บทที่ 4.
การวิเคราะห์การถดถอย
Regression Analysis

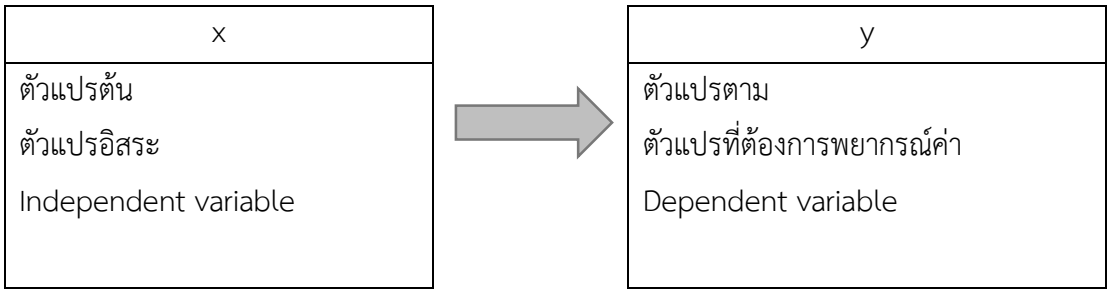
4.1 แนวคิดการทำ regression analysis

Regression analysis คือการวิเคราะห์หาความสัมพันธ์ของข้อมูล เมื่อเราต้องการประมาณการณ์ค่าของข้อมูลที่เราไม่รู้จัก ที่จริงในชีวิตประจำวันเราอาจรู้จัก regression analysis แบบไม่รู้ตัว เช่น เริ่มจากตัวอย่างง่ายๆคือ มีเงิน 10 บาทซื้อขนมได้ 1 ชิ้น ถ้ามีเงิน 50 บาทจะซื้อได้กี่ชิ้น กรณีนี้เราสามารถคำนวณได้ง่ายๆจากการเทียบบัญญัติไตรยางค์ เพราะความสัมพันธ์ของข้อมูล 2 ชนิดนี้ (เงิน และราคาขนม) เป็นความสัมพันธ์แบบคงที่ 10 บาท ซื้อได้ 1 ชิ้น ดังนั้น 50 บาท ซื้อได้ 5 ชิ้น

แต่หลายๆครั้งข้อมูลไม่ได้มีความสัมพันธ์กันแบบคงที่แบบนี้ เช่น ถ้า 10 วินาที เราวิ่งได้ 50 เมตร 100 วินาที เราจะวิ่งได้กี่เมตร ถ้าเราใช้วิธีเทียบบัญญัติไตรยางศ์เหมือนหาราคาขนม เราคงตอบว่า 100 วินาที วิ่งได้ 500 เมตร แต่ในความเป็นจริง ยิ่งวิ่งนานขึ้น เรายิ่งเหนื่อยมากขึ้น ทำให้ระยะทางที่วิ่งได้ลดลง เราอาจวิ่งได้แค่ 200 หรือ 300 เมตรเท่านั้น จะเห็นความความสัมพันธ์ของข้อมูล 2 ชนิดนี้ (เวลา และระยะทาง) ไม่มีความคงที่ และบางทีอาจมีปัจจัยอื่นเข้ามาเกี่ยวข้องเช่น อุณภูมิระหว่างวัน น้ำหนักตัวของผู้วิ่ง ส่วนสูงของผู้วิ่ง อายุของผู้วิ่ง ปัจจัยอื่น ๆ เหล่านี้ล้วนส่งผลกับสิ่งที่เราสนใจ (ระยะทาง) ทั้งนั้น

นี่คือความสำคัญของการทำ regression analysis คือการวิเคราะห์ข้อมูลจากข้อมูลที่เรามี (ตัวแปร – variable) เพื่อพยากรณ์ค่าหรือทำนาย (predict) ค่าที่เราสนใจ จากตัวอย่างเรื่องขนม ข้อมูลที่เรามีคือ เงิน ข้อมูลที่เราสนใจคือ จำนวนขนมที่ซื้อได้ หรือจากตัวอย่างเรื่องการวิ่ง ข้อมูลที่เรามีคือ เวลาที่ใช้วิ่ง อุณภูมิ น้ำหนัก ส่วนสูง อายุ ข้อมูลที่เราสนใจพยากรณ์คือ ระยะทางที่วิ่งได้

โดยสรุปให้เข้าใจง่ายๆ การทำ regression analysis คือ **การสร้างสมการชุดหนึ่งเพื่อทำนายค่าสิ่งที่เราต้องการ** เช่น หลังจากวิเคราะห์แล้ว การคำนวณหาระยะทางที่วิ่งได้อาจเป็น y = 2.5(x) + 40 โดย y คือระยะทางที่วิ่งได้ และ x คือเวลาที่ใช้ในการวิ่ง เป็นต้น

