



สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น)
Technology Promotion Association (Thailand-Japan)

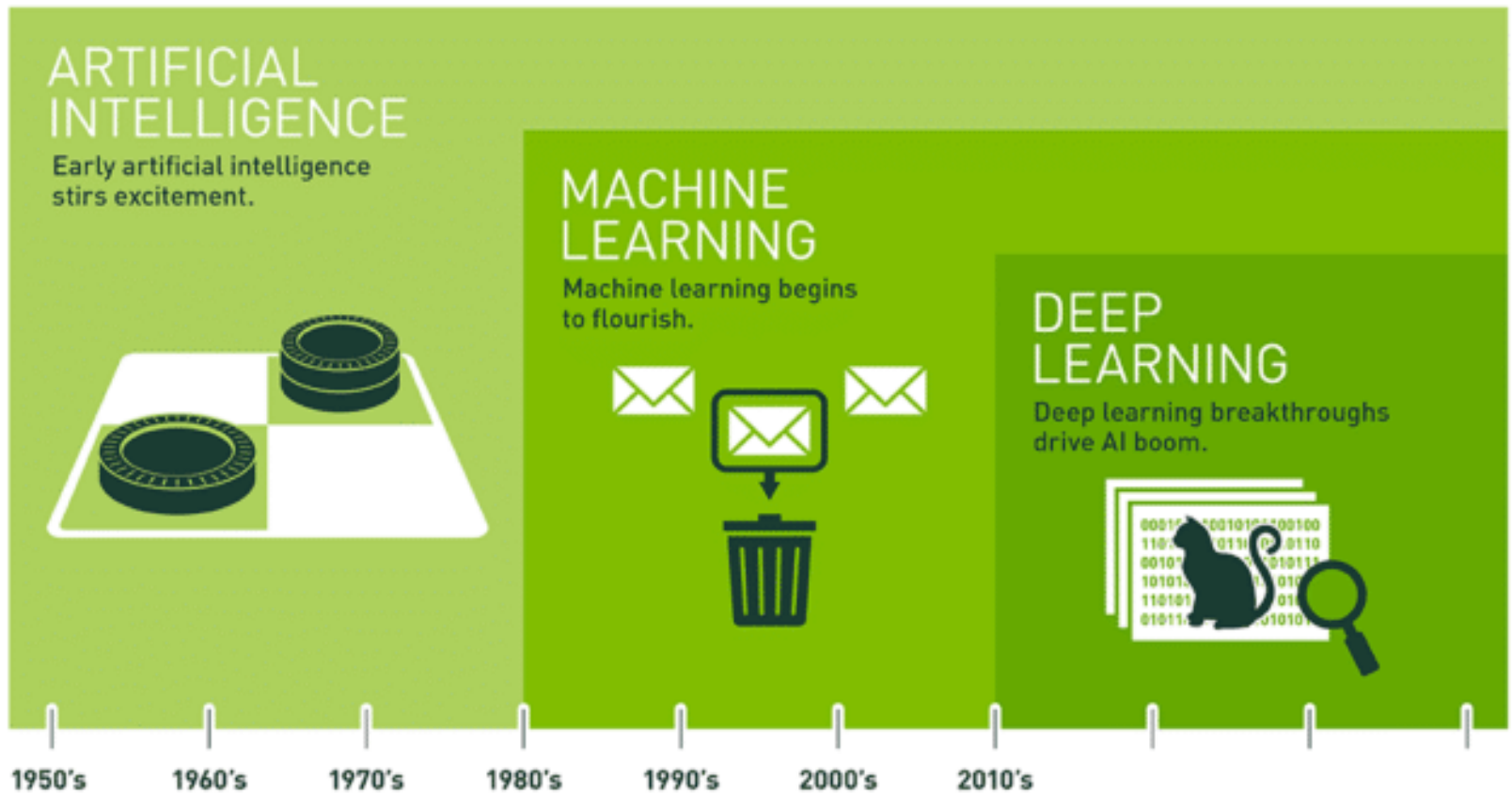


สถาบันเทคโนโลยีไทย-ญี่ปุ่น
Thai-Nichi Institute of Technology
泰日工業大学

AI (Artificial Intelligence)

AI

2



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

- AI ย่อมาจาก **Artificial Intelligence** โดยสามารถแปลเป็นภาษาได้ว่า ปัญญาประดิษฐ์ เทคโนโลยีที่สามารถบริหารจัดการข้อมูลมหาศาล ทำการประเมินผล วิเคราะห์ข้อมูลต่างๆ จนกลายเป็นเครื่องมือสำหรับการใช้งานที่มีประสิทธิภาพมากที่สุด ที่พร้อมพัฒนาตนเองอยู่เสมอ
- เริ่มใช้งานระบบปัญญาประดิษฐ์ ได้มีการสร้างขึ้นครั้งแรกเมื่อช่วงปี ค.ศ. 1956 โดยนักวิทยาศาสตร์คอมพิวเตอร์จากสหรัฐอเมริกา มีชื่อว่า **John McCarthy** ที่ได้พัฒนาจนสามารถสร้างเครื่องจักรที่มีความชาญฉลาดและแนวคิดแบบมนุษย์ได้เป็นเครื่องแรกนั่นเอง

Machine Learning

4

- **Machine Learning** คือ การทำให้ระบบคอมพิวเตอร์สามารถประมวลผล คาดการณ์ ตัดสินปัญหาต่างๆ ด้วยตนเองผ่านการเรียนรู้ชุดข้อมูลที่ป้อนเข้าไป
- ตัวอย่างเกี่ยวกับอัลกิริทึมของ **Machine Learning** ที่พบเห็นได้ในชีวิตประจำวัน ก็คือแพลตฟอร์มสตรีมเพลงต่างๆ ที่มักจะแนะนำเพลงใหม่ๆ หรือศิลปินให้กับผู้ใช้ โดยอัลกอรืทึมของ **Machine Learning** จะทำการวิเคราะห์จากข้อมูลเพลงหรือศิลปินที่ผู้ใช้ชื่นชอบ เทียบกับผู้ฟังอื่นที่มีแนวโน้มการฟังเพลงแบบเดียวกัน หลายธุรกิจที่มีระบบแนะนำมักจะใช้เทคนิคนี้เช่นเดียวกัน

Machine Learning

5

- Machine Learning มักมาควบคู่กับระบบและการ Coding ที่สลับซับซ้อน
- “Machine Learning” ความหมายของมันก็คือบางอย่างที่แสดงผลการทำงานเดียวกับข้อมูลที่ถูกป้อน และให้ผลลัพธ์ที่ดีขึ้นเรื่อย ๆ เมื่อเวลาผ่านไป เหมือนกับคุณมีไฟฉายที่มักเปิดเองอัตโนมัติเมื่อคุณพูดประมาณว่า “ที่นี่มืดจังเลยนะ” ต่อให้คุณไม่ได้พูดประโยคเดียวกันนี้ ไฟฉายนั้นก็ยังคงเปิดเอง

Deep learning

6

- **Deep Learning** คือ การจำลองระบบการประมวลผลของเซลล์ประสาทและสมองของมนุษย์ กล่าวได้ว่าเป็นการเลียนแบบการทำงานของระบบสมองมนุษย์
- การทำงานของ **Deep Learning** จะใช้โครงสร้างที่เหมือนกับเซลล์ประสาทในสมองของมนุษย์มาประเมินผลเพื่อให้ผลลัพธ์ที่ต้องการ สามารถที่จะประมวลผลได้อย่างแม่นยำ รวดเร็วและตรงพลิ่งเป็นอย่างมาก

ChatGPT

7

- ChatGPT is a natural language processing tool driven by AI technology that allows you to have human-like conversations and much more with the chatbot. The language model can answer questions and assist you with tasks, such as composing emails, essays, and code.

ChatGPT

8

⚡ GPT-3.5

⚡ GPT-4 🔒

ChatGPT



Examples

"Explain quantum computing in simple terms" →

"Got any creative ideas for a 10 year old's birthday?" →

"How do I make an HTTP request in Javascript?" →



Capabilities

Remembers what user said earlier in the conversation

Allows user to provide follow-up corrections

Trained to decline inappropriate requests



Limitations

May occasionally generate incorrect information

May occasionally produce harmful instructions or biased content

Limited knowledge of world and events after 2021

what is chatgpt



Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. [ChatGPT August 3 Version](#)

<https://chat.openai.com/>

Machine Learning

9

- Machine Learning แบ่งออกได้ 3 ประเภท
 - Supervised Learning
 - Unsupervised Learning
 - Reinforcement Learning

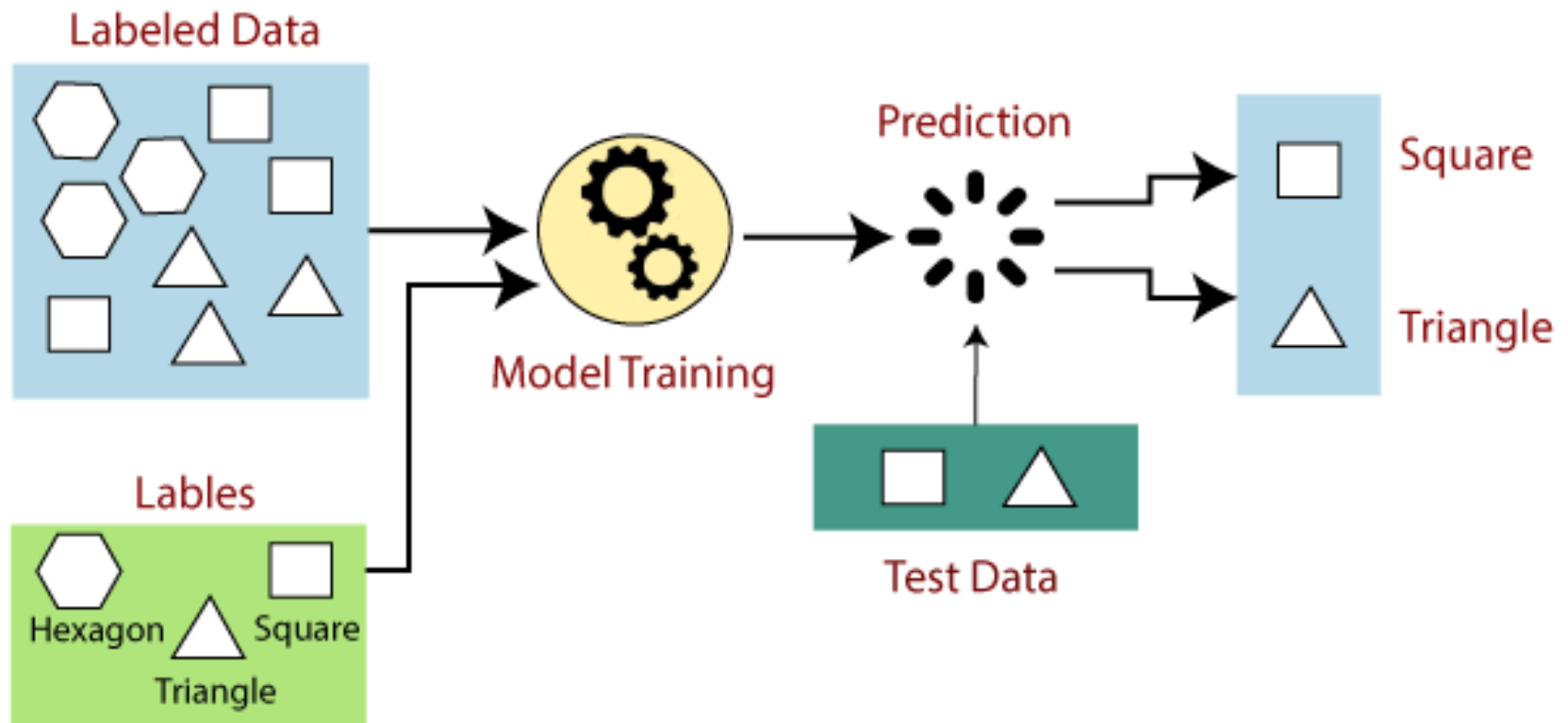
Supervised Learning

10

- **Supervised Learning** คือ การสร้างโมเดลเพื่อแปลงข้อมูล **input** เป็น **target** บางอย่าง ตัวอย่างง่ายที่สุดคือ **classification** กับ **regression**
- **Classification** คือมี **target** เป็นชนิดของข้อมูล เช่น เรียนรู้ว่า **email** เป็น **spam** หรือไม่เป็น **spam**
- **Regression** คือมี **target** เป็นตัวเลข เช่น เรียนรู้การประมาณราคาที่ดินจากปัจจัยแวดล้อม
- การจะสร้างโมเดลประเภทนี้ขึ้นมา ต้องมีชุดข้อมูลที่มีทั้ง **input** และ **target** ซึ่งจัดหามาโดยมนุษย์ เช่น การสร้าง **spam filter** ต้องรวบรวมข้อมูล **email** จำนวนมากและให้คนมาดูว่าอันไหนเป็น **spam** บ้าง แล้วนำมาสร้างโมเดล **spam filter** จากข้อมูลเหล่านี้

