

ตัวอย่าง (2)

<b><math>x_1</math></b>	<b><math>x_2</math></b>	<b><math>x_3</math></b>	<b><math>x_4</math></b>
-2	3	6	4
1	2	5	1
3	-1	-2	2
-1	-3	1	5

# No Multicollinearity

## ตัวอย่าง (2)

$$\begin{array}{c|c|c|c|c|c} \mathbf{x}_3 & \approx & \hat{\mathbf{x}}_3 & = & -0.34 & \\ \hline 6 & & 7.04 & & & \\ \hline 5 & & 2.75 & & & \\ \hline -2 & & -0.84 & & & \\ \hline 1 & & 0.19 & & & \\ \hline \end{array} \quad \begin{array}{c|c|c|c|c|c} \mathbf{x}_1 & & \mathbf{x}_2 & & \mathbf{x}_4 & \\ \hline -2 & & 3 & & 4 & \\ \hline 1 & & 2 & & 1 & \\ \hline 3 & & -1 & & 2 & \\ \hline -1 & & -3 & & 5 & \\ \hline \end{array} \quad \begin{array}{c|c} +1.20 & +0.69 \\ \hline \end{array}$$

# No Multicollinearity

ตัวอย่าง (2)

$\mathbf{x}_3$	$\hat{\mathbf{x}}_3$
6	7.04
5	2.75
-2	-0.84
1	0.19

$\approx$

$$R_{\mathbf{x}_3}^2 = 0.80$$

**“Multicollinearity”**

# No Multicollinearity

เกณฑ์การวัดค่า  $R^2$

$R^2 > 0.6$

$R^2 > 0.8$

$R^2 > 0.9$

# No Multicollinearity

ตัวอย่าง (2)

<b><math>x_1</math></b>	<b><math>x_2</math></b>	<b><math>x_3</math></b>	<b><math>x_4</math></b>
-2	3	6	4
1	2	5	1
3	-1	-2	2
-1	-3	1	5

# No Multicollinearity

ตัวอย่าง (2)

$x_1$	$x_2$	$x_4$
-2	3	4
1	2	1
3	-1	2
-1	-3	5

# No Multicollinearity

เครื่องมือทางคณิตศาสตร์ที่ใช้ตรวจสอบความสัมพันธ์เชิงเส้น  
ของตัวแปรตัวนึงมีชื่อว่า **Variance Inflation Factor (VIF)**

Feature Reading : Variance Inflation Factor (VIF)

# No Multicollinearity

## Note

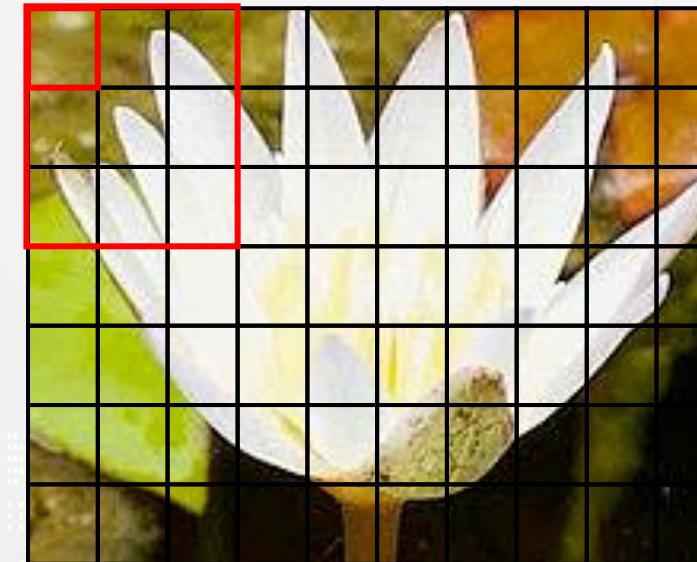
**“ในงานวิจัยสมัยใหม่ โดยเฉพาะด้าน neural network หรือ deep learning (ซึ่งมีรากฐานมาจาก linear regression โดยตรง) การที่เราจะรักษา assumption ข้อนี้ไว้ แทบจะเป็นไปไม่ได้เลย ”**

# No Multicollinearity

## Note

เพรา: feature ในโลกความจริงนั้นมีความขึ้นต่อกันสูง ยกตัวอย่างเช่น

### 1. ข้อมูลรูปภาพ



Ref : [https://en.wikipedia.org/wiki/Digital\\_image\\_processing#/media/File:Lotus\\_free.jpg](https://en.wikipedia.org/wiki/Digital_image_processing#/media/File:Lotus_free.jpg)

# No Multicollinearity

## Note

2. Feature ที่เกิดจากการทำ one hot encoding (ซึ่งก่อให้เกิด linearly dependent แน่นอน)

$x_0$	$x_1$	$x_2$	$x_3$
1	0	1	0
1	0	0	1
1	1	0	0
1	1	0	0

# No Multicollinearity

## Note

3. Feature ที่เกิดจากการ encode ในลักษณะที่คล้ายกับ one hot encoding เช่น count vectorization สำหรับข้อมูลที่เป็นภาษา

	and	document	first	is	one	second	the	third	this
This is the first document	0	1	1	1	0	0	1	0	1
This document is the second document	0	2	0	1	0	1	1	0	1
And this is the third one.	1	0	0	1	1	0	1	1	1
Is this the first document	0	1	1	1	0	0	1	0	1

# No Multicollinearity

Q : ถ้า feature ที่ใช้มี multicollinearity จะส่งผลอย่างไร?

# No Multicollinearity

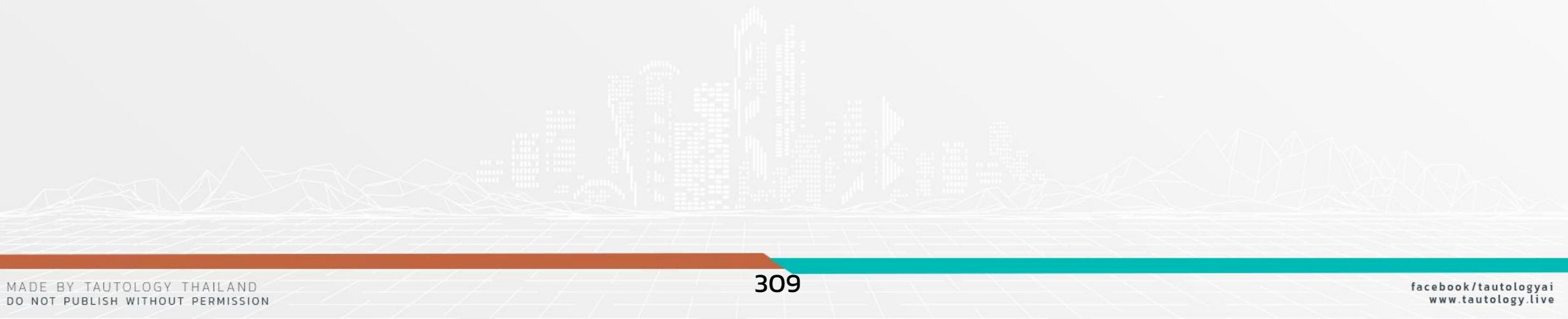
- **Effect of Multicollinearity**

- **Effect of Linearly Dependent**

# No Multicollinearity

- **Effect of Multicollinearity**

- ✓ ไม่สามารถวัด feature importance ด้วย p-value ได้
- ✓ ไม่ส่งผลกระทบต่อ performance ของ model



# No Multicollinearity

- **Effect of Linearly Dependent**

- ไม่สามารถวัดรากของ feature importance ด้วย p-value ได้
- ✗ มีโอกาสสูงที่จะส่งผลกระทบต่อ performance ของ model

# Assumption

- Linear Relationship**
- Normality of Residuals**
- Homoscedasticity**
- No Missing Features**
- No Multicollinearity**

# Model Improvement





# Problem with Linearly Dependent



# Problem with Linearly Dependent

- ถ้า  $X_b$  เป็น linearly dependent

$$X_b = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

$$x_0 = x_1 + x_2 + x_3$$

# Problem with Linearly Dependent

แล้ว  $X_b^T X_b$  เป็น linearly dependent

$$X_b^T X_b = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

# Problem with Linearly Dependent

- ถ้า  $X_b^T X_b$  เป็น linearly dependent

$$X_b^T X_b = \begin{bmatrix} 3 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$x_0 = x_1 + x_2 + x_3$$

# Problem with Linearly Dependent

แล้ว  $X_b^T X_b$  ไม่สามารถหา inverse ได้

**Invertible Matrix Theorem.** Let  $A$  be an  $n \times n$  matrix, and let  $T: \mathbf{R}^n \rightarrow \mathbf{R}^n$  be the matrix transformation  $T(x) = Ax$ . The following statements are equivalent:

1.  $A$  is invertible.
2.  $A$  has  $n$  pivots.
3.  $\text{Nul}(A) = \{0\}$ .
4. The columns of  $A$  are linearly independent.
5. The columns of  $A$  span  $\mathbf{R}^n$ .
6.  $Ax = b$  has a unique solution for each  $b$  in  $\mathbf{R}^n$ .
7.  $T$  is invertible.
8.  $T$  is one-to-one.
9.  $T$  is onto.

Ref : <https://textbooks.math.gatech.edu/ila/invertible-matrix-thm.html>

# Problem with Linearly Dependent

**ดังนี้** เราจะไม่สามารถคำนวณ normal equation ได้

## Normal Equation

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$



# Problem with Linearly Dependent

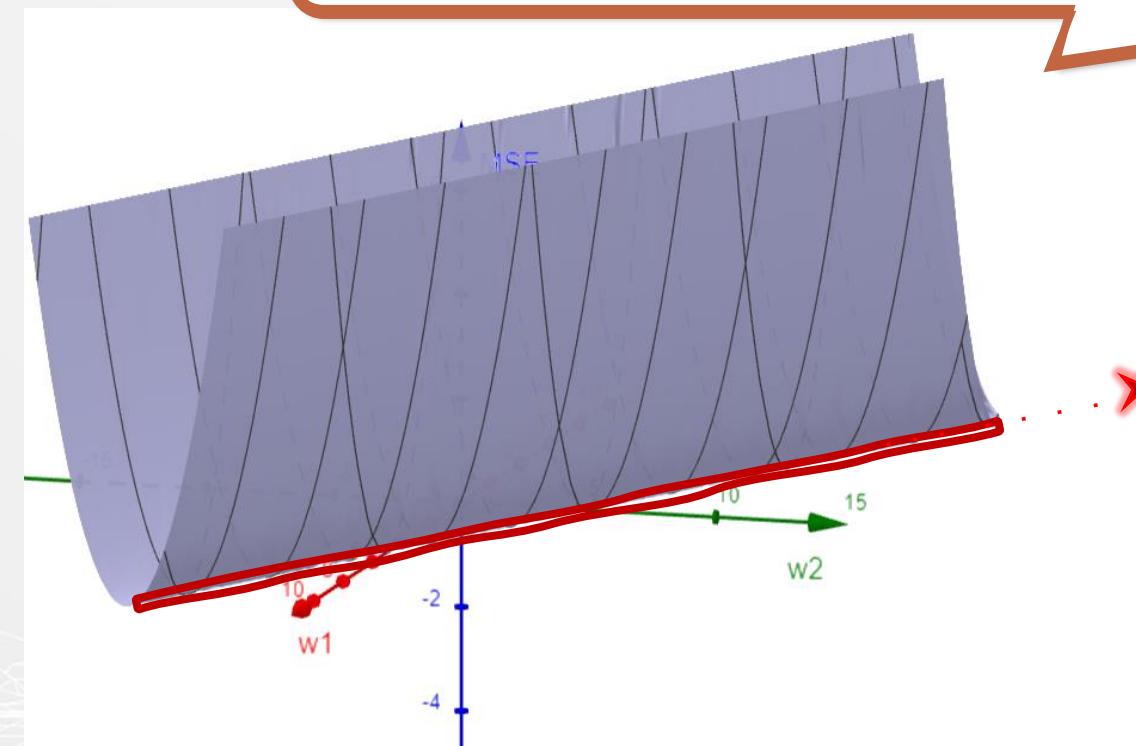
**Q:** ถ้า  $X_b^T X_b$  inverse ไม่ได้ แปลว่าไม่มีคำตอบรึเปล่า?

## Normal Equation

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$

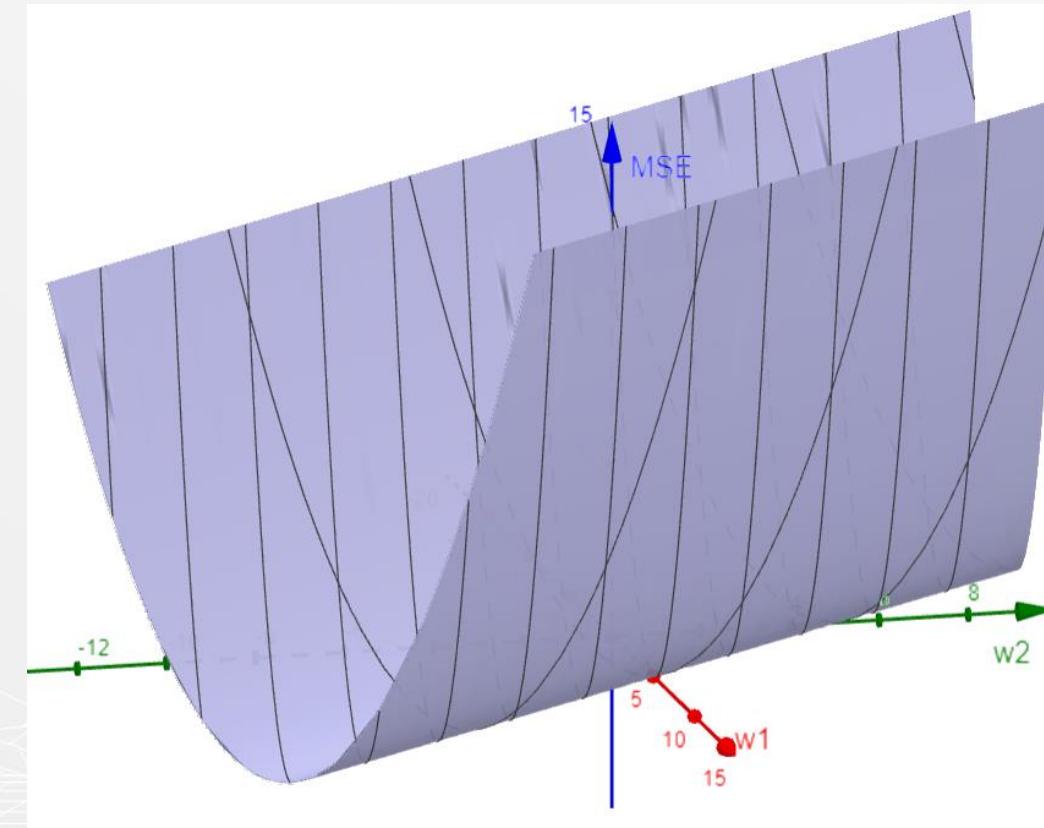
# Problem with Linearly Dependent

A: มีคำตอบ และคำตอบมีมากมายมาก



# Problem with Linearly Dependent

กราฟนี้มารอย่างไร?



# Problem with Linearly Dependent

## ตัวอย่างการหาสมการร่องคำตอบ

$x_1$	$x_2$	$x_3$	y
0	1	0	3.8
0	0	1	5.1
1	0	0	3
0	1	0	4.2
0	0	1	4.9

ตารางแสดง dataset ที่ feature linearly dependent

# Problem with Linearly Dependent

## Normal Equation

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$

$$(X_b^T X_b) \mathbf{w} = X_b^T \mathbf{y}$$

# Problem with Linearly Dependent

$$(X_b^T X_b) \mathbf{w} = X_b^T \mathbf{y}$$



$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

# Problem with Linearly Dependent

$$(X_b^T X_b) \mathbf{w} = X_b^T \mathbf{y}$$

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>
0	1	0	3.8
0	0	1	5.1
1	0	0	3
0	1	0	4.2
0	0	1	4.9



$$X_b = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 3.8 \\ 5.1 \\ 3 \\ 4.2 \\ 4.9 \end{bmatrix}$$

# Problem with Linearly Dependent

เราสามารถคำนวณ  $X_b^T X_b$  และ  $X_b^T y$  ได้ดังนี้

$$\bullet X_b^T X_b = \begin{bmatrix} 5 & 1 & 2 & 2 \\ 1 & 1 & 0 & 0 \\ 2 & 0 & 2 & 0 \\ 2 & 0 & 0 & 2 \end{bmatrix}$$

$$\bullet X_b^T y = \begin{bmatrix} 21 \\ 3 \\ 8 \\ 10 \end{bmatrix}$$

# Problem with Linearly Dependent

จาก normal equation เราจะได้ว่า

$$(X_b^T X_b) \mathbf{w} = X_b^T \mathbf{y}$$

$$\begin{bmatrix} 5 & 1 & 2 & 2 \\ 1 & 1 & 0 & 0 \\ 2 & 0 & 2 & 0 \\ 2 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 21 \\ 3 \\ 8 \\ 10 \end{bmatrix}$$

$$\begin{bmatrix} 5w_0 + w_1 + 2w_2 + 2w_3 \\ w_0 + w_1 \\ 2w_0 + 2w_2 \\ 2w_0 + 2w_3 \end{bmatrix} = \begin{bmatrix} 21 \\ 3 \\ 8 \\ 10 \end{bmatrix}$$

# Problem with Linearly Dependent

จากนั้นทำการแก้สมการโดยตัวแปร

$$5w_0 + w_1 + 2w_2 + 2w_3 = 21$$

$$w_0 + w_1 = 3$$

$$2w_0 + 2w_2 = 8$$

$$2w_0 + 2w_3 = 10$$

เราจะได้สมการของคำตอบเป็น

$$w_0 = 5 - w_3$$

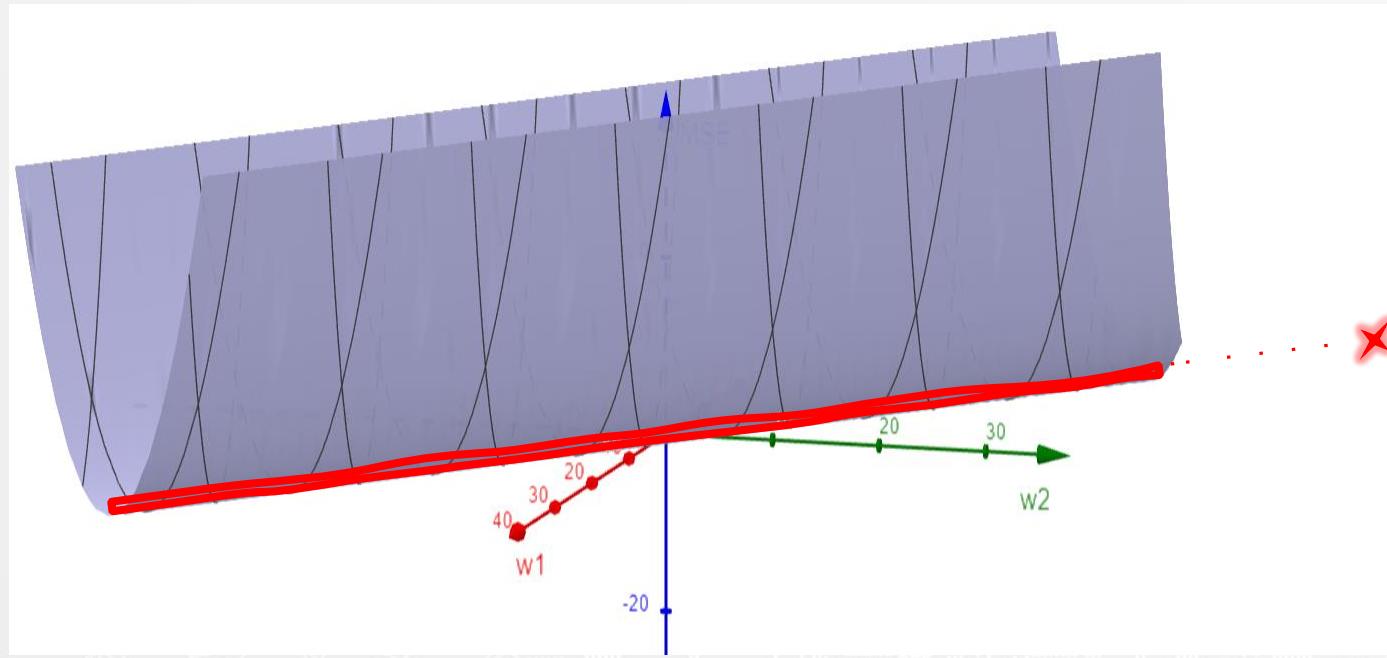
$$w_1 = -2 + w_3$$

$$w_2 = -1 + w_3$$

$$w_3 \in \mathbb{R}$$

# Problem with Linearly Dependent

ซึ่งจะสามารถ plot เป็นกราฟได้ดังนี้



สมการร่องคำตอบ

$$\begin{aligned}w_0 &= 5 - w_3 \\w_1 &= -2 + w_3 \\w_2 &= -1 + w_3 \\w_3 &\in \mathbb{R}\end{aligned}$$

# Problem with Linearly Dependent

Q : แล้วเราจะคำนวณได้อย่างไร?

ในเมื่อ  $X_b^T X_b$  ไม่สามารถหา inverse ได้

## Normal Equation

$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$

# Problem with Linearly Dependent

**A:** เราสามารถใช้ pseudo inverse ในการหา inverse ของ  $X_b^T X_b$  แทนได้

## Normal Equation

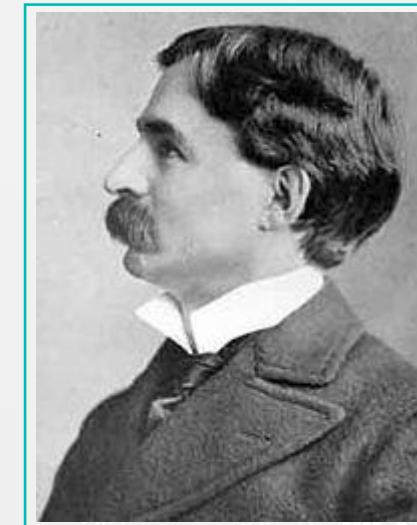
$$\mathbf{w} = (X_b^T X_b)^{-1} X_b^T \mathbf{y}$$



# Problem with Linearly Dependent

ชื่ง pseudo inverse กี่เราจะใช้มีชื่อเรียกว่า

**“Moore-Penrose”**



**Note :** sklearn ใช้ Moore-Penrose เช่นกัน

# Problem with Linearly Dependent

“ จากระสบการณ์ทำงานจริง เมื่อ  $X_b$  มี feature ที่เป็น linearly dependent จำนวนมาก จะทำให้การหาค่าตอบด้วย Moore-Penrose มีปัญหา และนำมาซึ่งค่าตอบที่มีขนาดห์มา ”

# Problem with Linearly Dependent

บางครั้งการฝึกเรียนของ Moore-Penrose นำมาซึ่ง **ค่าตอบที่มีขนาดหิ่ม**

```
1 for coef in reg.coef_:
2     print(coef)
```

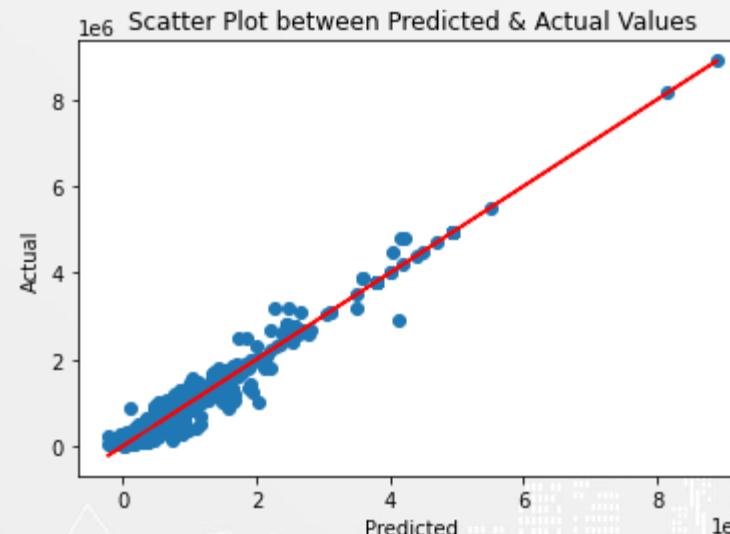
```
-0.4229530799092025
-3727805854.664848
-3727880650.457581
-7401406299.58381
-12440201347.051556
-12439757389.52999
-12439097475.793343
-12438372051.419
-12439847778.274954
:
:
```

รูปแสดงค่า weight ของ dataset car price

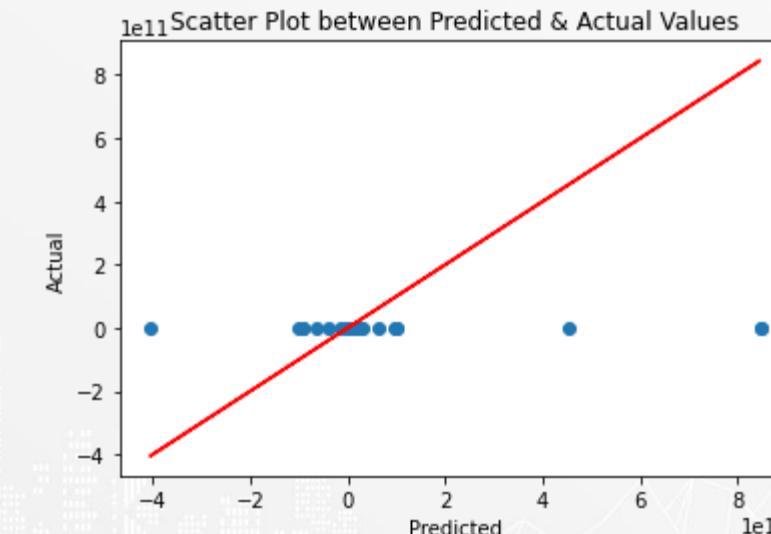
# Problem with Linearly Dependent

การที่ **คำตอบมีขนาดฟื้น** จะทำให้

**“performance ของ model แย่ เมื่อเจอกับ unseen data”**



กราฟแสดงข้อมูลระหว่างค่าจริง และค่าพยากรณ์  
บน training set ของ car price



กราฟแสดงข้อมูลระหว่างค่าจริง และค่าพยากรณ์  
บน test set ของ car price

# Problem with Linearly Dependent

ตัวอย่าง

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>
0	1	0	3.8
0	0	1	5.1
1	0	0	3
0	1	0	4.2
0	0	1	4.9

ตารางแสดง dataset ที่ feature linearly dependent

# Problem with Linearly Dependent

ตัวอย่าง

$$w_0 = 5 - w_3$$

$$w_1 = -2 + w_3$$

$$w_2 = -1 + w_3$$

$$w_3 \in \mathbb{R}$$

# Problem with Linearly Dependent

## ตัวอย่าง

ให้  $w_3 = 1,000,000$

จะได้  $w_0 = 5 - 1,000,000 = -999,995$

$w_1 = -2 + 1,000,000 = 999,998$

$w_2 = -1 + 1,000,000 = 999,999$

# Problem with Linearly Dependent

ตัวอย่าง

$$\hat{y} = -999,995 + 999,998 x_1 + 999,999 x_2 + 1,000,000 x_3$$

- Training set

$x_1$	$x_2$	$x_3$	$y$	$\hat{y}$
0	1	0	3.8	4
0	0	1	5.1	5
1	0	0	3	3
0	1	0	4.2	4
0	0	1	4.9	5

# Problem with Linearly Dependent

## ตัวอย่าง

$$\hat{y} = -999,995 + 999,998 x_1 + 999,999 x_2 + 1,000,000 x_3$$

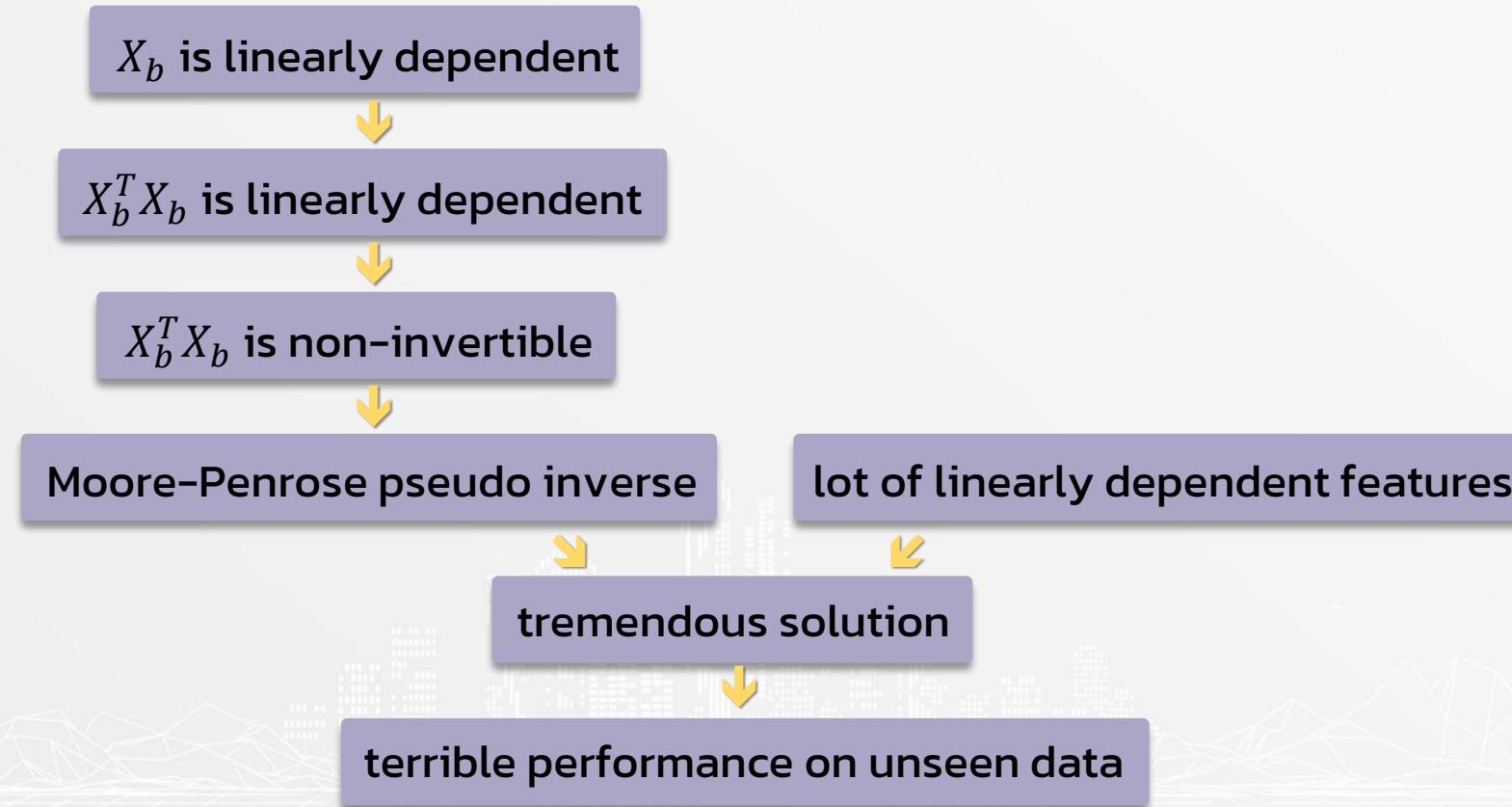
- Test set

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>	<b><math>\hat{y}</math></b>
0	0	0	1	-999,995