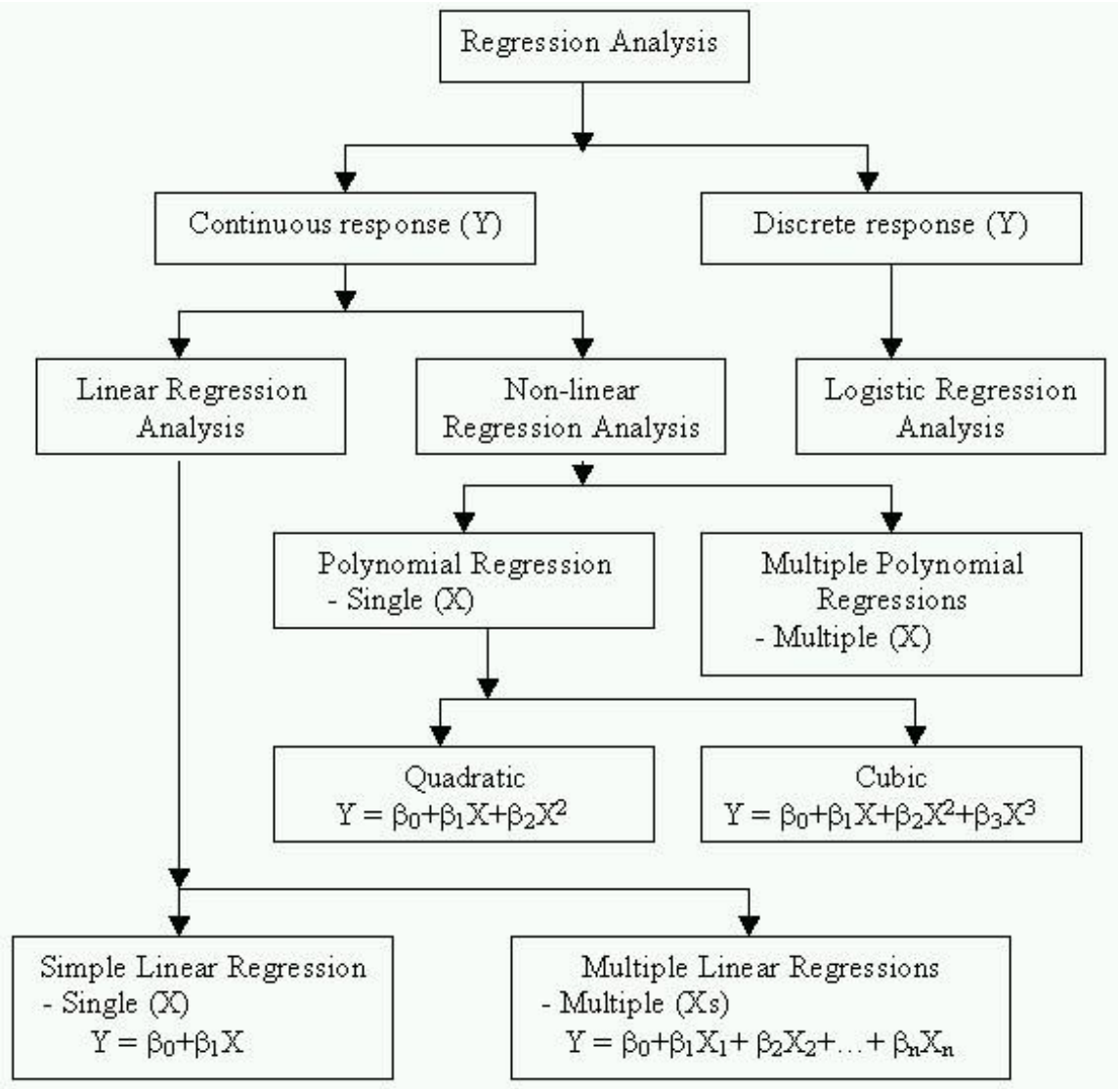


4.1.1 ตัวอย่างการประยุกต์ใช้ Regression analysis

4.1.2 รูปแบบการทำ Regression analysis

การทำ regression analysis มีระดับความยากง่ายแตกต่างกัน ขึ้นอยู่กับลักษณะข้อมูล จำนวนตัวแปรต้น และรูปแบบความสัมพันธ์ของตัวแปรต้นและตัวแปรตาม เนื้อหาบทนี้เราจะเรียน 4 รูปแบบคือ

1. Simple linear regression
2. Multiple linear regression
3. Polynomial regression
4. Logistic regression



ภาพจาก https://sites.google.com/site/mystatistics01/regression-correlation-analysis/regression-analysis

4.2 Simple linear regression การถดถอยเชิงเส้นอย่างง่าย

Simple linear regression คือรูปแบบความสัมพันธ์ระหว่างข้อมูล 2 ชนิด นั่นคือมีเพียง 1 ตัวแปร ที่ส่งผลกับข้อมูลที่เราสนใจ เช่น

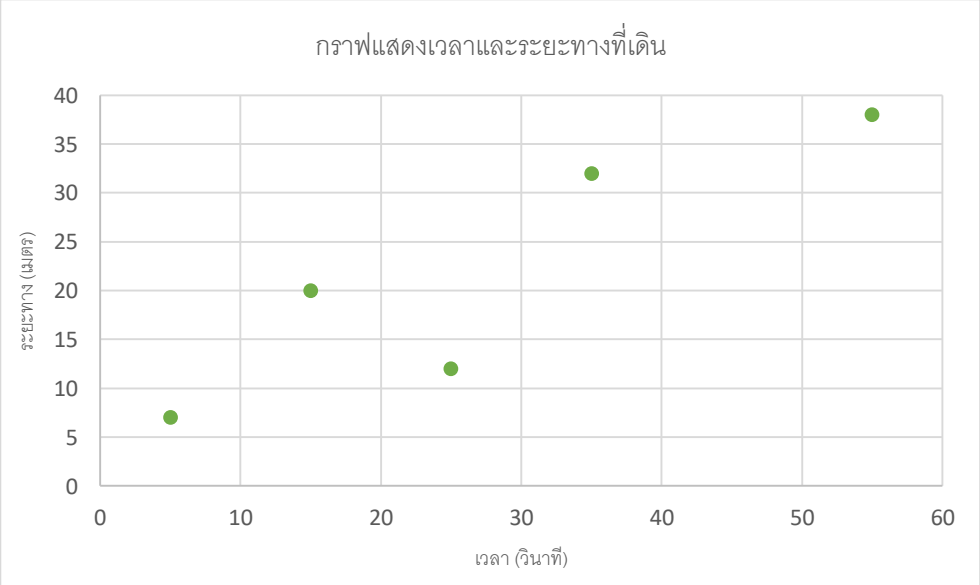
- เวลาที่ใช้และระยะทางที่วิ่งได้
- จำนวนแคลอรี่ที่กินแต่ละวันกับน้ำหนักตัวที่เพิ่มขึ้น
- เงินที่ลงทุนซื้อโฆษณาบน Facebook กับยอดขายสินค้าที่เพิ่มขึ้น
- เงินเดือนกับยอดการ shopping