Supervised Learning

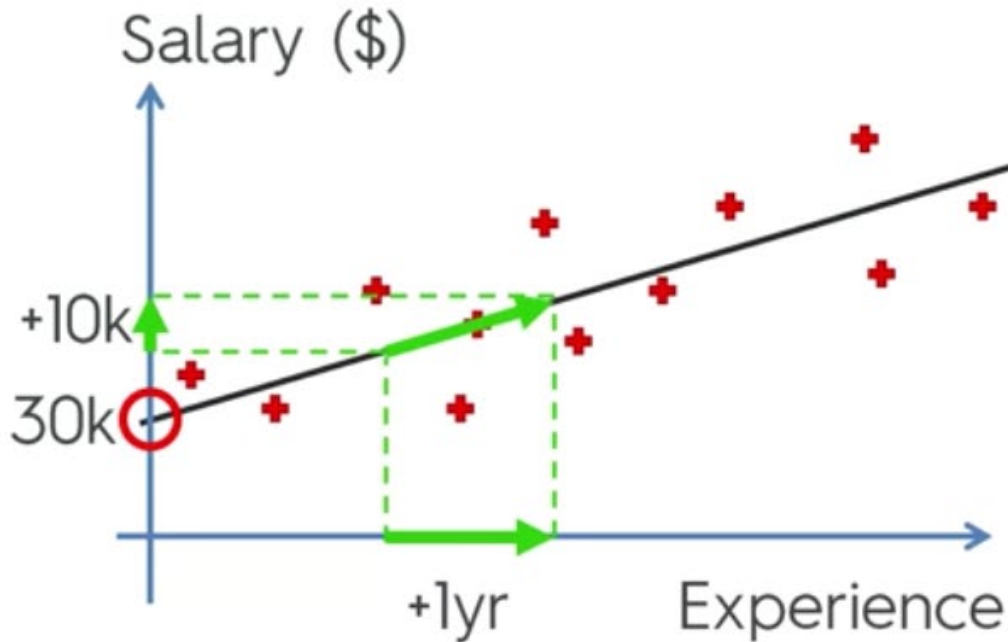
11



Supervised Learning

12

Simple Linear Regression:



$$y = b_0 + b_1 * x$$



$$\text{Salary} = b_0 + b_1 * \text{Experience}$$

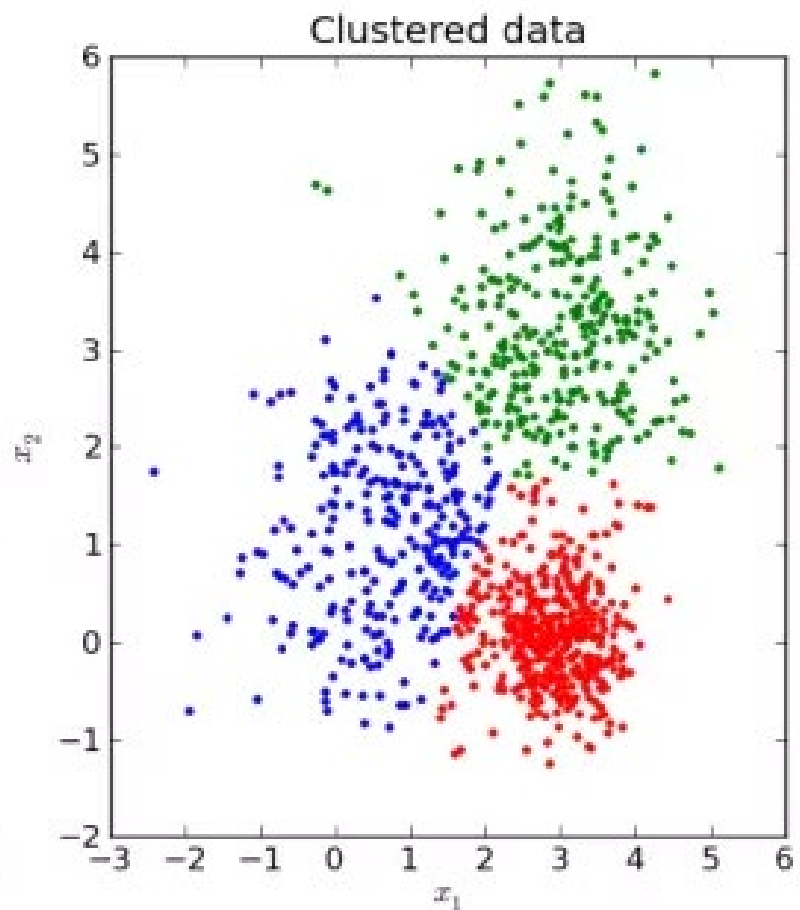
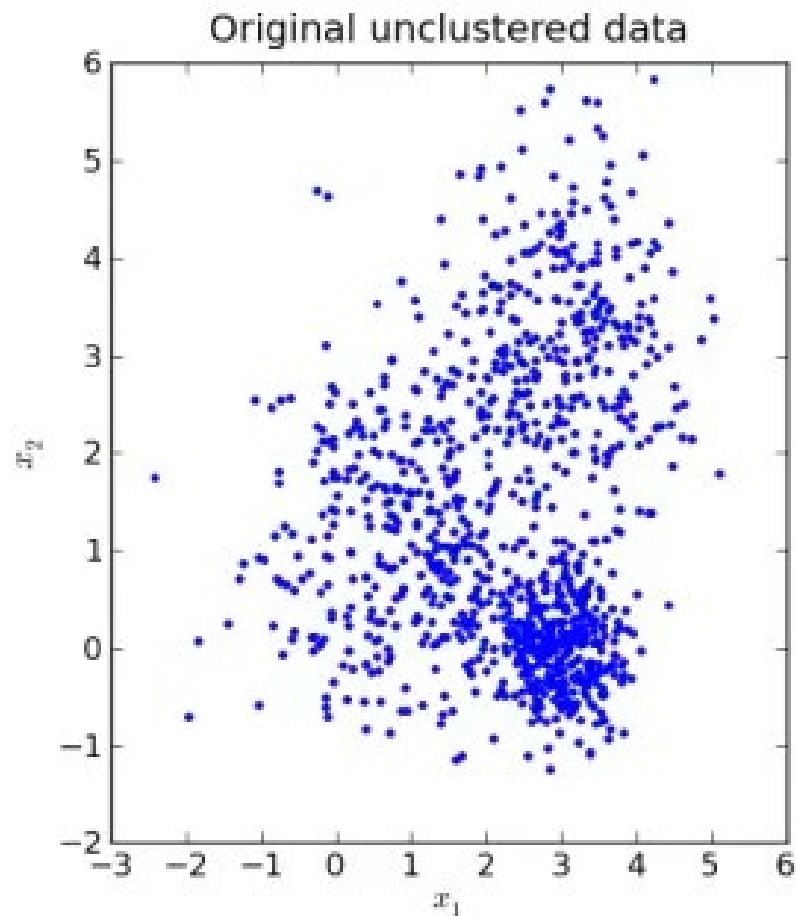
Unsupervised Learning

13

- **Unsupervised Learning** คือ การสร้างโมเดลโดยใช้ข้อมูล **input** เพียงอย่างเดียว ไม่มี **target**
- **Clustering** การจัดกลุ่มข้อมูลตามคุณลักษณะ เช่น การจัดกลุ่มลูกค้าตามพฤติกรรมการซื้อของ
- การสร้างโมเดลประเภทนี้ขึ้นมา ใช้เพียงข้อมูล **input** อย่างเดียว ไม่ต้องจัดหา **target** เช่น โมเดลการจัดกลุ่มลูกค้า เราไม่ต้องรู้มาก่อนว่าจะมีกลุ่มอะไรบ้าง

Unsupervised Learning

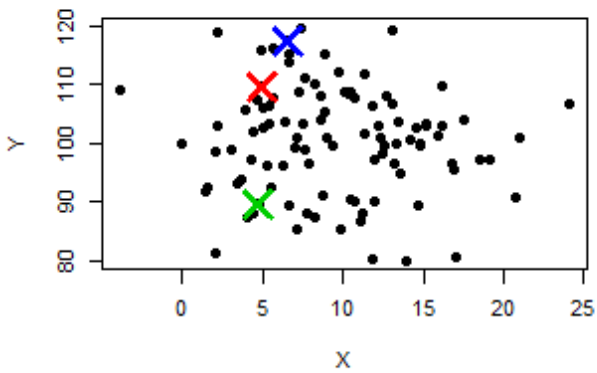
14



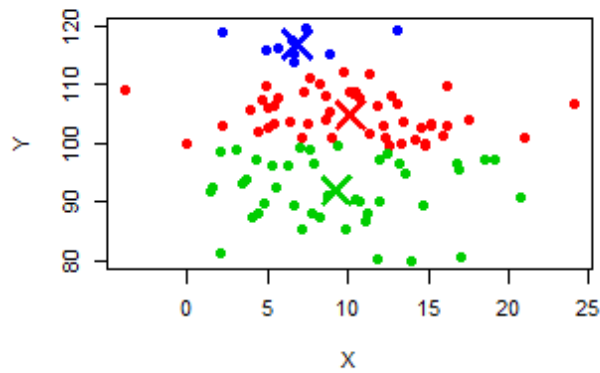
Unsupervised Learning

15

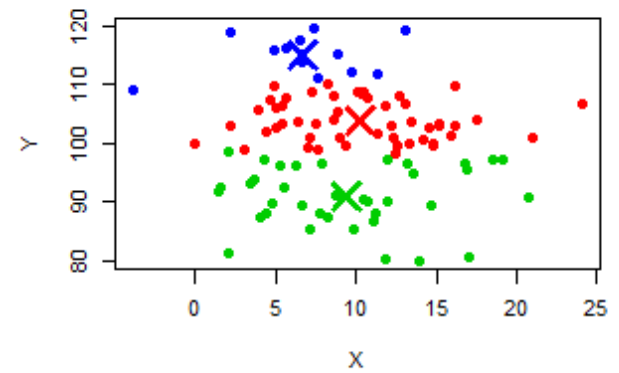
Iteration 1



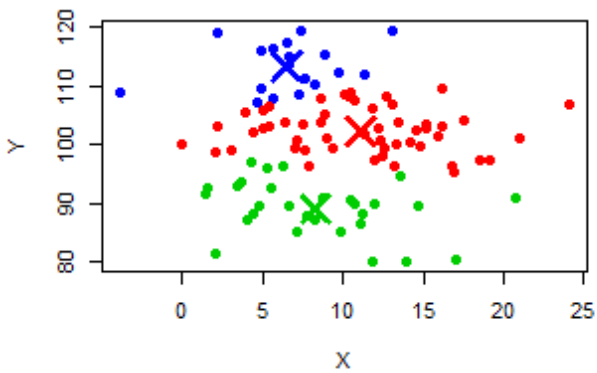
Iteration 2



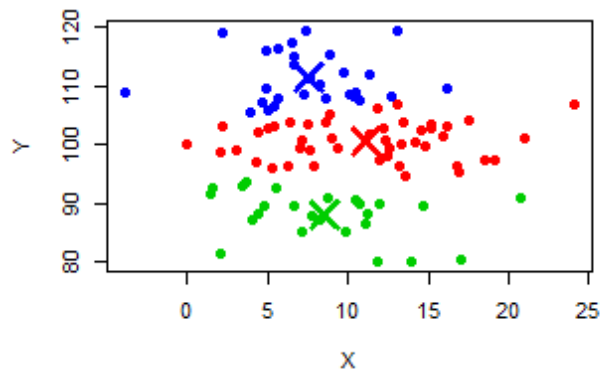
Iteration 3



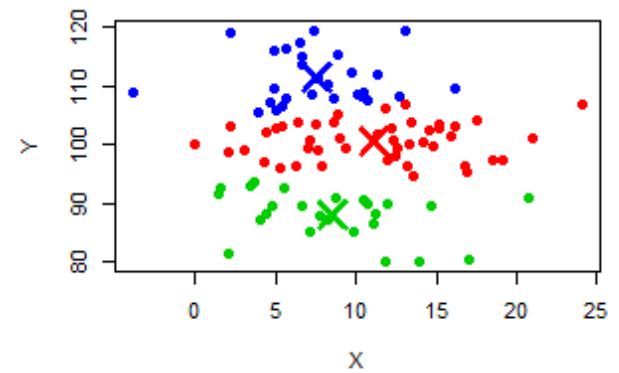
Iteration 6



Iteration 9



Converged!



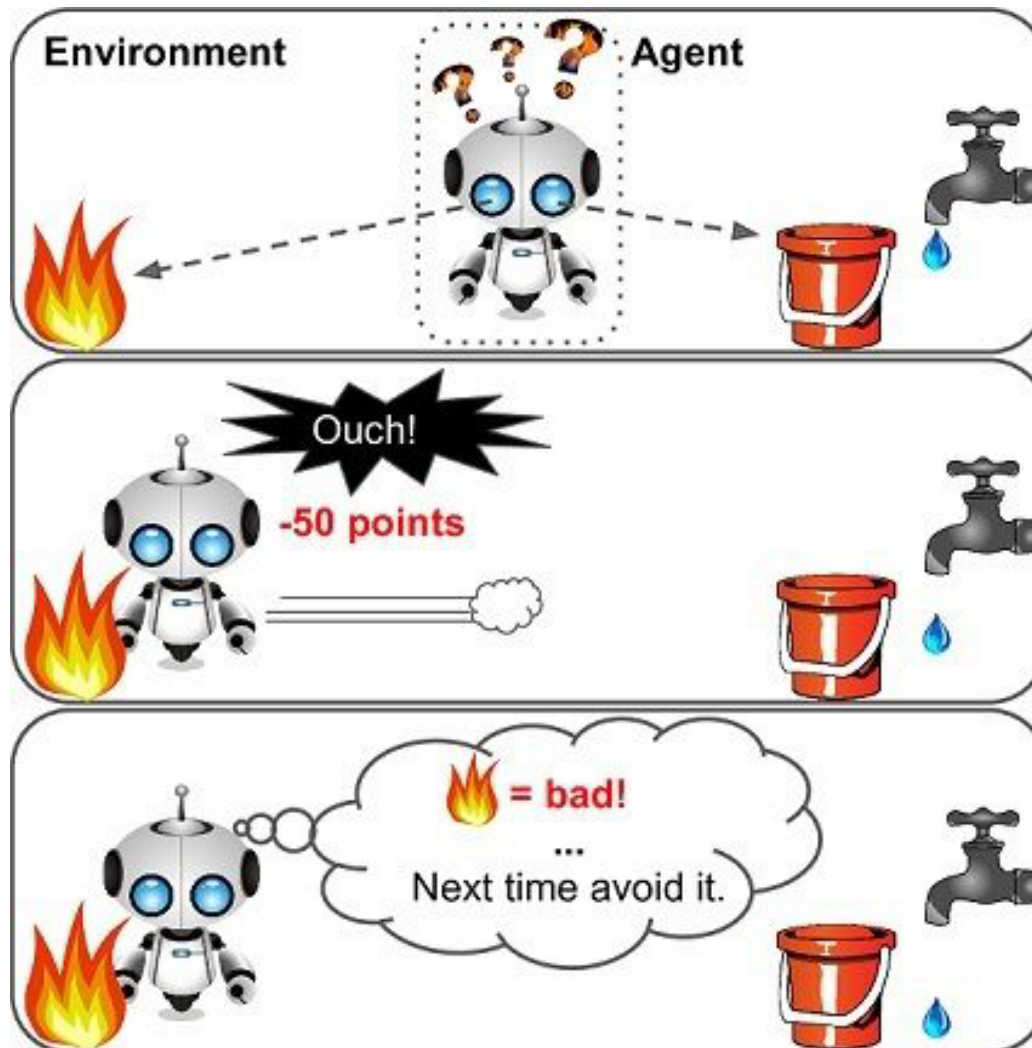
Reinforcement Learning

16

- **Reinforcement Learning** คือ การเรียนรู้แบบเสริมกำลัง ที่มีการเรียนรู้สิ่งต่าง ๆ จาก **Agent** (ผู้กระทำ **Action**) ภายใต้การเลือกกระทำสิ่งต่าง ๆ ให้ได้ผลลัพธ์ที่มากที่สุด ผ่านการลองผิดลองถูก ภายใต้สถานการณ์หรือระบบจำลอง ที่พัฒนาระบบการตัดสินใจให้ดีขึ้นเรื่อย ๆ เช่น การเล่นเกมโกะให้ชนะผู้เล่นระดับโลก ไปจนถึงการพิจารณาเลือกซื้อสินทรัพย์ และการลงทุนในรูปแบบต่าง ๆ เป็นต้น