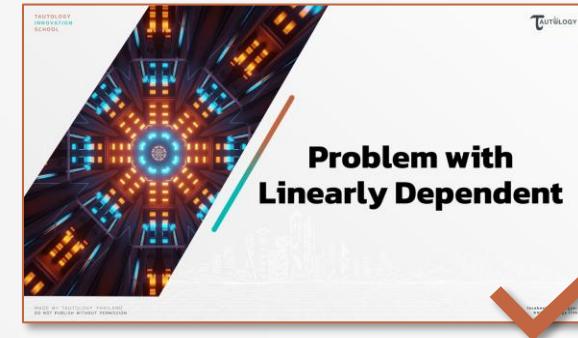
“performance ของ model แม้  
เมื่อเจอกับ unseen data”

# Problem with Linearly Dependent

## Conclusion



# Model Improvement



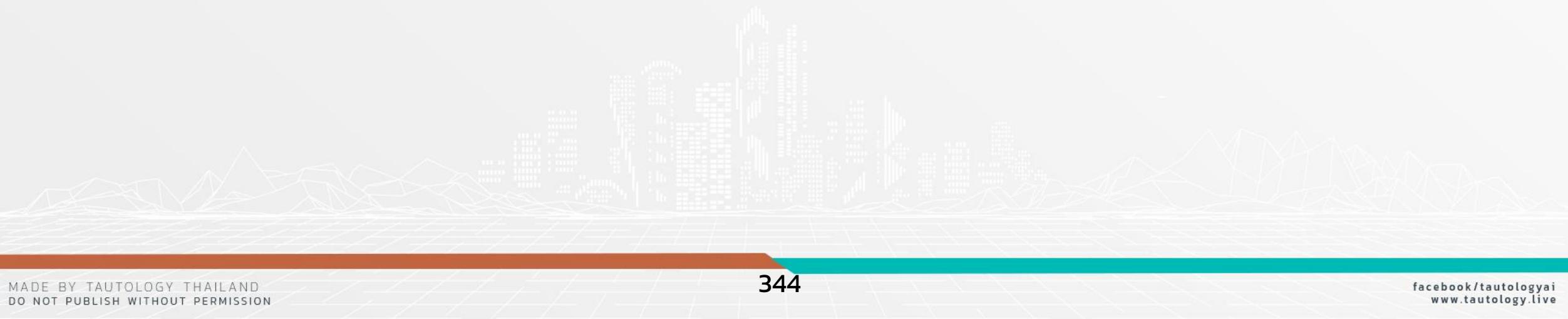


# Solution

# Solution

เพื่อแก้ปัญหานี้ นักคณิตศาสตร์จึงเกิดแนวคิดว่า

“ อยากหาคำตอบ ( $w$ ) บนร่องน้ำที่มีขนาดเล็กที่สุด ”



# Solution

$$\sum_{j=0}^p w_j$$

$$\sum_{j=0}^p w_j^2$$

$$\sum_{j=0}^p |w_j|$$

# Solution

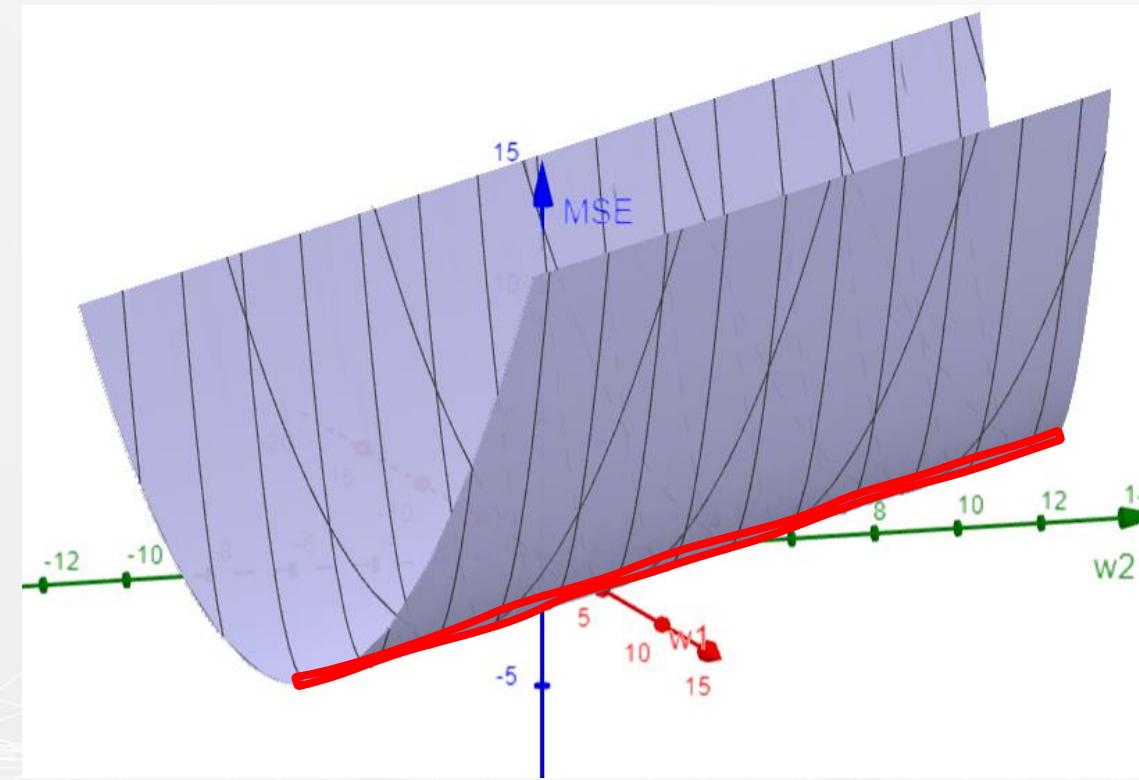
$$\sum_{j=0}^p w_j$$

$$\sum_{j=0}^p w_j^2$$

$$\sum_{j=0}^p |w_j|$$

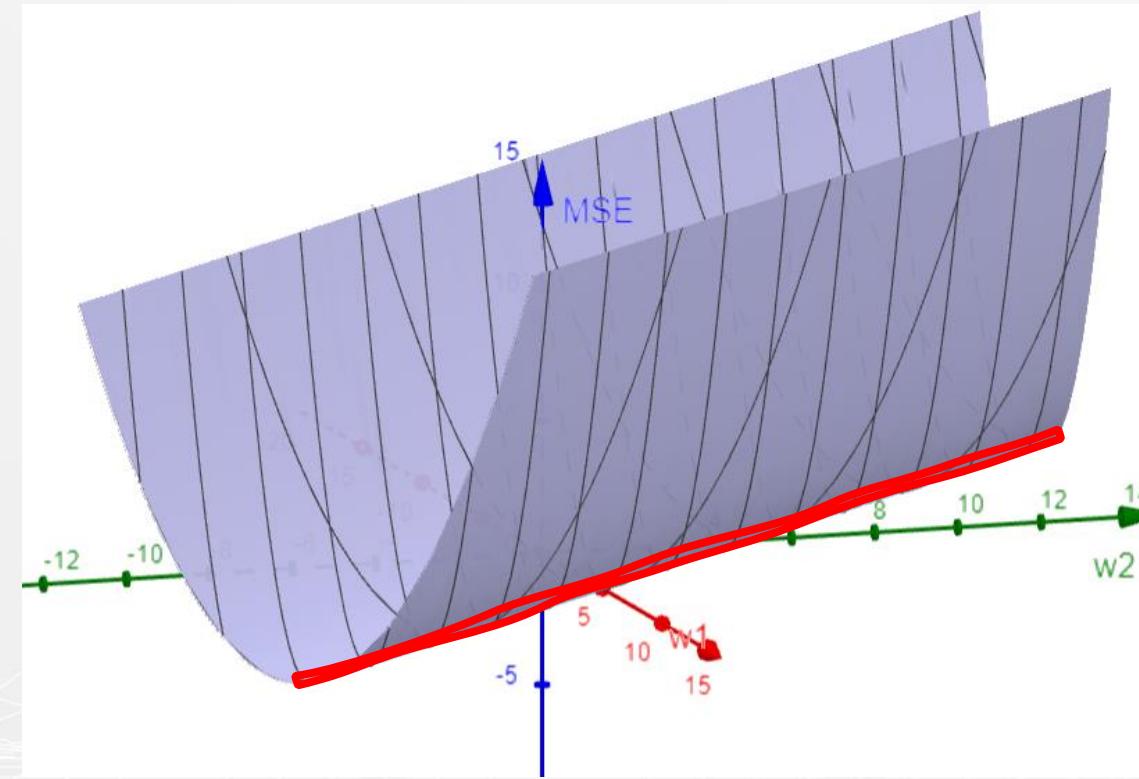
# Solution

“ คำตอบ ( $w$ ) บนร่องที่มีขนาดเล็กที่สุด ”



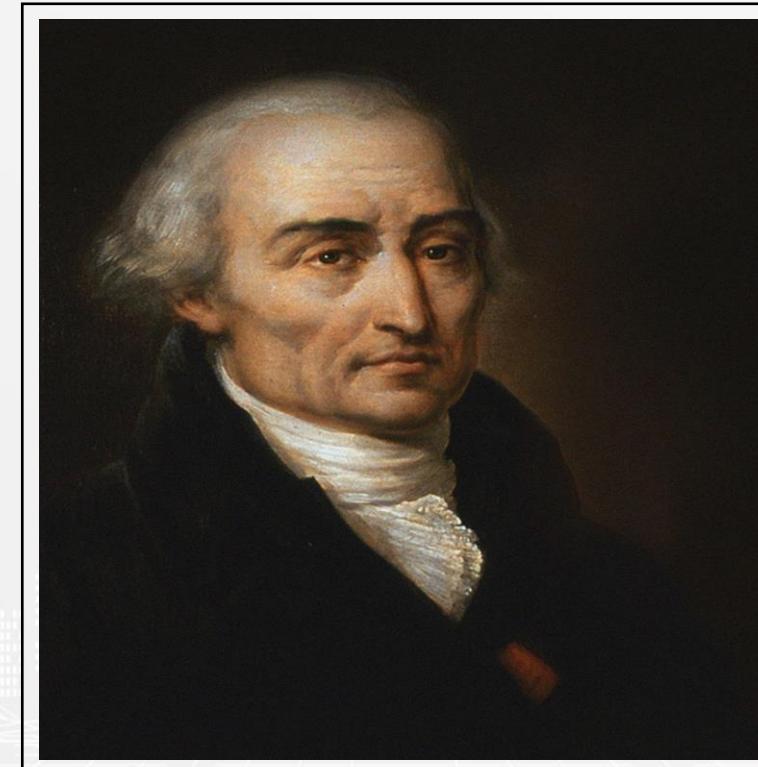
# Solution

“ คำตอบ ( $w$ ) บรร่องที่  $\sum_{j=0}^p w_j^2$  มีค่าน้อยที่สุด ”



# Solution

คณิตศาสตร์ที่เราใช้แก้ปัญหานี้คือ **Lagrange Multipliers**



# Solution

**minimize**  $\sum_{j=0}^p w_j^2$

**subject to** สมการร่องคำตอบ

# Solution

ตัวอย่าง

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>
0	1	0	4
0	0	1	5
1	0	0	3

ตารางแสดง dataset ที่ feature linearly dependent

# Solution

ຕັວອຍ່າງ

$$w_0 = 5 - w_3$$

$$w_1 = -2 + w_3$$

$$w_2 = -1 + w_3$$

$$w_3 \in \mathbb{R}$$

# Solution

ตัวอย่าง

**minimize**  $\sum_{j=0}^p w_j^2$

**subject to**  $w_0 = 5 - w_3$

$w_1 = -2 + w_3$

$w_2 = -1 + w_3$

# Solution

ตัวอย่าง

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 1 \\ 2 \end{bmatrix}$$

# Solution

ตัวอย่าง

$$\hat{y} = 3 + 0x_1 + x_2 + 2x_3$$

- Training set

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>	<b><math>\hat{y}</math></b>
0	1	0	4	4
0	0	1	5	5
1	0	0	3	3

# Solution

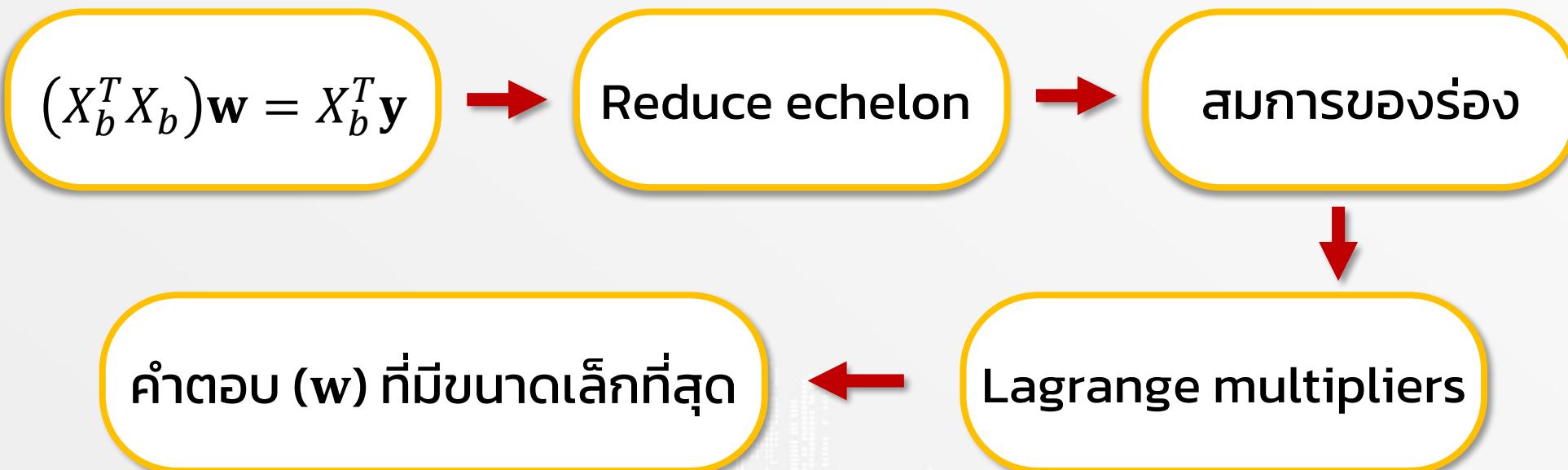
ตัวอย่าง

$$\hat{y} = 3 + 0x_1 + x_2 + 2x_3$$

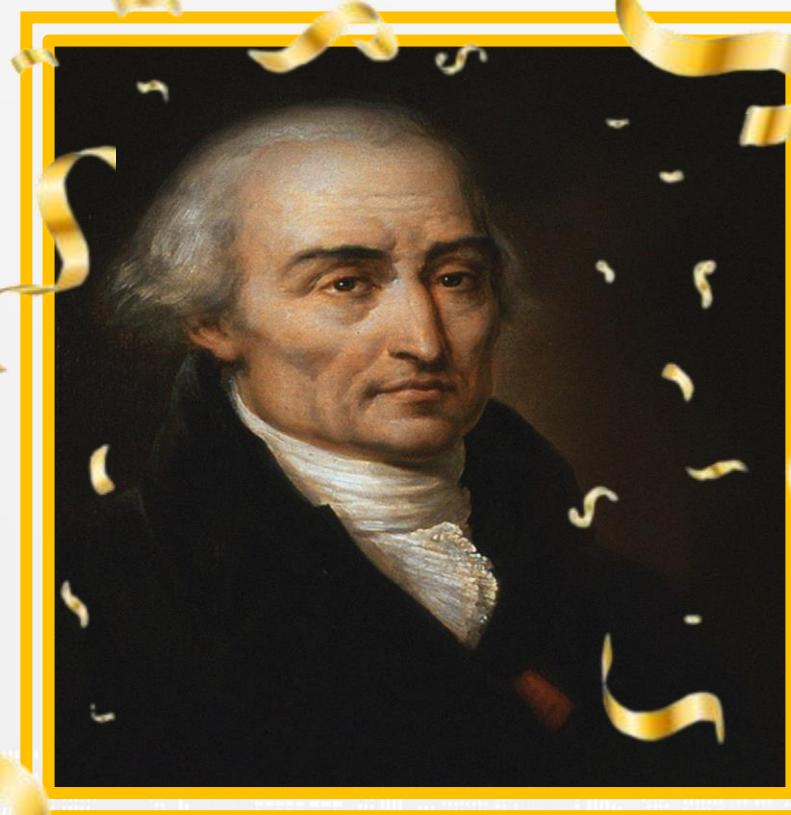
- Test set

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>	<b><math>\hat{y}</math></b>
0	0	0	1	3

# Solution



# Solution



# Solution

จากการพิจารณา นักคณิตศาสตร์คันพบร่วม

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



**สมการของร่อง**

# Solution

นักคณิตศาสตร์จึงเกิดแนวคิดต่อไปนี้

**minimize**  $\sum_{j=0}^p w_j^2$

**subject to** สมการของร่อง



**minimize**  $\sum_{j=0}^p w_j^2$

**subject to**  $\sum_{i=1}^n (y_i - \hat{y}_i)^2 = c$

# Solution

**minimize**  $\sum_{j=0}^p w_j^2$

**subject to**  $\sum_{i=1}^n (y_i - \hat{y}_i)^2 = c$



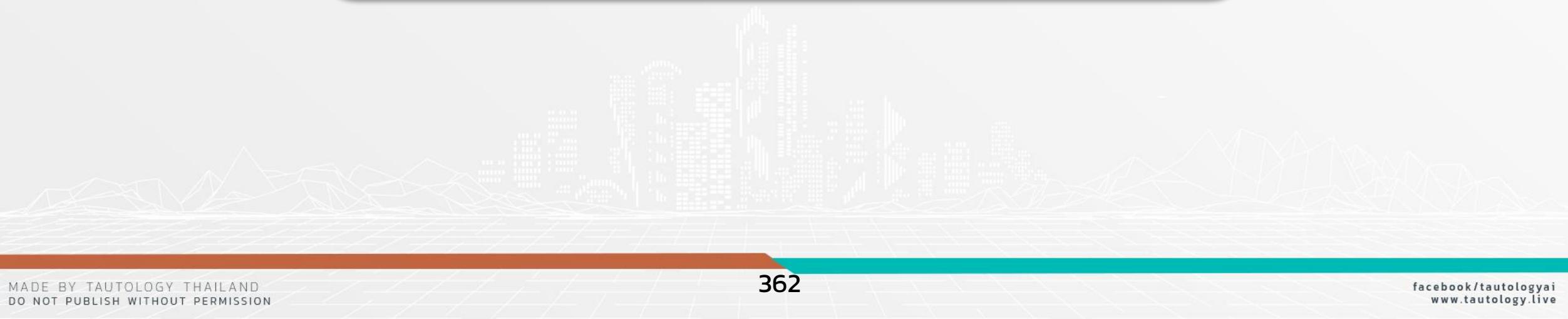
**minimize**  $Cost (\nabla Cost = 0)$

ໄລຍ້  $Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$

# Solution

**minimize**  $Cost (\nabla Cost = 0)$

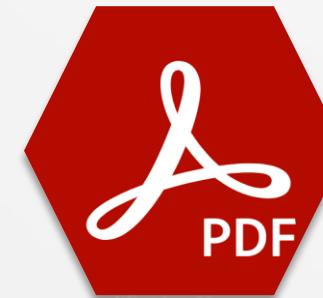
$$\text{ໄດຍກົດ} Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



# Solution



Derivation of Equivalent

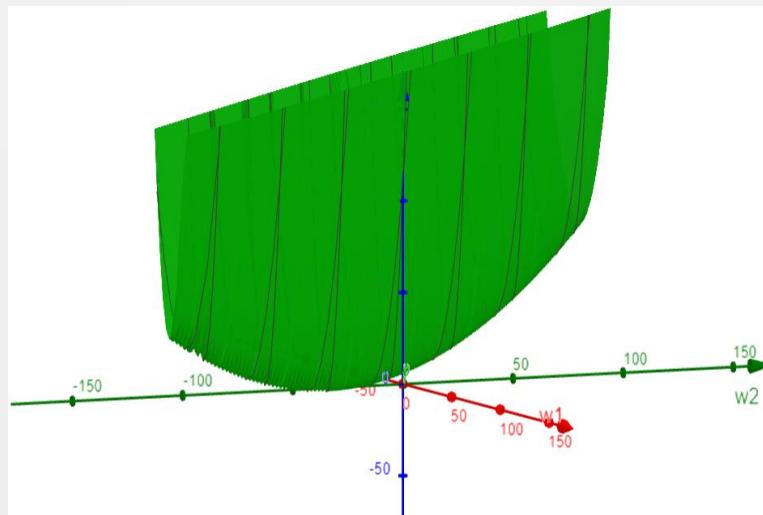


Open File  
**Derive\_Equivalent.pdf**

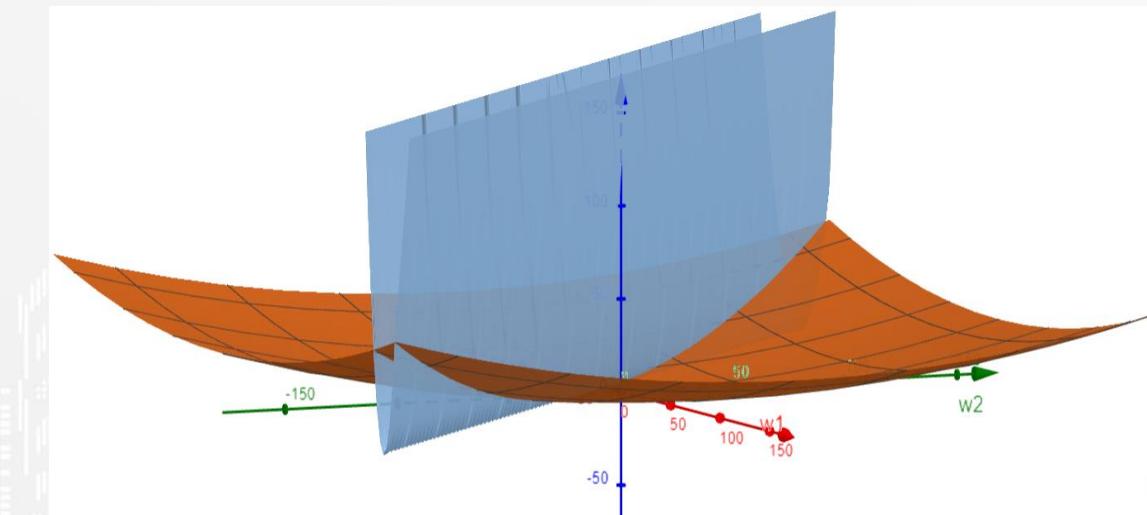
# Solution

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

$$\lambda = 0.001$$



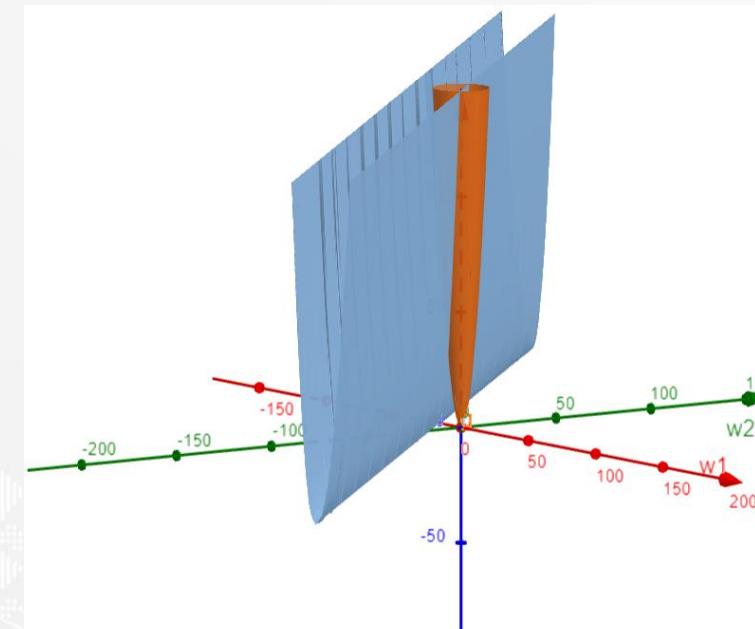
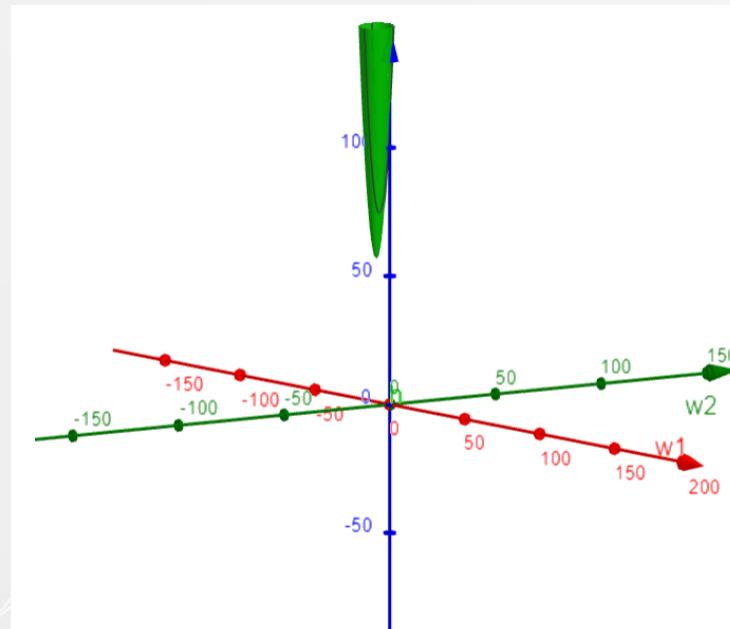
=



# Solution

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

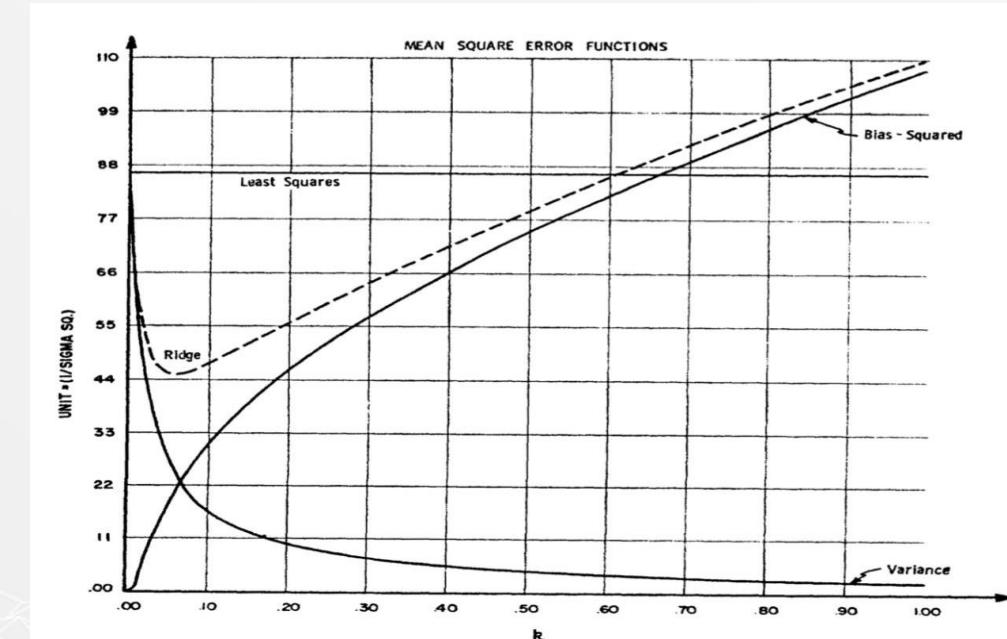
$$\lambda = 1$$



# Solution

ในปี 1970 นักคณิตศาสตร์ ได้ค้นพบสมบัติอันยิ่งใหญ่ของการ minimize

$$\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



Ref : Ridge Regression : Biased Estimation for Nonorthogonal Problems

# Solution

“ นักคณิตศาสตร์คันพบร่วมกันว่า การใส่  $\lambda > 0$  จะทำให้ error ของ model ลดลงกว่าการไม่ใส่  $\lambda$  ( $\lambda = 0$ ) สำหรับ data ทุกประเภท ”

\*ยกเว้น  $y = \hat{y}$  (model ไม่มี error)

# Solution

ตัวอย่าง

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>
0	1	0	3.8
0	0	1	5.1
1	0	0	3
0	1	0	4.2
0	0	1	4.9

ตารางแสดง dataset ที่ feature linearly dependent

# Solution

ตัวอย่าง

**minimize**  $Cost$  ( $\nabla Cost = 0$ )

โดยที่  $Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$

# Solution

## ตัวอย่าง

กำหนดให้  $\lambda = 10^{-16}$

ดังนั้น เราจะได้

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 0.5 \\ 1.5 \\ 2.5 \end{bmatrix}$$

# Solution

ตัวอย่าง

$$\hat{y} = 2.5 + 0.5x_1 + 1.5x_2 + 2.5x_3$$

- Training set

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>	<b><math>\hat{y}</math></b>
0	1	0	3.8	4
0	0	1	5.1	5
1	0	0	3	3
0	1	0	4.2	4
0	0	1	4.9	5

# Solution

ตัวอย่าง

$$\hat{y} = 2.5 + 0.5x_1 + 1.5x_2 + 2.5x_3$$

- Test set

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>x<sub>3</sub></b>	<b>y</b>	<b><math>\hat{y}</math></b>
0	0	0	1	2.5

# Solution

**“ Ridge Regression ”**

# Solution

**minimize**  $\sum_{j=0}^p w_j^2$   
**subject to** สมการของร่อง



**minimize**  $\sum_{j=0}^p w_j^2$   
**subject to**  $\sum_{i=1}^n (y_i - \hat{y}_i)^2 = c$

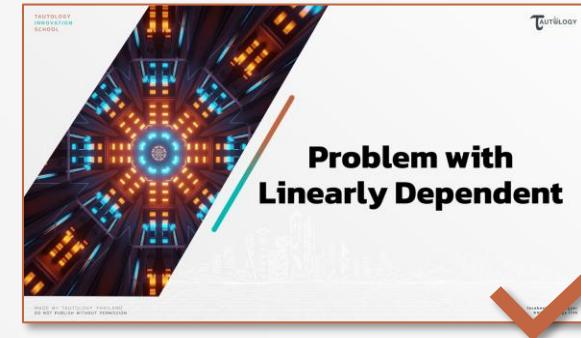


**Ridge Regression**



**minimize**  $Cost (\nabla Cost = 0)$   
โดยที่  $Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$

# Model Improvement



# Regularization

# Regularization

What is  
Regularization?