4.2.1 แนวคิด linear regression แบบ least square

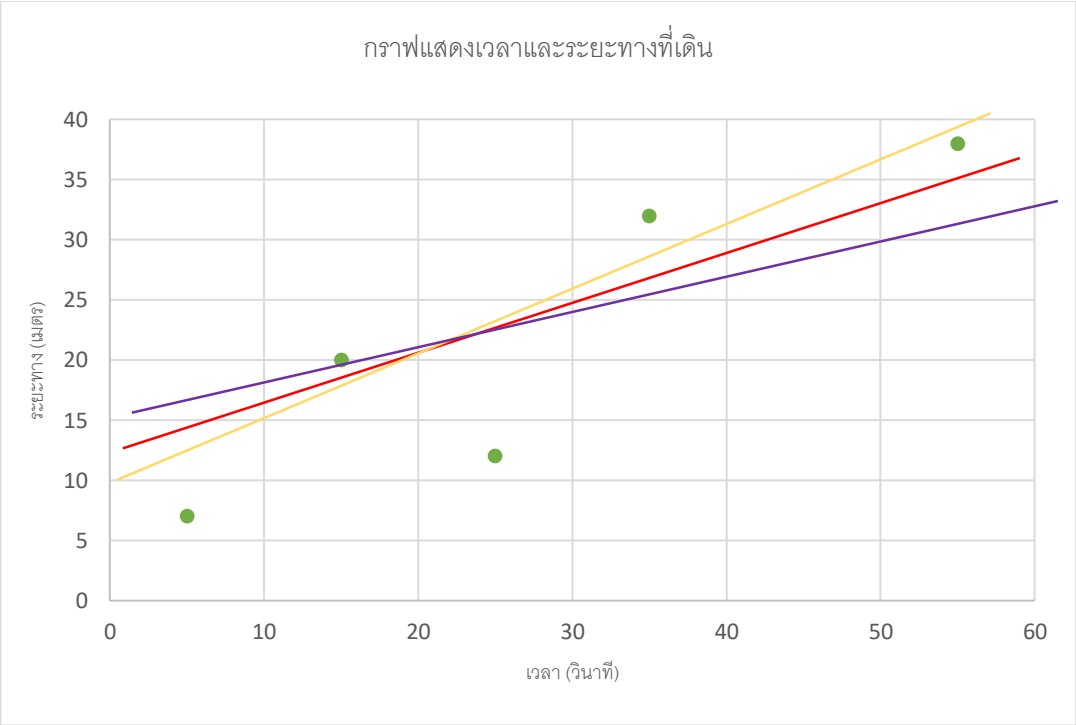
สมมติเราเก็บข้อมูลของการเดินว่าระยะทางที่เดินได้ใช้เวลากี่วินาที ข้อมูลแสดงดังนี้

ระยะทาง (เมตร) (Y)	เวลา (วินาที) (X)
7	5
20	15
12	25
32	35
38	55

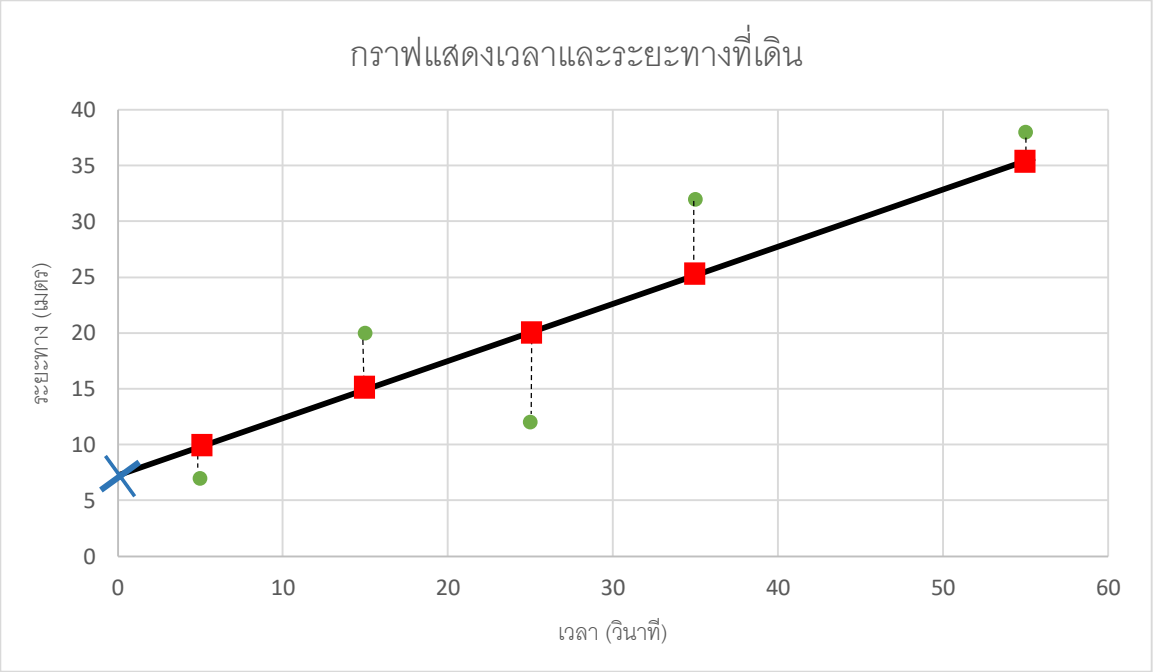
จากข้อมูลสามารถสร้างกราฟแสดงความสัมพันธ์ของระยะทาง (Y) และ เวลา (X) ได้ดังนี้



จะเห็นว่า ถ้าเราต้องการประเมิณว่า ในเวลา 100 วินาที จะเดินได้กี่เมตร เราไม่สามารถตอบได้ทันทีจากข้อมูลที่มีอยู่ เราต้องสร้างเส้นตรงที่เป็นตัวแทนของกลุ่มข้อมูลเหล่านี้เพื่อช่วยในการคำนวณ เราอาจสร้างเส้นตรงได้ 3 เส้น ดังนี้



ปัญหาคือ เราไม่รู้ว่า เส้นตรงใดคือเส้นตรงที่ดีที่สุด แนวคิดการทำ linear regression คือ เส้นตรงที่ดีที่สุดคือเส้นที่ทำให้เกิดความคลาดเคลื่อน (error) น้อยที่สุด เพื่อใช้ในการคาดการณ์ข้อมูล



ความหมายของแต่ละจุดในกราฟคือ

สัญลักษณ์	ชื่อเรียก	ความหมาย
———	linear regression equation $\hat{y} = bx_i + a$	เส้นประมาณการณ์ linear regression
●	Actual response, y_i	ค่าข้อมูลจริง
■	Predicted response, $f(x_i)$ หรือ $\hat{y}$ โดย $\hat{y}_i = bx_i + a$	ค่าข้อมูลที่ได้จากการประมาณการณ์
– – – ·	Residuals $y_i - \hat{y}$	ค่าความคลาดเคลื่อน หรือความผิดพลาด (error) คำนวณจาก observed value – predicted value
×	y intercept (a)	จุดตัดแกน y คือ หรือค่าของ y เมื่อ x มีค่าเป็น 0