Reinforcement Learning

17



- 1 Observe
- 2 Select action using policy
- 3 Action!
- 4 Get reward or penalty
- 5 Update policy (learning step)
- 6 Iterate until an optimal policy is found

Reinforcement Learning

18



Reinforcement Learning

19

Summary [\[edit\]](#)

Game	Date	Black	White	Result	Moves
1	9 March 2016	Lee Sedol	AlphaGo	Lee Sedol resigned	186 Game 1 ↗
2	10 March 2016	AlphaGo	Lee Sedol	Lee Sedol resigned	211 Game 2 ↗
3	12 March 2016	Lee Sedol	AlphaGo	Lee Sedol resigned	176 Game 3 ↗
4	13 March 2016	AlphaGo	Lee Sedol	AlphaGo resigned	180 Game 4 ↗
5	15 March 2016	Lee Sedol ^{[note 1]}	AlphaGo	Lee Sedol resigned	280 Game 5 ↗
Result: AlphaGo 4 – 1 Lee Sedol					



สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น)
Technology Promotion Association (Thailand-Japan)



สถาบันเทคโนโลยีไทย-ญี่ปุ่น
Thai-Nichi Institute of Technology
泰日工業大学

Linear regression

Linear regression

21

- Linear Regression จัดอยู่ในกลุ่ม Supervised Learning
- ใช้ในการทำนายผลแบบ Regression ซึ่งเป็นการทำนายตัวเลขที่มีค่าไม่แน่นอน

Linear regression

22

- **Linear regression** คือ อัลกอริทึมหนึ่งในการวิเคราะห์สถิติ และเครื่องมือทางคณิตศาสตร์ที่ใช้ในการประมาณค่าของตัวแปรตาม (dependent variable) จากตัวแปรอิสระ (independent variable) ที่เป็นเส้นตรง
- ใช้รูปแบบสมการเชิงเส้น (**linear equation**) เพื่อพยากรณ์หรือสร้างรูปแบบทางคณิตศาสตร์เพื่อใช้ในการคาดการณ์ผลลัพธ์ในอนาคตของข้อมูล
- ในกรณีของ **linear regression** แบบเดี่ยว (**simple linear regression**) จะมีตัวแปรอิสระเพียงตัวเดียว

Linear regression

23

- เป้าหมายของ **linear regression** คือ การหาความสัมพันธ์ระหว่างตัวแปรอิสระกับตัวแปรตามแบบเชิงเส้น
- สร้างสมการเชิงเส้นที่ใช้ในการประมาณค่าตัวแปรตามโดยป้อนค่าของตัวแปรอิสระ

Linear regression

24

- สมการของ **Linear regression** สำหรับกรณีของ **simple linear regression** (ตัวแปรอิสระเพียงตัวเดียว) สามารถแสดงได้ดังนี้

$$y = \beta_0 + \beta_1 x$$

Linear regression

25

$$y = \beta_0 + \beta_1 x$$

- y คือตัวแปรตาม (**dependent variable**) ที่เราต้องการประมาณค่าหรือพยากรณ์
- x คือตัวแปรอิสระ (**independent variable**) ที่เราใช้ในการประมาณค่าหรือพยากรณ์
- β_0 และ β_1 เป็นค่าคงที่ (**coefficients**) ที่ใช้ปรับสมการให้เข้ากับข้อมูลที่มีอยู่

Linear regression

26

- กำหนดให้ $x = [1 \ 2 \ 3 \ 4 \ 6]$ และ $y = [5 \ 7 \ 9 \ 11 \ 15]$
เมื่อนำพิกัด x, y มาพล็อตกราฟ โดยใช้ ภาษา **python** ดังนี้

```
import numpy as np #numpy เป็นชื่อของ library ที่ใช้ในการคำนวณทางคณิตศาสตร์
import matplotlib.pyplot as plt #matplotlib เป็นชื่อของ library ที่นิยมใช้มากที่สุดในการพล็อตกราฟสองมิติจาก array

x = np.array([1, 2, 3, 4, 6])
y = np.array([5, 7, 9, 11, 15])

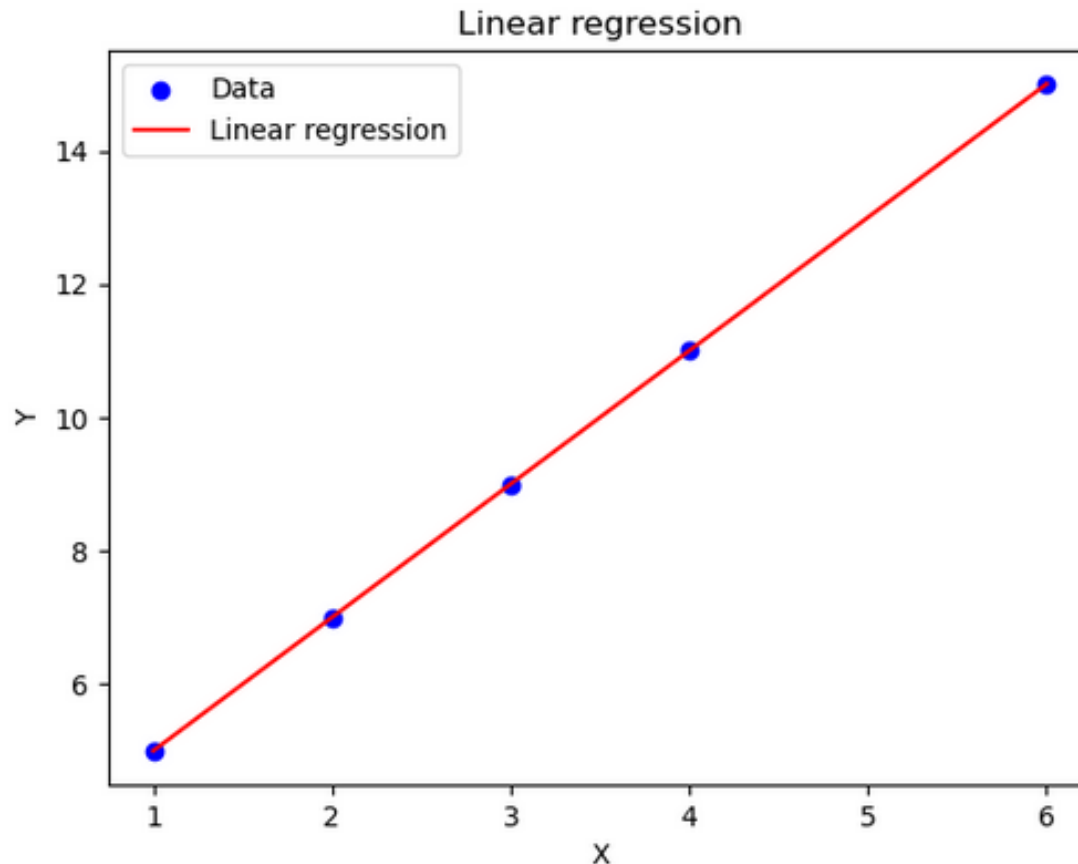
coefficients = np.polyfit(x, y, 1)
m = coefficients[0] # ความชัน (slope)
b = coefficients[1] # จุดตัดแกน y (intercept)
regression_line = m * x + b

plt.scatter(x, y, color='blue', label='Data')
plt.plot(x, regression_line, color='red', label='Linear regression')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Linear regression')
plt.legend()
plt.show()
```

Linear regression

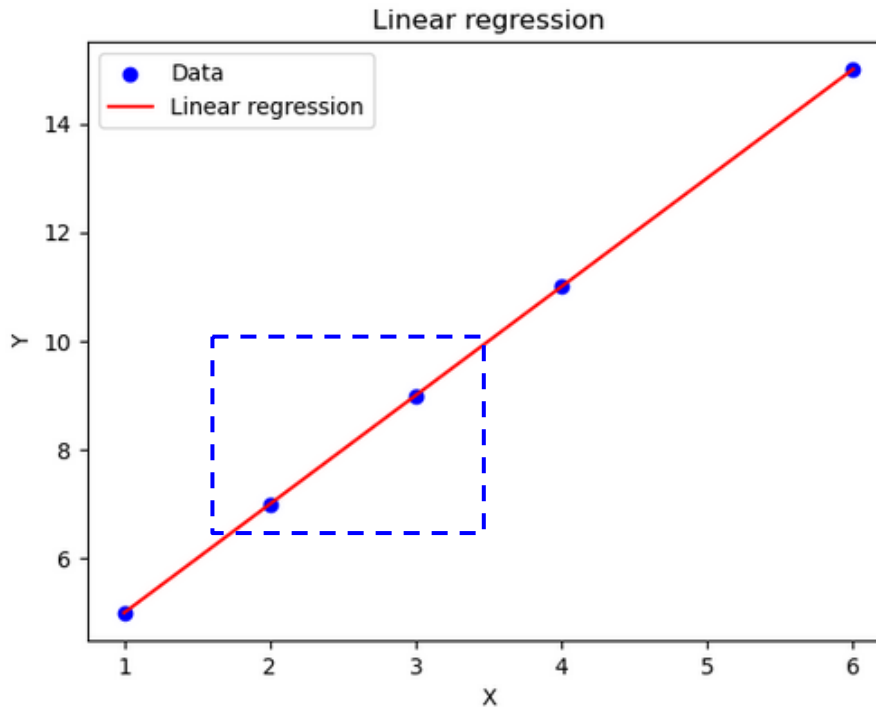
27

- กำหนดให้ $x = [1 \ 2 \ 3 \ 4 \ 6]$ และ $y = [5 \ 7 \ 9 \ 11 \ 15]$
เมื่อนำพิกัด x, y มาพล็อตกราฟ โดยใช้ ภาษา **python** ดังนี้



Linear regression

28



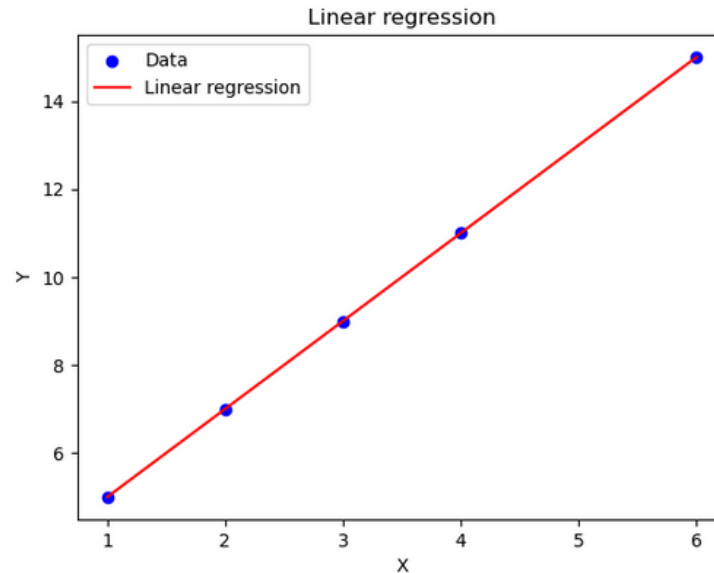
$$y = \beta_0 + \beta_1 x$$

คำนวณหาค่าความชันของสมการเส้นตรงได้ดังนี้

$$\beta_1 = \frac{y_2 - y_1}{x_2 - x_1} = \frac{9 - 7}{3 - 2} = 2$$

Linear regression

29



$$y = \beta_0 + \beta_1 x$$

จากนั้น นำค่า x และ y จากคู่อันดับไปคำนวณค่าจุดตัดแกน y (β_1)
อย่างเช่น พิจารณาที่คู่อันดับ (2,7) จะได้

$$7 = \beta_0 + 2 \cdot 2$$

$$\beta_0 = 3$$

$$y = 3 + 2x$$

Linear regression

30

$$y = 2x + 3$$

เมื่ออยากรู้ค่า y ที่ค่า x อื่น ๆ สามารถคำนวณได้ดังตัวอย่างนี้ ในกรณีที่
อยากรู้ค่า y ที่ค่า x เท่ากับ 7 นำค่า x เท่ากับ 7 แทนในสมการเส้นตรงจะได้
ค่า y ดังนี้

$$y = 2 \cdot 7 + 3 = 17$$

Linear Regression และความสัมพันธ์ของข้อมูล

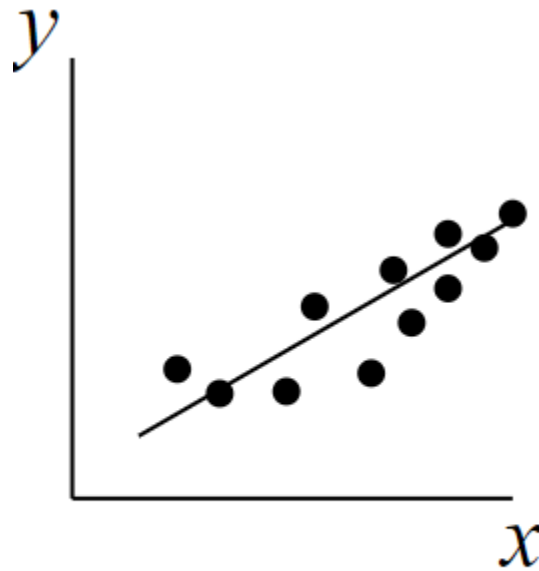
31

- **Linear Regression** จัดอยู่ในกลุ่ม **Supervised Learning** โดยข้อมูลตัวอย่างที่นำมา **Train Model** ชนิดนี้ จะต้องมีการลัพท์ที่เป็นผลลัพธ์
- การพิจารณาว่าควรเลือกใช้โมเดล **Linear Regression** ในการทำนายผลหรือไม่
- ต้องดูจากข้อมูลตัวอย่างที่มีอยู่ ซึ่งหากข้อมูลนั้นให้ผลลัพธ์ออกมาเป็นตัวเลข แต่อาจเปลี่ยนแปลงในทิศทางเดียวกัน (เมื่อตัวหนึ่งเพิ่มอีกตัวหนึ่งก็เพิ่มตามกัน) หรือเปลี่ยนแปลงในทิศทางตรงกันข้ามก็ได้ (เมื่อตัวหนึ่งเพิ่ม อีกตัวหนึ่งลดลง) ก็คาดการณ์ในเบื้องต้นได้ว่า น่าจะใช้ **Linear Regression** ได้