Ridge Regression

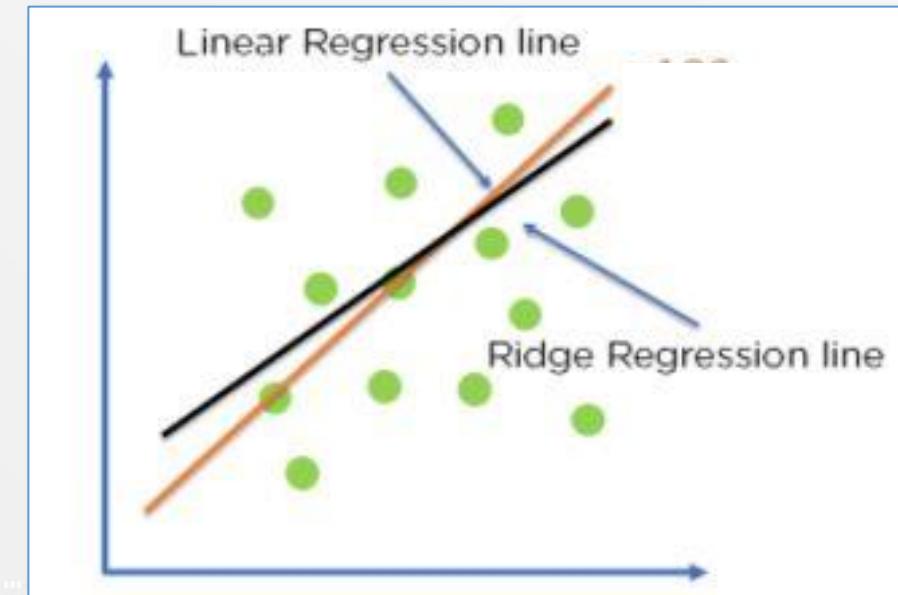
Lasso Regression

Elastic Net

Conclusion

# What is Regularization?

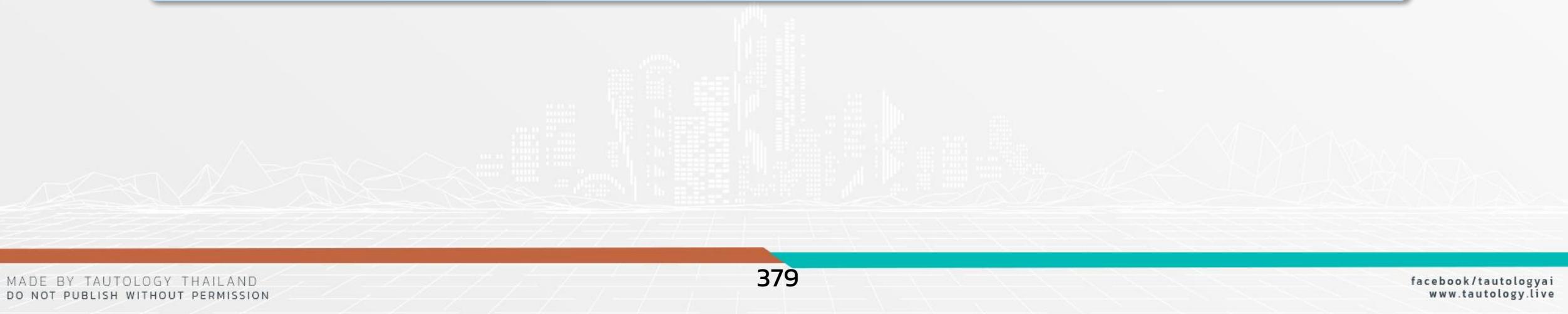
Regularization คือ วิธีการสร้าง model ให้มีความ general มากขึ้น (ลดปัญหา overfitting)



Ref: <https://www.simplilearn.com/tutorials/machine-learning-tutorial/regularization-in-machine-learning>

# What is Regularization?

“ ช่วยแก้ปัญหา performance แย่ เมื่อเจอกับ unseen data ทำให้ model มี performance ดีขึ้นสำหรับ data ทุกประเภท (linearly dependent, multicollinearity, no multicollinearity) ”



# What is Regularization?

สรุปแบบสรุปสำหรับผู้ที่ข้ามเนื้อหา Problem & Solution

Linearly dependent  $\rightarrow$  มีโอกาสสูงที่ performance ของ model จะไม่ดี



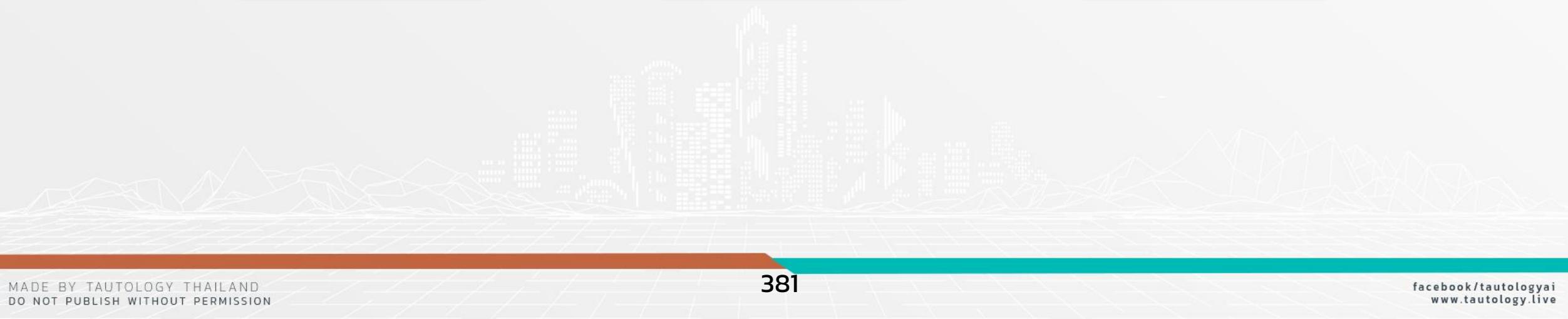
Ridge Regression  
(Regularization)

# What is Regularization?

**Ridge  
Regression**

**Lasso  
Regression**

**Elastic Net**



# Regularization

**What is  
Regularization?**



Ridge Regression

Lasso Regression

Elastic Net

Conclusion

# Ridge Regression

- What is Ridge Regression?
- Geometric View
- Properties
- Model Creation
- How to find Lambda
- Code

# What is Ridge Regression?

Ridge regression คือ การทำ regularization โดยมี cost function เป็น

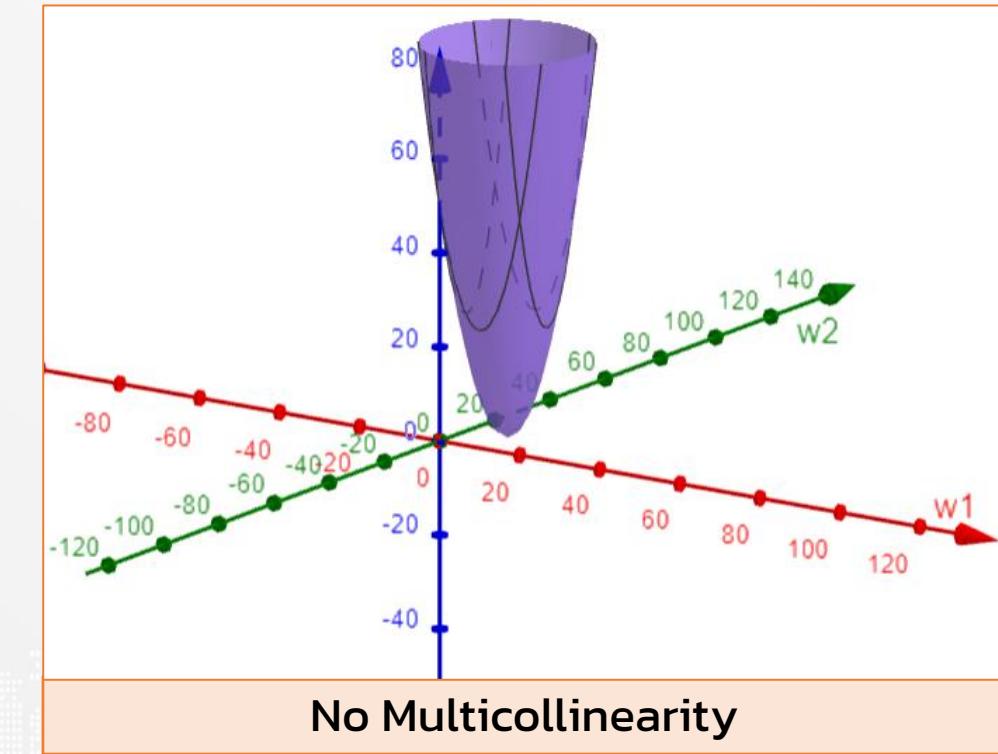
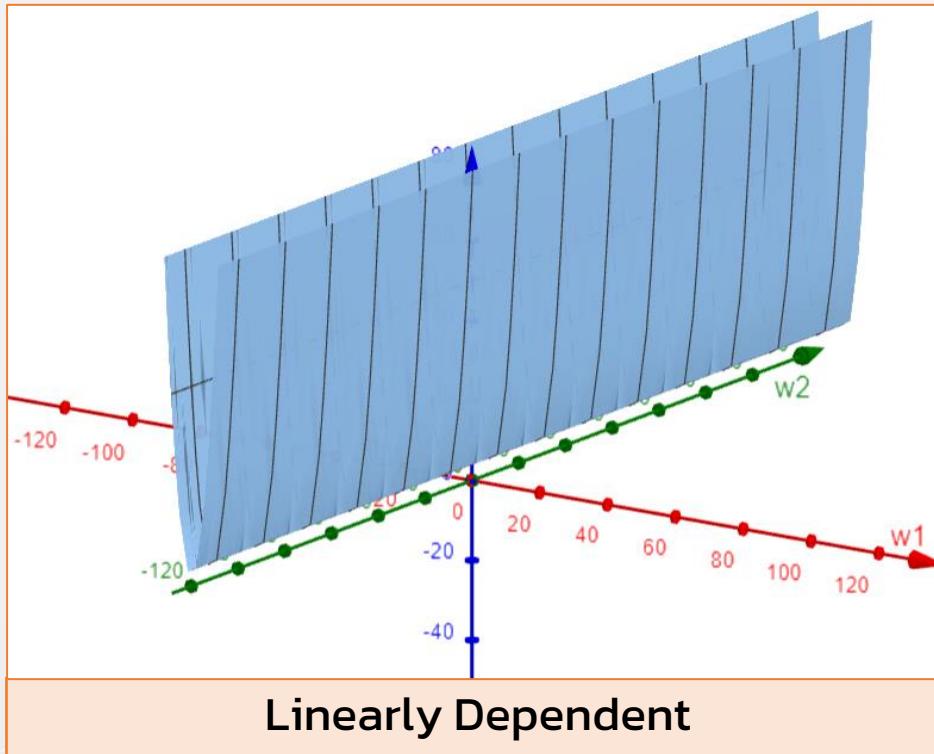
$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

# Ridge Regression

## **What is Ridge Regression?**

- Geometric View
- Properties
- Model Creation
- How to find Lambda
- Code

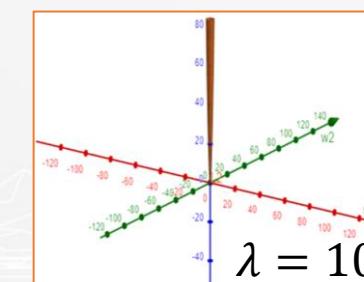
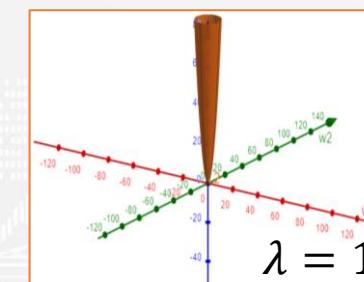
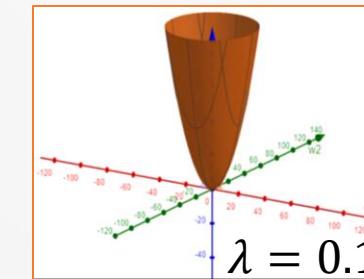
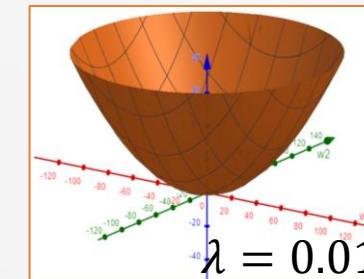
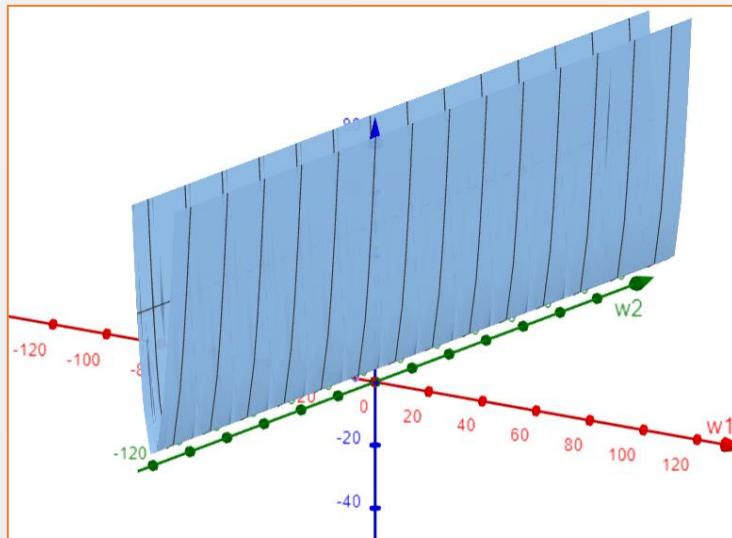
# Geometric View



# Geometric View

## Linearly Dependent

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



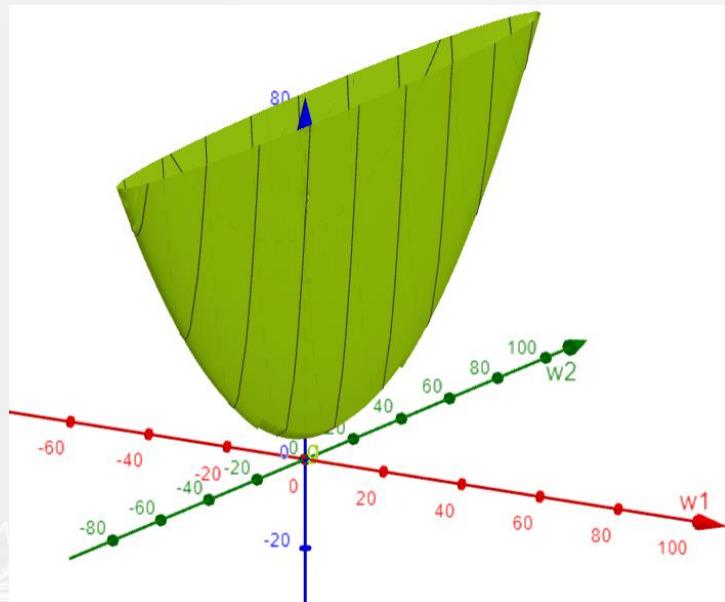
<https://www.geogebra.org/3d/mzbky4wt>

# Geometric View

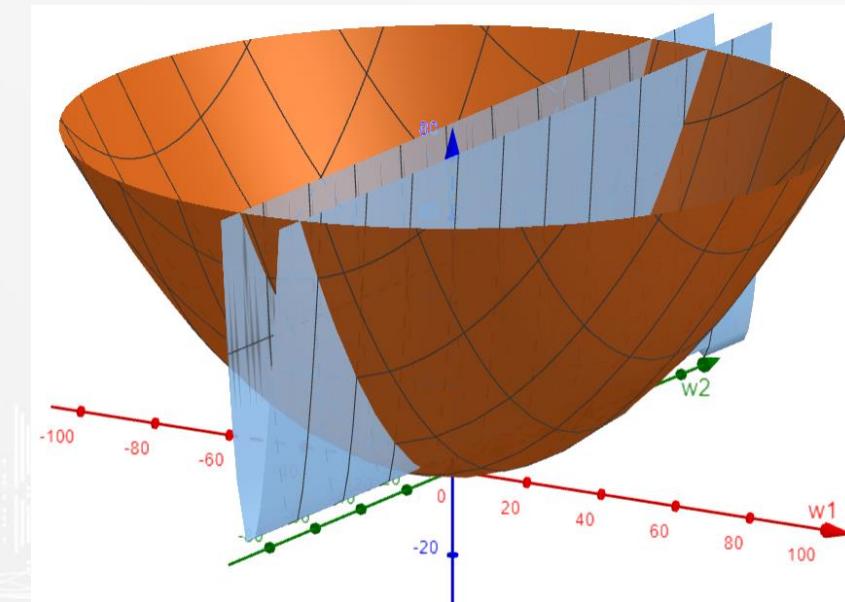
Linearly Dependent

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

$$\lambda = 0.01$$



=

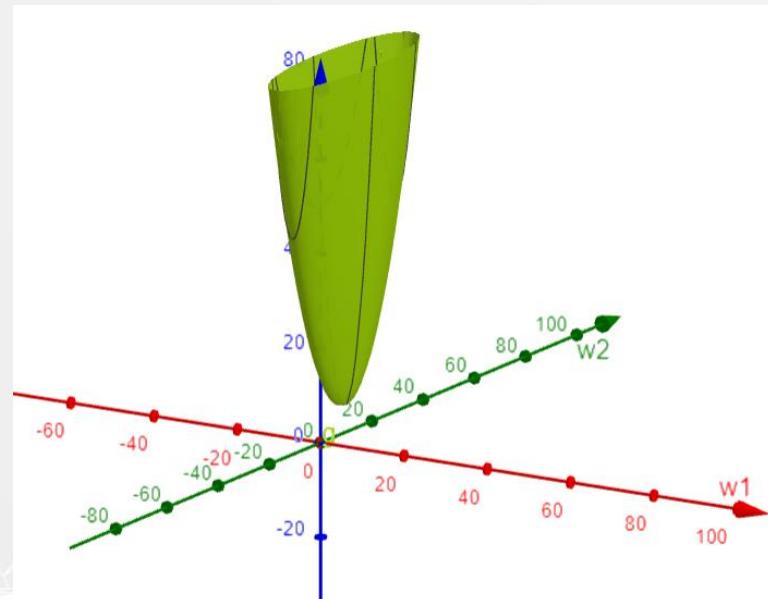


# Geometric View

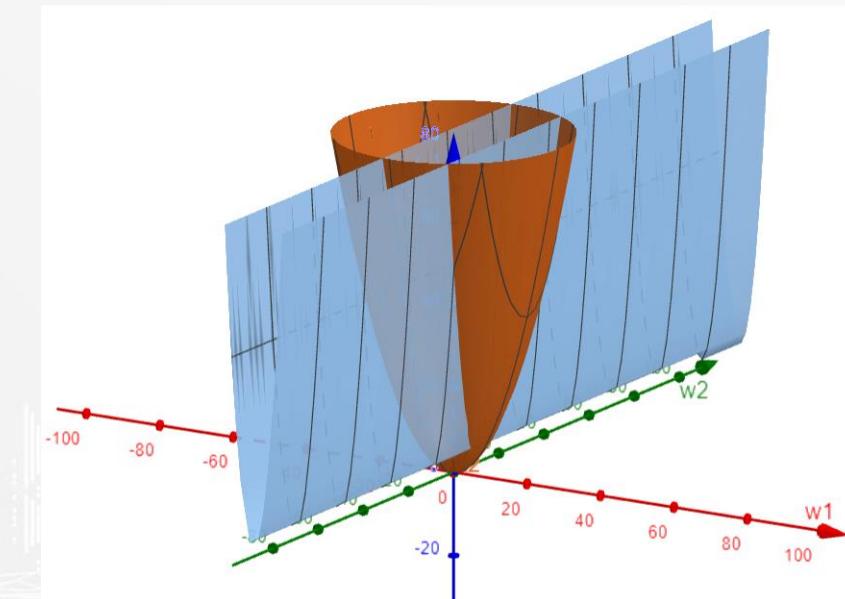
Linearly Dependent

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

$$\lambda = 0.1$$



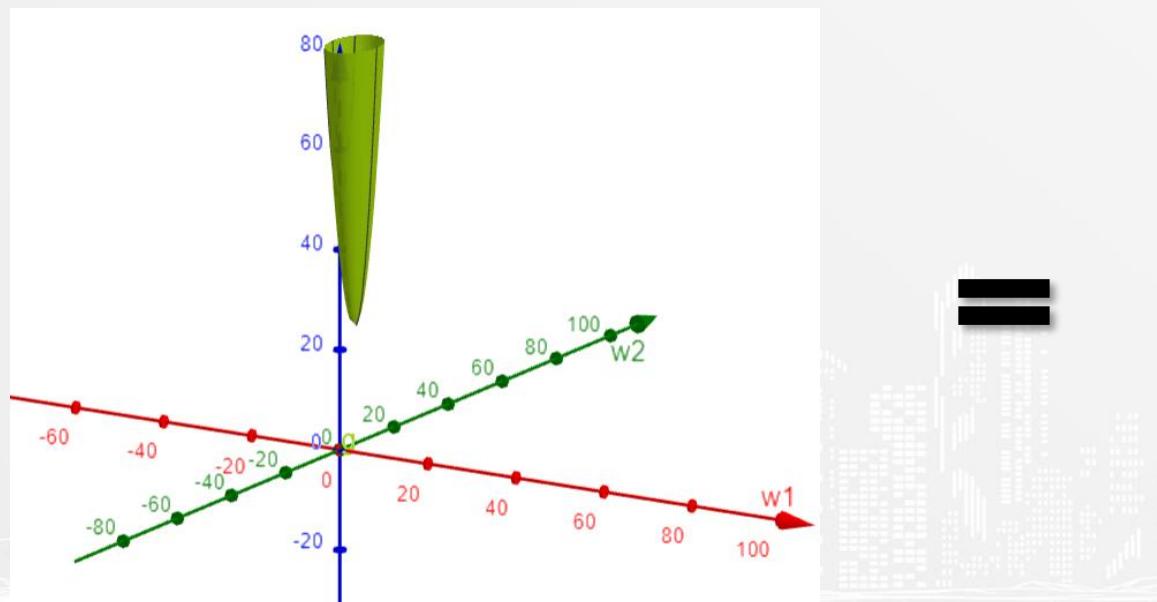
=



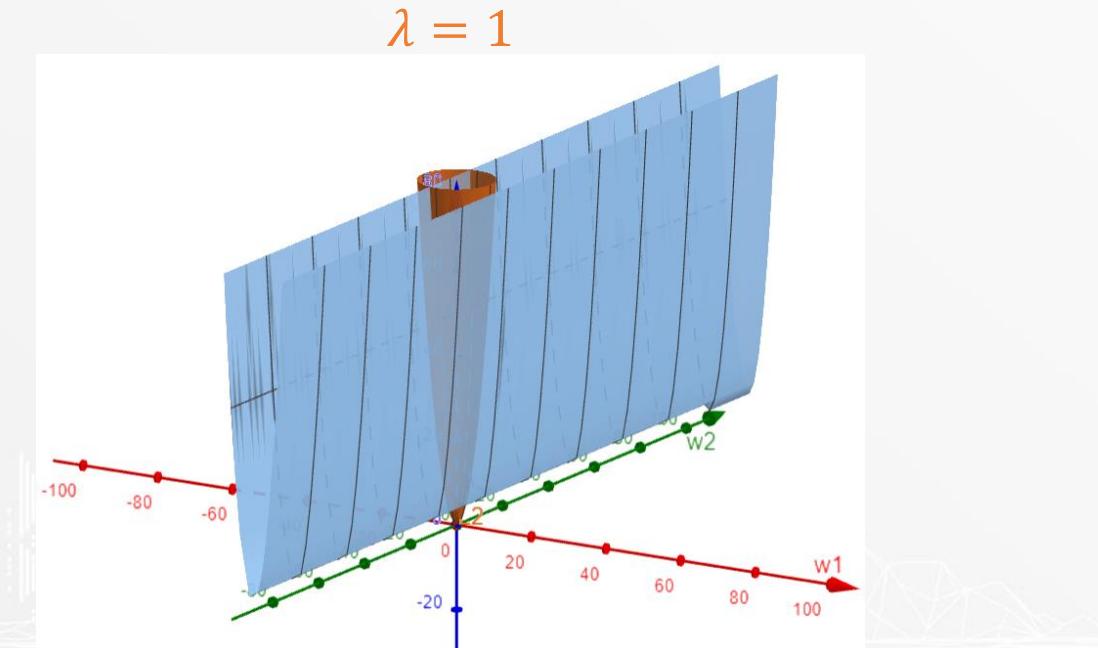
# Geometric View

## Linearly Dependent

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



=

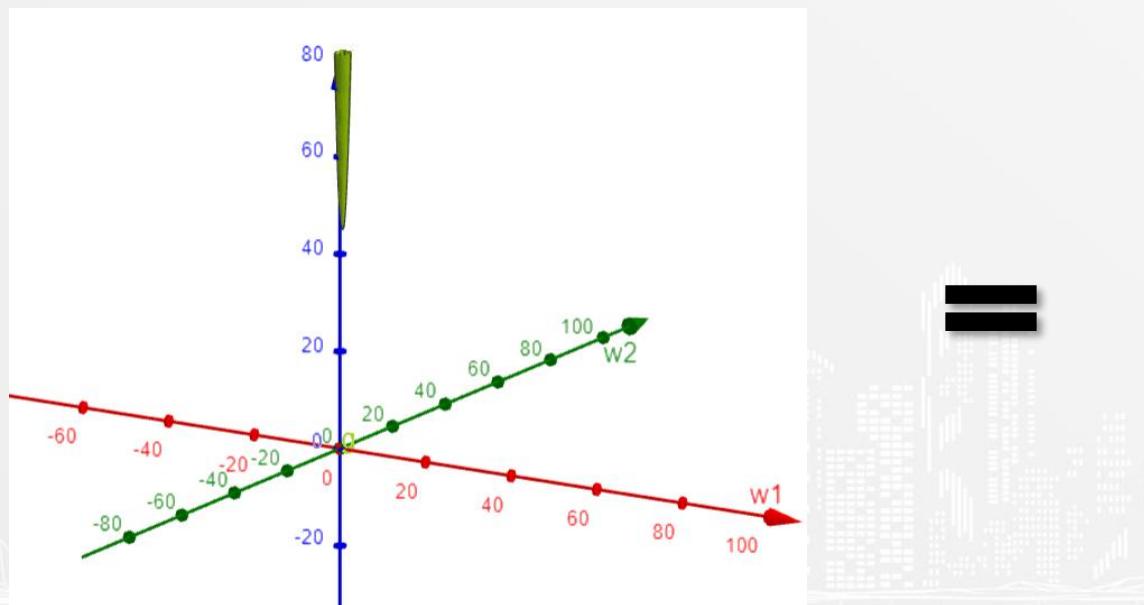


$$\lambda = 1$$

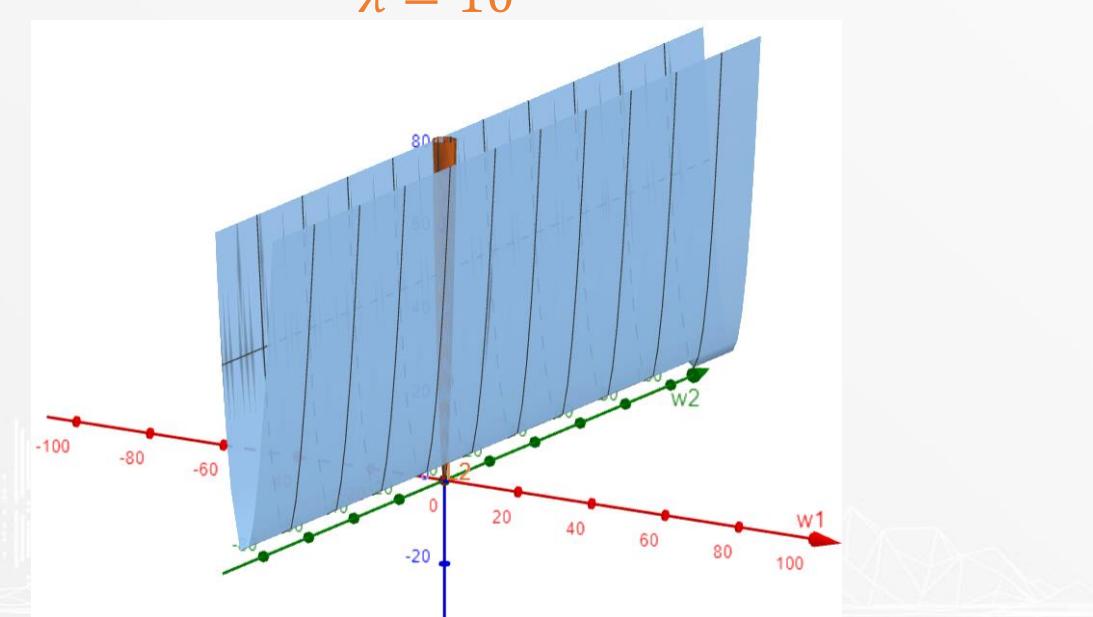
# Geometric View

## Linearly Dependent

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



391



# Geometric View

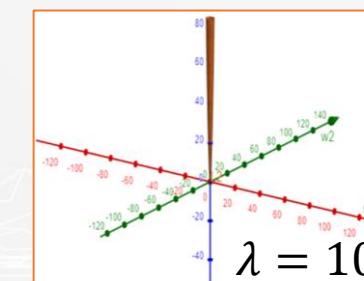
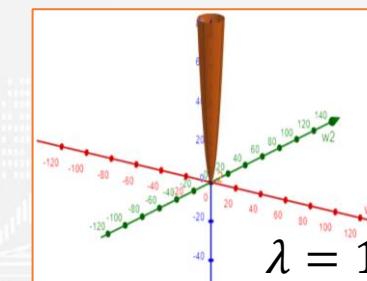
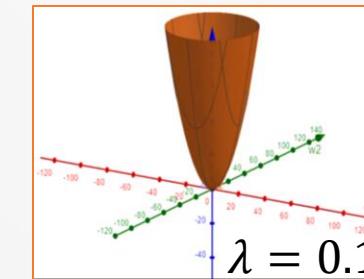
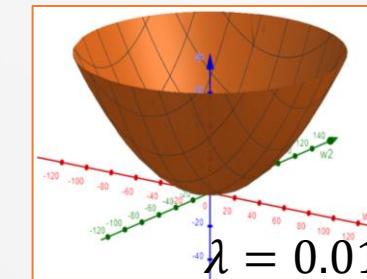
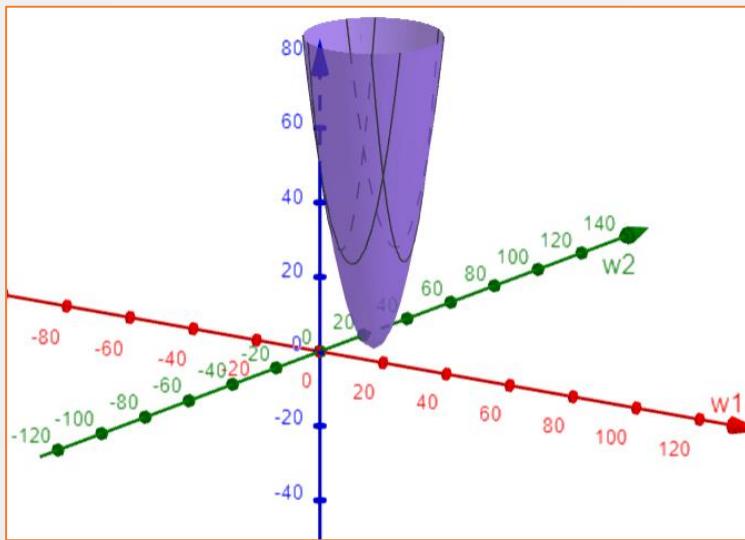
## Linearly Dependent