แนวคิด Least Squares Method คือเส้นตรงที่ดีที่สุดคือ เส้นที่ทำให้เกิดความคลาดเคลื่อนน้อยที่สุด หรือผลต่างของค่าจริงกับค่าประมาณการณ์น้อยที่สุด นั่นคือหาเส้นตรงที่

$$\min \sum (y_i - \hat{y}_i)^2$$

เส้นตรงเขียนอยู่ในรูปสมการแบบ linear equation คือ

$$Y = bX + a$$

ค่า b และ a คำนวณได้จาก

$$b = \frac{\sum(x_i - \bar{x})\ (y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

$$a = \bar{Y} - b\,\bar{X}$$

โดย

Y คือตัวแปรตาม คือค่าที่เราสนใจเป็นหลัก และต้องการทำนายค่า ในตัวอย่างนี้คือระยะทางที่เดินได้

X คือตัวแปรต้น คือตัวแปรที่ส่งผลกับค่าของ Y ในตัวอย่างนี้คือ เวลาที่ใช้ในการเดิน

a คือ Intercept หรือจุดตัดแกน Y หรือค่า bias

b คือ weight หรือค่าความชันของเส้น หรือค่าของ x ทุก 1 หน่วยที่เปลี่ยนไป ที่ส่งผลต่อ y

จากข้อมูลการเดินนี้สามารถสร้างสมการ linear regression ได้ ดูในตัวอย่าง 4.1

แต่ก่อนที่จะดูตัวอย่างการหาสมการ linear regression เราดูการหาค่าทางสถิติที่เป็นตัวบอกความเหมาะสมของโมเดลที่ใช้วิเคราะห์กันก่อน คือค่า Coefficient of determination (R^2) และค่า Mean squared error (MSE)

4.2.2 Coefficient of determination (R^2 - R-squared)

Coefficient of determination คือ ค่าสัมประสิทธิ์แสดงการตัดสินใจ หรืออธิบายง่ายๆว่าเป็นค่าที่ใช้ตรวจสอบว่าข้อมูล 2 ตัวที่เรานำมาหาความสัมพันธ์นี้ มีความสัมพันธ์กันในเชิงเส้นตรงหรือไม่ โดยใช้ค่า R^2 (R-squared)

ค่า R^2 มีค่าอยู่ระหว่าง 0 – 1 แต่โดยปกติจะรายงานเป็นเปอร์เซ็นต์ ค่า R^2 ยิ่งสูงแสดงว่าข้อมูลนั้นมีความสัมพันธ์กันเชิงเส้นตรงมาก นั่นคือส่งผลให้การประมาณการณ์มีความถูกต้องมาก

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

โดยที่

$$SS_{res} = \sum_i^n (y_i - \hat{y}_i)^2 \text{ หรือคือค่า } \sum_i^n (error)^2$$

$$SS_{tot} = \sum_i^n (y_i - \bar{y})^2$$

SS_{res} คือ residual sum of squares

SS_{tot} คือ total sum of squares

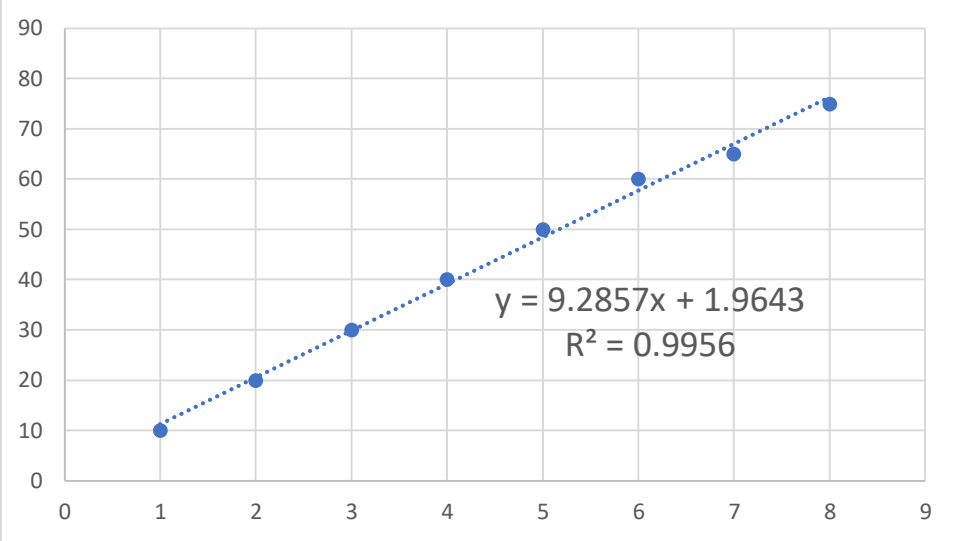
ถ้าค่า R^2 มีค่าน้อยมาก แสดงว่าข้อมูล 2 ชุดนี้ไม่มีความสัมพันธ์กันในเชิงเส้นตรง แสดงว่าเราไม่ควรใช้ simple linear regression ในการวิเคราะห์ข้อมูล อาจเป็นไปได้ว่าต้องเพิ่มตัวแปรอื่น ๆ เช่นใช้ multiple linear regression หรือใช้ regression รูปแบบอื่น ๆ เลยก็ได้