Linear Regression และความสัมพันธ์ของข้อมูล

32

- กรณีที่ข้อมูลมีการเปลี่ยนแปลงไปในทางเดียวกัน (เมื่อตัวหนึ่งเพิ่ม อีกตัวหนึ่งก็เพิ่มขึ้น) หรือเราเรียกว่ามีความสัมพันธ์ต่อกันในเชิงบวก (Positive Correlation)

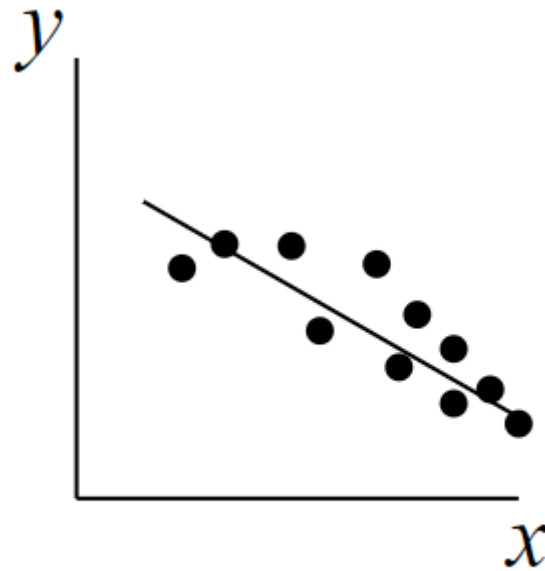


Positive

Linear Regression และความสัมพันธ์ของข้อมูล

33

- กรณีที่ข้อมูลมีการเปลี่ยนแปลงไปในทางตรงกันข้าม (เมื่อตัวหนึ่งเพิ่มอีกตัวหนึ่งลด) หรือเราเรียกว่ามีความสัมพันธ์ต่อกันในเชิงลบ (Negative Correlation)

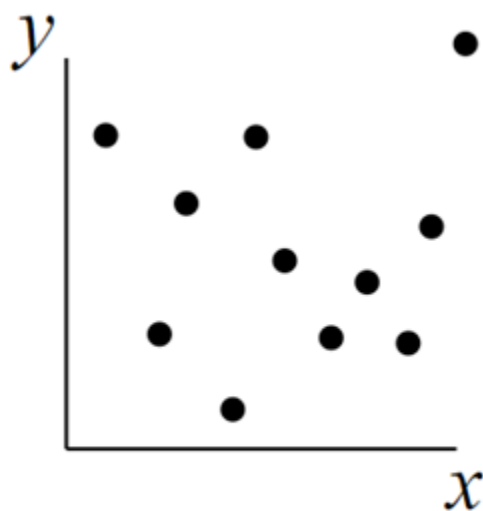


Negative

Linear Regression และความสัมพันธ์ของข้อมูล

34

- กรณีที่ข้อมูลไม่มีการเปลี่ยนแปลงตามกันอย่างชัดเจน (เมื่อตัวหนึ่งเพิ่มหรือลด อีกตัวหนึ่งอาจเปลี่ยนแปลงน้อยมาก หรือมีแนวโน้มที่ไม่แน่นอน) ซึ่งอาจเรียกว่า ไม่มีความสัมพันธ์ต่อกัน ไม่ควรทำนายด้วย Linear Regression



No correlation

Linear Regression และความสัมพันธ์ของข้อมูล

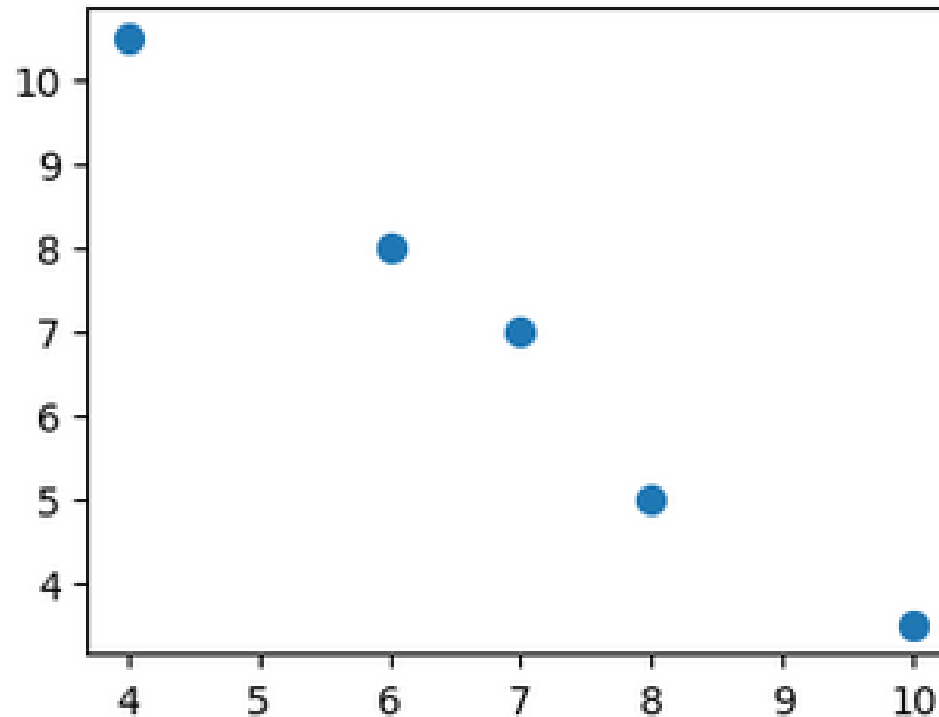
35

```
import matplotlib.pyplot as plt
import numpy as np
data = [[4, 10.5],[6,8],[7, 7],[8, 5],[10, 3.5]]
arr = np.array(data)
x = arr[:, 0]
y = arr[:, 1]
plt.figure(figsize=(4, 3))
plt.scatter(x, y, s = 50)
plt.show( )
```

```
import pandas as pd #pandas เป็นชื่อของ library ที่ถูกสร้างขึ้นมาเพื่อการจัดการและวิเคราะห์ข้อมูลตาราง
data = [[4, 10.5],[6,8],[7, 7],[8, 5],[10, 3.5]]
df = pd.DataFrame(data, columns=['x' , 'y'])
display(df.corr().round(3))
```

Linear Regression และความสัมพันธ์ของข้อมูล

36



	x	y
x	1.000	-0.991
y	-0.991	1.000

Linear Regression และความสัมพันธ์ของข้อมูล

37

```
import numpy as np #numpy เป็นชื่อของ library ที่ใช้ในการคำนวณทางคณิตศาสตร์
import matplotlib.pyplot as plt #matplotlib เป็นชื่อของ library ที่นิยมใช้มากที่สุดในการพลอตกราฟสองมิติจาก array

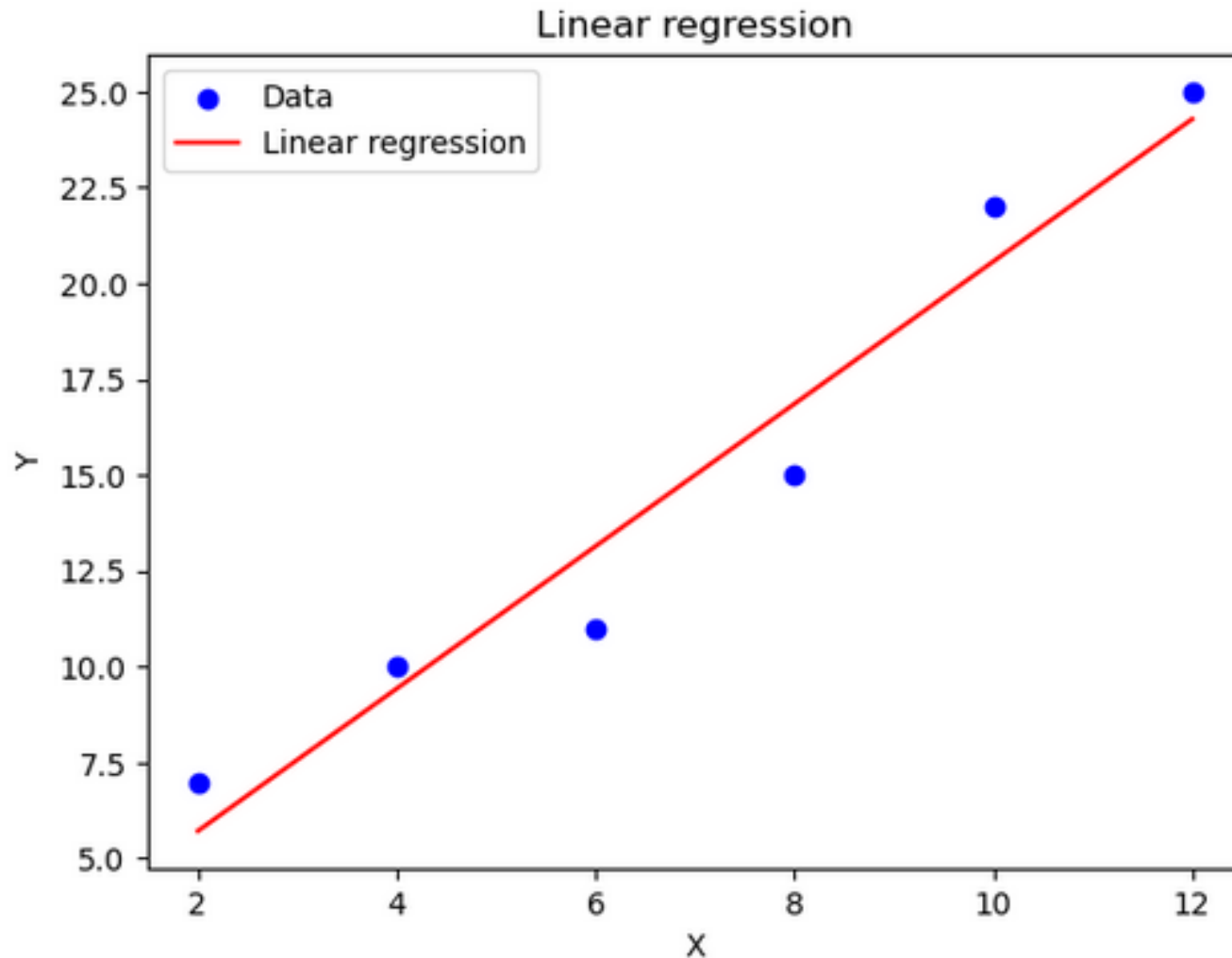
x = np.array([2, 4, 6, 8, 10, 12])
y = np.array([7, 10, 11, 15, 22, 25])

coefficients = np.polyfit(x, y, 1)
m = coefficients[0] # ความชัน (slope)
b = coefficients[1] # จุดตัดแกน y (intercept)
regression_line = m * x + b

plt.scatter(x, y, color='blue', label='Data')
plt.plot(x, regression_line, color='red', label='Linear regression')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Linear regression')
plt.legend()
plt.show()
```

Linear Regression และความสัมพันธ์ของข้อมูล

38



Simple Linear Regression

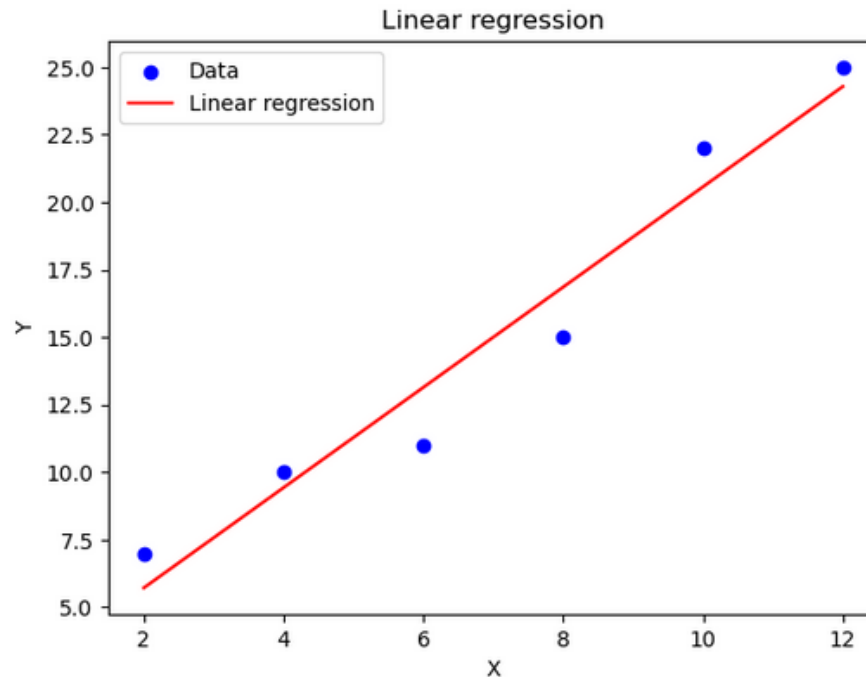
39

- Simple Linear Regression ใช้ในการทำนายผลสำหรับกรณีข้อมูลตัวอย่างประกอบด้วยคอลัมน์ที่เป็น **Feature** หรือตัวแปรอิสระ (**x: Independence Variable**) เพียงอันเดียว และคอลัมน์ที่เป็น **Target** หรือผลลัพธ์หรือตัวแปรตาม (**y: Dependence Variable**) มีค่าเป็นตัวเลข
- ทั้งตัวแปรอิสระและตัวแปรตาม มีความสัมพันธ์กับแบบ **Positive** หรือ **Negative** อย่างไรอย่างหนึ่ง