- จุดตាំส្តុដីមែន មានឈូរភាគរវាងសមការរំងកំពេប និង ជុំកំណើន
- កំពេបនេះបានផ្តល់ឱ្យសមការរំងកំពេប បើនជុំតាំស្តុដីមែន
- យើង  $\lambda$  មាត្រា ជុំតាំស្តុដីមែនយើង ក្រុម្ភាស្តុកំណើន
- យើង  $\lambda$  បានយើង ជុំតាំស្តុដីមែនយើង ក្រុម្ភាស្តុសមការរំងកំពេប

# Geometric View

## No Multicollinearity

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



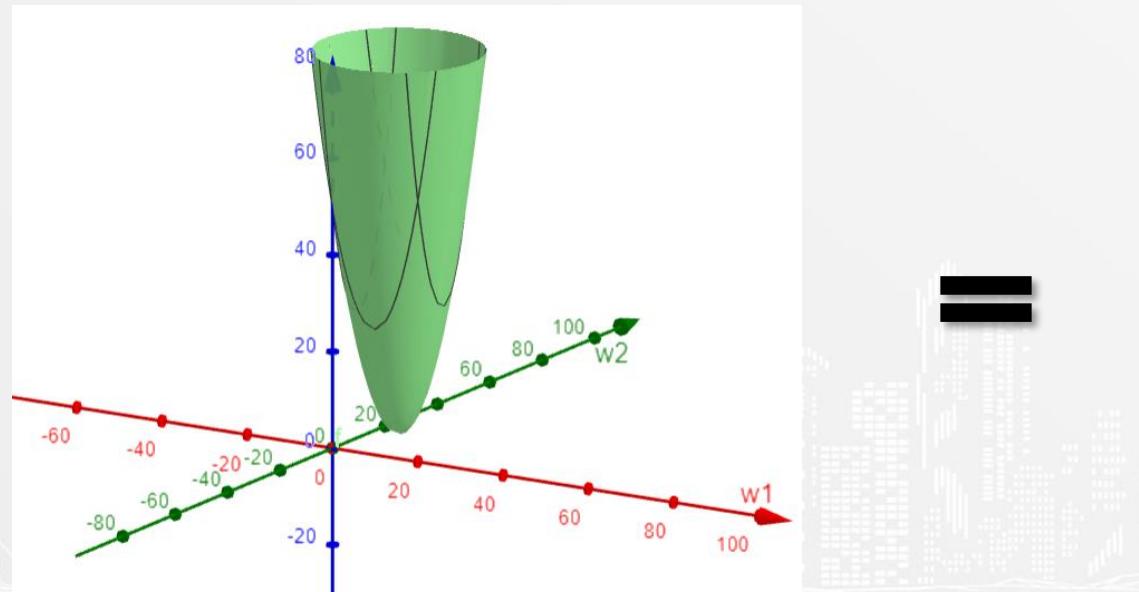
<https://www.geogebra.org/3d/mzbky4wt>

# Geometric View

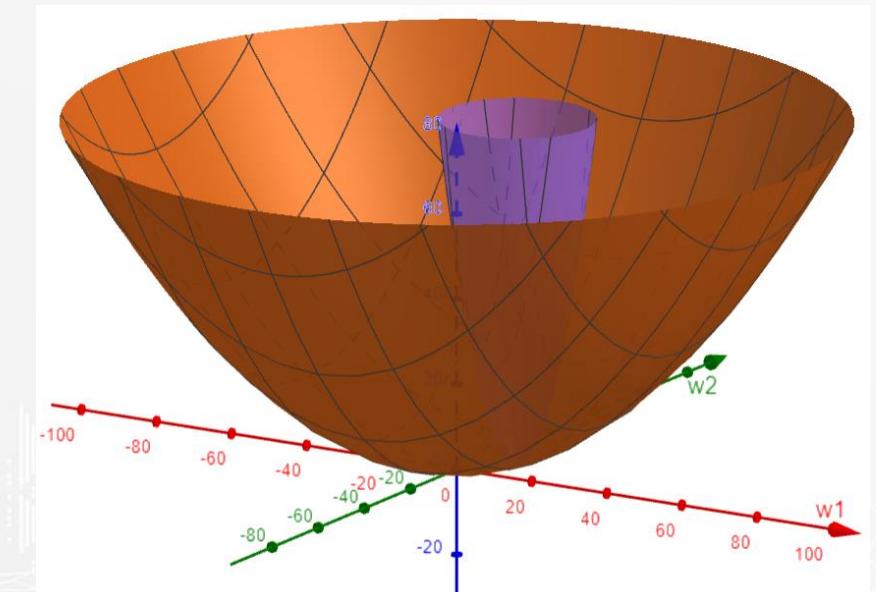
No Multicollinearity

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$

$$\lambda = 0.01$$



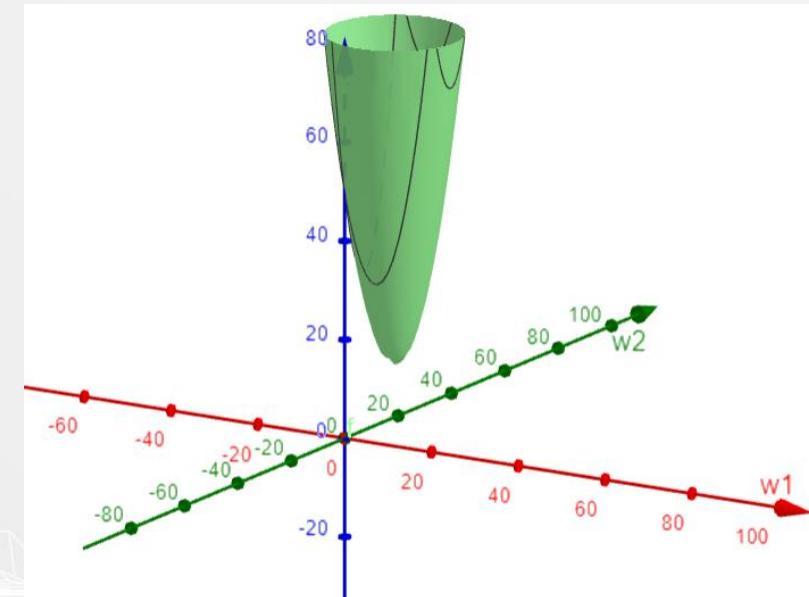
=



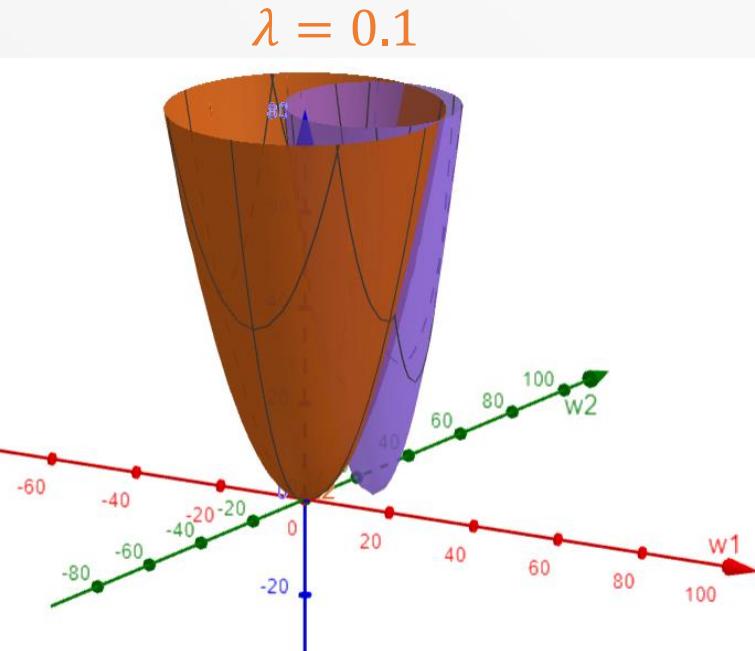
# Geometric View

No Multicollinearity

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



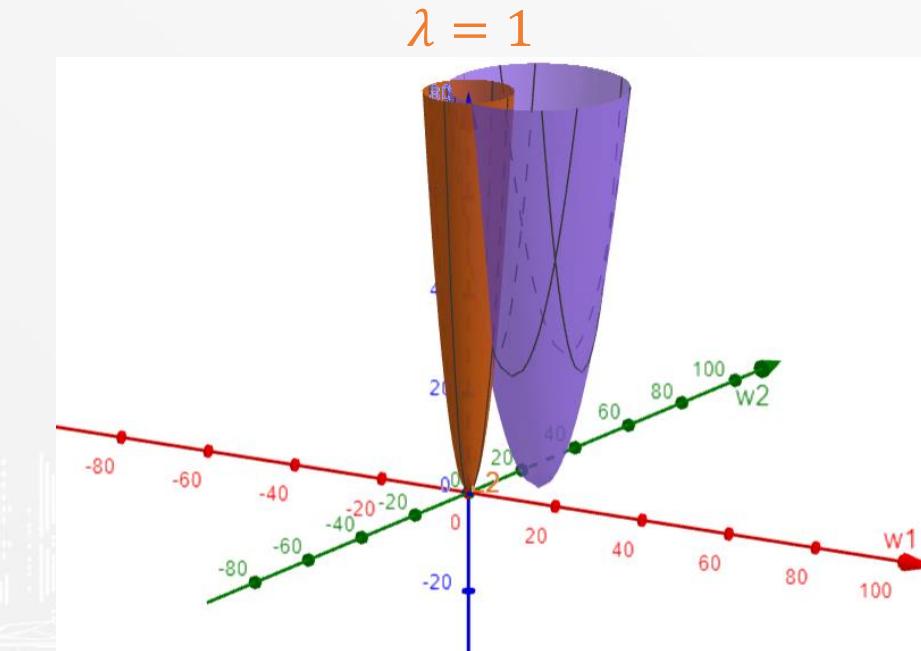
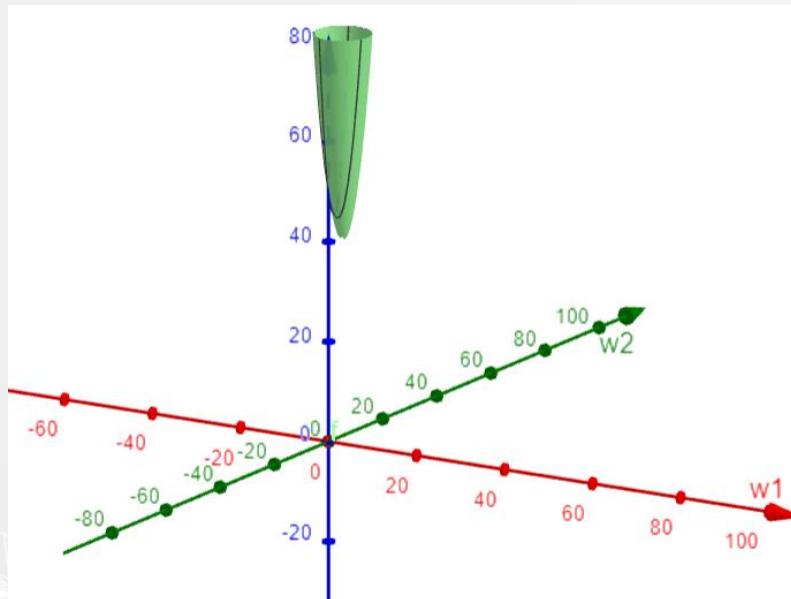
=



# Geometric View

No Multicollinearity

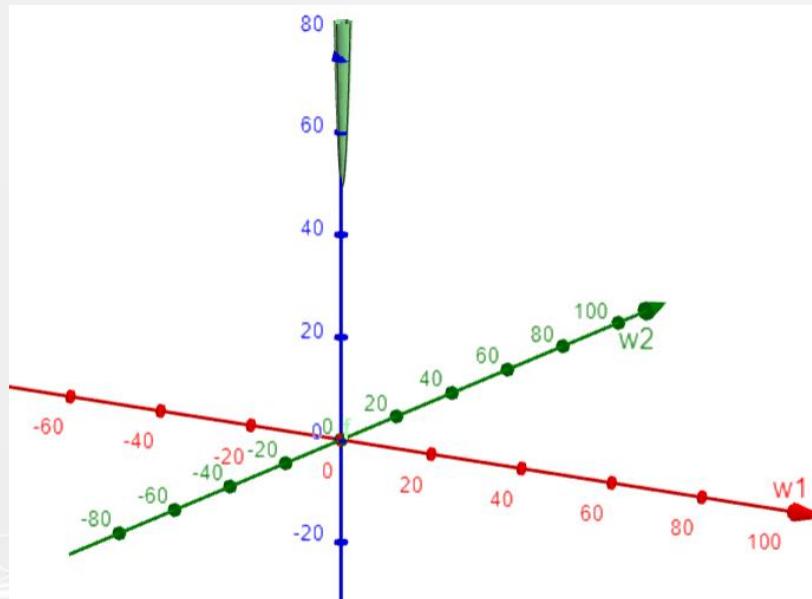
$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



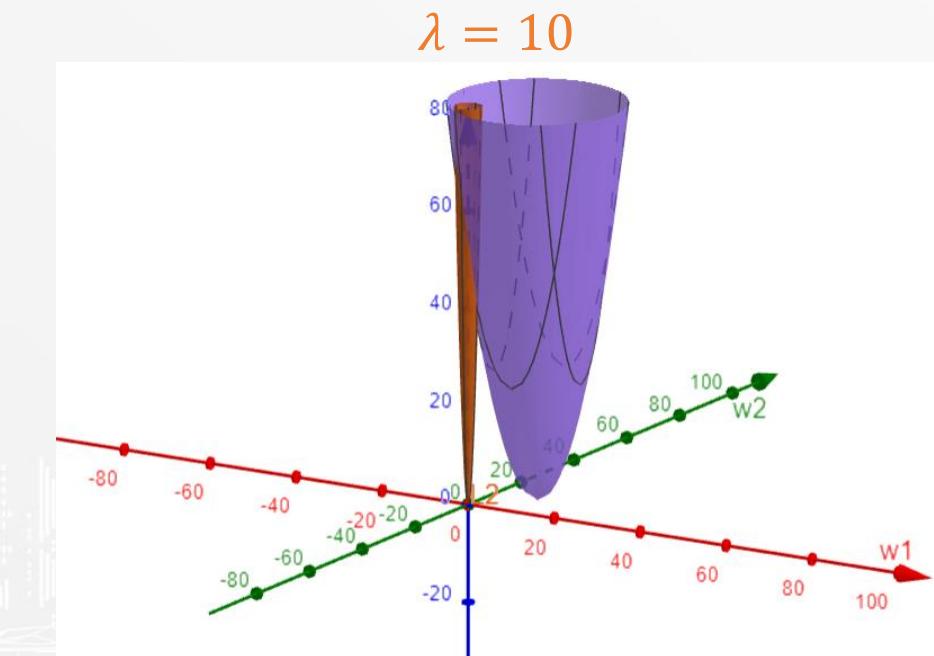
# Geometric View

No Multicollinearity

$$Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



=



# Geometric View

## No Multicollinearity

- จุดตាំส្តុដីមែន មានឈូរភាពរវាងជុំតាំស្តុដីមែន និងជុំតាំស្តុដីកំណើន
- យើង លេខ មេរក ជុំតាំស្តុដីមែនយើង ក្រោមឈូរភាពរវាងជុំតាំស្តុដីកំណើន
- យើង លេខ បានឈូរ ជុំតាំស្តុដីមែនយើង ក្រោមឈូរភាពរវាងជុំតាំស្តុដីកំណើន

# Ridge Regression

**What is Ridge Regression?**

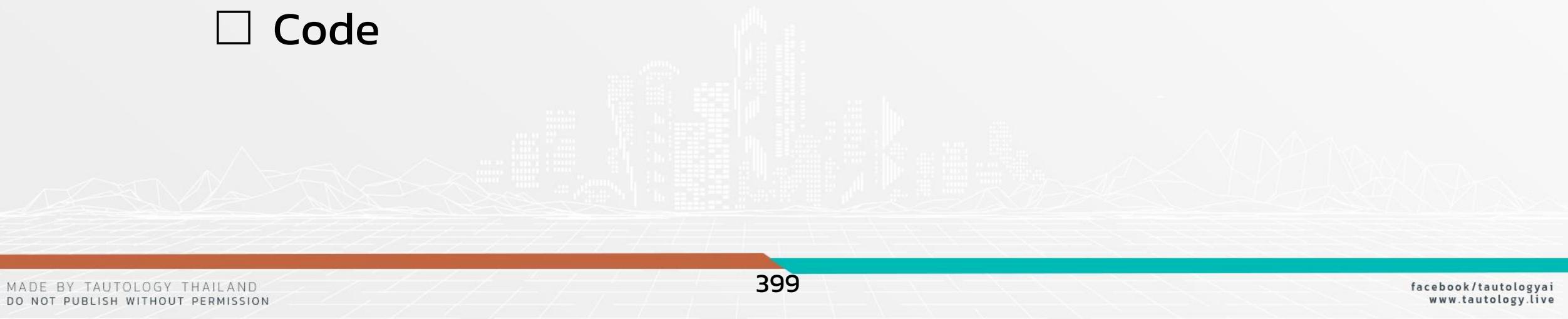
**Geometric View**

**Properties**

**Model Creation**

**How to find Lambda**

**Code**



# Properties

“ การใส่  $\lambda > 0$  จะทำให้ error ของ model ลดลงกว่าการ  
ไม่ใส่  $\lambda$  ( $\lambda = 0$ ) สำหรับ data ทุกประเภท ”

\*ยกเว้น  $y = \hat{y}$  (model ไม่มี error)

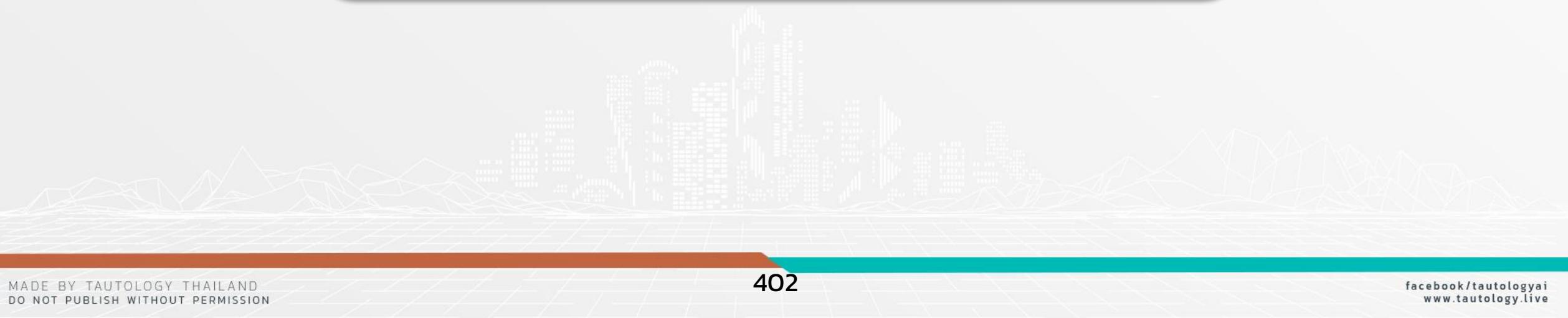
# Ridge Regression

- What is Ridge Regression?**
- Geometric View**
- Properties**
- Model Creation
- How to find Lambda
- Code

# Model Creation

$$\nabla Cost = 0$$

$$\text{โดยที่ } Cost = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$$



# Model Creation

$$\mathbf{w} = (X_b^T X_b + \lambda \mathbf{I})^{-1} X_b^T \mathbf{y}$$

# Model Creation



Derivation of Ridge Regression



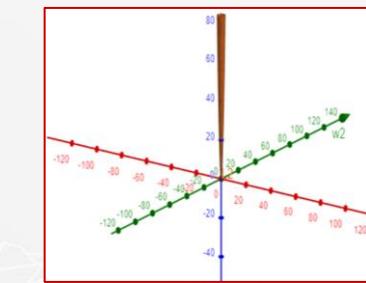
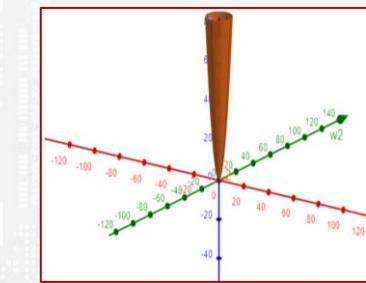
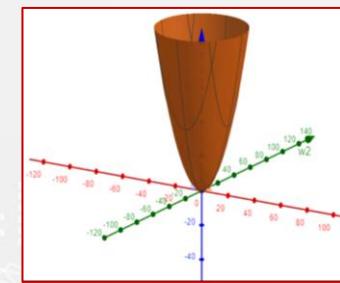
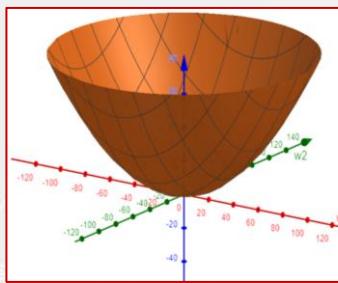
Open File  
**Derive\_Ridge.pdf**

# Ridge Regression

- What is Ridge Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda
- Code

# How to find Lambda

Q: แล้วเราจะ  $\lambda$  ที่ดีที่สุดได้อย่างไร?



# How to find Lambda

**A: ໃຊ້ Cross Validation**

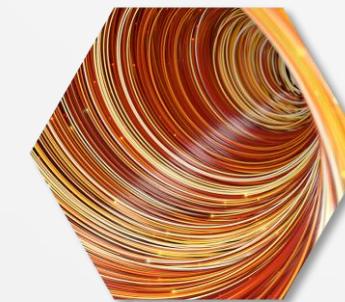
# How to find Lambda

$\lambda$	Fold					Mean	Rank
	1	2	3	4	5		
$\lambda = 0.01$	0.78	0.77	0.72	0.80	0.78	0.77	2
$\lambda = 0.1$	0.80	0.88	0.78	0.72	0.80	0.78	1
$\lambda = 1$	0.74	0.72	0.76	0.75	0.78	0.75	4
$\lambda = 10$	0.82	0.74	0.74	0.74	0.70	0.76	3

# How to find Lambda



For more information



## Cross Validation

# Ridge Regression

- What is Ridge Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda**
- Code**

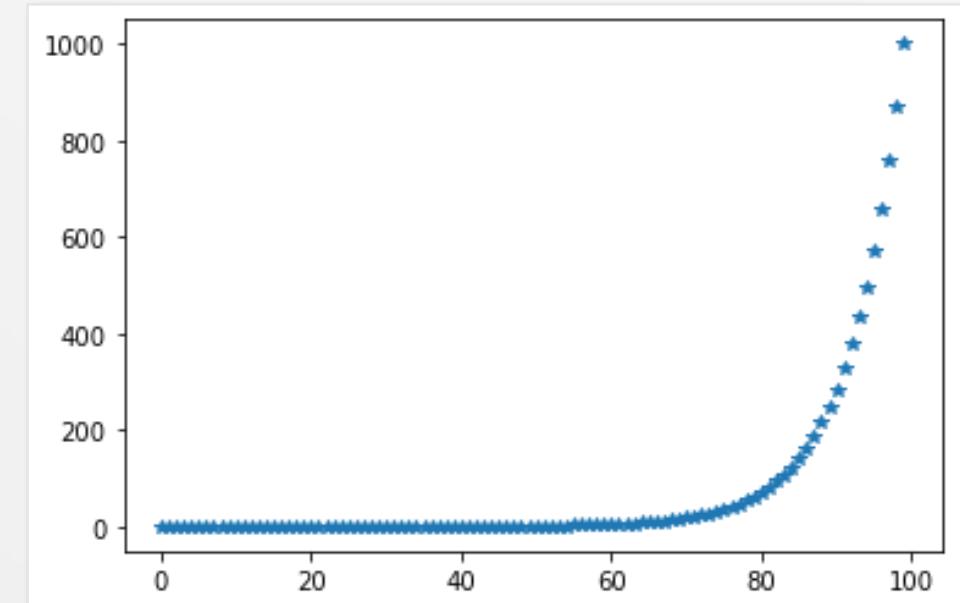
# Code

## ตัวอย่าง code สำหรับ ridge regression

$x_1$	$x_2$	y
0	1	3.9
2	1	7.7
1	1	6.2
2	0	5.2
3	1	9.8

# Code

- เก็บค่า  $\lambda$  ที่ต้องการพิจารณาไว้ในตัวแปรชื่อ alphas



```
1 alphas = np.logspace(-3, 3, num=100)
```

# Code

- Code สำหรับสร้าง model จากข้อมูลของเราโดยที่

$$X = \begin{bmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 1 \\ 2 & 0 \\ 3 & 1 \end{bmatrix}, \quad y = \begin{bmatrix} 3.9 \\ 7.7 \\ 6.2 \\ 5.2 \\ 4.8 \end{bmatrix}$$

```
1 reg = RidgeCV(alphas=alphas, cv=5)
2 reg.fit(X, y)
```

# Code

- ค่า  $\lambda$  ที่ดีที่สุด ถูกเก็บไว้ใน attribute ชื่อ `alpha_`

```
1 reg.alpha_
```

```
0.001
```

# Code

- ค่า  $w_0$  ถูกเก็บไว้ใน attribute ชื่อ intercept\_

```
1 reg.intercept_
```

```
1.3639553659622141
```

# Code

- ค่า  $w_1, \dots, w_p$  จะเก็บไว้ใน attribute ชื่อ `coef_`

```
1 reg.coef_
```

```
array([1.91935049, 2.6563548 ])
```

# Code



Code for ridge regression



Open File  
**Regularization.ipynb**

# Ridge Regression

- What is Ridge Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda**
- Code**

# Regularization

**What is  
Regularization?**



**Ridge Regression**



**Lasso Regression**



**Elastic Net**



**Conclusion**



# Lasso Regression

- What is Lasso Regression?
- Geometric View
- Properties
- Model Creation
- How to find Lambda
- Code

# What is Lasso Regression?

Lasso regression คือ การทำ regularization โดยมี cost function เป็น

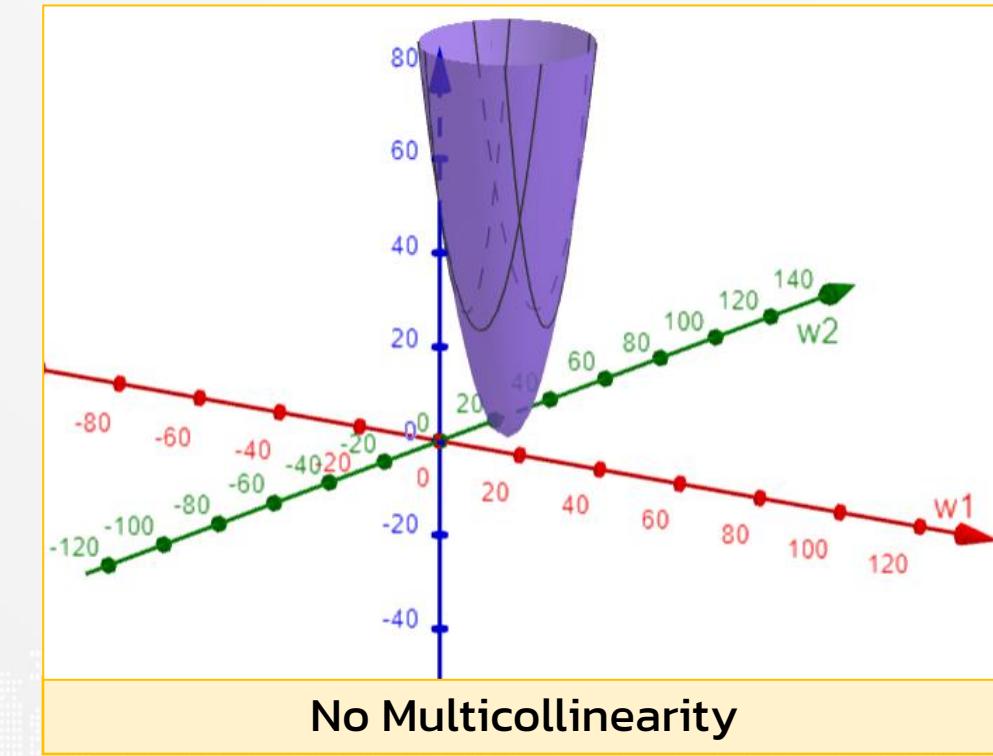
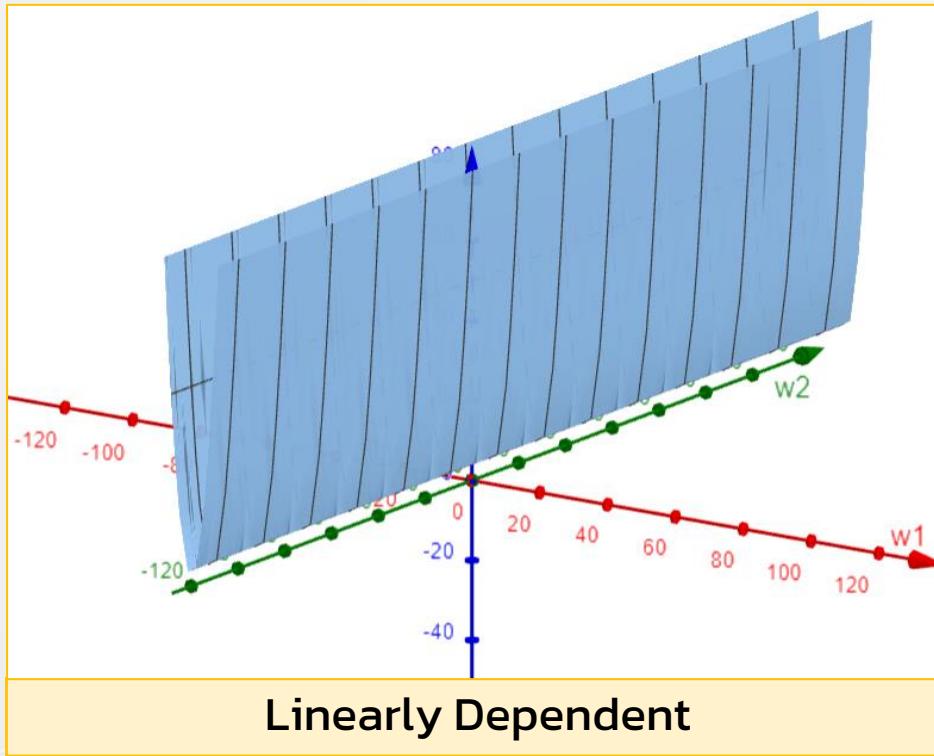
$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$

