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

Logistic Regression

Logistic Regression

- Logistic Regression จัดอยู่ในกลุ่ม Supervised Learning ที่ทำนายผลแบบ Classification (ถึงแม้ชื่อโมเดลจะมีคำว่า Regression ก็ตาม)

Logistic Regression

- $\log_b(x) = y$ 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

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

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

Logistic Regression

- ส่วน Natural Logarithm คือกรณีที่เลขฐานของมันมีค่าเป็น 2.718281828459 หรือเรียกว่า Euler's number ซึ่งโดยทั่วไป เรามักใช้เพียงทศนิยม 2 - 3 ตำแหน่ง
- $\log_{2.718}(10) = a$ หมายถึง $2.718^a = 10$ นั่นคือ $a = 2.302$ 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

- 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

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

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

Logistic Regression

- ความน่าจะเป็นต้องมีค่าอยู่ระหว่าง 0 - 1 ($0 \leq P \leq 1$)

เช่น ถ้าเราโยนเหรียญ 1 ครั้ง ความน่าจะเป็นที่จะได้ หัว หรือ ก้อย เท่ากับ 1/2 หรือ การทอยลูกเต๋าแต่ละครั้ง ความน่าจะเป็นที่จะได้แต้มที่ต้องการ เท่ากับ 1/6 และ ถ้าความน่าจะเป็นที่เกิดเหตุการณ์นั้น = P แสดงว่า ความน่าจะเป็นที่จะ **ไม่** เกิดเหตุการณ์นั้น = 1 - P

เช่น โอกาสที่สลากกินแบ่งแต่ละใบจะถูกรางวัลเลขท้าย 2 ตัว (00 - 99) เท่ากับ 1/100 = 0.01

ดังนั้น โอกาสที่จะ **ไม่** ถูกรางวัลเลขท้าย 2 ตัว จึงเท่ากับ 1 - 0.01 = 0.99

Logistic Regression

- Odds Ratio เป็นอัตราส่วนระหว่าง ความน่าจะเป็นที่จะเกิดสิ่งใดสิ่งหนึ่ง กับ ความน่าจะเป็นที่จะ **ไม่** เกิด สิ่งนั้น

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

Logistic Regression

- Logit Function ก็คือ Natural Logarithm ของ Odds Ratio

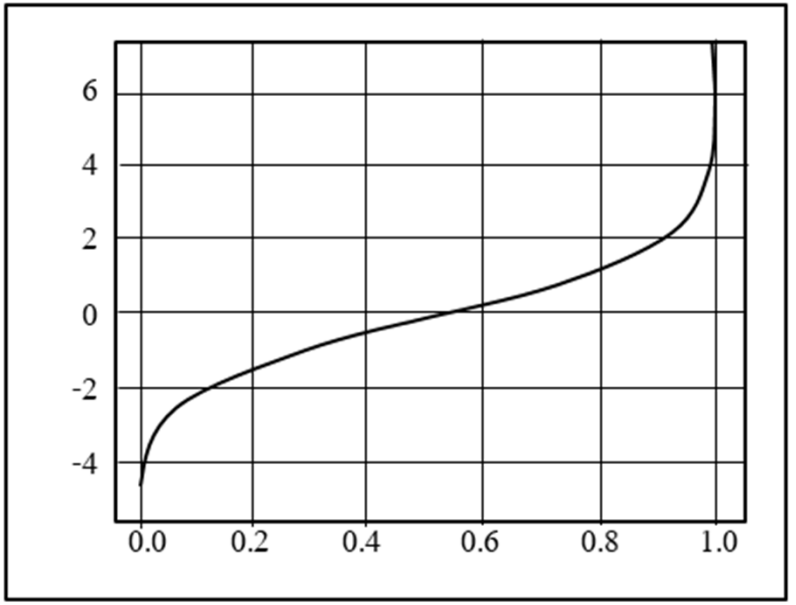
$$\text{logit} = \ln\left(\frac{P}{1-P}\right)$$