Simple Linear Regression

40

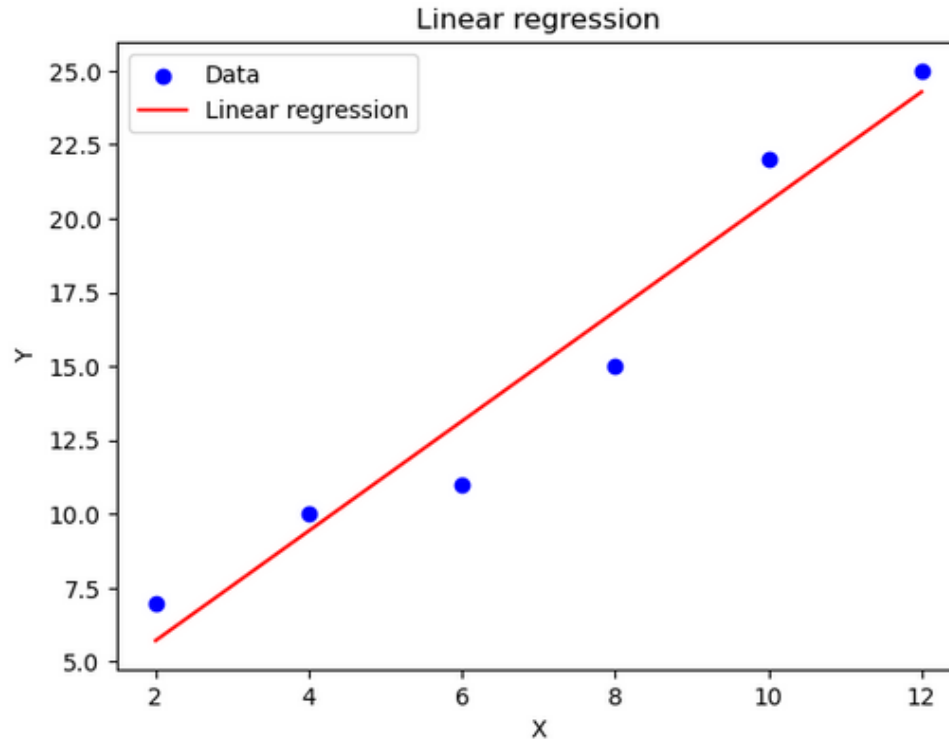
x	2	4	6	8	10	12
y	7	10	11	15	22	25



- จะเห็นว่าจุดต่าง ๆ ไม่ได้อยู่บนเส้นตรง Regression Line ซึ่งเส้นตรงดังกล่าวเป็นเพียงเส้นที่ใช้ในการประมาณค่า

Simple Linear Regression

41



- เราไม่สามารถนำข้อมูลตัวอย่างที่มีอยู่มาใช้ในการคำนวณหา β_0 และ β_1 ด้วยวิธีการเดียวกับสมการเส้นตรง $y = mx + c$ ได้ แต่ต้องใช้สูตรการคำนวณที่แตกต่างออกไป

Linear Regression และความสัมพันธ์ของข้อมูล

42

$$y = \beta_0 + \beta_1 x$$

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Linear Regression และความสัมพันธ์ของข้อมูล

43

- x_i = คือรายการข้อมูลแต่ละค่าในคอลัมน์ที่เป็น Feature หรือตัวแปรอิสระ
- \bar{x} = คือค่าเฉลี่ยของข้อมูลในคอลัมน์ที่เป็น Feature
- y_i = คือรายการข้อมูลแต่ละค่าในคอลัมน์ที่เป็น Target หรือตัวแปรตาม หรือผลลัพธ์นั่นเอง
- \bar{y} = คือค่าเฉลี่ยของข้อมูลในคอลัมน์ที่เป็น Target

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Linear Regression และความสัมพันธ์ของข้อมูล

44

x	2	4	6	8	10	12
y	7	10	11	15	22	25

	A	B	C	D	E	
ลำดับ	x_i	y_i	$x_i - \bar{x}$	$(x_i - \bar{x})^2$	$y_i - \bar{y}$	$C * E$
1	2	7	$2 - 7 = -5$	25	$7 - 15 = -8$	40
2	4	10	-3	9	-5	15
3	6	11	-1	1	-4	4
4	8	15	1	1	0	0
5	10	22	3	9	7	21
6	12	25	5	25	10	50
ผลรวม	42	90		70		130
ค่าเฉลี่ย	$\bar{x} = 7$	$\bar{y} = 15$				

Linear Regression และความสัมพันธ์ของข้อมูล

45

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_1 = 130/70 = 1.85$$

$$\beta_0 = 15 - (1.85 * 7) = 2.05$$

- สมการที่ใช้สำหรับการทำนายผลของข้อมูลชุดนี้คือ $y = 2.05 + 1.85x$
- หากเราต้องการการทำนายผล เมื่อกำหนด x เป็นค่าอื่น ๆ ก็นำมาแทนลงในสมการ $y = 2.05 + 1.85x$

Linear Regression และความสัมพันธ์ของข้อมูล

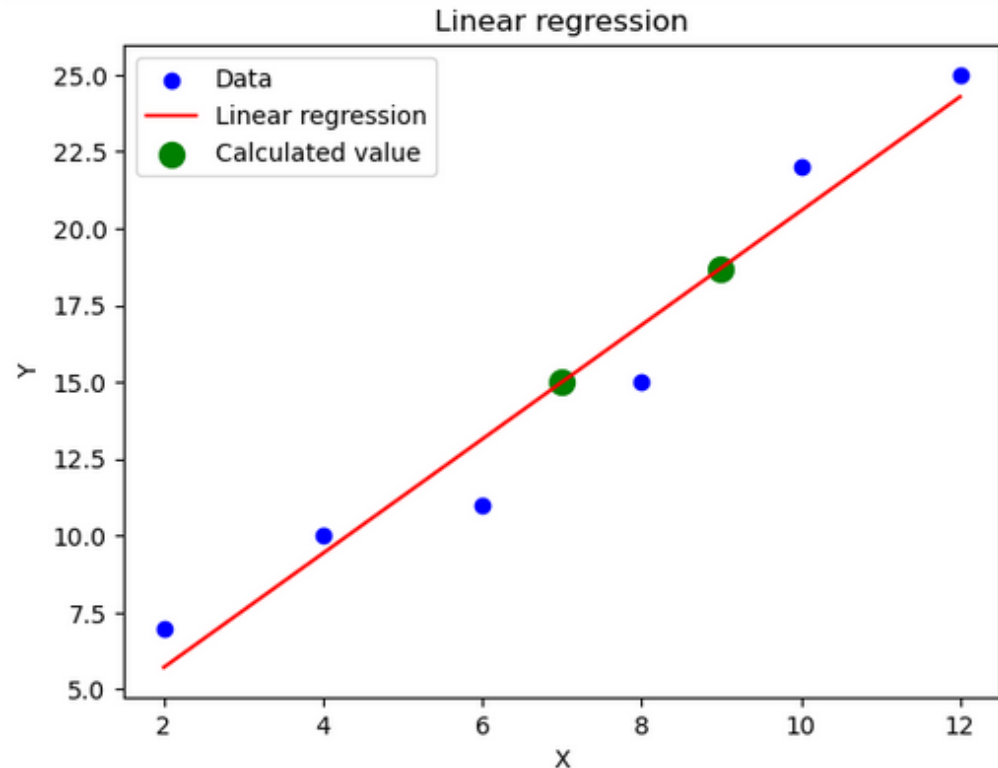
46

- หากเราต้องการทำนายผล เมื่อกำหนด x เป็นค่าอื่น ๆ ก็นำมาแทนลงในสมการ

$$y = 2.05 + 1.85x$$

$$y_7 = 2.05 + (1.85 * 7) = 15$$

$$y_9 = 2.05 + (1.85 * 9) = 18.7$$



การทำนายผลด้วย **Scikit-Learn**

47

- **Scikit-learn** เป็นโมดูลหนึ่งของภาษาโปรแกรมมิ่ง **Python** เป็นแพ็คเกจที่รวบรวม **Library** ด้าน **Machine Learning** เอาไว้ และถูกออกแบบมาให้ทำงานร่วมกับ **Library** ของภาษาโปรแกรมมิ่ง **Python** อย่าง **Numpy** และ **Scipy** ได้ดี

การทำนายผลด้วย **Scikit-Learn**

J.P.Morgan

Scikit-learn is an indispensable part of the Python machine learning toolkit at JPMorgan. It is very widely used across all parts of the bank for classification, predictive analytics, and very many other machine learning tasks. Its straightforward API, its breadth of algorithms, and the quality of its documentation combine to make scikit-learn simultaneously very approachable and very powerful.

Stephen Simmons, VP, Athena Research, JPMorgan

J.P.Morgan

Spotify

Scikit-learn provides a toolbox with solid implementations of a bunch of state-of-the-art models and makes it easy to plug them into existing applications. We've been using it quite a lot for music recommendations at Spotify and I think it's the most well-designed ML package I've seen so far.

Erik Bernhardsson, Engineering Manager Music Discovery & Machine Learning, Spotify



การทำนายผลด้วย **Scikit-Learn**

49

- การใช้งาน **Scikit-Learn** เพื่อทำนายผลของโมเดลแบบ **Simple Linear Regression** มีแนวทางดังนี้
- คลาสสำหรับโมเดลนี้คือ **LinearRegression** โดยต้องอิมพอร์ตจากโมดูล **sklearn.linear_model** แล้วสร้างอินสแตนซ์ของมัน

```
from sklearn.linear_model import LinearRegression
```

```
...
```

```
model = LinearRegression () #สร้างอินสแตนซ์ของคลาส LinearRegression
```

Linear Regression และความสัมพันธ์ของข้อมูล

50

- ในเบื้องต้น เราอาจสมมติข้อมูลขึ้นมาเองในแบบลิสต์หรืออาร์เรย์ 2 มิติ โดยข้อกำหนดที่สำคัญคือ
 - ▣ ค่าในแต่ละแถวของคอลัมน์ที่เป็น **Feature (x)** ต้องอยู่ในรูปแบบอาร์เรย์ 2 มิติ แม้แต่ละแถวจะมีค่าเดียวกันก็ตาม
 - ▣ ค่าในแต่ละแถวของคอลัมน์ที่เป็น **Target (y)** อาจอยู่ในรูปแบบอาร์เรย์มิติเดียว หรือ 2 มิติก็ได้
 - ▣ ข้อมูลต้องมีความสัมพันธ์แบบ **Positive** หรือ **Negative Correlation** ใดๆอย่างหนึ่ง

Linear Regression และความสัมพันธ์ของข้อมูล

51

- ตัวอย่าง การทำนายผลด้วย **Scikit-Learn** ของข้อมูลตัวอย่างชุดเดิมที่ใช้ประกอบการอธิบายไปก่อนหน้านี้

```
from sklearn.linear_model import LinearRegression
import numpy as np
x = [2, 4, 6, 8, 10, 12]
x = np.array(x).reshape (-1, 1)
y = [7, 10, 11, 15, 22, 25]
model = LinearRegression()
model.fit(x, y)

y_7 = model.predict([[7]])
print('x = 7, y =', y_7[0])
y_9 = model.predict([[9]])
print('x = 9, y =', y_9[0])
print('intercept =', model.intercept_)
print('coefficient =', model.coef_[0])
```

x	2	4	6	8	10	12
y	7	10	11	15	22	25

```
x = 7, y = 15.0
x = 9, y = 18.714285714285715
intercept = 2.0
coefficient = 1.8571428571428572
```

$$y = 2.05 + 1.85x$$

$$y_7 = 2.05 + (1.85 * 7) = 15$$

$$y_9 = 2.05 + (1.85 * 9) = 18.7$$

Linear Regression และความสัมพันธ์ของข้อมูล

52

- ตัวอย่าง สมมติว่าเรามีข้อมูลเกี่ยวกับราคาที่ดิน ในหน่วยล้านบาท ซึ่งสัมพันธ์กับจำนวนปีที่ผ่านมาดังตาราง หากเราต้องการทำนายผลราคาที่ดินในปีที่ 17 และ 20 สามารถทำได้ดังนี้

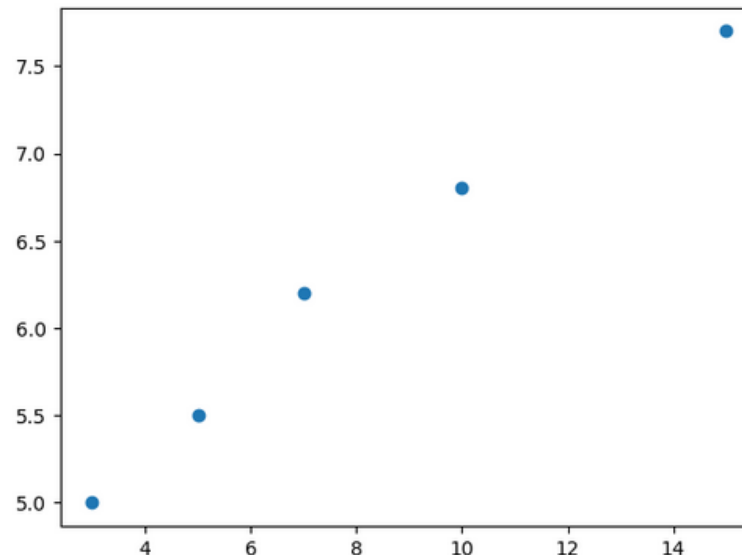
years	3	5	7	10	15
land price (million baht)	5	5.5	6.2	6.8	7.7