# Lasso Regression

## **What is Lasso Regression?**

- Geometric View
- Properties
- Model Creation
- How to find Lambda
- Code

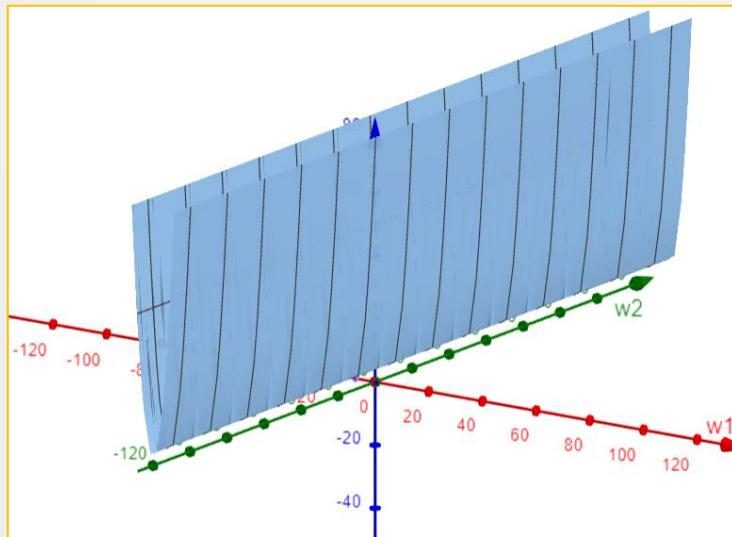
# Geometric View



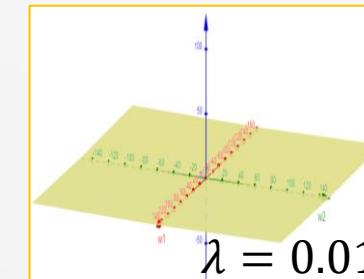
# Geometric View

## Linearly Dependent

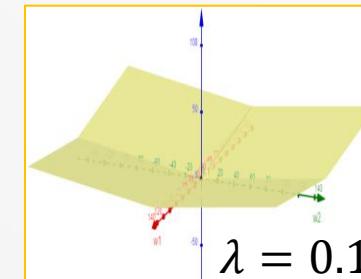
$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



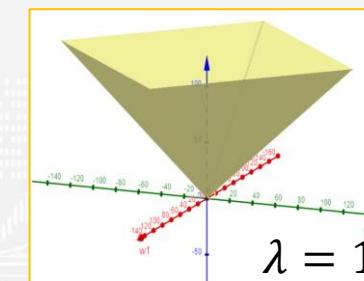
<https://www.geogebra.org/m/c87cckmk>



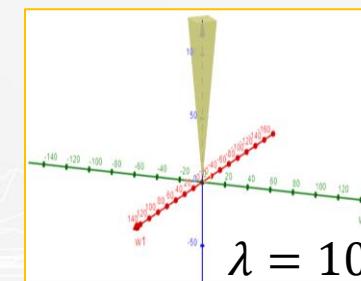
$\lambda = 0.01$



$\lambda = 0.1$



$\lambda = 1$



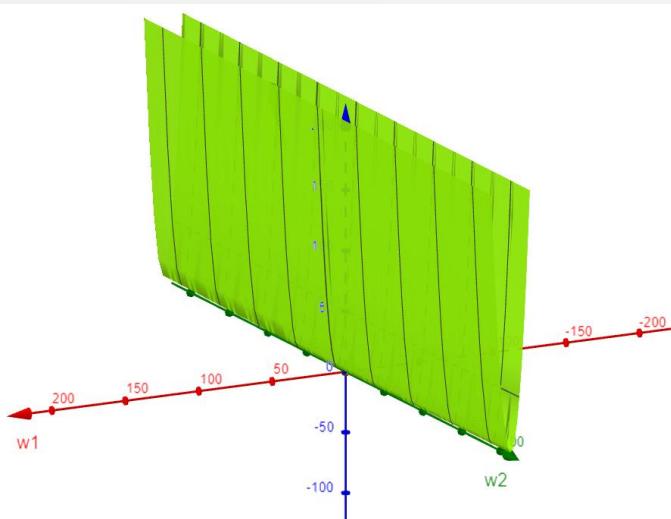
$\lambda = 10$

# Geometric View

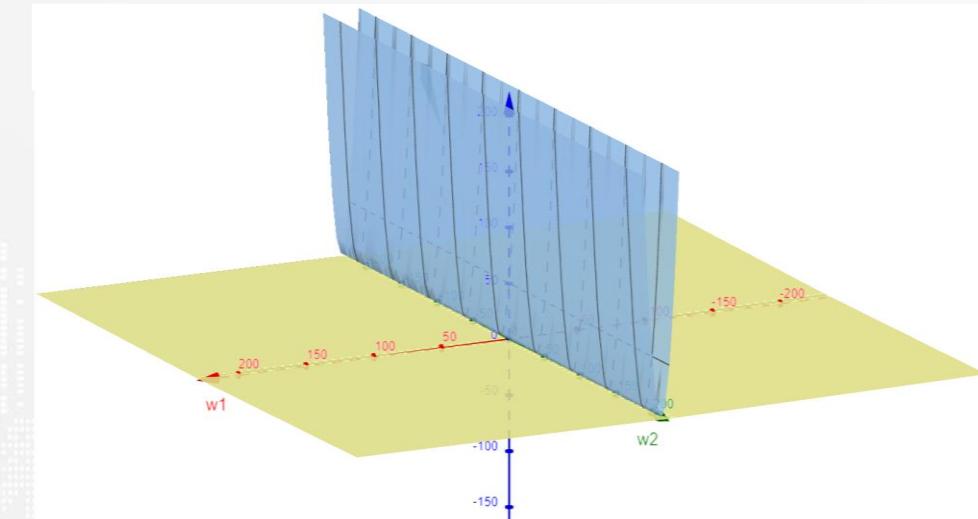
Linearly Dependent

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$

$$\lambda = 0.01$$



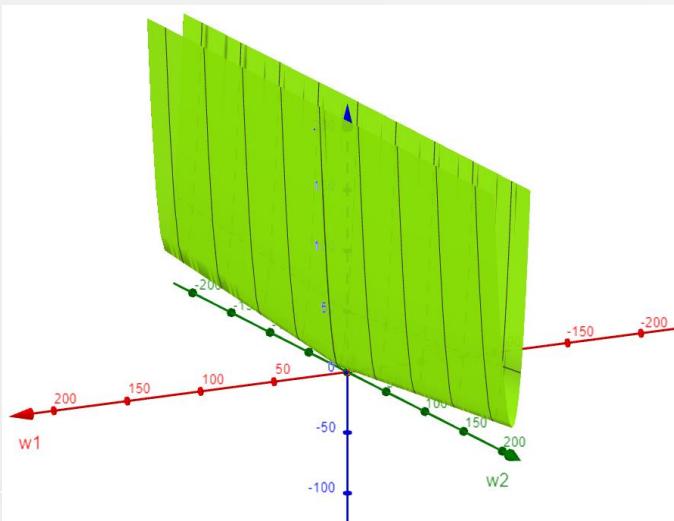
=



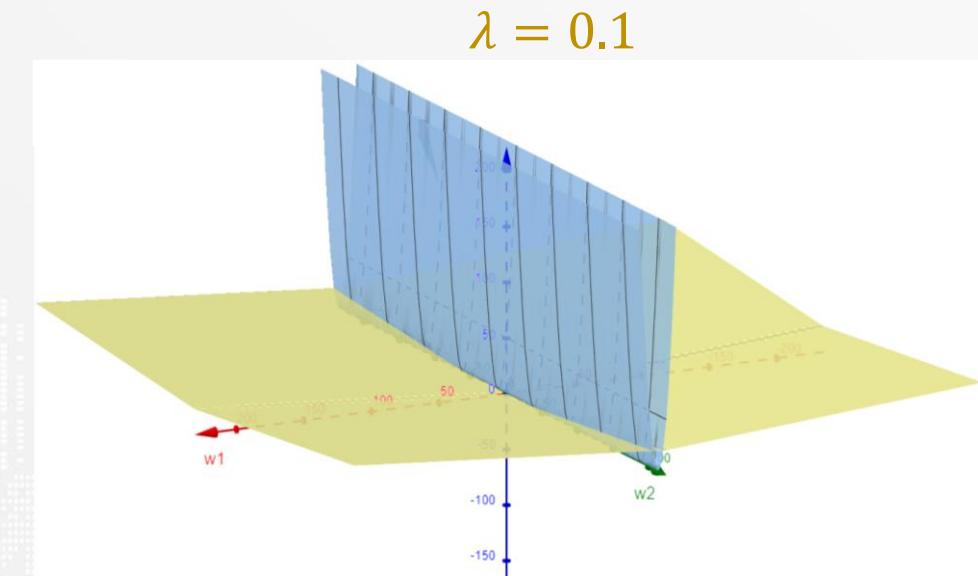
# Geometric View

Linearly Dependent

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



=

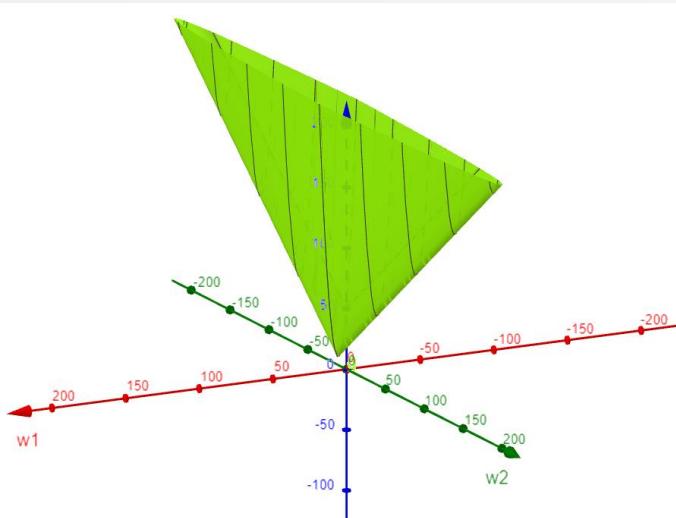


$\lambda = 0.1$

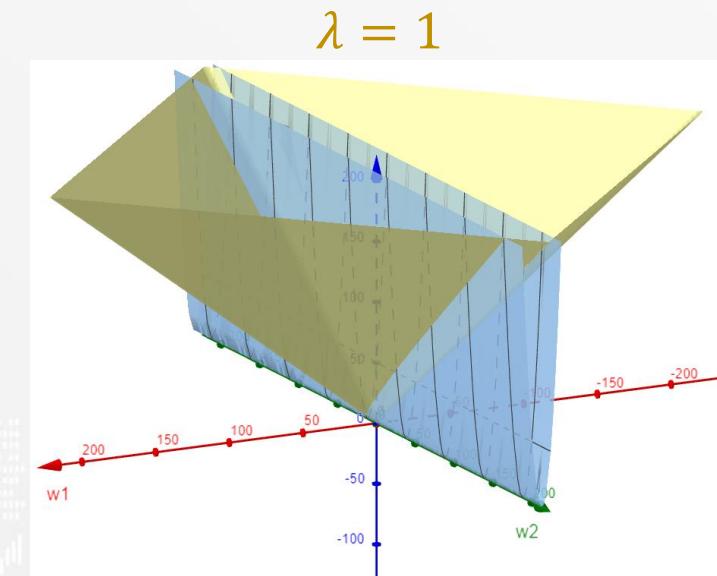
# Geometric View

Linearly Dependent

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



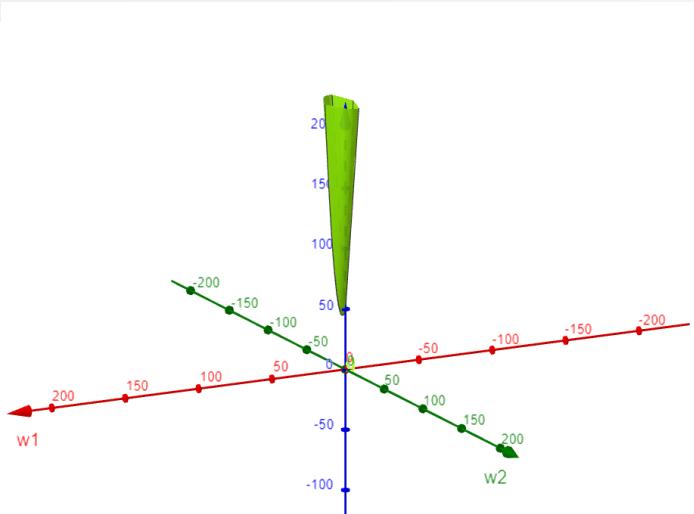
=



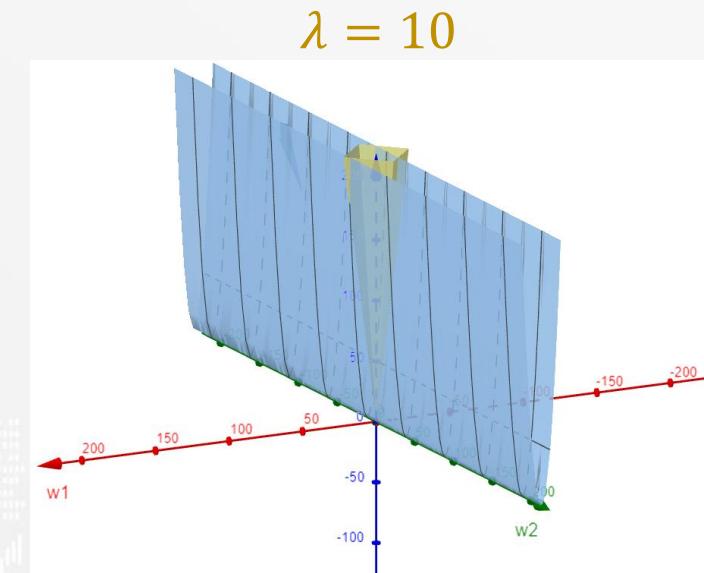
# Geometric View

## Linearly Dependent

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



=



# Geometric View

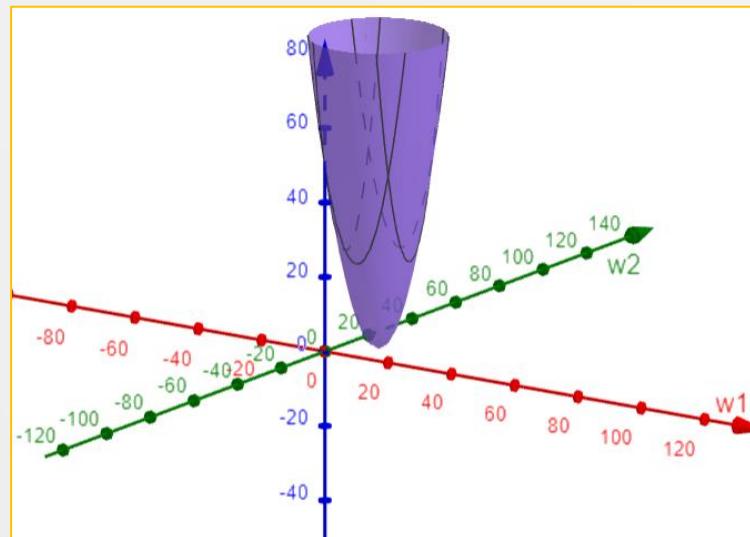
## Linearly Dependent

- จุดตាំส្តុដីមែន មានឈូរភាគរវាងសមការរំងកំពេប និង ជុំកំណើន
- កំពេបនេះបានផ្តល់ឱ្យសមការរំងកំពេប បើនជុំតាំស្តុដីមែន
- យើង  $\lambda$  មាត្រា ជុំតាំស្តុដីមែនយើង ក្រុម្ភាស្តុកំណើន
- យើង  $\lambda$  បានយើង ជុំតាំស្តុដីមែនយើង ក្រុម្ភាស្តុសមការរំងកំពេប

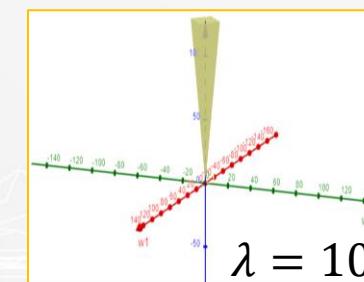
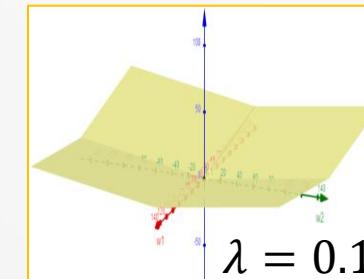
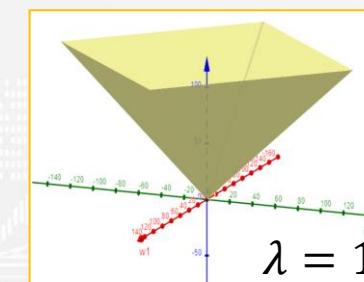
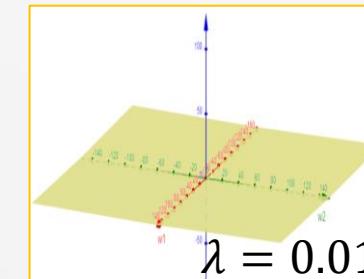
# Geometric View

## No Multicollinearity

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



<https://www.geogebra.org/m/c87cckmk>

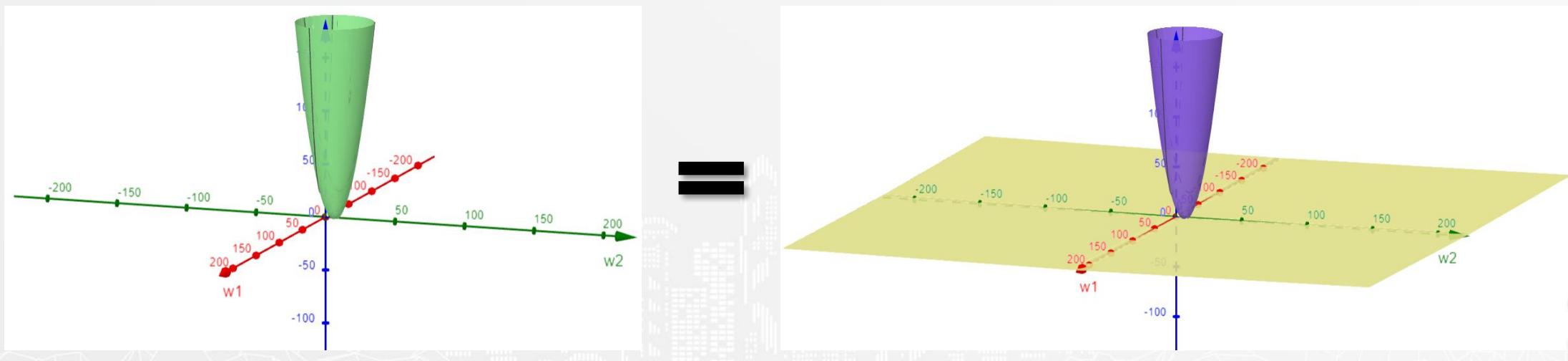


# Geometric View

No Multicollinearity

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$

$$\lambda = 0.01$$

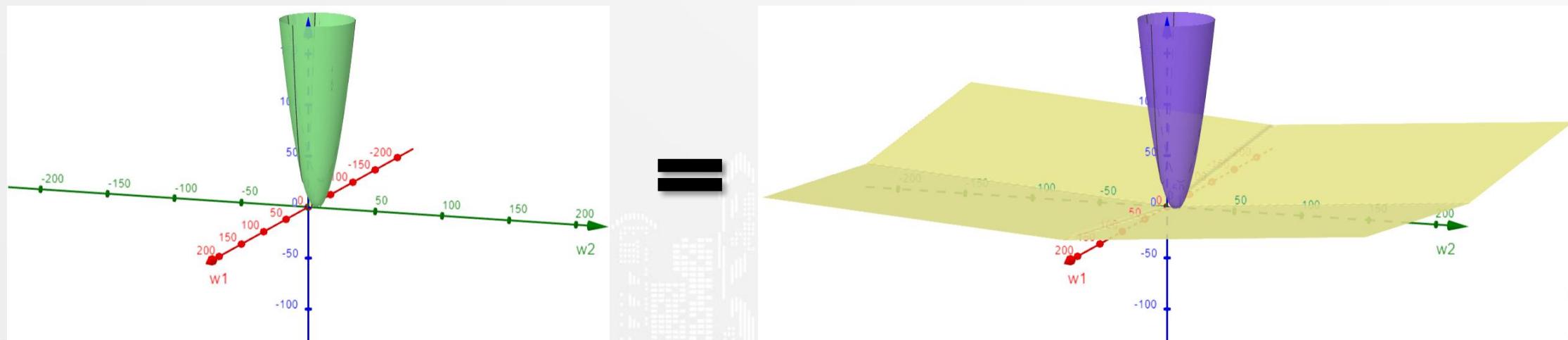


# Geometric View

No Multicollinearity

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$

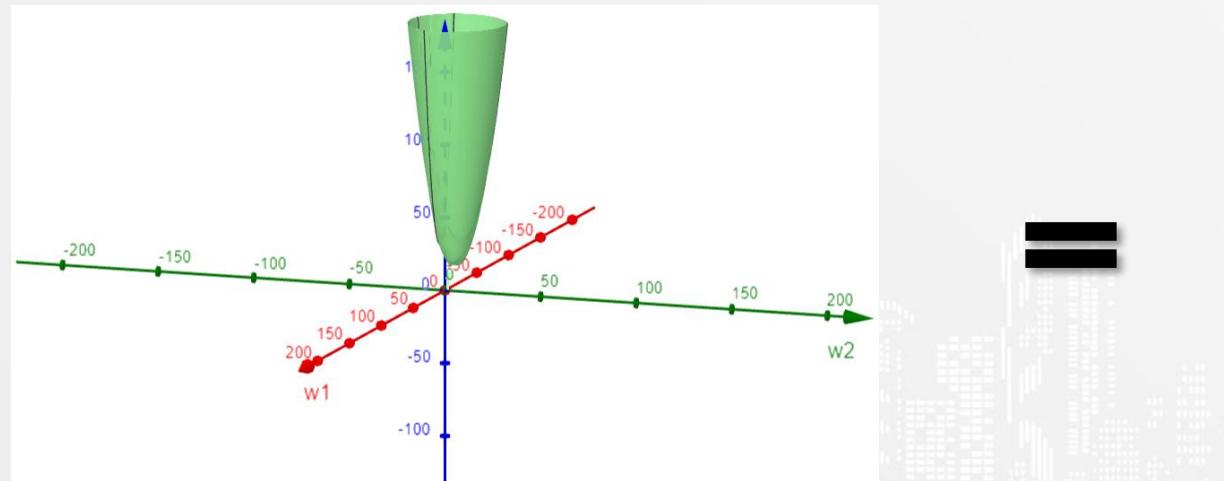
$$\lambda = 0.1$$



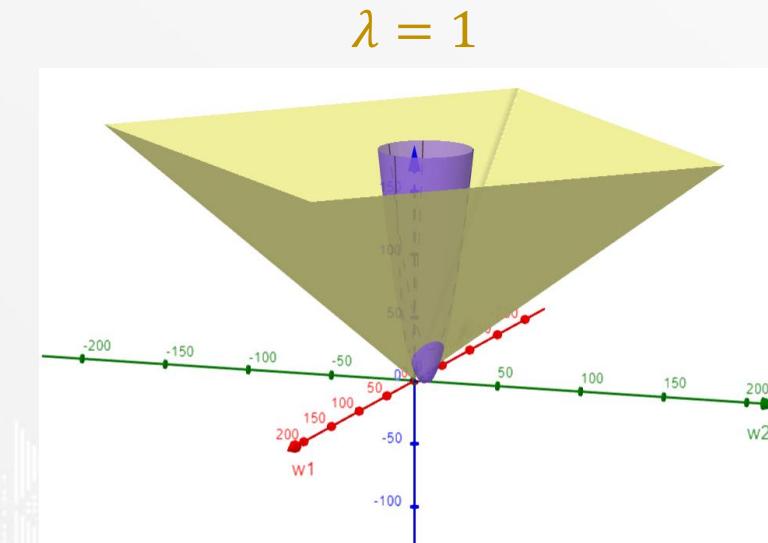
# Geometric View

No Multicollinearity

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



=

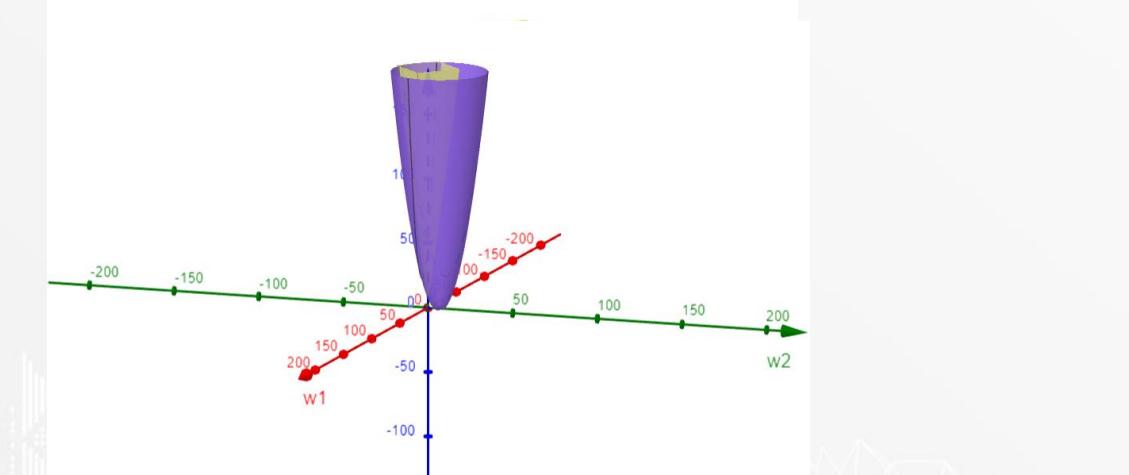
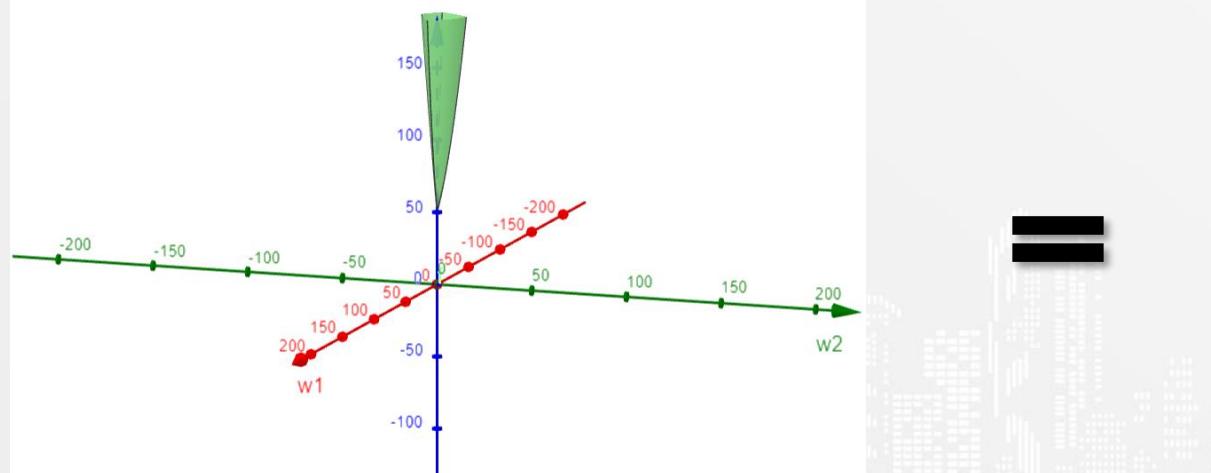


# Geometric View

No Multicollinearity

$$Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$

$$\lambda = 10$$



# Geometric View

## No Multicollinearity

- จุดตាំส្តុដីមែន មានឈូរភាពរវាងជុំតាំស្តុដីមែន និងជុំតាំស្តុដីកំណើន
- យើង លេខ មេរក ជុំតាំស្តុដីមែនយើង ក្រុមចុំតាំស្តុដីកំណើន
- យើង លេខ បានឈូរ ជុំតាំស្តុដីមែនយើង ក្រុមចុំតាំស្តុដីកំណើន