- 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

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

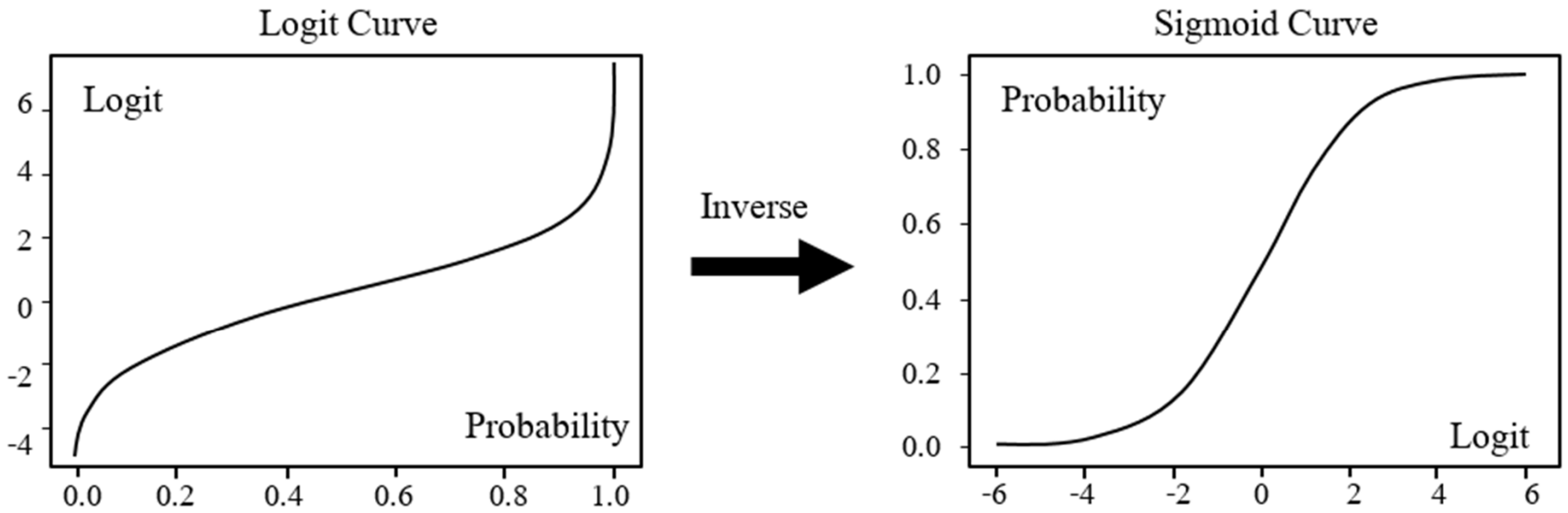
```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 0.999, num=100)
y = np.log(x/(1-x))
plt.plot(x, y)
plt.grid()
plt.show()
```



Logistic Regression

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

Logistic Regression



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

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

Logistic Regression

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

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

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

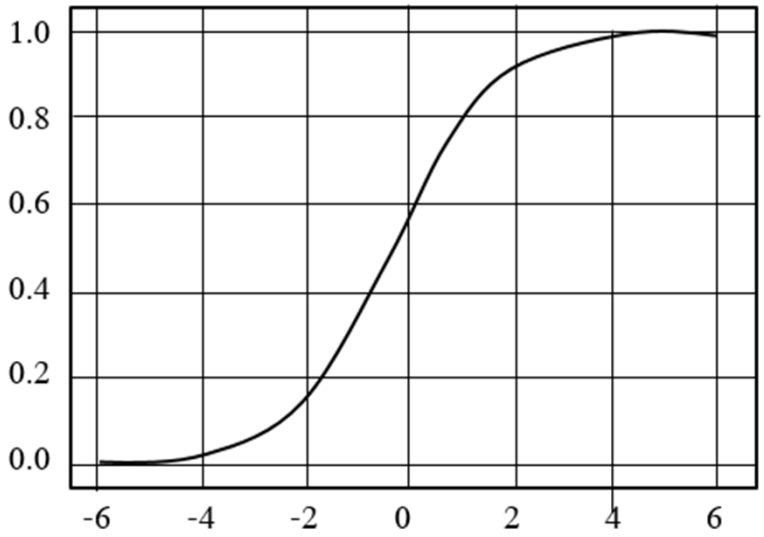
$$P = \frac{1}{1+\frac{1}{e^{L}}} = \frac{1}{1+e^{-L}}$$

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

Logistic Regression

- 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

- Sigmoid Function จะถูกนำไปใช้สำหรับการทำนายผลแบบ Logistic Regression ด้วยเหตุนี้ เราอาจ เรียก Sigmoid Function อีกอย่างว่าเป็น Logistic Function

Logistic Regression

- Logistic Regression เป็นโมเดลที่จัดอยู่ในประเภท Supervise Learning สำหรับการทำนายผลแบบ Classification ถึงแม้ชื่อโมเดลจะลงท้ายด้วยคำว่า Regression ก็ตาม โดยข้อมูลตัวอย่างแต่ละรายการจะมี 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

x	y
5.5	No
6.5	Yes
8.1	Yes
3.2	No
7.5	Yes

x1	x2	y
8	6	1
3	5	0
4	9	0
5	8	1
9	9	1

Logistic Regression

- กรณีที่มีตัวแปรอิสระเพียงตัวเดียว

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

- กรณีที่มีตัวแปรอิสระหลายตัว

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

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

Logistic Regression

- ถ้าความน่าจะเป็นขึ้นกับตัวแปรอิสระ x1, x2, x3, ... แสดงว่า Logit Function ก็มีรูปแบบเหมือนกับ สมการของ Linear Regression