Linear Regression และความสัมพันธ์ของข้อมูล

53

- จากการพิจารณาข้อมูลในตารางด้วยสายตา เราอาจจะคาดคะเนได้ว่า ข้อมูลทั้ง 2 มีความสัมพันธ์กันแบบ Positive แต่เพื่อให้แน่ใจ เราลอง plot กราฟดู ได้รูปกราฟดังนี้

years	3	5	7	10	15
land price (million baht)	5	5.5	6.2	6.8	7.7



Linear Regression และความสัมพันธ์ของข้อมูล

54

```
import matplotlib.pyplot as plt
x = [3, 5, 7, 10, 15]
y = [5, 5.5, 6.2, 6.8, 7.7]
plt.scatter(x, y)
plt.show( )

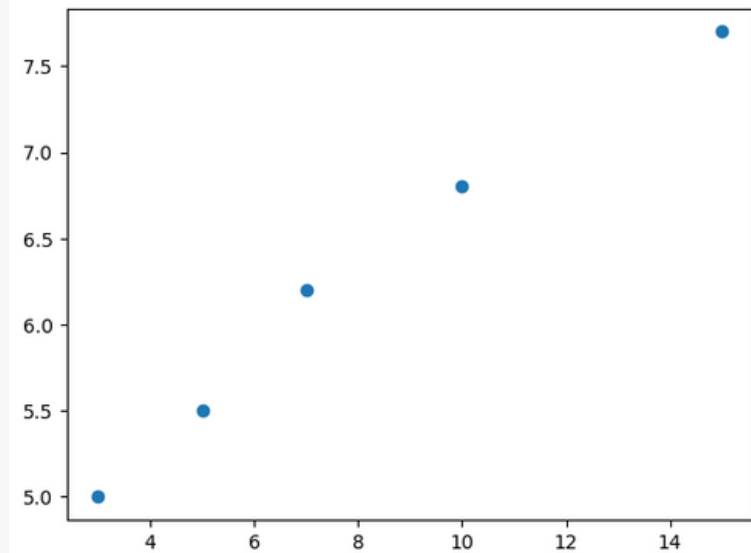
x = np.array(x).reshape(-1 , 1)
model = LinearRegression()
model.fit (x , y)
x_predict = [[17], [20]] #ทำนายราคาที่ดินปีที่ 17 และ 20
y_predict = model.predict(x_predict)

ic = '{:.2f}'.format(model.intercept_)
ce = '{:.2f}'.format(model.coef_[0])

print(f'y = {ic} + {ce}x')

price_17 = '{:.2f}'.format(y_predict[0])
price_20 = '{:.2f}'.format(y_predict[1])

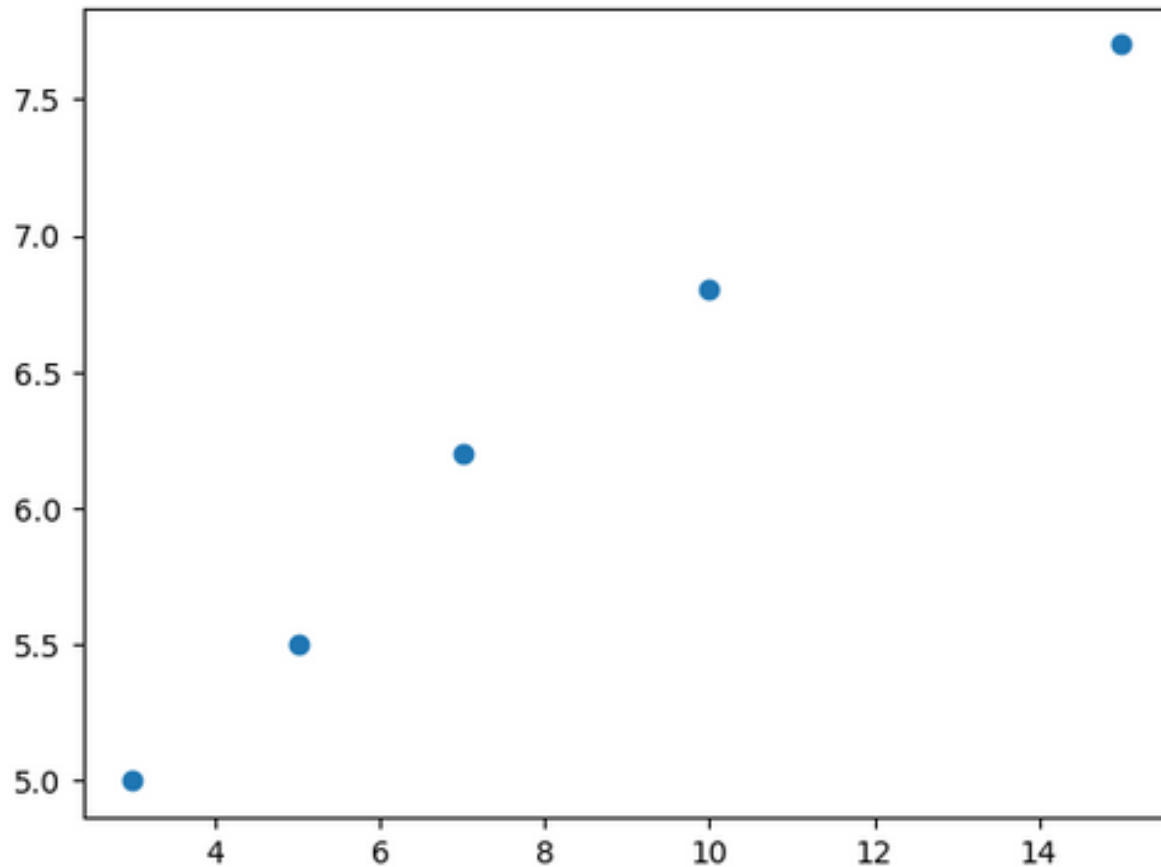
print (f'year 17, land price is {price_17} million baht')
print (f'year 20, land price is {price_20} million baht')
```



$y = 4.44 + 0.23x$
year 17, land price is 8.27 million baht
year 20, land price is 8.94 million baht

Linear Regression และความสัมพันธ์ของข้อมูล

55



$$y = 4.44 + 0.23x$$

year 17, land price is 8.27 million baht

year 20, land price is 8.94 million baht

Question

56

- หากเรามีข้อมูลเกี่ยวกับเงินเดือน ในหน่วยบาท ซึ่งสัมพันธ์กับจำนวนปีที่ทำงาน ให้ทำนายเงินเดือนเมื่อทำงานครบ 10, 12 และ 15 ปี พร้อมทั้งสมการที่ใช้สำหรับการทำนายผล

years	1	3	7	8
Salary (baht)	25000	27000	34000	35000

Question

57

- ข้อมูลเกี่ยวกับการออกกำลังกายเป็นระยะเวลาในหน่วยนาที สัมพันธ์กับจำนวนแคลอรีที่ถูกเผาผลาญไป ดังตาราง ให้ทำนายจำนวนแคลอรีที่ถูกเผาผลาญ เมื่อเวลาผ่านไป 28, 43 และ 60 นาที พร้อมทั้งสมการที่ใช้สำหรับการทำนายผล

minutes	5	10	15	20	25	30	35	40	45	50	55
calories_burned	8	42	102	113	148	206	202	230	360	354	427



สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น)
Technology Promotion Association (Thailand-Japan)



สถาบันเทคโนโลยีไทย-ญี่ปุ่น
Thai-Nichi Institute of Technology
泰日工業大学

Logistic Regression

Logistic Regression

59

- Logistic Regression จัดอยู่ในกลุ่ม Supervised Learning ที่ทำนายผลแบบ Classification (แม้ชื่อโมเดลจะมีคำว่า Regression ก็ตาม)
- นอกจากนี้ ก็ยังจะกล่าวถึงการประเมินโมเดลจาก Confusion Matrix ซึ่งสามารถนำไปใช้กับโมเดลชนิดอื่น ๆ ที่ทำนายผลแบบ Classification ได้ทั้งหมด

Logistic Regression

60

- $\log_b(x) = y$ หมายถึง $b^y = x$
- $\log_2(8) = 3$ เพราะ $2^3 = 8$
- $\log_2(32) = a$ มีความหมายว่า $2^a = 32$ นั่นคือ $a = 5$
 เพราะ $2^5 = 32$
- ในกรณีที่เป็น **logarithm** ฐาน 10 เราอาจไม่ระบุเลขฐานก็ได้
- $\log_{10}(100)$ อาจเขียนเป็น $\log(100)$

Logistic Regression

61

- ส่วน Natural Logarithm คือกรณีที่เลขฐานของมันมีค่าเป็น 2.718281828459 หรือเรียกว่า Euler's number ซึ่งโดยทั่วไป เรามักใช้เพียงทศนิยม 2 - 3 ตำแหน่ง
- $\log_{2.718}(10) = \alpha$ หมายถึง $2.718^\alpha = 10$ นั่นคือ $\alpha = 2.302$ เพราะ $2.718^{2.302} \approx 10$
- โดยส่วนใหญ่เราจะแทนเลขฐานด้วยตัว e เช่น $\log_e(100) = 4.605$ หรือเขียนแทน \log_e ด้วยสัญลักษณ์ \ln ในลักษณะดังนี้

$$\log_e(x) = \ln(x)$$

$$\ln(100) = 4.605 \quad \text{เพราะ } e^{4.605} = 2.718^{4.605} \approx 100$$

Logistic Regression

62

- สูตรที่น่าสนใจอื่น ๆ ของ Logarithm
- $\ln(a * b) = \ln(a) + \ln(b)$
- $\ln(a/b) = \ln(a) - \ln(b)$
- $\ln(a^b) = b * \ln(a)$

Logistic Regression

63

- Odds Ratio และ Logit Function เป็นพื้นฐานที่จะนำไปสู่ Logistic Function
- โอกาสหรือความน่าจะเป็น (Probability) ที่จะได้ผลลัพธ์ที่เราต้องการ คือ (ให้ P แทนความน่าจะเป็นที่สิ่งนั้นจะเกิดขึ้น)

$$P = \frac{\text{จำนวนเหตุการณ์ที่จะเกิดสิ่งนั้น}}{\text{จำนวนเหตุการณ์ที่เป็นไปได้ทั้งหมด}}$$

Logistic Regression

64

- ความน่าจะเป็นต้องมีค่าอยู่ระหว่าง $0 - 1$ ($0 \leq P \leq 1$)
- ถ้าเราโยนเหรียญ 1 ครั้ง ความน่าจะเป็นที่จะได้ หัว หรือ ก้อย เท่ากับ $1/2$
- การทอยลูกเต๋าแต่ละครั้ง ความน่าจะเป็นที่จะได้แต้มที่ต้องการ เท่ากับ $1/6$
- ถ้าความน่าจะเป็นที่เกิดเหตุการณ์นั้น = P แสดงว่า ความน่าจะเป็นที่จะ ไม่เกิดเหตุการณ์นั้น = $1 - P$
- โอกาสที่สลากกินแบ่งแต่ละใบจะถูกรางวัลเลขท้าย 2 ตัว (00 - 99) เท่ากับ $1/100 = 0.01$
- โอกาสที่จะ ไม่ถูกรางวัลเลขท้าย 2 ตัว จึงเท่ากับ $1 - 0.01 = 0.99$

Logistic Regression

65

- **Odds Ratio** เป็นอัตราส่วนระหว่าง ความน่าจะเป็นที่จะเกิดสิ่งใดสิ่งหนึ่ง กับ ความน่าจะเป็นที่จะไม่เกิดสิ่งนั้น

$$\text{odds} = \frac{\text{ความน่าจะเป็นที่จะเกิดสิ่งนั้น}}{\text{ความน่าจะเป็นที่จะไม่เกิดสิ่งนั้น}} = \frac{P}{1 - P}$$

Logistic Regression

66

- Logit Function ก็คือ Natural Logarithm ของ Odds Ratio

$$\text{logit} = \ln\left(\frac{P}{1 - P}\right)$$

- เนื่องจาก P มีค่าระหว่าง $0 - 1$ ดังนั้น ถ้าเรานำไปแทนค่าเพื่อดูขอบเขตของ **logit** จะได้ดังนี้

- $\text{logit}(0) = \ln(0 / (1 - 0)) = \ln(0) = -\infty$

- $\text{logit}(1) = \ln(1 / (1 - 1)) = \ln(\infty) = \infty$

Logistic Regression

67

- **logit** จะมีค่าระหว่าง $(-\infty, \infty)$ ซึ่งหากเราวาดกราฟด้วย **Matplotlib** ก็จะเป็นดังภาพถัดไป หรือเรียกว่า **Logit Curve**

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

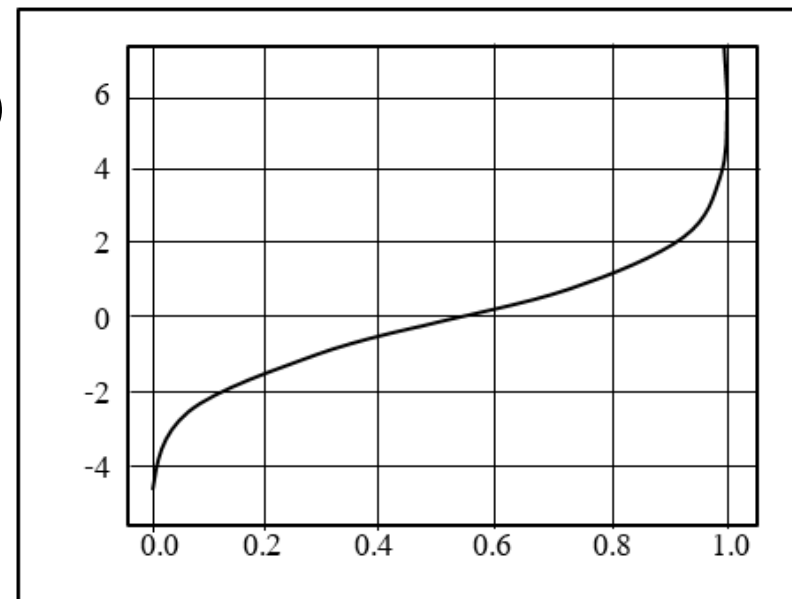
```
x = np.linspace(0, 0.999, num=100)
```

```
y = np.log(x/(1-x))
```

```
plt.plot(x, y)
```

```
plt.grid()
```

```
plt.show()
```