# Lasso Regression

**What is Lasso Regression?**

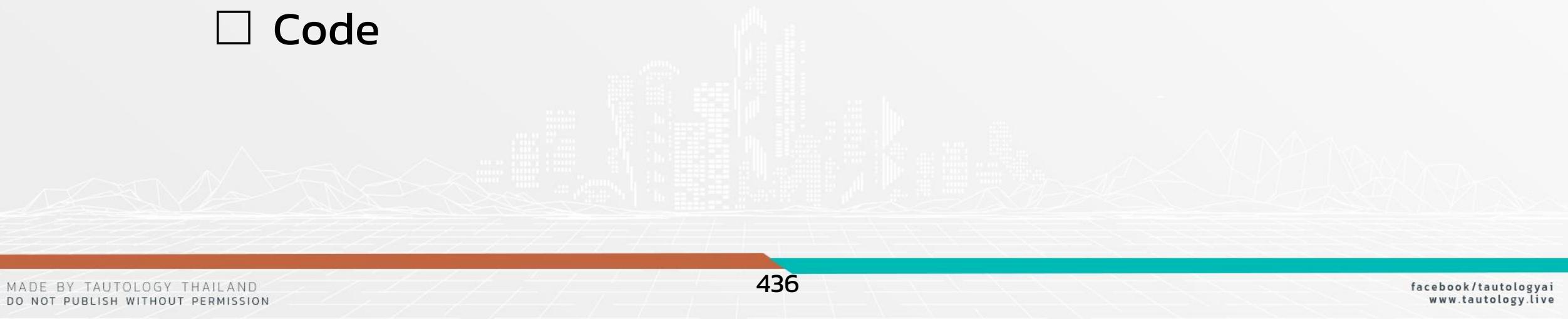
**Geometric View**

**Properties**

**Model Creation**

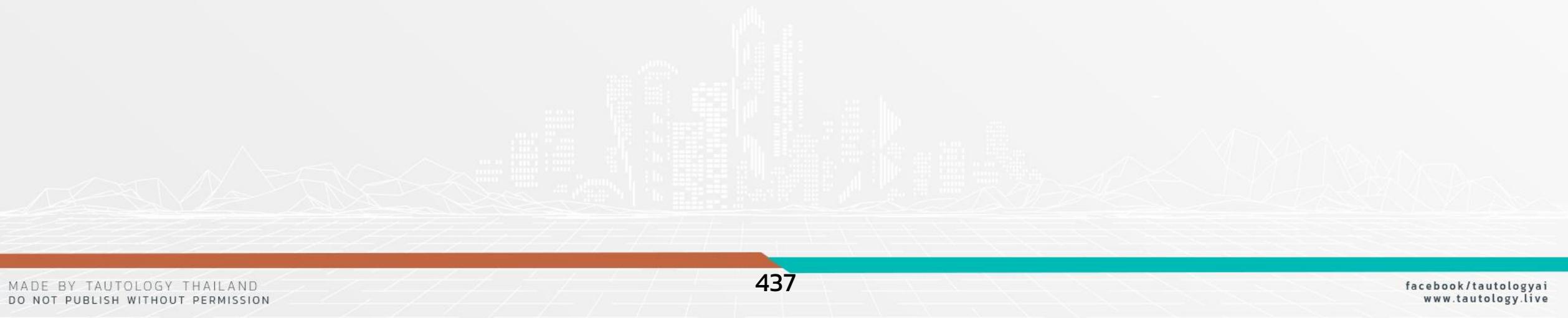
**How to find Lambda**

**Code**



# Properties

1. Generalization
2. Feature Selection



# Properties

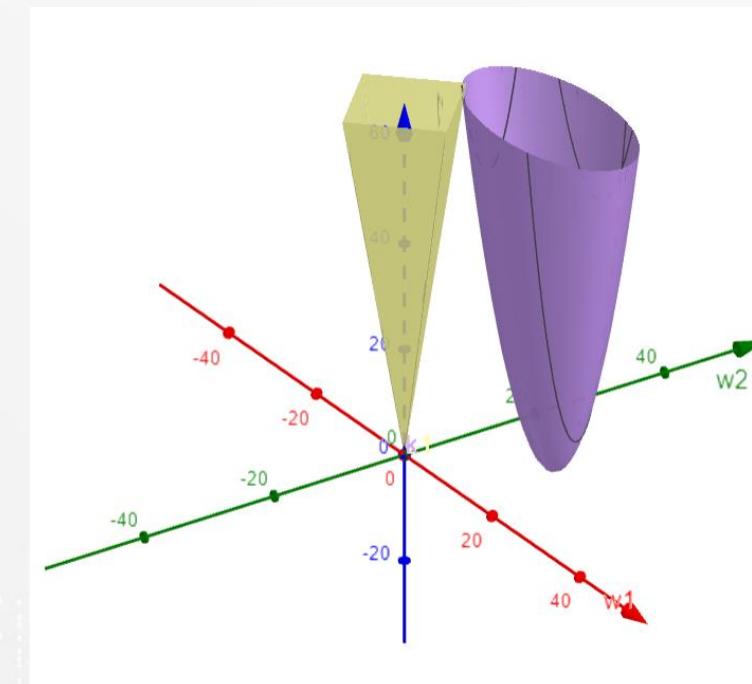
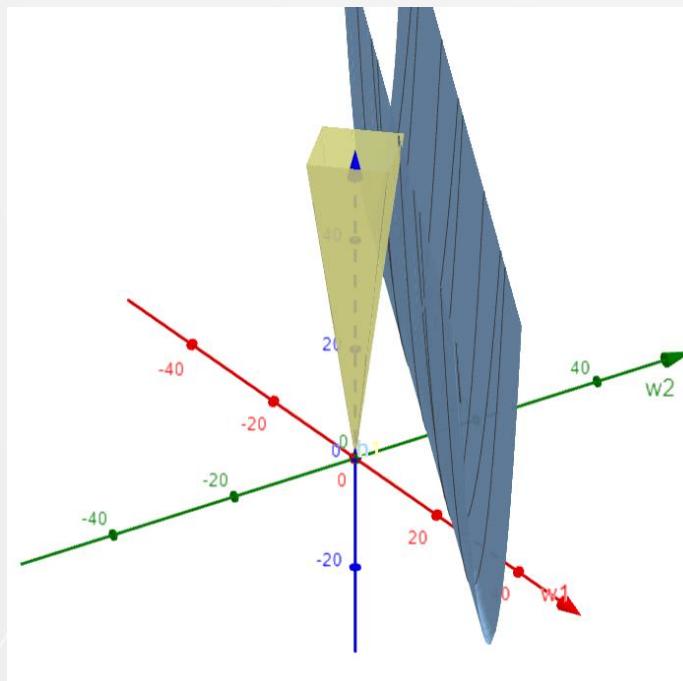
## 1. Generalization

“ การใส่  $\lambda > 0$  จะทำให้ error ของ model ลดลงกว่าการ  
ไม่ใส่  $\lambda$  ( $\lambda = 0$ ) สำหรับ data ทุกประเภท ”

\*ยกเว้น  $y = \hat{y}$  (model ไม่มี error)

# Properties

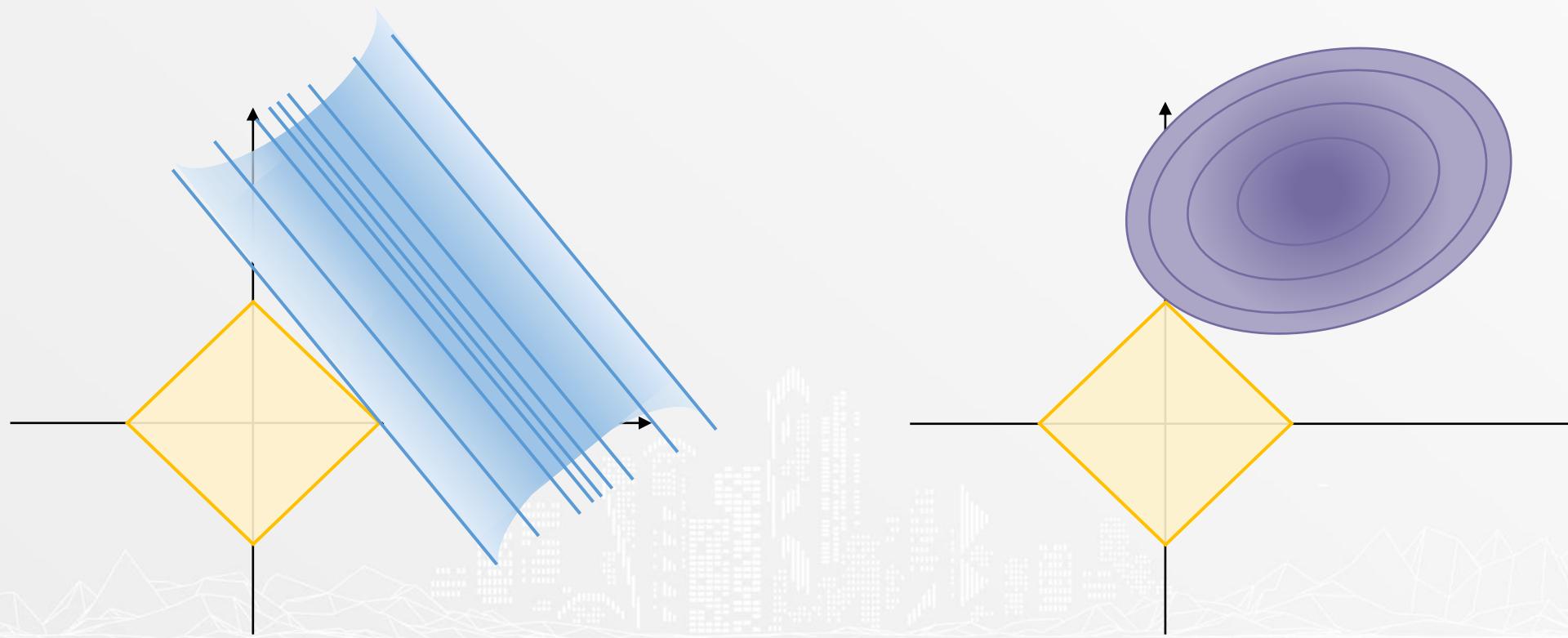
## 2. Feature Selection



<https://www.geogebra.org/m/c87cckmk>

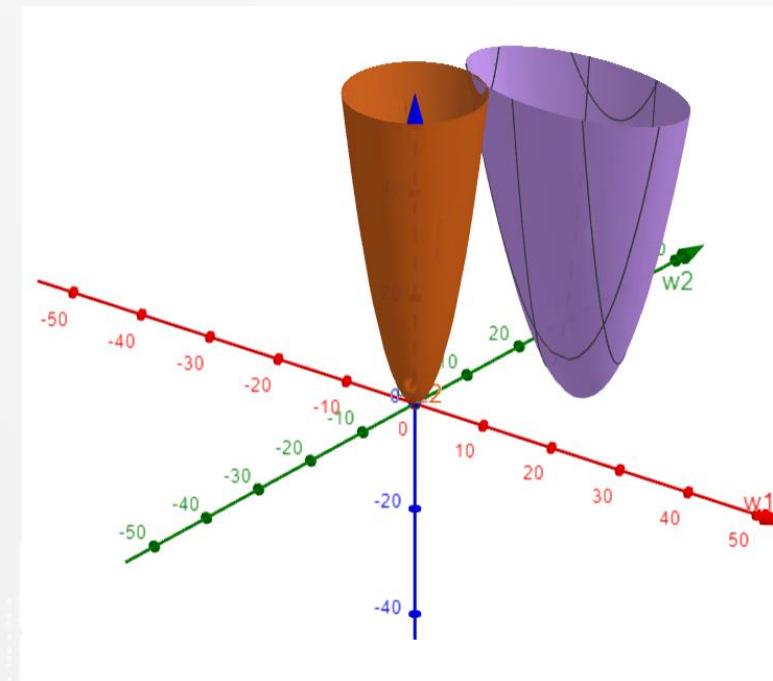
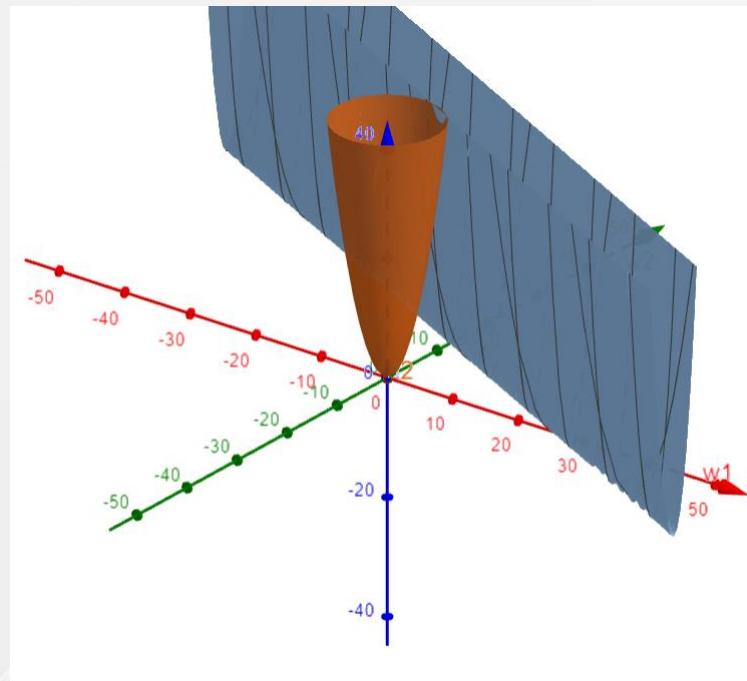
# Properties

## 2. Feature Selection



# Properties

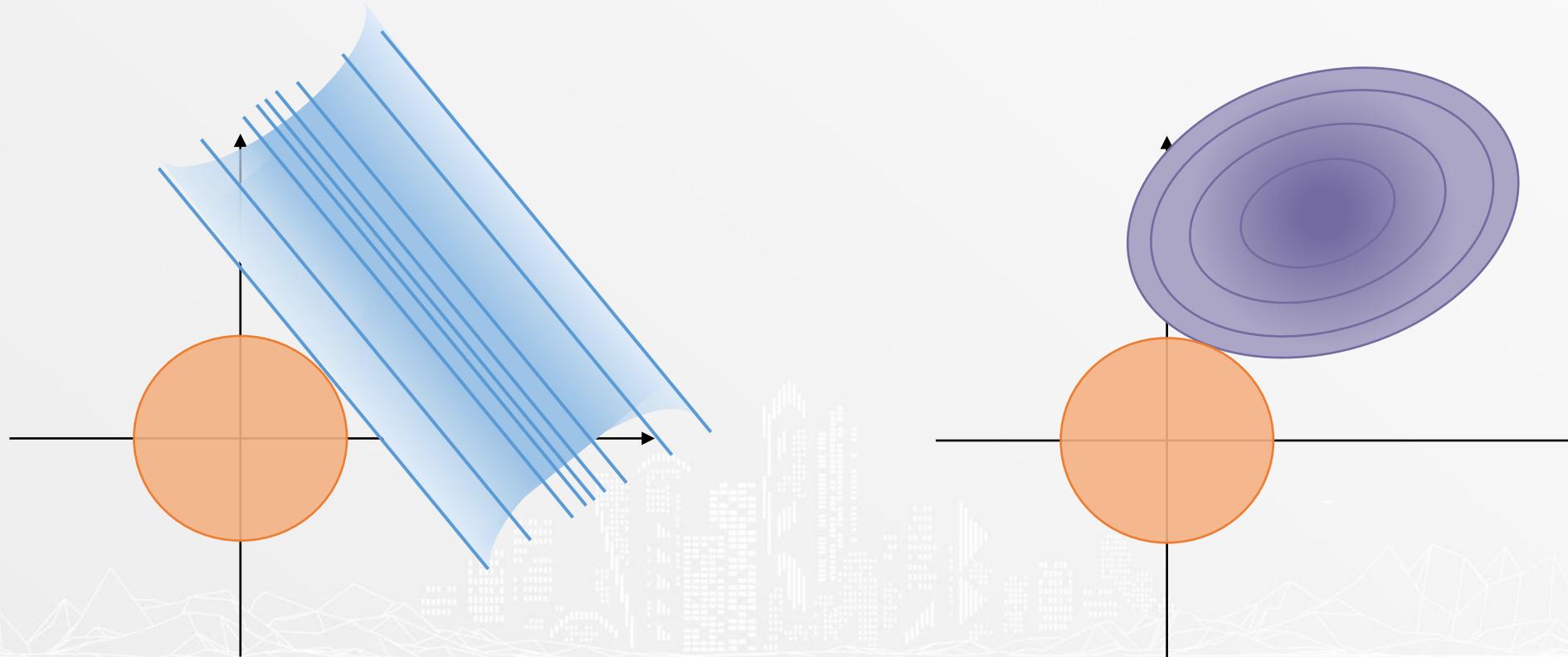
## 2. Feature Selection



<https://www.geogebra.org/3d/mzbky4wt>

# Properties

## 2. Feature Selection



# Lasso Regression

- What is Lasso Regression?**
- Geometric View**
- Properties**
- Model Creation
- How to find Lambda
- Code

# Model Creation

$$\nabla Cost = 0$$

$$\text{ໄດຍກີ } Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p |w_j|$$



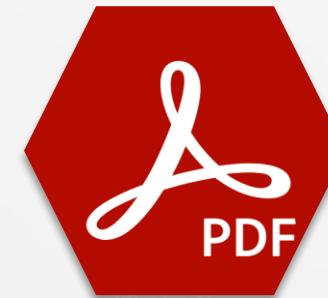
# Model Creation

$$X_b^T X_b \mathbf{w} + \lambda \nabla |\mathbf{w}| = X_b^T \mathbf{y}$$

# Model Creation



Derivation of Lasso Regression



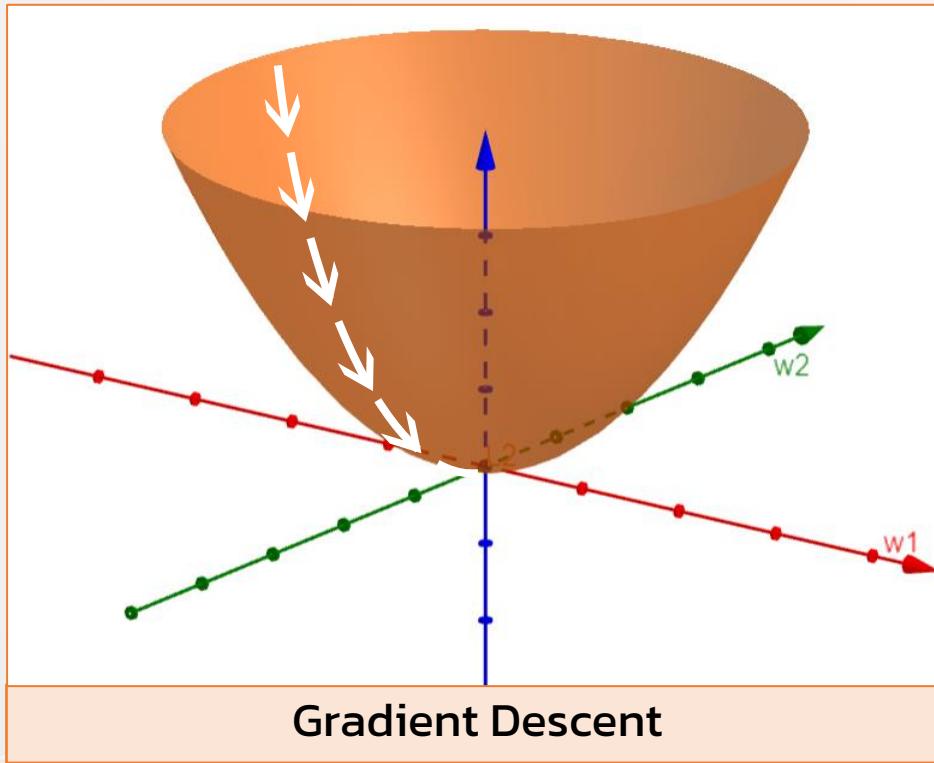
Open File  
**Derive\_Lasso.pdf**

# Model Creation

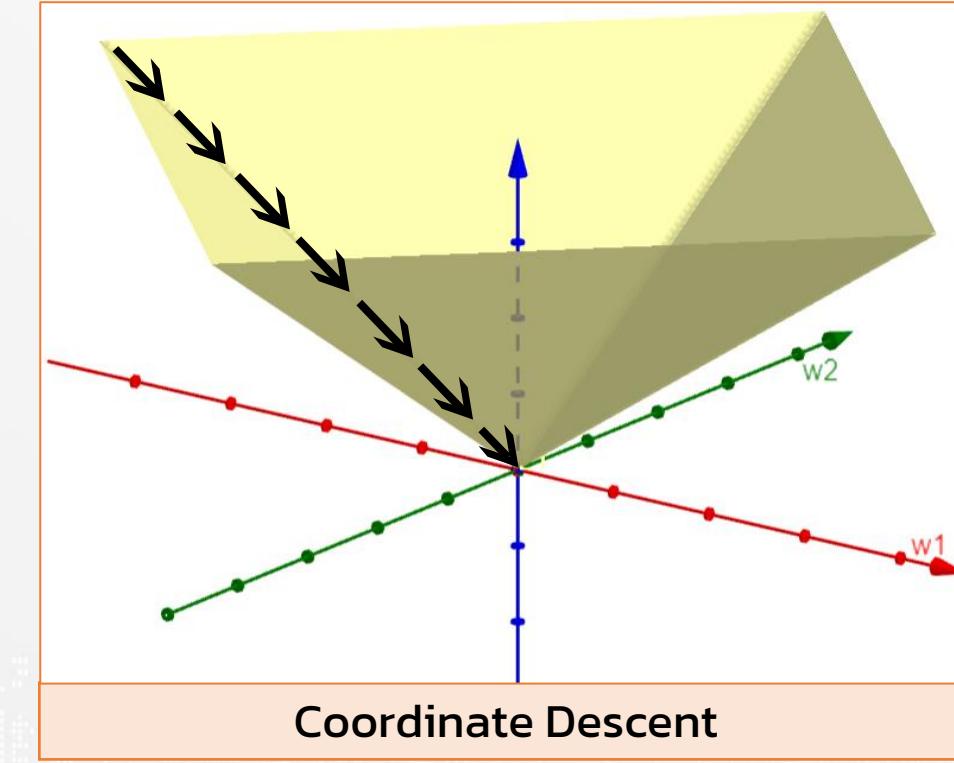
$$X_b^T X_b \mathbf{w} + \lambda \nabla |\mathbf{w}| = X_b^T \mathbf{y}$$



# Model Creation



Gradient Descent



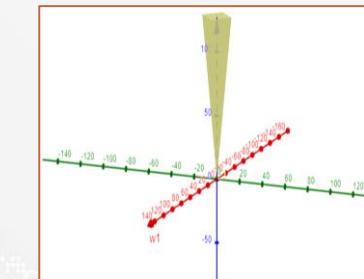
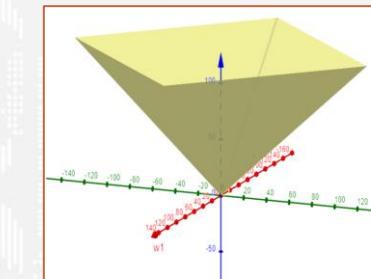
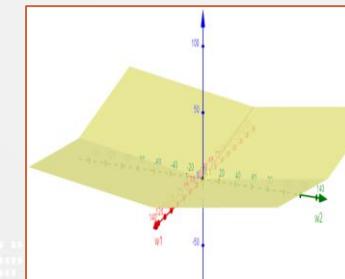
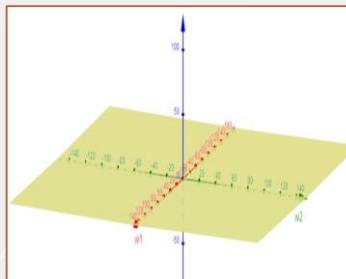
Coordinate Descent

# Lasso Regression

- What is Lasso Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda
- Code

# How to find Lambda

Q : แล้วเราจะ  $\lambda$  ที่ดีที่สุดได้อย่างไร?



# How to find Lambda

A: ใช้ Cross Validation

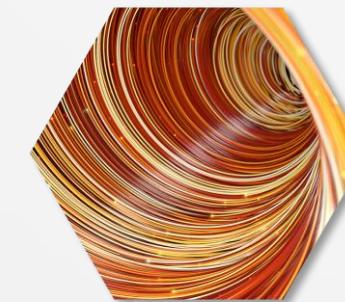
# How to find Lambda

$\lambda$	Fold					Mean	Rank
	1	2	3	4	5		
$\lambda = 0.01$	0.78	0.77	0.72	0.80	0.78	0.77	2
$\lambda = 0.1$	0.80	0.88	0.78	0.72	0.80	0.78	1
$\lambda = 1$	0.74	0.72	0.76	0.75	0.78	0.75	4
$\lambda = 10$	0.82	0.74	0.74	0.74	0.70	0.76	3

# How to find Lambda



For more information



## Cross Validation

# Lasso Regression

- What is Lasso Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda**
- Code**

# Code

## ตัวอย่าง code สำหรับ lasso regression

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>y</b>
0	1	3.9
2	1	7.7
1	1	6.2
2	0	5.2
3	1	9.8

# Code

- Code สำหรับสร้าง model จากข้อมูลของเราโดยที่

$$X = \begin{bmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 1 \\ 2 & 0 \\ 3 & 1 \end{bmatrix}, \quad y = \begin{bmatrix} 3.9 \\ 7.7 \\ 6.2 \\ 5.2 \\ 4.8 \end{bmatrix}$$

```
1 reg = LassoCV(n_alphas=100, cv=5)
2 reg.fit(X, y)
```

# Code

- ค่า  $\lambda$  ที่ดีที่สุด ถูกเก็บไว้ใน attribute ชื่อ `alpha_`

```
1 | reg.alpha_
```

```
0.001784000000000006
```

# Code

- ค่า  $w_0$  ถูกเก็บไว้ใน attribute ชื่อ intercept\_

```
1 reg.intercept_
```

```
1.374272891314945
```

# Code

- ค่า  $w_1, \dots, w_p$  จะเก็บไว้ใน attribute ชื่อ `coef_`

```
1 reg.coef_
array([1.91732355, 2.64751178])
```

# Code



Code for lasso regression



Open File  
**Regularization.ipynb**

# Lasso Regression

- What is Lasso Regression?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda**
- Code**

# Regularization

**What is  
Regularization?**



**Ridge Regression**



**Lasso Regression**



**Elastic Net**

**Conclusion**

# Elastic Net

- What is Elastic Net?
- Geometric View
- Properties
- Model Creation
- How to find Lambda &  $l1_{ratio}$
- Code

# What is Elastic Net?

Elastic Net คือ การทำ regularization โดยมี cost function เป็น

$$\begin{aligned} Cost = & \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda(l1_{ratio}) \sum_{j=0}^p |w_j| + \\ & \frac{1}{2} \lambda(1 - l1_{ratio}) \sum_{j=0}^p w_j^2 \end{aligned}$$

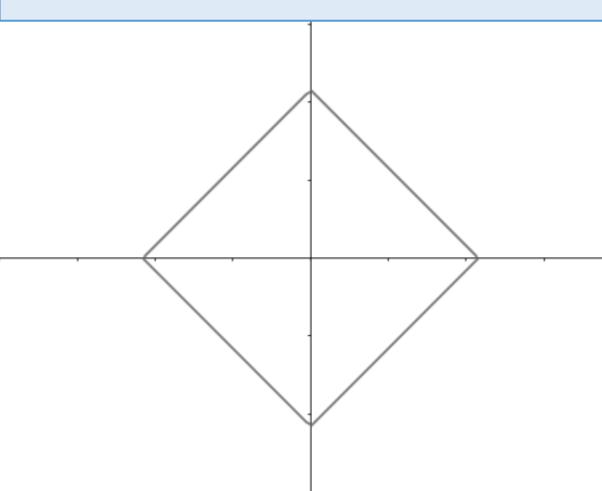
# Elastic Net

## **What is Elastic Net?**

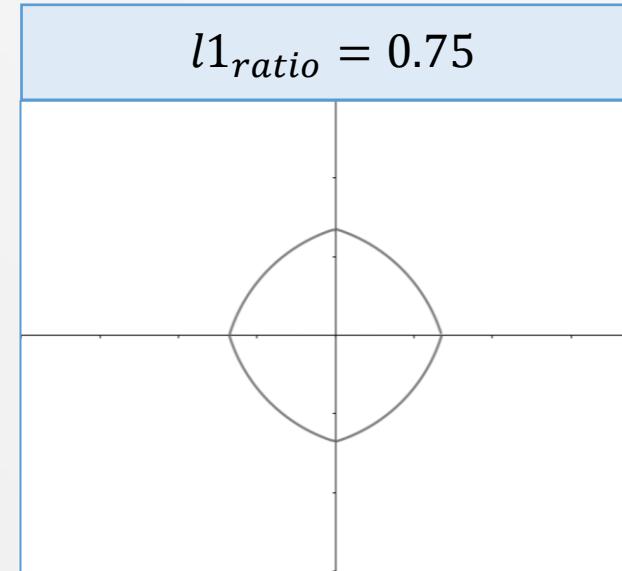
- Geometric View
- Properties
- Model Creation
- How to find Lambda &  $l1_{ratio}$
- Code

# Geometric View

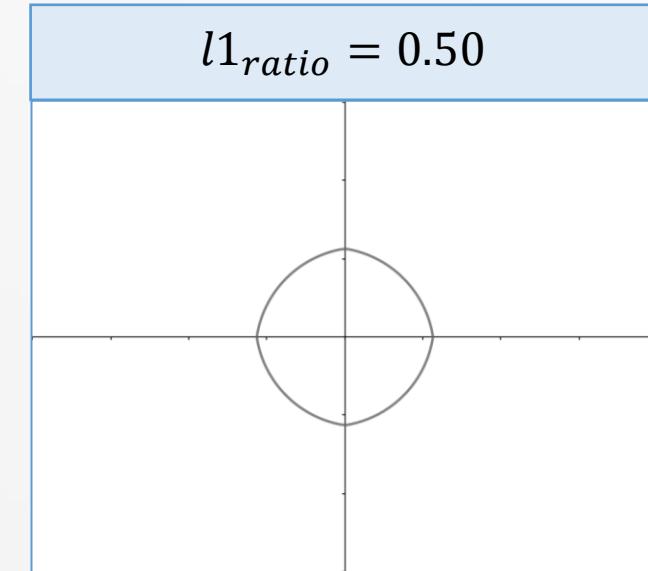
$l1_{ratio} = 1$



$l1_{ratio} = 0.75$

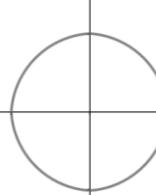


$l1_{ratio} = 0.50$

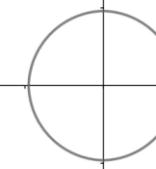


# Geometric View

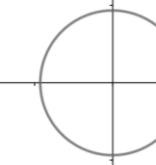
$l1_{ratio} = 0.25$



$l1_{ratio} = 0.10$



$l1_{ratio} = 0.01$



# Geometric View

## Linearly Dependent

- จุดตាំส្តុដឹង ធនូយូរេអវ៉ាងសមការរំងកំពេប និង ជុំកំណើន
- កំពេបនេះបានផ្តល់ឱ្យសមការបង្ហីរំងកំពេប ឬ ជុំកំណើនដីយុ
- យើង  $\lambda$  មាត្រា ជុំកំណើន ឱ្យក្នុងក្នុងកំណើន
- យើង  $\lambda$  បាន ជុំកំណើន ឱ្យក្នុងក្នុងកំពេប

# Geometric View

## No Multicollinearity

- จุดตាំส្តុដីមែន មានឈូរភាពរវាងជុំតាំស្តុដីមែន និងជុំតាំស្តុដីកំណើន
- យើង លេខ មេរក ជុំតាំស្តុដីមែនយើង ក្រោមឈូរភាពរវាងជុំតាំស្តុដីកំណើន
- យើង លេខ បានឈូរ ជុំតាំស្តុដីមែនយើង ក្រោមឈូរភាពរវាងជុំតាំស្តុដីកំណើន

# Elastic Net

- What is Elastic Net?**
- Geometric View**
- Properties
- Model Creation
- How to find Lambda &  $l1_{ratio}$
- Code