$$\text{logit} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$$

$$= \beta_0 + \Sigma \beta i x i$$

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

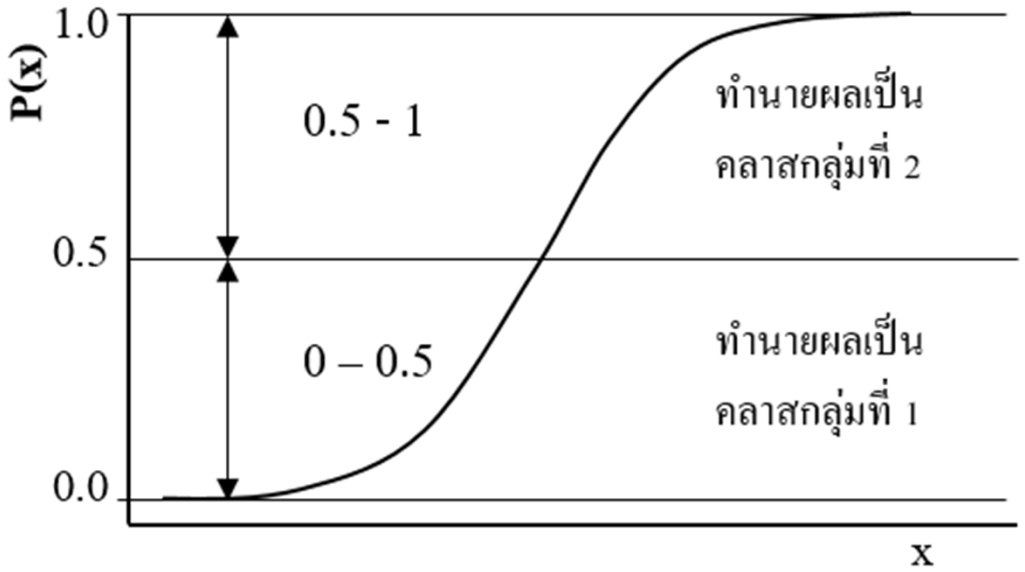
$$P(x) = \frac{1}{1 + e^{-(\beta_0 + \Sigma \beta_i x_i)}}$$

Logistic Regression

- Logistic Regression ก็คือการหาความน่าจะเป็นด้วย Sigmoid Function ทั้งนี้หากเราทราบค่า intercept และ coefficient ก็สามารถนำตัวแปรอิสระ x1, x2, ... มาแทนค่าในฟังก์ชัน ก็จะได้ค่าความน่าจะเป็นออกมา แต่อย่างไรก็ตาม ความน่าจะเป็นต้องมีค่าระหว่าง 0 - 1 ในขณะที่ผลลัพธ์ของ Logistic Regression จะต้องอยู่ในรูปแบบ Classification ที่จำแนกได้ 2 กลุ่ม เช่น Yes/No ซึ่งไม่สอดคล้องกัน ดังนั้น จึงต้องนำค่าความน่าจะเป็นที่ได้ไปทำการเปรียบเทียบต่อ ดังนี้

Logistic Regression

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



Logistic Regression

- จากสมการของ $P(x)$ การคำนวณค่า β_0 และ β_i ค่อนข้างซับซ้อน
- เราจึงนำ **Scikit-Learn** มาช่วยในการคำนวณ

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

Logistic Regression

- ตัวอย่างที่ 1 หากเรามีข้อมูลดังตารางในภาพ ถ้านำมา Train Model แบบ Logistic Regression สำหรับการทำนายผลก็จะได้ดังนี้

x1	x2	y
8	6	yes
3	5	no
4	9	no
5	8	yes
9	9	yes

Logistic Regression


```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()

x = [[8, 6], [3, 5], [4, 9], [5, 8], [9, 9]] #[[x1, x2]. ...]

y = [‘yes’, ‘no’, ‘no’, ‘yes’, ‘yes’]

model.fit(x, y)

#ทำนายผลเมื่อ x1=4, x2=4, y=? และ x1=5, x2=5, y=?

x_predict = [[4, 4], [5, 5]]

y_predict = model.predict(x_predict)
```

Logistic Regression


```
print(‘m x = [4, 4], y =‘, y_predict[0])

print(‘m x = [5, 5], y =‘, y_predict[1]) print()

print( ‘ความน่าจะเป็น’)

prob = model.predict_proba (x_predict)

print(prob)
```

Logistic Regression

- แบบฝึกหัด 1 หากผลการสอบ ผ่าน หรือ ไม่ผ่าน ขึ้นอยู่กับชั่วโมงที่เรียน กับ ชั่วโมงการพักผ่อน ดังตาราง ให้ทำนายว่า หากเรียน 7 ชั่วโมง พักผ่อน 1 ชั่วโมง จะสอบผ่านหรือไม่

Learn	Relax	Test
5	4	pass
3	6	not pass
9	2	pass
2	9	pass
4	5	not pass