Logistic Regression

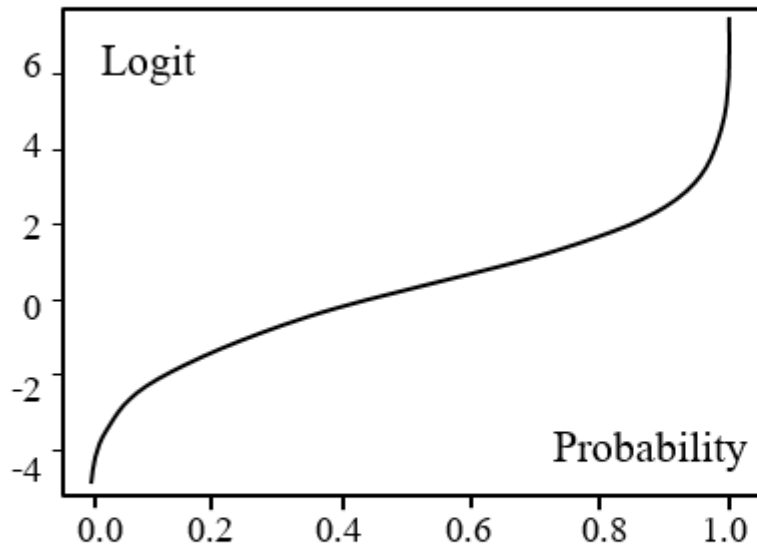
68

- **Sigmoid Function** คือฟังก์ชันที่ใช้ในการแปลงช่วงตัวเลขจาก $(-\infty, \infty)$ ไปเป็นช่วงตัวเลขระหว่าง $(0, 1)$ ซึ่งจากกราฟ **Logit Curve** ที่มีค่าระหว่าง $(-\infty, \infty)$ ดังที่กล่าวมา หากนำไปแปลงเป็นช่วงระหว่าง $(0, 1)$ ก็จะได้ **Sigmoid Curve** ในลักษณะดังภาพถัดไป

Logistic Regression

69

Logit Curve

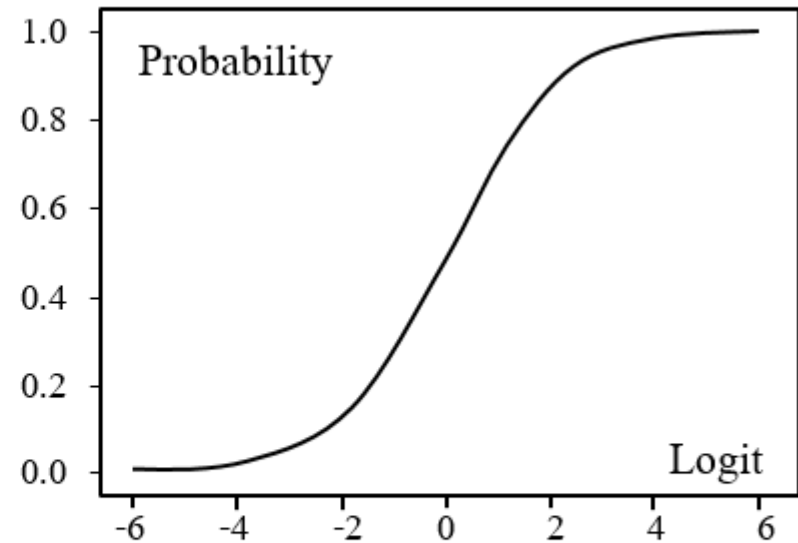


$$\text{logit} = \ln\left(\frac{P}{1-P}\right)$$

Inverse



Sigmoid Curve



$$P = \frac{1}{1 + e^{-\text{logit}}}$$

Logistic Regression

70

$$L = \ln \left[\frac{P}{1 - P} \right]$$

$$e^L = \frac{P}{1 - P}$$

$$\frac{1 - P}{P} = \frac{1}{e^L}$$

$$e^L - Pe^L = P$$

$$e^L = P + Pe^L = P(1 + e^L)$$

$$P = \frac{e^L}{1 + e^L} = \frac{1}{\frac{1}{e^L} + 1} = \frac{1}{e^{-L} + 1}$$

$$\text{sigmoid} = \frac{1}{1 + e^{-L}}$$

Logistic Regression

71

- **Sigmoid Function** ก็เหมือนกับการหาความน่าจะเป็นที่จะเกิดเหตุการณ์อย่างใดอย่างหนึ่ง ซึ่งมีค่าระหว่าง **0 – 1**

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

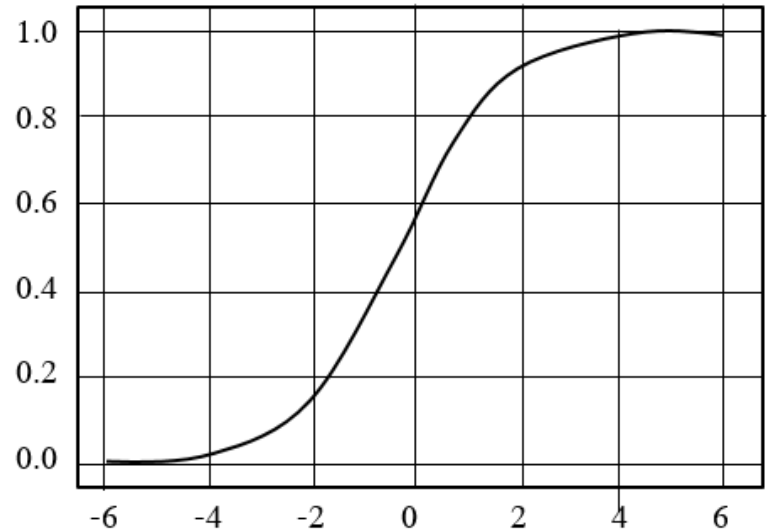
```
x = np.linspace (-6, 6, 121)
```

```
y = 1 / (1 + np.exp(-x))
```

```
plt.plot(x, y)
```

```
plt.grid()
```

```
plt.show()
```



Logistic Regression

72

- Sigmoid Function จะถูกนำไปใช้สำหรับการทำนายผลแบบ Logistic Regression ด้วยเหตุนี้ เราอาจ เรียก Sigmoid Function อีกอย่างว่าเป็น Logistic Function
- Logistic Regression เป็นโมเดลที่จัดอยู่ในประเภท Supervised Learning สำหรับการทำนายผลแบบ Classification แม้ชื่อโมเดลจะลงท้ายด้วยคำว่า Regression ก็ตาม

Logistic Regression

73

- ข้อมูลตัวอย่างแต่ละรายการจะมี คอลัมน์ที่เป็น **Target** (หรือผลลัพธ์ หรือตัวแปรตาม : y) ที่สามารถจำแนกประเภทหรือแบ่งได้เป็น 2 กลุ่ม เช่น Yes/No, True/False, Success/Fail, High/Low, Male/Female, R/X, 1/0
- ส่วนคอลัมน์ที่เป็น **Feature** หรือตัวแปรอิสระ (x) อาจมีตั้งแต่ 1 คอลัมน์ขึ้นไป เช่นเดียวกับ **Linear Regression**

Logistic Regression

74

x	y
4	No
7.2	Yes
5.1	Yes
3.2	No
8	Yes

x1	x2	y
7	5	0
3	4	1
2	8	0
6	7	1
8	7	1

Logistic Regression

75

- กรณีที่มีตัวแปรอิสระเพียงตัวเดียว

$$y = \beta_0 + \beta_1 x$$

$$y = 3.2 + 1.5x$$

- กรณีที่มีตัวแปรอิสระหลายตัว

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k$$

$$y = \beta_0 + \sum \beta_i x_i$$

$$y = 5.2 + 3.5x_1 + 4x_2 + 7x_3$$

Logistic Regression

76

- ถ้าความน่าจะเป็นขึ้นกับตัวแปรอิสระ x_1, x_2, x_3, \dots แสดงว่า **Logit Function** ก็มีรูปแบบเหมือนกับ สมการของ **Linear Regression**

$$\begin{aligned}\text{logit} &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k \\ &= \beta_0 + \sum \beta_i x_i\end{aligned}$$

$$P = \frac{1}{1 + e^{-\text{logit}}}$$

$$P(x) = \frac{1}{1 + e^{-(\beta_0 + \sum \beta_i x_i)}}$$

Logistic Regression

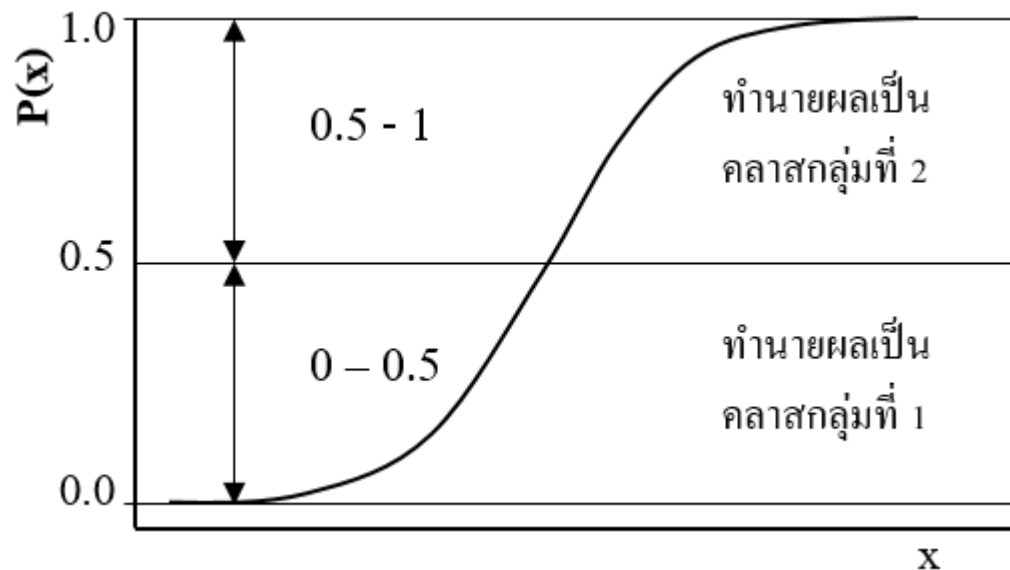
77

- **Logistic Regression** ก็คือการหาความน่าจะเป็นด้วย **Sigmoid Function** ทั้งนี้หากเราทราบค่า **intercept** และ **coefficient** ก็สามารถนำตัวแปรอิสระ x_1, x_2, \dots มาแทนค่าในฟังก์ชัน ก็จะได้ค่าความน่าจะเป็นออกมา
- อย่างไรก็ตาม ความน่าจะเป็นต้องมีค่าระหว่าง **0 - 1** ในขณะที่ผลลัพธ์ของ **Logistic Regression** จะต้องอยู่ในรูปแบบ **Classification** ที่จำแนกได้ **2** กลุ่ม เช่น **Yes/No** ซึ่งไม่สอดคล้องกัน จึงต้องนำค่าความน่าจะเป็นที่ได้ไปทำการเปรียบเทียบต่อ ดังนี้

Logistic Regression

78

- หากความน่าจะเป็นมีค่าระหว่าง $0 - 0.5$ จะทำนายผลเป็นคลาสกลุ่มที่ 1
- หากความน่าจะเป็นมีค่าระหว่าง $0.5 - 1$ จะทำนายผลเป็นคลาสกลุ่มที่ 2



□ จากสมการของ $P(\mathbf{x})$ ต่อไปนี้เป็นขั้นตอนการคำนวณค่า β_0 และ β_i

$$P(\mathbf{x}) = \frac{1}{1 + e^{-(\beta_0 + \sum \beta_i x_i)}}$$

$$y = \beta_0 + \sum \beta_i x_i$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$A = \sum x_1^2 - \frac{(\sum x_1)^2}{N}$$

$$B = \sum x_2^2 - \frac{(\sum x_2)^2}{N}$$

$$C = \sum x_1 y - \frac{(\sum x_1)(\sum y)}{N}$$

$$D = \sum x_2 y - \frac{(\sum x_2)(\sum y)}{N}$$

$$E = \sum x_1 x_2 - \frac{(\sum x_1)(\sum x_2)}{N}$$

x_1	x_2	y
7	5	0
3	4	1
2	8	0
6	7	1
8	7	1

$$\beta_1 = \frac{(B)(C) - (E)(D)}{(A)(B) - (E)^2}$$

$$\beta_2 = \frac{(A)(D) - (E)(C)}{(A)(B) - (E)^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}_1 - \beta_2 \bar{x}_2$$

Logistic Regression

80

- จากสมการของ $P(\mathbf{x})$ การคำนวณค่า β_0 และ β_i ค่อนข้างใช้เวลาในการคำนวณ
- เราสามารถนำ **Scikit-Learn** มาช่วยในการคำนวณ

Logistic Regression

81

- ตัวอย่างที่ 1 จากการสำรวจนักเรียนกลุ่มหนึ่งพบว่า การสอบผ่าน **Pass** หรือสอบตก **Fail** ขึ้นกับชั่วโมงการเรียน (**hours_study**) ดังตาราง ถ้าเรานำมา **Train Model** แบบ **Logistic Regression** แล้วทำนายผล ในกรณีที่ศึกษา 3.2, 3.75 และ 7 ชั่วโมง จะแสดงขั้นตอนได้ดังนี้

hours_study	pass
0	Fail
1	Fail
2	Fail
3	Fail
4	Pass
5	Pass
6	Pass

Logistic Regression

82

```
import numpy as np
from sklearn.linear_model import LogisticRegression

x = [[0], [1], [2], [3], [4], [5], [6]] #[[x1]. ...]
y = ['Fail', 'Fail', 'Fail', 'Fail', 'Pass', 'Pass', 'Pass']
model = LogisticRegression()
model.fit(x, y)

x_predict = [3.2, 3.75, 7]
x_predict = np.array(x_predict).reshape(-1, 1)
y_predict = model.predict(x_predict)
print('Prediction:')

for (i, xp) in enumerate(x_predict):
    print(f'study: {xp[0]} hours =>', y_predict[i])
print()

print('Logistic Function:')
ic = '{:.2f}'.format(model.intercept_[0])
ce = '{:.2f}'.format(model.coef_[0,0])
print(f'P(x) = 1 / (1 + exp({ic} + ({ce})x))')
```