ข้อมูลที่มีความสัมพันธ์กันในเชิงเส้นตรงอาจสัมพันธ์แบบตามกัน หรือแบบผกผันก็ได้

ตัวอย่างข้อมูล กราฟ สมการเชิงเส้น และ ค่า R^2 แบบต่างๆ

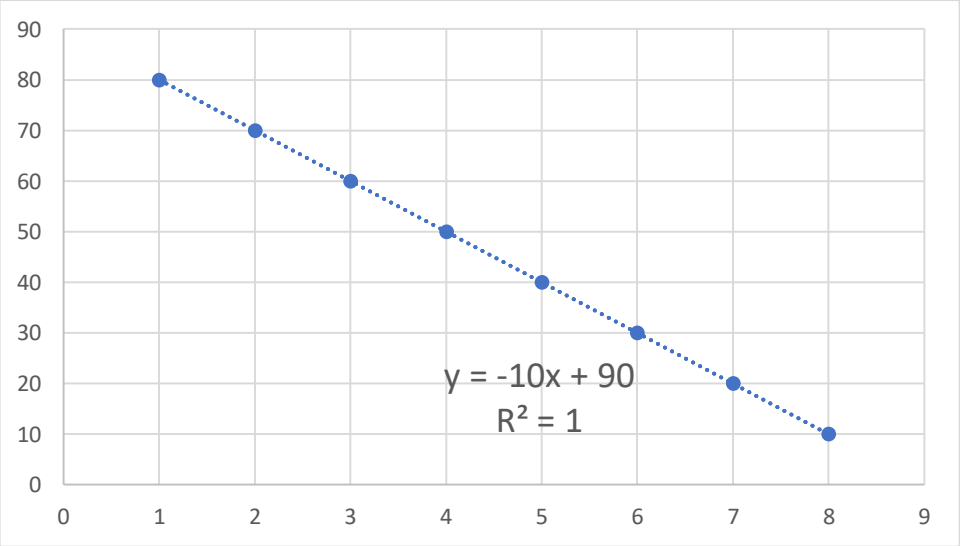
1. ข้อมูลที่สัมพันธ์เชิงเส้นแบบตามกัน

Y	X
10	1
20	2
30	3
40	4
50	5
60	6
65	7
75	8



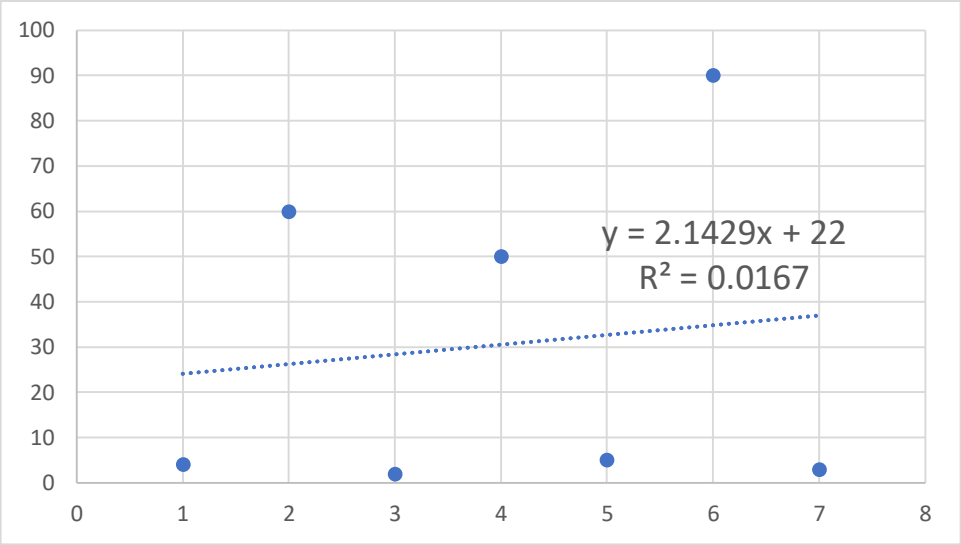
2. ข้อมูลที่สัมพันธ์เชิงเส้นแบบผกผัน

Y	X
10	1
20	2
30	3
40	4
50	5
60	6
65	7
75	8



3. ข้อมูลที่ไม่มีความสัมพันธ์เชิงเส้น

Y	X
4	1
60	2
2	3
50	4
5	5
90	6
3	7
8	8



4.2.3 การคำนวณความผิดพลาด mean squared error

การคำนวณค่าความผิดพลาด หรือ error เป็นการตรวจสอบว่ารูปแบบการวิเคราะห์ที่ใช้วิเคราะห์ข้อมูล (โมเดล – model) มีความแม่นยำมากน้อยเพียงใด ถ้า model นั้นมีความแม่นยำสูงค่าความผิดพลาดต้องมีค่าน้อย ค่าที่ใช้ตรวจสอบความผิดพลาดมีหลายค่า ในที่นี้เราจะใช้ค่า mean squared error (MSE) ซึ่งสามารถคำนวณได้จาก

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

โดยที่ n คือจำนวนข้อมูล

y_i คือค่าข้อมูลจริง

$\hat{y}_i$ คือค่าที่ได้จากการประมาณการณ์

ถ้าพิจารณาดี ๆ ค่า MSE คือการหาค่าเฉลี่ยของ $(error)^2$ นั่นเอง หรืออธิบายง่าย ๆ มันคือความแตกต่างระหว่างค่าจริงและค่าที่ได้จากการทำนาย

4.2.4 สรุปขั้นตอน linear regression

ที่เห็นมาทั้งหมดอาจดูเหมือนเยอะ เพราะเราแสดงให้เห็นถึงแนวคิดการทำ linear regression แต่ที่จริงแล้วการทำ linear regression ไม่ได้ยากขนาดนั้น อาจต้องคำนวณเยอะ แต่ไม่ได้ยากอย่างที่กลัว สรุปแนวคิดง่ายๆคือ

1. มีข้อมูลที่เราสนใจ ต้องการพยากรณ์ค่า ข้อมูลนี้คือ x และ y

2. สร้างโมเดล หรือสมการ Linear regression เพื่อใช้ในการพยากรณ์ จากข้อมูล x และ y ที่มี
   สร้างสมการ $y = bx + a$ โดยคำนวณจาก

$$b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$ และ

$$a = \bar{y} - b\,\bar{x}$$

นั่นคือถ้าเรารู้ค่า b และ a แล้ว และเรามีค่า x เราสามารถทำนายค่า y ได้

3. คำนวณค่า R^2 เพื่อวิเคราะห์ว่าข้อมูลมีความสัมพันธ์กันในเชิงเส้นตรงหรือไม่

ค่า $\boldsymbol{R^2}$ คำนวณจาก $$\boldsymbol{R^2 = 1 - \frac{SS_{res}}{SS_{tot}}}$$

โดยที่

$$\boldsymbol{SS_{res} = \sum_i (y_i - \hat{y}_i)^2 = \sum_i (error)^2}$$

$$\boldsymbol{SS_{tot} = \sum_i (y_i - \bar{y})^2}$$

4. คำนวณค่าความผิดพลาด (Mean squared error) เพื่อวิเคราะห์ว่าโมเดลนี้เหมาะสมกับข้อมูลชุดนี้หรือไม่

5. พยากรณ์หาค่า y จาก x ที่ต้องการ

หมายเหตุ ค่า R^2 และ MSE ไม่ใช่ค่าที่นำมาสร้างโมเดลการวิเคราะห์โดยตรง แต่เป็นค่าที่บอกความเหมาะสมของโมเดล

ตัวอย่าง 4.1 จากข้อมูลการเดิน ให้ทำงานต่อไปนี้

a. สร้างสมการ linear regression
b. คำนวณค่า coefficient of determination (R^2)
c. คำนวณค่า MSE และพิจารณาว่า สมการ regression นี้เหมาะสมหรือไม่
d. พยากรณ์ว่า ถ้าใช้เวลา 100 และ 150 วินาที จะเดินได้กี่เมตร