# Properties

1. Generalization
2. Soft Feature Selection



# Properties

## 1. Generalization

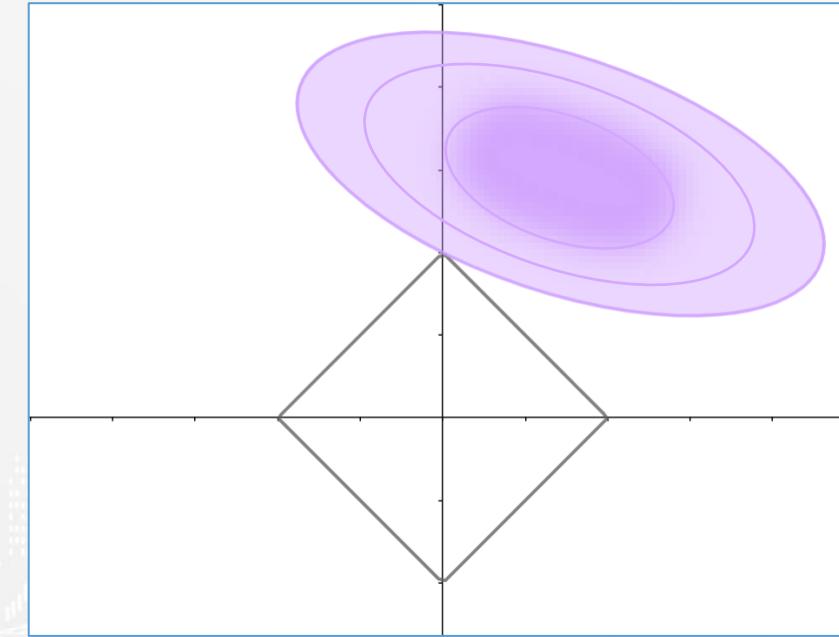
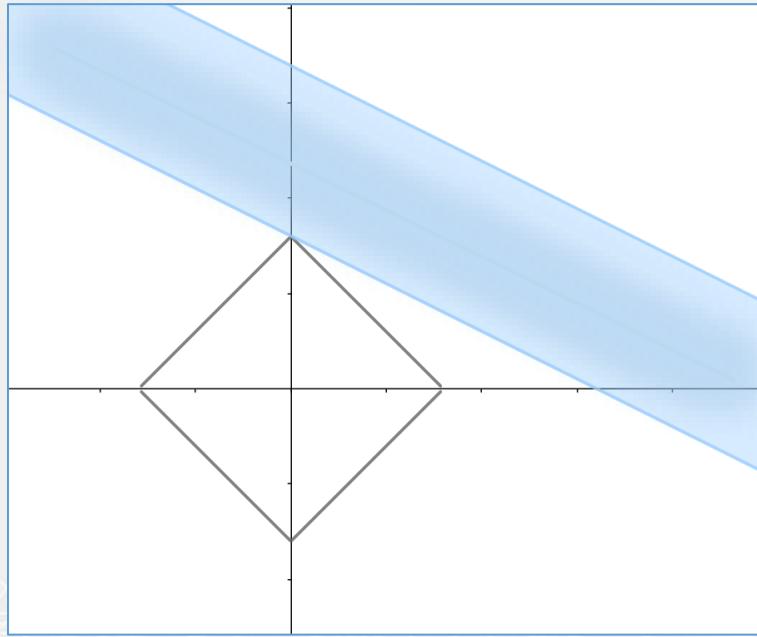
“ การใส่  $\lambda > 0$  จะทำให้ error ของ model ลดลงกว่าการ  
ไม่ใส่  $\lambda$  ( $\lambda = 0$ ) สำหรับ data ทุกประเภท ”

\*ยกเว้น  $y = \hat{y}$  (model ไม่มี error)

# Properties

## 2. Soft Feature Selection

- ◆  $l1_{ratio} = 1$

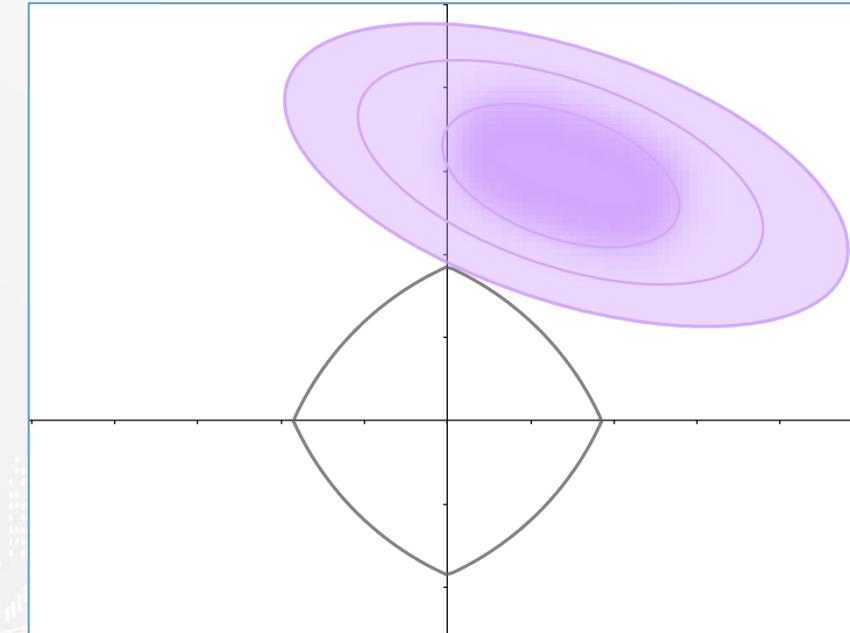
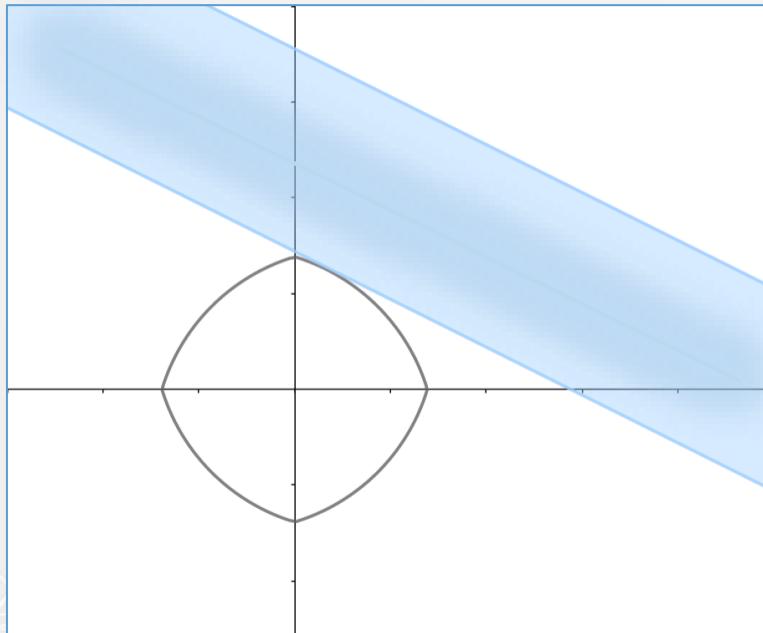


<https://www.geogebra.org/3d/pwcw4ay5>

# Properties

## 2. Soft Feature Selection

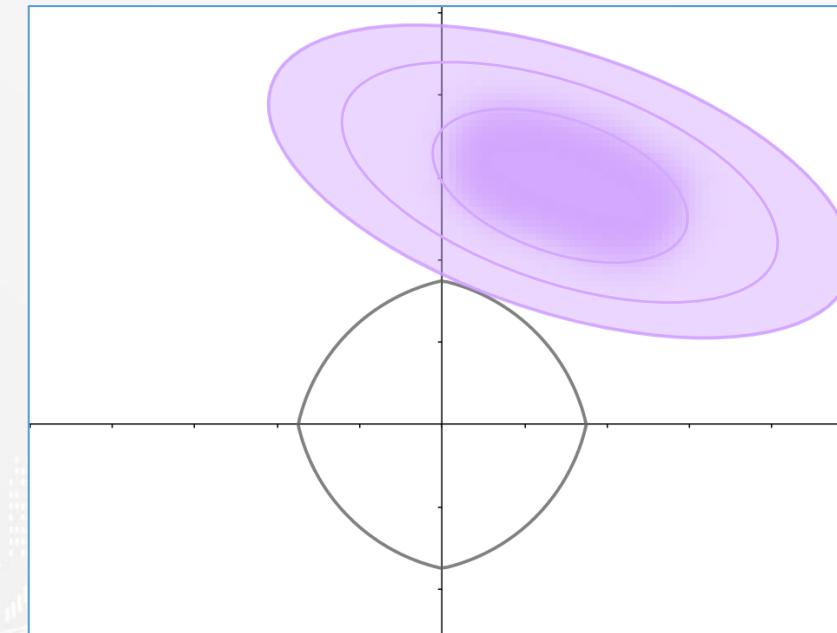
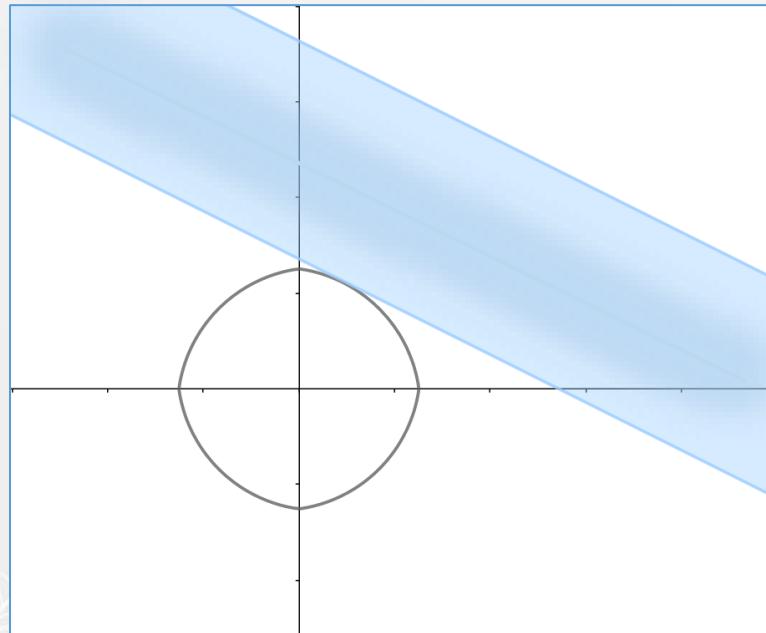
- ◆  $l1_{ratio} = 0.75$



# Properties

## 2. Soft Feature Selection

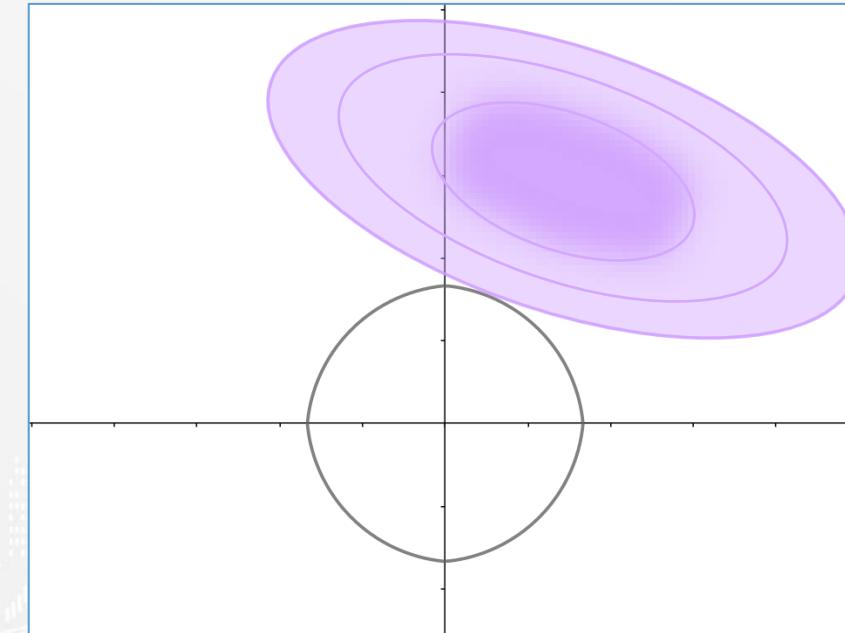
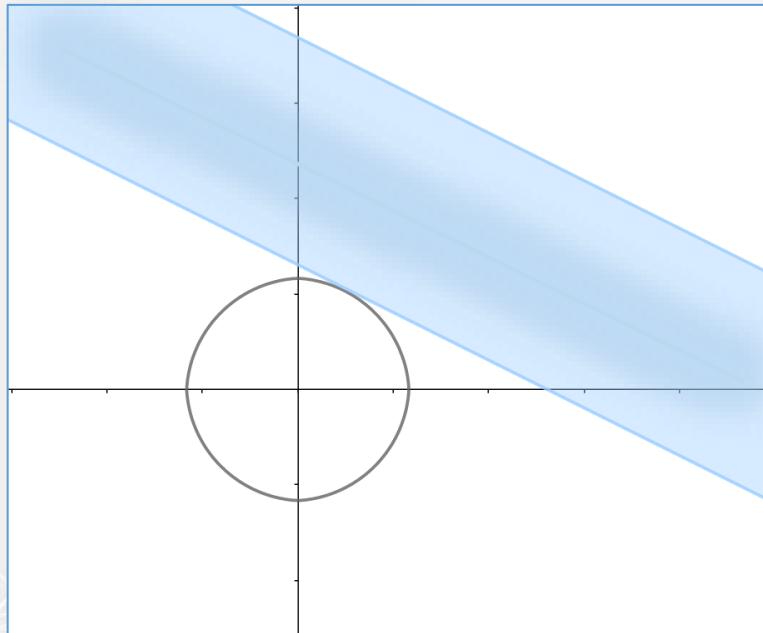
- ◆  $l1_{ratio} = 0.50$



# Properties

## 2. Soft Feature Selection

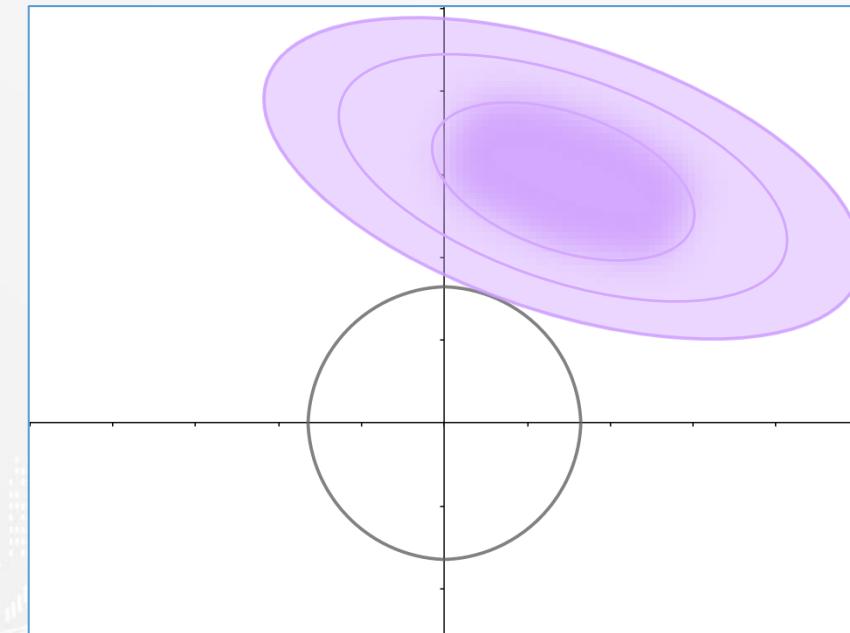
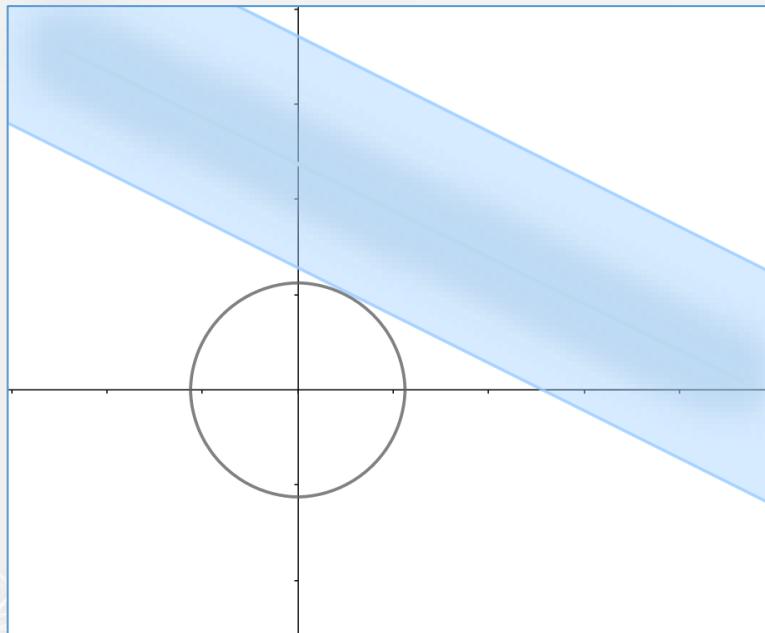
- ◆  $l1_{ratio} = 0.25$



# Properties

## 2. Soft Feature Selection

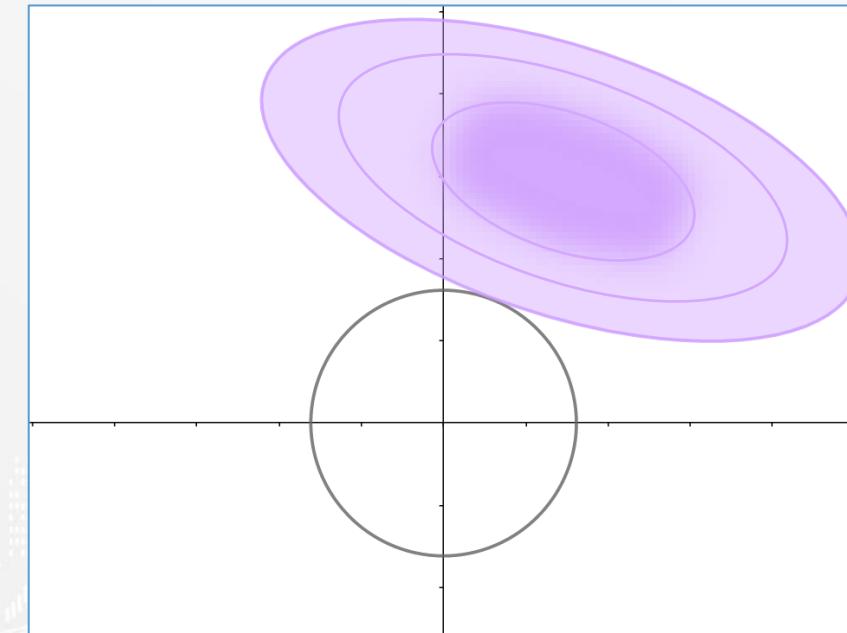
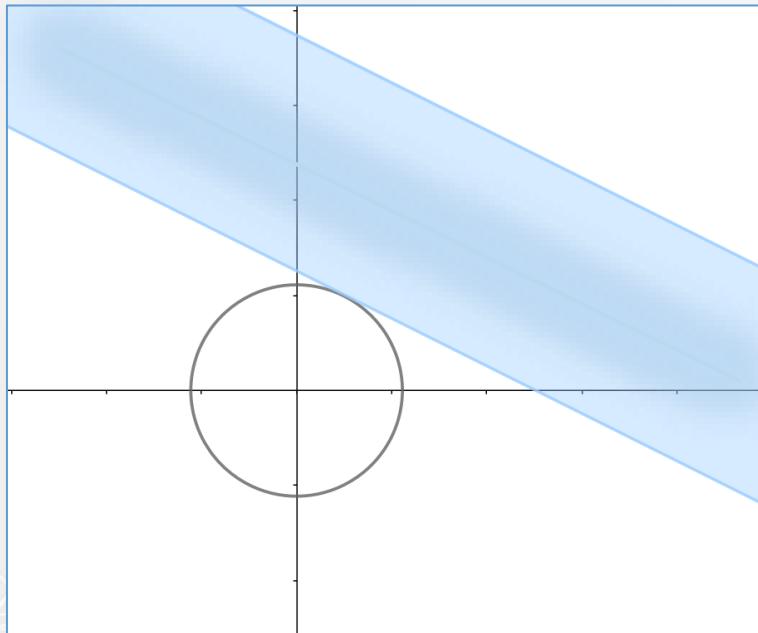
- ◆  $l1_{ratio} = 0.10$



# Properties

## 2. Soft Feature Selection

- ◆  $l1_{ratio} = 0.01$



# Elastic Net

- What is Elastic Net?**
- Geometric View**
- Properties**
- Model Creation
- How to find Lambda &  $l1_{ratio}$
- Code

# Model Creation

$$\nabla Cost = 0$$

ໄດຍ້  $Cost = \frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda(l1_{ratio}) \sum_{j=0}^p |w_j| + \frac{1}{2} \lambda(1 - l1_{ratio}) \sum_{j=0}^p w_j^2$

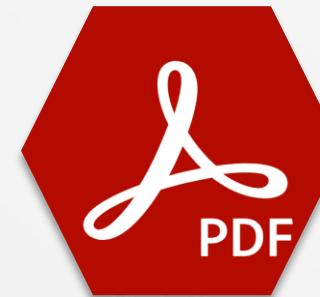
# Model Creation

$$\left( X_b^T X_b + \lambda(1 - l1_{ratio}) \right) \mathbf{w} + \lambda(l1_{ratio}) \nabla |\mathbf{w}| = X_b^T \mathbf{y}$$

# Model Creation



Derivation of Elastic Net



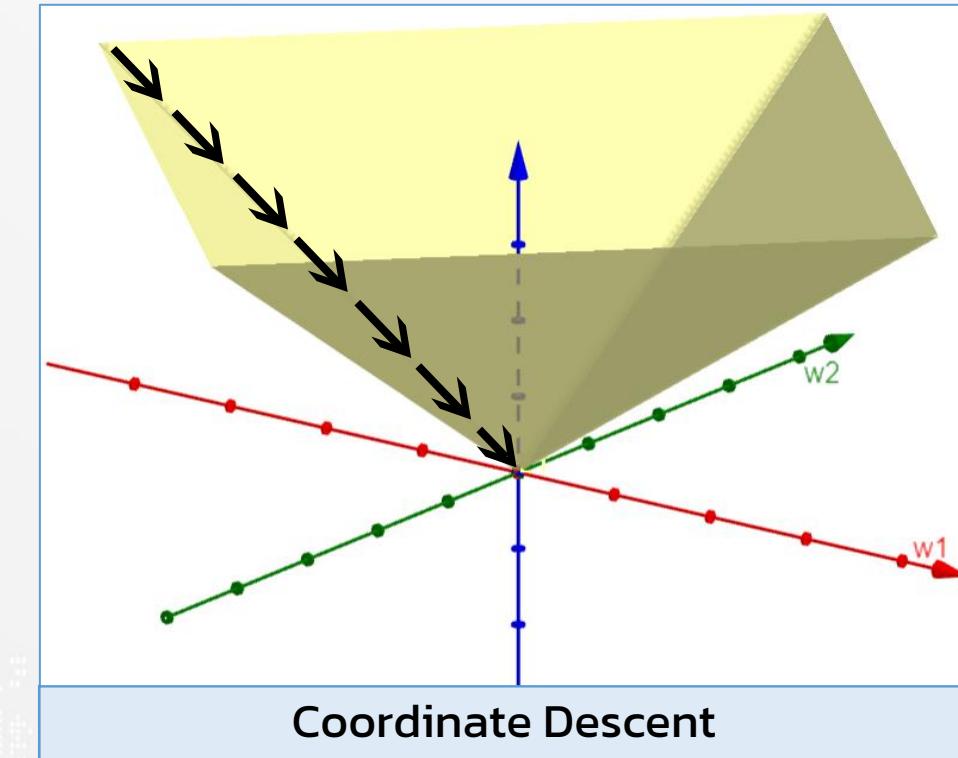
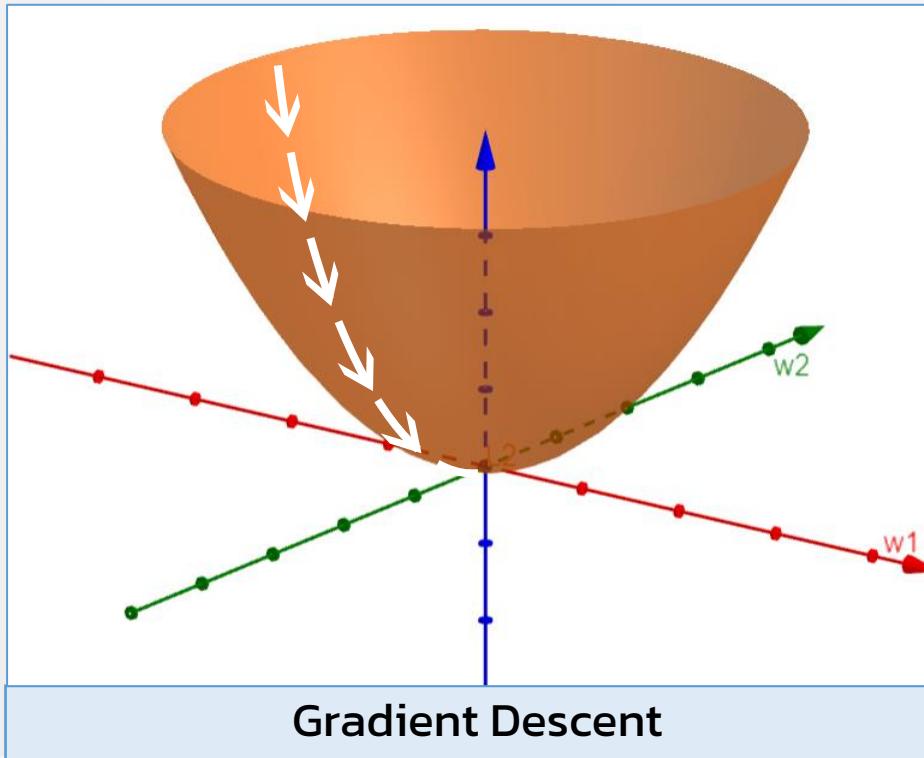
Open File  
**Derive\_Elastic Net.pdf**

# Model Creation

$$\left( X_b^T X_b + \lambda(1 - l1_{ratio}) \right) \mathbf{w} + \lambda(l1_{ratio}) \nabla |\mathbf{w}| = X_b^T \mathbf{y}$$



# Model Creation

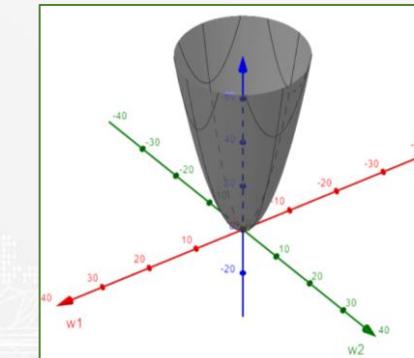
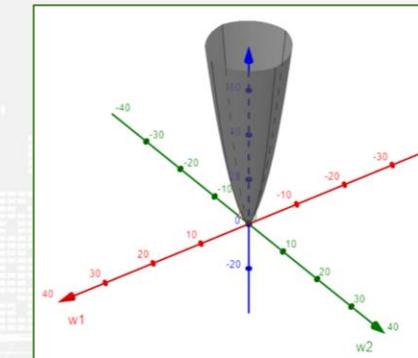
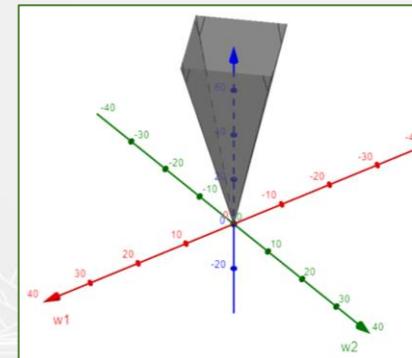


# Elastic Net

- What is Elastic Net?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda &  $l1_{ratio}$
- Code

# How to find Lambda & $l1_{ratio}$

Q: แล้วเราจะ  $\lambda$  และ  $l1_{ratio}$  ที่ดีที่สุดได้อย่างไร?