Logistic Regression

83

```
import numpy as np
from sklearn.linear_model import LogisticRegression

x = [[0], [1], [2], [3], [4], [5], [6]] #[[x1]. ...]
y = ['Fail', 'Fail', 'Fail', 'Fail', 'Pass', 'Pass', 'Pass']
model = LogisticRegression()
model.fit(x, y)
```

```
x_predict = [3.2, 3.75, 7]
x_predict = np.array(x_predict).reshape(-1, 1)
y_predict = model.predict(x_predict)
print('Prediction:')

for (i, xp) in enumerate(x_predict):
    print(f'study: {xp[0]} hours =>', y_predict[i])
print()
```

```
print('Logistic Function:')
ic = '{:.2f}'.format(model.intercept_[0])
ce = '{:.2f}'.format(model.coef_[0,0])
print(f'P(x) = 1 / (1 + exp({ic} + ({ce})x))')
```

Prediction:

study: 3.2 hours => Fail
study: 3.75 hours => Pass
study: 7.0 hours => Pass

Logistic Function:

$P(x) = 1 / (1 + \exp(-4.03 + (1.15)x))$

Logistic Regression

84

- ตัวอย่างที่ 2 หากเรามีข้อมูลดังตารางในภาพ ถ้านำมา **Train Model** แบบ **Logistic Regression** ให้ทำนายผลเมื่อ $x_1=3, x_2=3$ และ $x_1=5, x_2=6$

x_1	x_2	y
7	6	yes
2	4	no
5	8	yes
4	7	no
8	10	yes

Logistic Regression

85

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
x = [[7, 6], [2, 4], [5, 8], [4, 7], [8, 10]] #[[x1, x2]. ...]
y = ['yes', 'no', 'yes', 'no', 'yes']
model.fit(x, y)

#ทำนายผลเมื่อ x1=3, x2=3 และ x1=5, x2=6
x_predict = [[3, 3], [5, 6]]
y_predict = model.predict(x_predict)

print('X = [3, 3], y =', y_predict[0])
print('X = [5, 6], y =', y_predict[1])
print()
print('probability')
prob = model.predict_proba(x_predict)
print(prob)
print('      1-P      P')
```

Logistic Regression

86

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
x = [[7, 6], [2, 4], [5, 8], [4, 7], [8, 10]] #[[x1, x2]. ...]
y = ['yes', 'no', 'yes', 'no', 'yes']
model.fit(x, y)
```

```
#ทำนายผลเมื่อ x1=3, x2=3 และ x1=5, x2=6
x_predict = [[3, 3], [5, 6]]
y_predict = model.predict(x_predict)
```

```
print('X = [3, 3], y =', y_predict[0])
print('X = [5, 6], y =', y_predict[1])
print()
print('probabiltiy')
prob = model.predict_proba(x_predict)
print(prob)
print('          1-P          P')
```

X = [3, 3], y = no
X = [5, 6], y = yes

```
probabiltiy
[[0.93074277 0.06925723]
 [0.46291989 0.53708011]]
          1-P          P
```

Question

87

□ บริษัทแห่งหนึ่งได้ทดสอบการทำงานของเครื่องจักรพบว่า อายุของเครื่องจักร (หน่วยเป็น เดือน) และจำนวนชั่วโมงการทำงานต่อวัน มีผลต่อสมรรถนะของเครื่องจักร ดังตาราง ให้ทำนายผลในกรณีนี้

1) อายุของเครื่องจักรเท่ากับ 40 เดือน และทำงาน 3 ชั่วโมงต่อวัน

2) อายุของเครื่องจักรเท่ากับ 70 เดือน และทำงาน 2 ชั่วโมงต่อวัน

และให้หา **Logistic Function**

machine_age (months)	operate_hours_per_day	performance
59	4	Not good
15	4	Good
78	8	Not good
22	6	Good
38	5	Good
71	7	Not good
35	4	Not good
44	5	Good

Confusion Matrix

88

- ถ้าเราต้องการประเมินการทำนายผลของโมเดล หากเป็นกรณีของ **Linear Regression** ที่เราได้ศึกษามา ก็สามารถนำผลลัพธ์จริง (y_{true}) และผลการทำนาย (y_{predict}) มาคำนวณค่าต่าง ๆ ที่เกี่ยวข้องได้เลย เพราะมีค่าเป็นตัวเลข
- กรณีของ **Logistic Regression** ซึ่งผลการทำนายเป็นแบบ **Classification** เราจึงไม่สามารถทำเช่นนั้นได้ แต่ต้องใช้วิธีการนับจำนวนรายการที่ทำนายถูก ($y_{\text{true}} = y_{\text{predict}}$) และจำนวนรายการที่ทำนายผิด ($y_{\text{true}} \neq y_{\text{predict}}$) แล้วมาเปรียบเทียบกัน

Confusion Matrix

89

	Predict: Negative (N)	Predict: Positive (P)
Actual: Negative	TN	FP
Actual: Positive	FN	TP

Confusion Matrix

90

ผลลัพธ์จริง (Actual)	การทำนาย (Predict)	ผลการทำนาย	Confusion Matrix
Negative	Negative	True	True Negative (TN)
Negative	Positive	False	False Positive (FP)
Positive	Negative	False	False Negative (FN)
Positive	Positive	True	True Positive (TP)

Confusion Matrix

91

x	y_true	y_predict	Confusion Matrix
...	1	1	TP
...	1	0	FN
...	0	0	TN
...	1	1	TP
...	1	0	FN
...	0	0	TN
...	1	1	TP
...	0	0	TN
...	0	1	FP
...	0	0	TN

Model Evaluation

92

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Model Evaluation

93

□ **Accuracy** คือความแม่นยำในการทำนายผล โดยคิดจากจำนวนที่ทำนายถูกทั้งหมดหารด้วยจำนวนรายการทั้งหมดใน **Confusion Matrix**

□ **Accuracy** = $(20 + 60) / 100 = 0.8$ หรืออัตราการทำนายถูกเท่ากับ 80%

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{Total}}$$

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Model Evaluation

94

- **Precision** คือ อัตราส่วนของการทำนายเชิงบวกแล้วถูกต้อง (TP) ต่อจำนวนการทำนายเชิงบวกทั้งหมด

$$\text{Precision} = 60 / 70 = 0.86$$

$$\text{Precision} = \frac{\text{TP}}{\text{Predict_Positive}}$$

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Model Evaluation

95

- **Recall** (หรือ **Sensitivity** หรือ **True Positive Rate**) คือ ตราส่วนของการทำนายเชิงบวกแล้วถูกต้อง (TP) ต่อผลลัพธ์จริงในเชิงบวก (Actual Positive) ทั้งหมด

$$\text{Recall} = 60 / 70 = 0.86$$

$$\text{Recall} = \frac{\text{TP}}{\text{Actual_Positive}}$$

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Model Evaluation

96

- **F1 Score** เป็นค่าที่แสดงถึงระดับความสอดคล้องกันระหว่าง Precision และ Recall

$$F1 = 2 * (0.86 * 0.86) / (0.86 + 0.86) = 0.86$$

$$F1 = \frac{2 * (\text{precision} * \text{recall})}{\text{precision} + \text{recall}}$$

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Logistic Regression

97

- **Error Rate** (หรือ **Misclassification Rate**) คืออัตราความคลาดเคลื่อน ซึ่งคำนวณจากจำนวนที่ทำนายผิดทั้งหมด หารด้วยจำนวนรายการทั้งหมด
- จากจำนวนสมาชิกของ **Confusion Matrix** ที่ผ่านมา แสดงว่าอัตราความคลาดเคลื่อน คือ

$err = (10 + 10) / 100 \# 0.2$ หรืออัตราทำนายผิดเท่ากับ 20%

$$\text{Error_Rate} = \frac{\text{FP} + \text{FN}}{\text{Total}}$$

	Predict (N)	Predict (P)	รวม
Actual (N)	TN = 20	FP = 10	30
Actual (P)	FN = 10	TP = 60	70
รวม	30	70	100

Logistic Regression

98

- ค่าต่างๆ ดังที่กล่าวมา เราพิจารณาจากจำนวนที่ทำนายถูกต้องเป็นหลัก (ยกเว้น **Error Rate**) ดังนั้น ถ้าได้ตัวเลขที่มีค่ามากแสดงว่าจำนวนการทำนายผลถูกต้องก็มีมากเช่นกัน หรือกล่าวได้ว่า เป็นการทำนายผลมีความน่าเชื่อถือนั่นเอง

Logistic Regression

99

- บริษัทแห่งหนึ่งได้ทดสอบการทำงานของเครื่องจักรพบว่า อายุของเครื่องจักร (หน่วยเป็น เดือน) และจำนวนชั่วโมงการทำงานต่อวัน มีผลต่อสมรรถนะของเครื่องจักร ดังตารางไฟล์

machine_performance.xlsx ให้ทำนายผลในกรณีนี้ที่

- 1) อายุของเครื่องจักรเท่ากับ 20 เดือน และทำงาน 3 ชั่วโมงต่อวัน
- 2) อายุของเครื่องจักรเท่ากับ 50 เดือน และทำงาน 7 ชั่วโมงต่อวัน
- 3) อายุของเครื่องจักรเท่ากับ 40 เดือน และทำงาน 2 ชั่วโมงต่อวัน
- 4) อายุของเครื่องจักรเท่ากับ 80 เดือน และทำงาน 5 ชั่วโมงต่อวัน

และให้หา **Logistic Function** และ **Confusion Matrix**

	machine_age_months	work_hours_per_day	machine_performance
0	58	3	1
1	73	5	0
2	23	4	1
3	59	4	0
4	16	3	1
5	37	1	1
6	68	5	0
7	49	5	1
8	27	7	0
9	60	2	1
10	11	5	1
11	78	8	0
12	23	5	1
13	37	3	1
14	57	7	0
15	73	8	0
16	39	4	1
17	71	7	0
18	35	4	1
19	45	4	1

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

df = pd.read_excel(r'machine_performance.xlsx')
with pd.option_context('display.max_rows', 20): display(df)

x = df[['machine_age_months', 'work_hours_per_day']]
y = df['machine_performance']

x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0)

#scaling
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)

model = LogisticRegression()
model.fit(x_train, y_train)
```

```

x_pred = [[20, 3], [50, 7], [40, 2], [80, 5]]
#ข้อมูลที่จะทำนายผล ก็ต้องปรับสเกลเช่นเดียวกัน
x_pred_sc = scaler.transform(x_pred)
y_pred = model.predict(x_pred_sc)

print('Prediction:')
for (i, xp) in enumerate(x_pred):
    # คอสถมันน์ผลลัพธ์มีค่าเป็น 0/1 จึงเทียบเป็นสตริงที่เข้าใจได้
    machine_perf = 'No' if y_pred[i] == 0 else 'Yes'
    text = f'Machine Age: {xp[0]} Month(s), '
    text += f'Work: {xp[1]} Hour(s)/Day '
    text += f'=> Performance: {machine_perf}'
    print(text)
print()

print('Logistic Function:')
ic = '{:.2f}'.format(model.intercept_[0])
ce1 = '{:.2f}'.format(model.coef_[0, 0])
ce2 = '{:.2f}'.format(model.coef_[0, 1])

print(f'P(x) = 1 / (1 + exp({ic} + ({ce1})age + ({ce2})hour))')

```

```
### Model Evaluation ###
y_pred_test = model.predict(x_test)
print('Confusion Matrix:')
cfmx = confusion_matrix(y_test, y_pred_test)
print(cfm)
print()

print('Accuracy:', '{:.2f}'.format(accuracy_score(y_test, y_pred_test)))
print('Precision:', '{:.2f}'.format(precision_score(y_test, y_pred_test)))
print('Recall:', '{:.2f}'.format(recall_score(y_test, y_pred_test)))
print('F1 Score:', '{:.2f}'.format(f1_score(y_test, y_pred_test)))

#error_rate = (FP + FN) / Total
err = (cfmx[0, 1] + cfm[1, 0]) / y_test.count()
print('Error Rate', '{:.2f}'.format(err))
print()
```

Prediction:

Machine Age: 30 Month(s), Work: 6 Hour(s)/Day => Performance: Yes
Machine Age: 40 Month(s), Work: 8 Hour(s)/Day => Performance: No
Machine Age: 50 Month(s), Work: 5 Hour(s)/Day => Performance: No
Machine Age: 60 Month(s), Work: 3 Hour(s)/Day => Performance: Yes

Logistic Function:

$P(x) = 1 / (1 + \exp(0.60 + (-1.21)\text{age} + (-1.10)\text{hour}))$

Confusion Matrix:

```
[[1 1]
 [0 3]]
```

Accuracy: 0.80

Precision: 0.75

Recall: 1.00

F1 Score: 0.86

Error Rate 0.20

Question

105

- ร้านค้าแห่งหนึ่งได้จัดเก็บข้อมูลของผู้ที่เข้าสมัครสมาชิกของร้านไว้ในไฟล์ **Excel** แล้วพบว่า เพศ อายุ และเงินเดือน **gender age salary** มีผลต่อการสั่งซื้อสินค้า (**purchased**) โดยข้อมูลแสดงดังตารางไฟล์ **customer_info.xlsx** กำหนดให้ **gender** เท่ากับ 2 คือเพศหญิง ให้ทำนายผลในกรณีที่
- 1) เพศชาย อายุเท่ากับ 40 ปี และเงินเดือน 19500 บาท
 - 2) เพศหญิง อายุเท่ากับ 40 ปี และเงินเดือน 19500 บาท
 - 3) เพศหญิง อายุเท่ากับ 25 ปี และเงินเดือน 20500 บาท
 - 4) เพศชาย อายุเท่ากับ 60 ปี และเงินเดือน 40000 บาท
- และให้หา **Logistic Function** และ **Confusion Matrix**

Question

106

	gender	age	salary	purchased
0	1	19	19000	0
1	1	35	20000	0
2	2	26	43000	0
3	2	27	57000	0
...				
396	1	51	23000	1
397	2	50	20000	1
398	1	36	33000	0
399	2	49	36000	1
400 rows × 4 columns				