วิธีทำ จากข้อมูลการเดิน สามารถคำนวณค่าต่างๆเพื่อสร้าง linear equation ได้ดังนี้

	ระยะทาง (y)	เวลา (x)	$(X - \bar{X})(Y - \bar{Y})$	$(X - \bar{X})^2$
	7	5	325.6	484
	20	15	21.6	144
	12	25	19.6	4
	32	35	81.6	64
	38	55	453.6	784
average	**21.8**	**27**		
sum			**902**	**1480**

ต้องการสร้าง linear regression equation $Y = bX + a$

โดยที่

$$b = \frac{\sum(x_i - \bar{x})\,(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} = \frac{902}{1480} = 0.609$$

$$a = \bar{Y} - b\,\bar{X} = 21.8 - (0.609 * 27) = 5.357$$

a. linear regression equation คือ

$$Y = 0.609X + 5.357$$

เมื่อได้สมการแล้ว สามารถคำนวณค่า R^2 ได้ดังนี้

	ระยะทาง (y)	เวลา (x)	$(Y - \bar{Y})^2$	$\hat{y}$	$(y - \hat{y})^2$
	7	5	219.04	8.402	1.966
	20	15	3.24	14.492	30.338
	12	25	96.04	20.582	73.651
	32	35	104.04	26.672	28.388
	38	55	262.44	38.852	0.726
average	21.8	27			
sum			684.8	109	135.069

b. ค่า R^2 คำนวณจาก

$$R^2 = 1 - \frac{135.069}{684.8} = 0.80$$

c. คำนวณค่า MSE

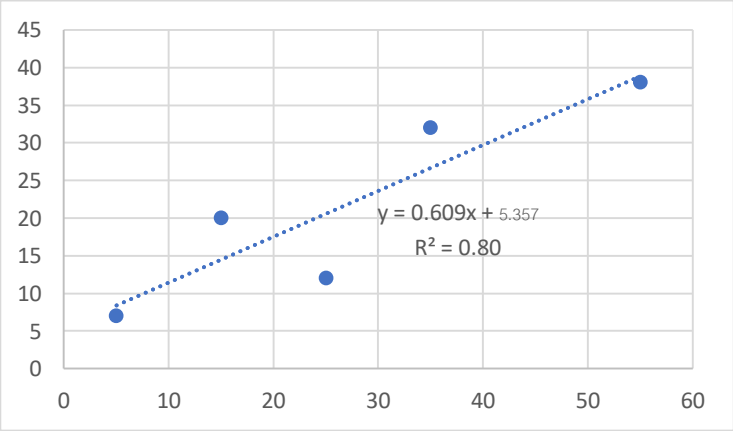
$$MSE = \left(\frac{1}{5}\right) 135.069 = 27.014$$

ค่า R^2 แสดงถึงความสัมพันธ์ระหว่างข้อมูล x และ y ชุดนี้ คือมีความสัมพันธ์ในเชิงเส้นตรง 80% และมีค่า MSE = 27 (อาจต้องเปรียบเทียบกับค่า error ที่ได้จากโมเดลรูปแบบอื่น ๆ) อย่างไรก็ตามข้อมูลชุดนี้มีความสัมพันธ์เชิงเส้นตรง 80% ซึ่งยอมรับได้

d. ทำนายค่าของ Y เมื่อ X มีค่าต่างๆ

ถ้าใช้เวลา 100 วินาที จะเดินได้ 0.609(100) + 5.357 = 66.257 เมตร

ถ้าใช้เวลา 150 วินาที จะเดินได้ 0.609(150) + 5.357 = 96.707 เมตร



ตัวอย่าง 4.2 จากตารางข้อมูลการวิ่งด้านล่าง ให้ทำงานต่อไปนี้

a. หา linear regression equation

b. คำนวณค่า R2 และค่า MSE

c. ในเวลา 20 และ 40 วินาที จะวิ่งได้ระยะทางกี่เมตร

d. ถ้าต้องการวิ่งได้ระยะทาง 500 เมตร ต้องใช้เวลาวิ่งกี่วินาที

คนที่	ระยะทาง (เมตร)	เวลา (วินาที)
1	5	3
2	9	2
3	12	7
4	11	2
5	12	6
6	10	4
7	8	5
8	4	3
9	25	9
10	30	10

วิธีทำ

จากข้อมูล สามารถคำนวณค่าสำหรับการสร้างสมการได้ดังนี้

คนที่	ระยะทาง (เมตร) Y	เวลา (วินาที) X	$(X-\bar{X})(Y-\bar{Y})$	$(X-\overline{X})^2$
1	5	3	15.96	4.41
2	9	2	11.16	9.61
3	12	7	-1.14	3.61
4	11	2	4.96	9.61
5	12	6	-0.54	0.81
6	10	4	2.86	1.21
7	8	5	0.46	0.01
8	4	3	18.06	4.41
9	25	9	48.36	15.21
10	30	10	85.26	24.01
average	**12.6**	**5.1**		
sum			**185.4**	**72.9**

เราต้องการสร้าง linear regression equation $Y = bX + a$

โดยที่

$$b = \frac{\sum(x_i-\bar{x})\,(y_i-\bar{y})}{\sum(x_i-\bar{x})^2} = \frac{185.4}{72.9} = 2.543$$

$$a = \bar{Y} - b\,\bar{X} = 12.6 - (2.54 * 5.1) = -0.369$$

a. linear regression equation คือ

$$Y = 2.543X - 0.369$$

b. คำนวณค่า R^2 และ MSE

คนที่	ระยะทาง (เมตร) Y	เวลา (วินาที) X	$(Y - \bar{Y})^2$	$\hat{y}$	$(y - \hat{y})^2$
1	5	3	57.76	7.26	5.108
2	9	2	12.96	4.717	18.344
3	12	7	0.36	17.432	29.507
4	11	2	2.56	4.717	39.476
5	12	6	0.36	14.889	8.346
6	10	4	6.76	9.803	0.039
7	8	5	21.16	12.346	18.888
8	4	3	73.96	7.26	10.628
9	25	9	153.76	22.518	6.16
10	30	10	302.76	25.061	24.394
average	**12.6**	**5.1**			
sum			**632.4**	**126.003**	**160.89**