# How to find Lambda & $l_1$ ratio

A: چیز Cross Validation

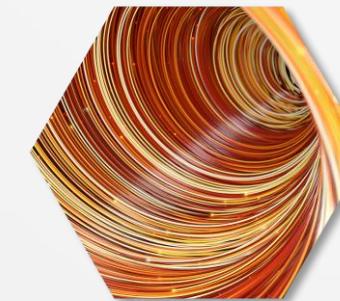
# How to find Lambda & $l1_{ratio}$

$l1_{ratio}$	$\lambda$	Fold					Mean	Rank
		1	2	3	4	5		
0.1	0.1	0.78	0.77	0.72	0.80	0.78	0.77	2
	1	0.80	0.88	0.78	0.72	0.80	0.78	1
0.5	0.1	0.73	0.70	0.75	0.76	0.77	0.74	5
	1	0.80	0.76	0.74	0.73	0.71	0.748	4
0.9	0.1	0.75	0.72	0.78	0.76	0.79	0.76	3
	1	0.78	0.82	0.77	0.70	0.78	0.77	2

# How to find Lambda & $l_1$ ratio



For more information



## Cross Validation

# Elastic Net

- What is Elastic Net?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda &  $l1_{ratio}$**
- Code**

# Code

## ตัวอย่าง code สำหรับ elastic net

<b>x<sub>1</sub></b>	<b>x<sub>2</sub></b>	<b>y</b>
0	1	3.9
2	1	7.7
1	1	6.2
2	0	5.2
3	1	9.8

# Code

- Code สำหรับสร้าง model จากข้อมูลของเราโดยที่

$$X = \begin{bmatrix} 0 & 1 \\ 2 & 1 \\ 1 & 1 \\ 2 & 0 \\ 3 & 1 \end{bmatrix}, \quad y = \begin{bmatrix} 3.9 \\ 7.7 \\ 6.2 \\ 5.2 \\ 4.8 \end{bmatrix}$$

```
1 l1_ratio = [0.01, 0.05, 0.1, 0.5, 0.7, 0.9, 0.95, 0.99, 1]
```

```
1 reg = ElasticNetCV(l1_ratio=l1_ratio, n_alphas=100, cv=5)
2 reg.fit(X, y)
```

# Code

- ค่า  $l1_{ratio}$  ที่ดีที่สุด ถูกเก็บไว้ใน attribute ชื่อ `l1_ratio_`

```
1 reg.l1_ratio_
```

```
1.0
```

# Code

- ค่า  $\lambda$  ที่ดีที่สุด ถูกเก็บไว้ใน attribute ชื่อ `alpha_`

```
1 reg.alpha_
```

```
0.001784000000000006
```

# Code

- ค่า  $w_0$  ถูกเก็บไว้ใน attribute ชื่อ intercept\_

```
1 reg.intercept_
```

```
1.374272891314945
```

# Code

- ค่า  $w_1, \dots, w_p$  จะเก็บไว้ใน attribute ชื่อ `coef_`

```
1 reg.coef_
array([1.91732355, 2.64751178])
```

# Code



Code for Elastic Net



Open File  
**Regularization.ipynb**

# Elastic Net

- What is Elastic Net?**
- Geometric View**
- Properties**
- Model Creation**
- How to find Lambda &  $l1_{ratio}$**
- Code**

# Regularization

**What is  
Regularization?**



**Ridge Regression**



**Lasso Regression**



**Elastic Net**



**Conclusion**



# Conclusion

	Normal Equation	Ridge Regression	Lasso Regression	Elastic Net
cost function	$\sum_{i=1}^n (y_i - \hat{y}_i)^2$	$\sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p w_j^2$	$\frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=0}^p  w_j $	$\frac{1}{2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda(l1_{ratio}) \sum_{j=0}^p  w_j  + \frac{1}{2} \lambda(1 - l1_{ratio}) \sum_{j=0}^p w_j^2$
generalization	✗	✓	✓	✓
$\lambda$	✗	✓	✓	✓
$l1_{ratio}$	✗	✗	✗	✓
hyperparameter tuning	✗	CV	CV	CV
Feature selection	✗	✗	✓	✓

# Regularization

**What is  
Regularization?**



**Ridge Regression**



**Lasso Regression**



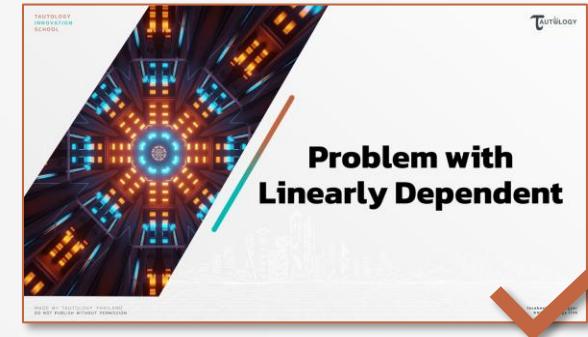
**Elastic Net**



**Conclusion**



# Model Improvement



# DL101 : Linear Regression



TAUTOLOGY  
INNOVATION  
SCHOOL

ADVANCED WORKSHOP



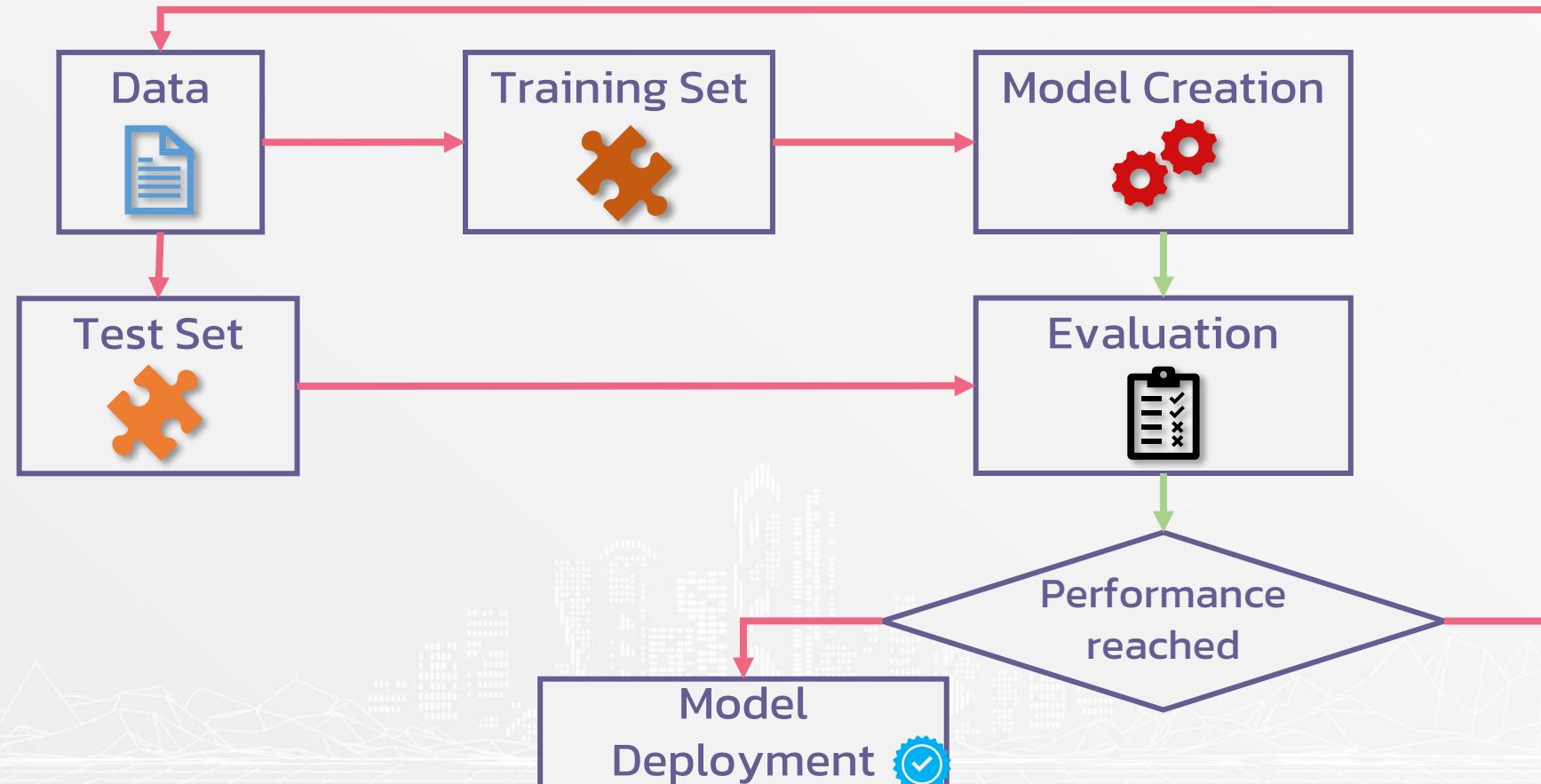
# ADVANCED WORKSHOP

BY TAUTOLOGY

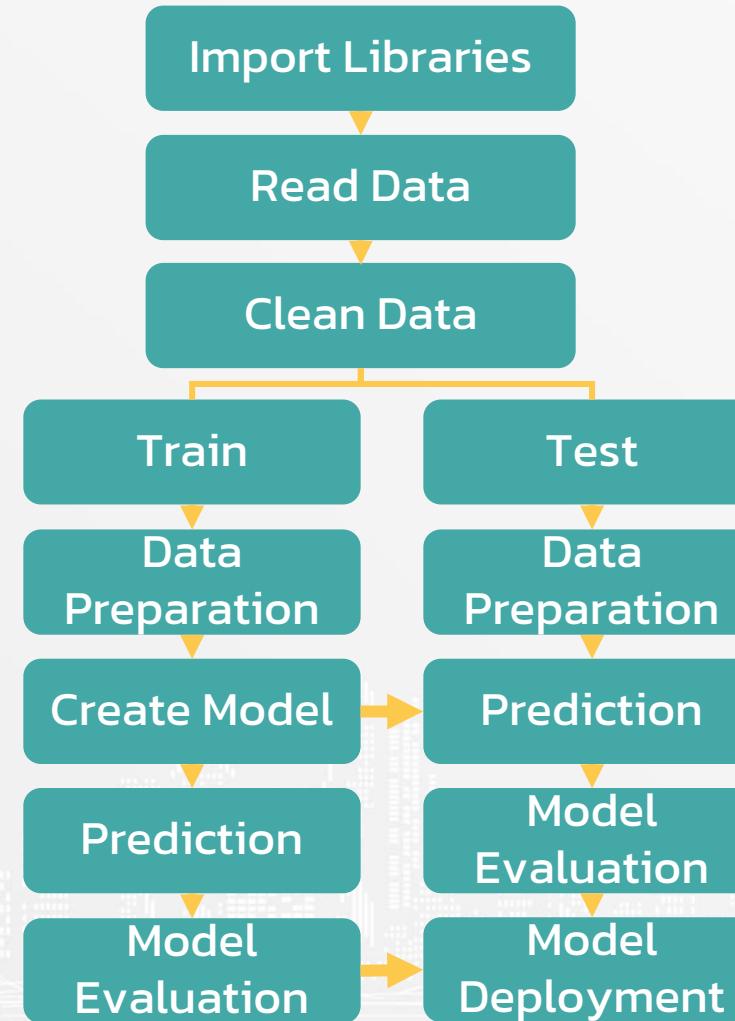
MADE BY TAUTOLOGY THAILAND  
DO NOT PUBLISH WITHOUT PERMISSION

facebook/tautologyai  
www.tautology.live

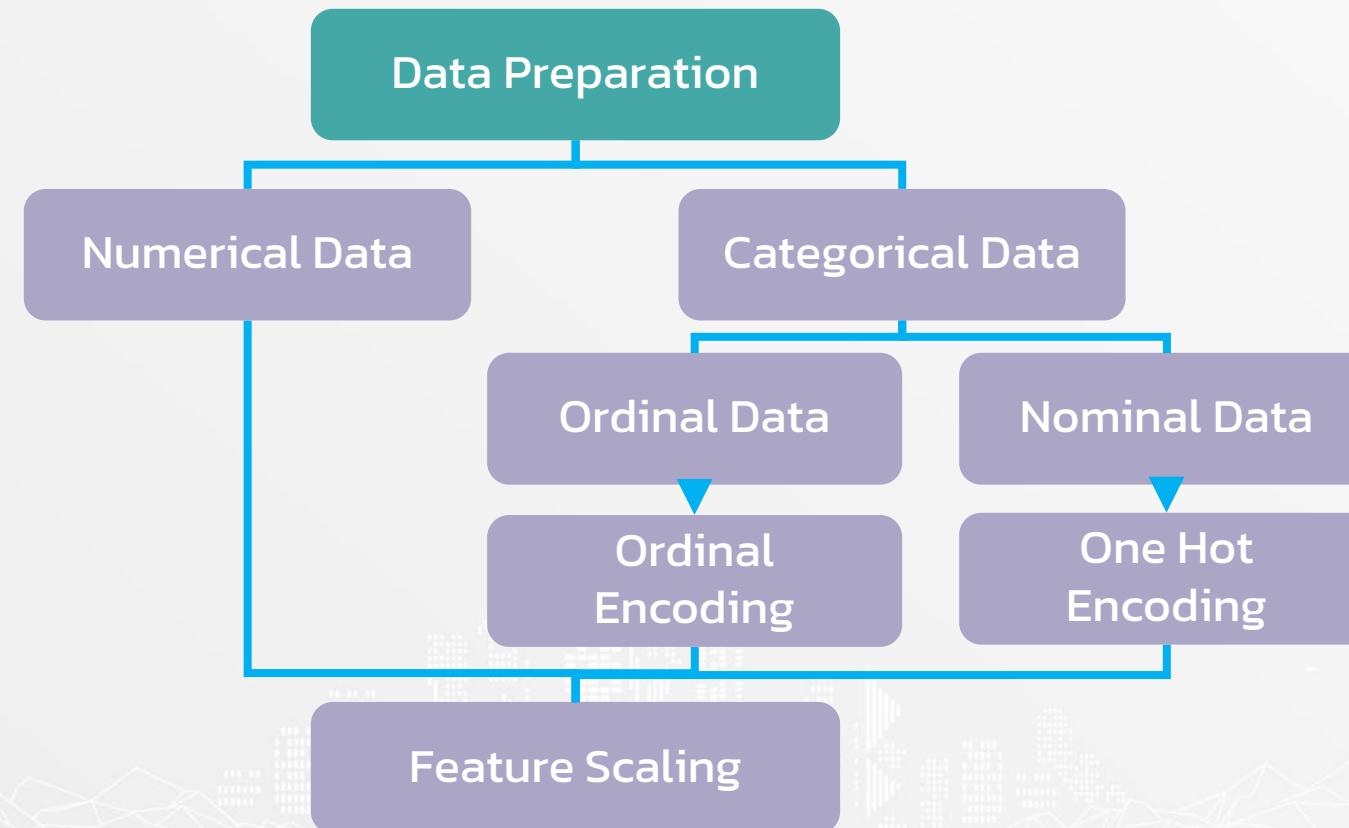
# Supervised Learning Workflow



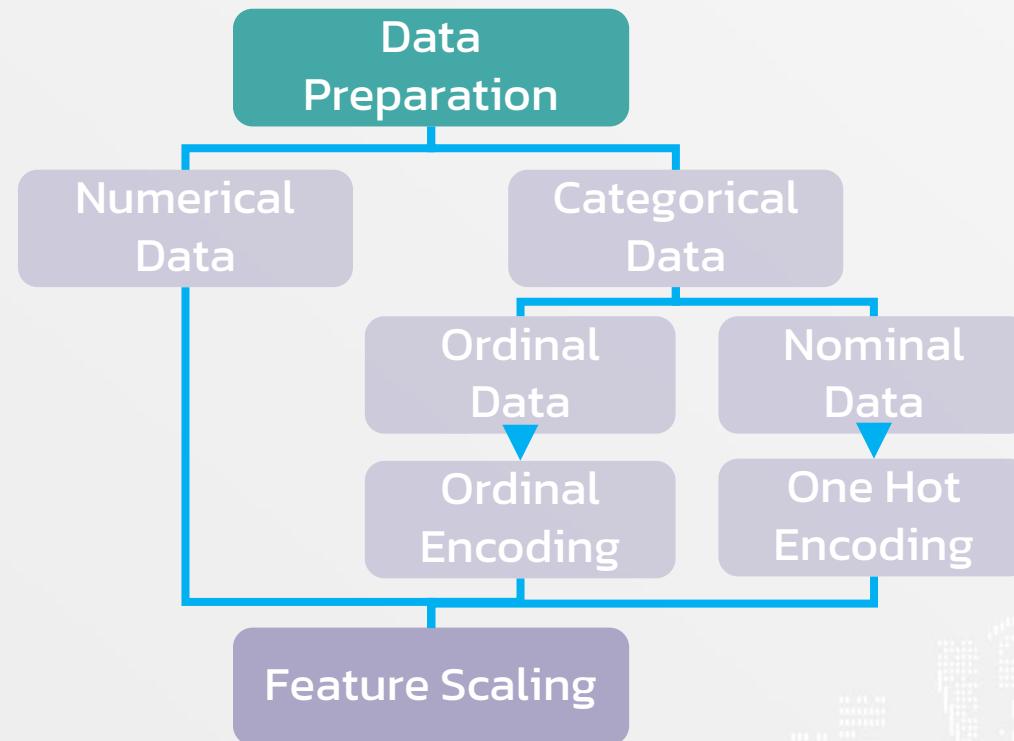
# Code Pipeline



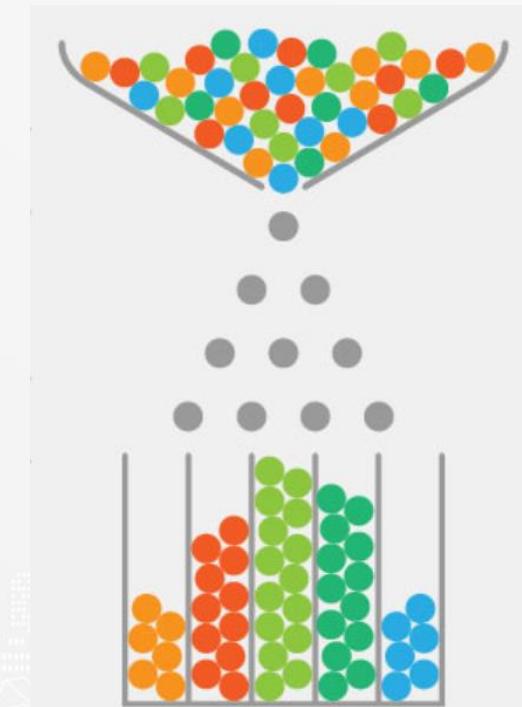
# Data Preparation



# Data Preparation



## Feature Scaling



# Code

- Feature Scaling for **training set**

```
1 | scaler = StandardScaler()  
2 | X_train_scaled = scaler.fit_transform(X_train)
```

- Feature Scaling for **test set**

```
1 | X_test_scaled = scaler.transform(X_test)
```

# Car Price

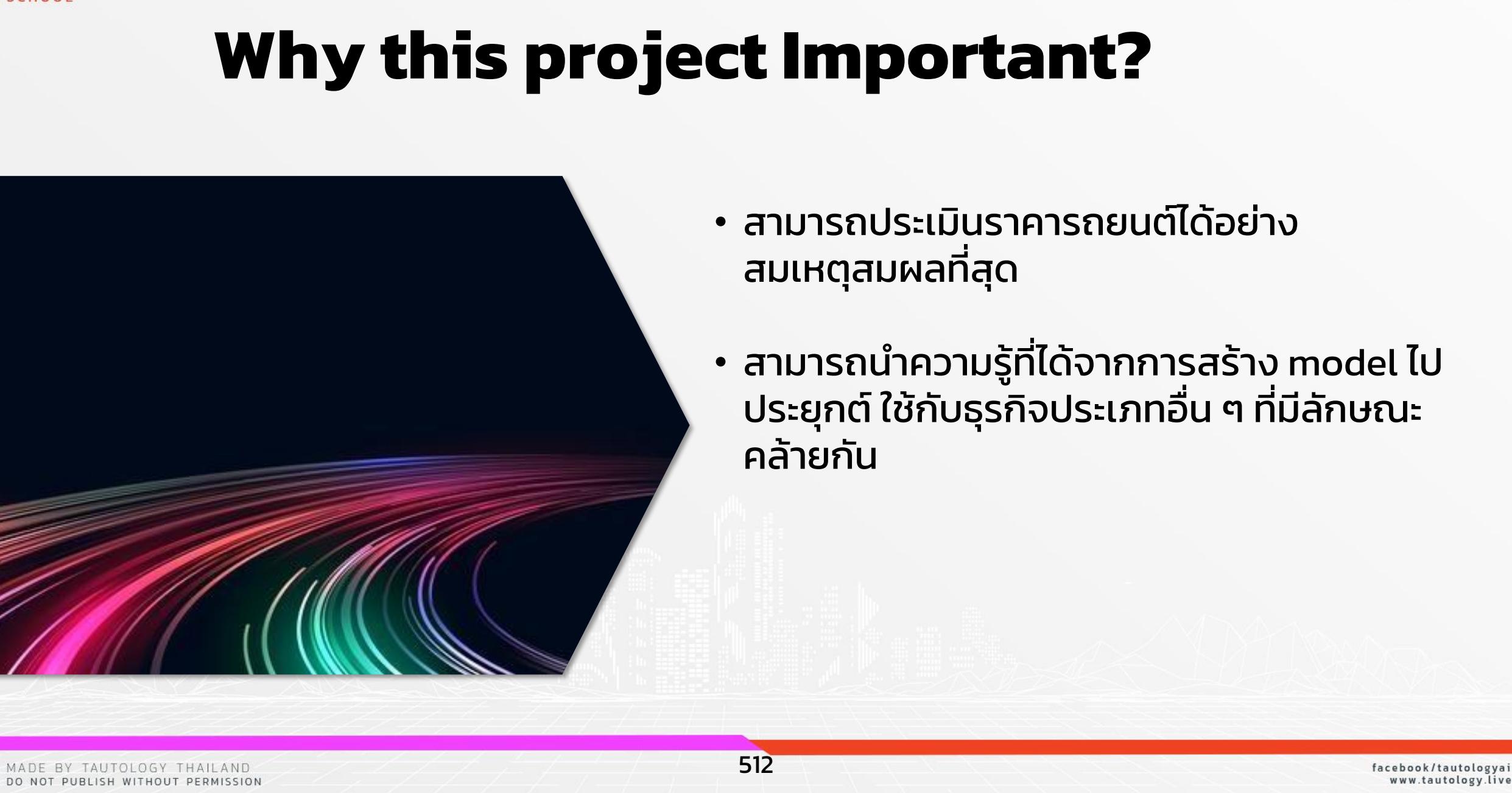
# Abstract

สร้าง model เพื่อประเมินราคาของรถยนต์มือสอง โดย feature ที่นำมาใช้ คือ ข้อมูลเกี่ยวกับรถยนต์คันนั้น ๆ และรูปแบบการขาย เช่น

- ยี่ห้อรถยนต์
- เลขกิโลของรถยนต์
- ขายด้วยตัวเอง หรือ ขายผ่านนายหน้า

# Why this project Important?

- สามารถประเมินราคารถยนต์ได้อย่างสมเหตุสมผลที่สุด
- สามารถนำความรู้ที่ได้จากการสร้าง model ไปประยุกต์ใช้กับธุรกิจประเภทอื่น ๆ ที่มีลักษณะคล้ายกัน

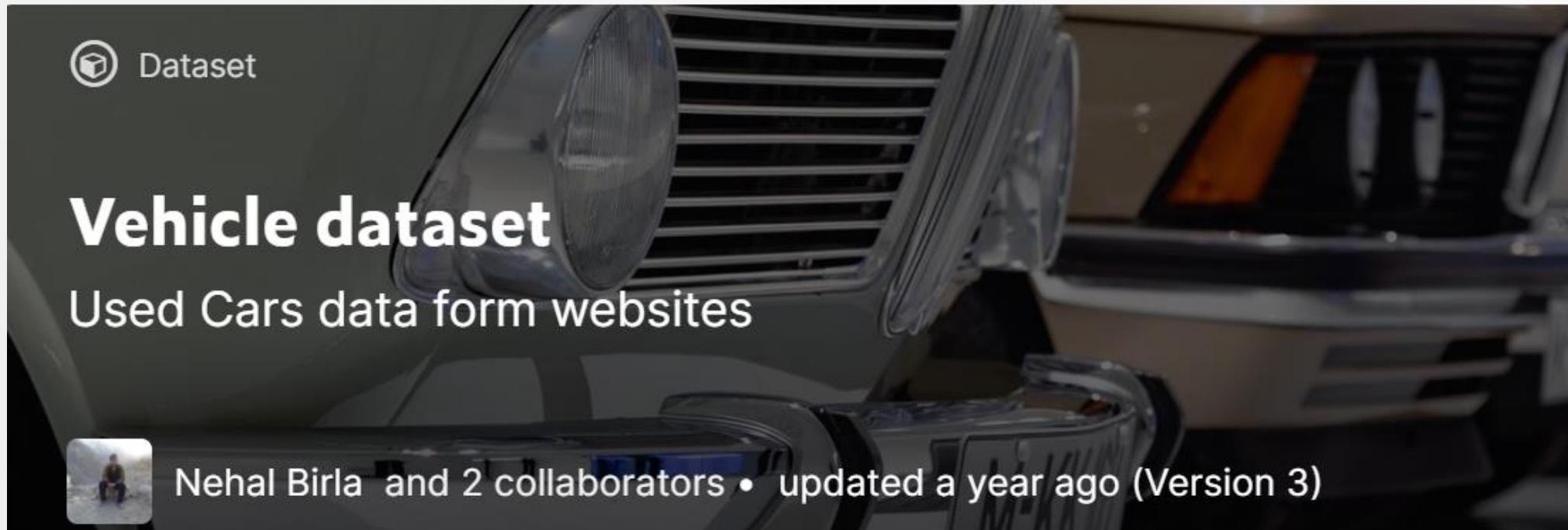


# Who this project is for?

- เจ้าของธุรกิจเต็ิมก๊อรด
- นักประเมินราคารถยนต์
- ผู้เกี่ยวข้องกับธุรกิจที่ต้องประเมินราคา เช่น ธุรกิจโรงรับจำนำ ร้านซื้อ/ขายเครื่องดูดน้ำมัน ธนาคาร
- นักวิเคราะห์ข้อมูล



# Car Price Dataset



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

# Car Price Dataset

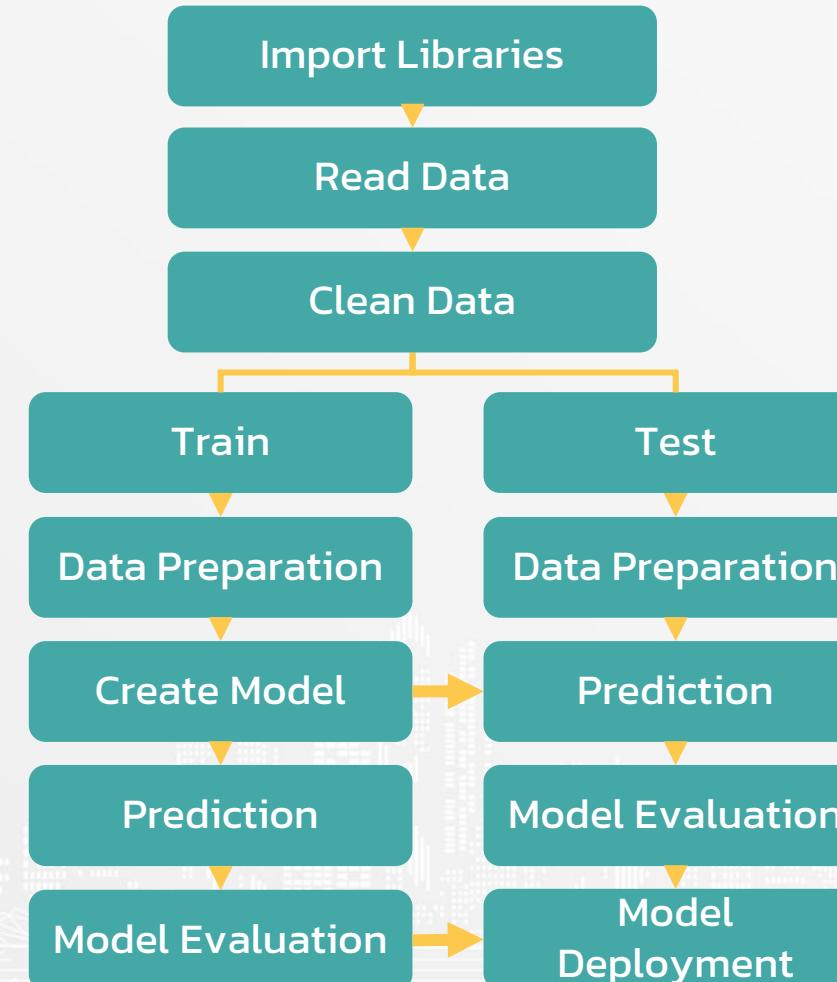
## Feature

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

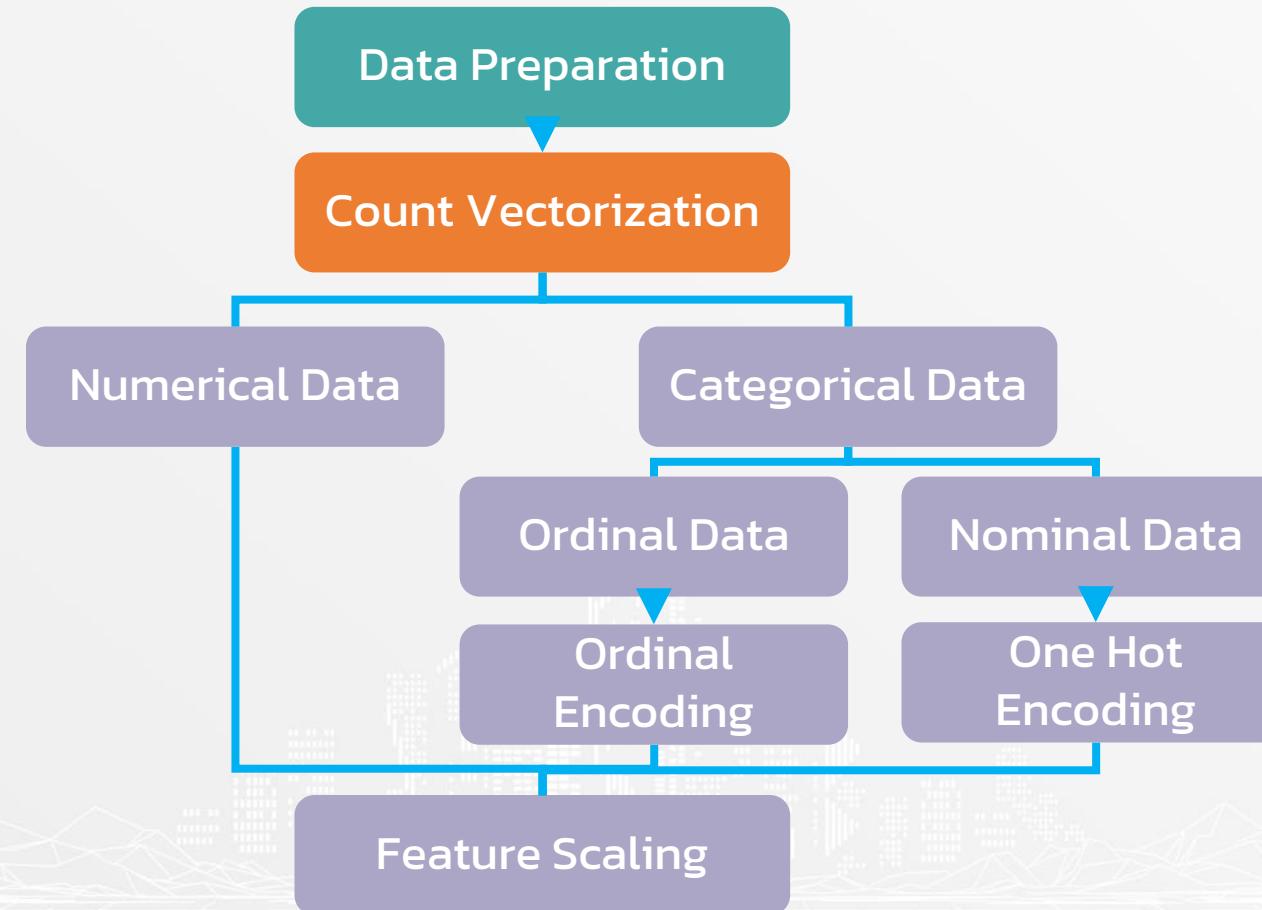
## Target

- selling\_price : ราคารถยนต์ที่ขายได้

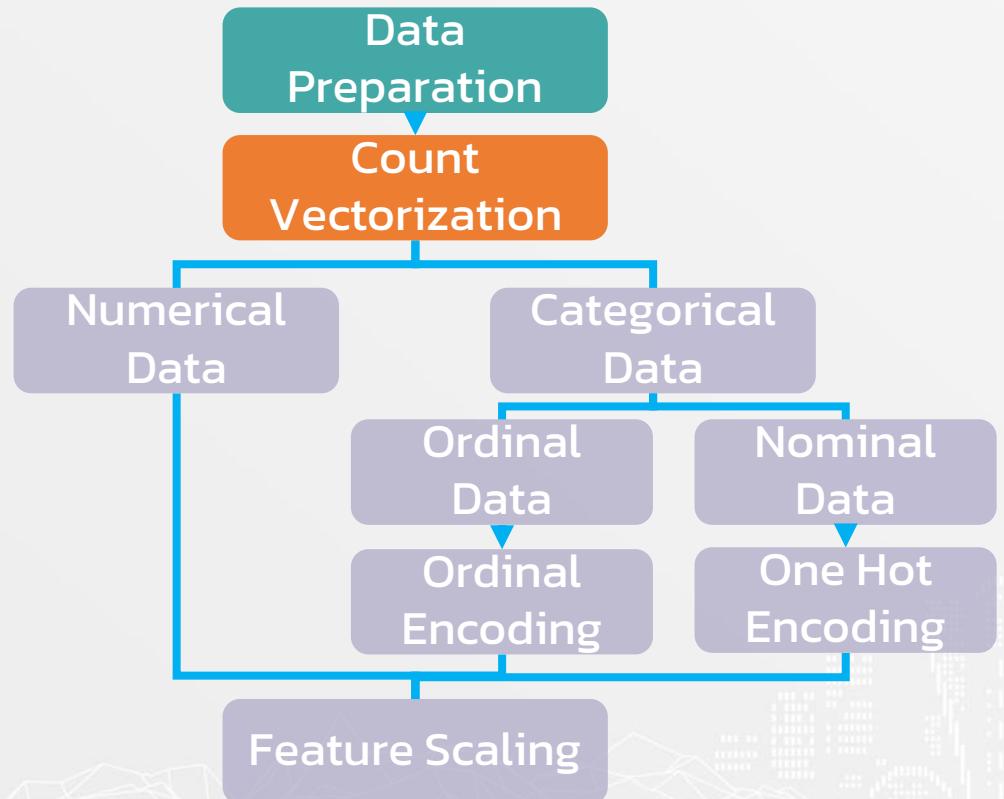
# What we learn from this project?



# 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



# DL101 : Linear Regression



# THANK YOU !

*We hope you enjoy our course*