คำนวณค่า R^2

$$R^2 = 1 - \frac{160.89}{632.4} = 0.745$$

คำนวณค่า MSE

$$MSE = \left(\frac{1}{10}\right)(160.89) = 16.089$$

c. ในเวลา 20 และ 40 วินาที จะวิ่งได้ระยะทางกี่เมตร

เวลา 20 วินาที วิ่งได้ระยะทาง Y = 2.543(20) – 0.369 = 50.43 เมตร

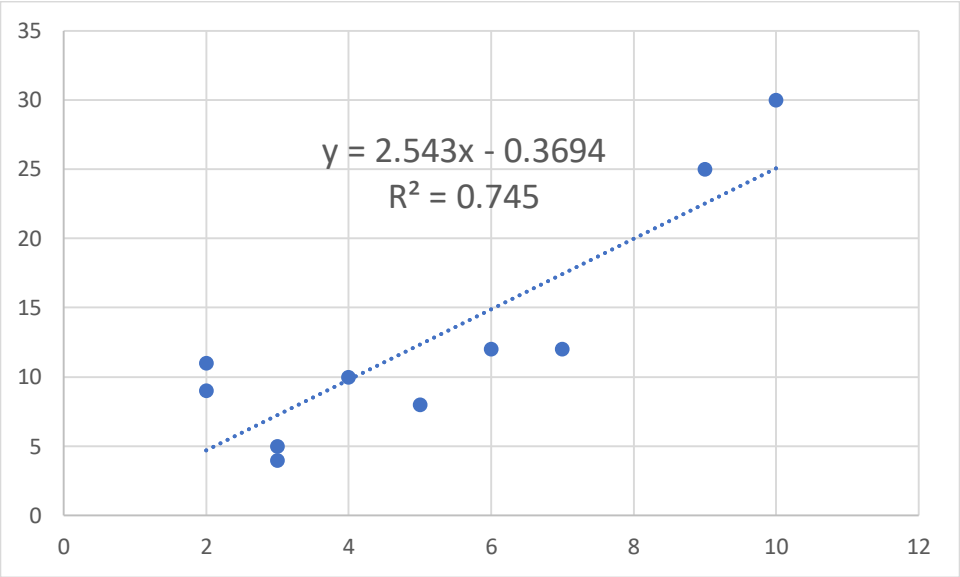
เวลา 40 วินาที วิ่งได้ระยะทาง Y = 2.543(40) – 0.369 = 101.35 เมตร

d. ถ้าต้องการวิ่งได้ระยะทาง 500 เมตร ต้องใช้เวลาวิ่งกี่วินาที

500 = 2.543(X) – 0.369

X = (500 + 0.369) / 2.543 = 196.763 วินาที

กราฟแสดงข้อมูลการวิ่งและ linear equation



**** ทำแบบฝึกหัดข้อ 1. (คำนวณเองไม่ใช้ python ยังไม่ต้องวาดกราฟ)**

4.3 python และ regression

โดยปกติขั้นตอนการวิเคราะห์ regression ไม่ว่าจะเป็น regression แบบใดก็ตาม (simple linear regression, multiple linear regression หรือ logistic regression) มี 3 ขั้นตอนคือ

1. **การเทรนข้อมูล** (training stage) ขั้นตอนนี้เป็นการนำข้อมูลที่มีมาวิเคราะห์ จะทำให้ได้โมเดลเพื่อใช้ในการทำนายผล
2. **การเทสข้อมูล** (testing stage) เป็นการนำโมเดลที่ได้จากขั้นตอนการเทรนมาตรวจสอบกับข้อมูลอีกชุดที่มีอยู่ เพื่อวิเคราะห์ว่าโมเดลที่ได้จากการเทรนด้วย regression รูปแบบนี้เหมาะสมหรือไม่ โดยใช้ค่า R^2 หรือ MSE ประกอบการพิจารณา
3. **การทำนายข้อมูลที่ต้องการ** (Prediction) นำโมเดลที่ได้จากการเทรนมาทำนายค่าข้อมูลที่ต้องการ

ข้อมูลที่มีจะถูกแบ่งออกเพื่อใช้ในการเทรนและเทสตามอัตราส่วนที่เหมาะสม เช่นแบ่งเป็น 80% เพื่อเทรน และ 20% เพื่อเทส

ตัวอย่าง ถ้ามีข้อมูลการเดินของคน 100 คน

คนที่	ระยะทาง (เมตร) Y	เวลา (วินาที) X
1	5	3
2	9	2
...		
81	10	4
...		
100	30	10

ใช้ข้อมูลของ 80 คนในการเทรนโมเดล

ใช้ข้อมูลของ 20 คนเพื่อเทสโมเดล

เมื่อได้โมเดลแล้ว นำโมเดลนั้นมาพยากรณ์ข้อมูลที่ต้องการ เช่น ต้องการรู้ว่า ในเวลา 20, 50, 180 นาที จะเดินได้ระยะทางกี่เมตร

สำหรับคอสนี้ เพื่อให้การทำงานไม่ซับซ้อน **เราจะไม่แบ่งข้อมูลออกเป็น training data และ testing data** แต่จะใช้ข้อมูลทั้งหมดเพื่อเทรน ได้โมเดล และนำโมเดลนั้นมาทำนายผล

4.3.1 ขั้นตอนการเขียน python เพื่อวิเคราะห์ regression

* ควรตั้งชื่อตัวแปรตามที่กำหนด ☺ *

step1: import packages

เราใช้ sklearn package ในการวิเคราะห์ regression

step2: อ่าน data

อ่านข้อมูลอาจเป็น numpy array หรืออ่านข้อมูลจาก csv file สร้างตัวแปรคือ

x_train เป็น array ขนาด (n_samples, n_features)

y_train เป็น array ขนาด (n_samples) โดย

- n_samples คือจำนวนข้อมูลที่มี หรือที่ต้องการนำมาเทรน
- n_features คือจำนวนตัวแปรต้นหรือ attribute กรณีของ simply regression มี 1 ตัวแปร

step3: สร้าง regression model

สร้าง model ตามรูปแบบการวิเคราะห์ เช่น LinearRegression

ใช้ model ทำการเทรน ค่า x_train, y_train ด้วยฟังก์ชัน model.fit(x_train, y_train)

step4: วิเคราะห์ผลความถูกต้องของโมเดล

เมื่อเทรนเสร็จจะได้ค่า

model.coef_ คือค่า coefficient หรือค่า b

model.intercept_ คือค่า intercept หรือค่า a

regression equation คือ $Y = bX + a$

นำโมเดลที่ได้จากการเทรน มาทำนายค่า x_train ข้อมูลที่ได้จากการทำนายคือตัวแปร y_pred

y_pred = model.predict(x_train)

เพื่อใช้คำนวณค่า R^2 และ MSE ของข้อมูล Training data คือ

r2 = r2_score(y_train, y_pred) คือค่า R^2

mse = mean_squared_error(y_train, y_pred) คือค่า MSE

step5: ทำนายค่า unseen data

ถ้ามีค่าข้อมูลอื่นๆที่ต้องการทำนาย สามารถสร้างตัวแปร x_new

x_new เป็น array ขนาด (x_samples, n_features) โดย x_samples คือจำนวยนข้อมูลใหม่ที่ต้องการให้ทำนายผล

และทำนายค่าโดยใช้คำสั่ง model.predict(x_new)

step6: plot graph

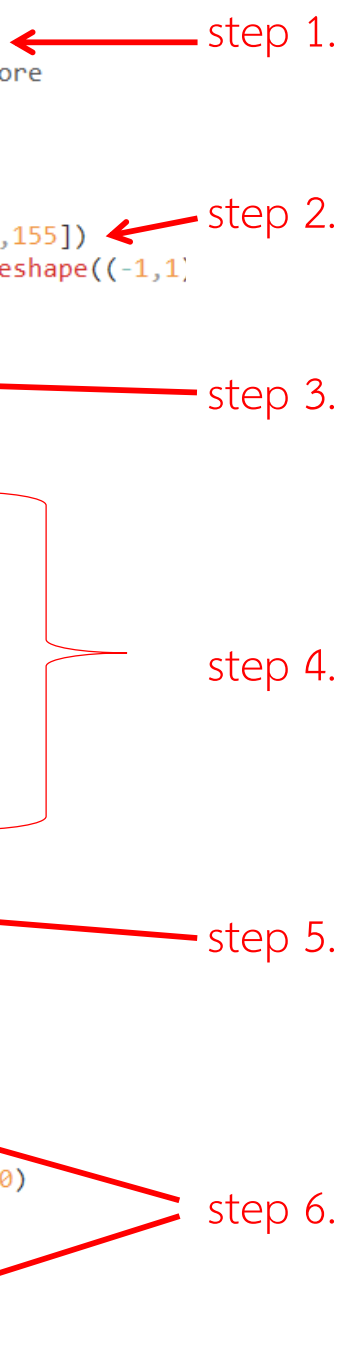
สร้างกราฟเพื่อแสดงข้อมูลทั้งหมด ข้อมูลที่ทำนาย เส้น regression

ตัวอย่าง 4.3 มีข้อมูลการวิ่งจำนวน 10 records เป็นความสัมพันธ์ระหว่างเวลาและระยะทาง เขียน python เพื่อ

a. สร้าง linear regression equation และ แสดง output สมการ ด้วยรูปแบบ $Y = bX + a$
b. คำนวณและ output ค่า R^2 และ MSE ของ Training data
c. วิเคราะห์ค่า R^2 และ MSE ของโมเดล ข้อมูลชุดนี้เหมาะสมจะใช้ linear regression หรือไม่
d. สร้างกราฟแสดงข้อมูลทั้งหมด x, y เป็นจุดสีเขียว และ เส้น linear regression เป็นเส้นสีน้ำเงิน
e. พยากรณ์ว่าถ้าใช้เวลา 80, 100 และ 150 วินาที จะวิ่งได้ระยะทางเท่าใด
f. **คำถามพิเศษ** (มีคะแนนพิเศษให้)
   - ตัวแปร y_pred คือค่าอะไรตอนเราคำนวณ linear regression แบบ manual
   - บรรทัดที่ 39 เราสามารถคำนวณค่า y_pred โดยไม่ใช้คำสั่ง
     y_pred = model.predict(x_new) ได้อย่างไร

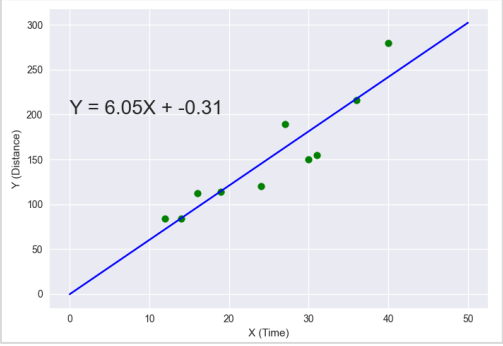
ระยะทาง (เมตร) (Y)	เวลา (วินาที) (X)
280	40
84	12
150	30
120	24
189	27
84	14
114	19
216	36
112	16
155	31

```python
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
plt.style.use('seaborn')

#step2: read data
y_train = np.array([280,84,150,120,189,84,114,216,112,155])
x_train = np.array([40,12,30,24,27,14,19,36,16,31]).reshape((-1,1)

#step3: train Regression model
model = LinearRegression()
model.fit(x_train, y_train)

#step4: analyse model
b = model.coef_
a = model.intercept_
print("Coefficient(b)\t : %.2f" % (b))
print("Intercept(a)\t : %.2f" % (a))
print("Linear eqauation:\t Y = %.2fX + %.2f" % (b,a))

y_pred = model.predict(x_train)
r2 = r2_score(y_train, y_pred)
MSE = mean_squared_error(y_train, y_pred)
print("R2\t : %.2f" % (r2))
print("MSE\t : %.2f" % (MSE))

#step5: predict unseen data
x_new =np.array([80,100,150]).reshape(-1,1)
y_pred_new = model.predict(x_new)
print("\nPredicted response of X:")
for i in range(x_new.shape[0]):
    print("%i \t %.2f" %(x_new[i][0], y_pred_new[i]))
#step6: draw regression graph
plt.scatter(x_train, y_train,  color='green')
plt.plot(x_train.flatten(), y_pred, color='blue')
plt.text(0,200,"Y = %.2fX + %.2f" % (b,a), fontsize = 20)

plt.xlabel('X (Time)')
plt.ylabel('Y (Distance)')
plt.show()

#step6: draw regression graph of differen range
x1 = np.linspace(0,50,2).reshape(-1,1)
y_pred_1 = model.predict(x1)

plt.scatter(x_train, y_train,  color='green')
plt.plot(x1, y_pred_1, color='blue')
plt.text(0,200,"Y = %.2fX + %.2f" % (b,a), fontsize = 20)

plt.xlabel('X (Time)')
plt.ylabel('Y (Distance)')
plt.show()
```



ขั้นตอน	การทำงาน
1.	import library ที่ใช้ในการทำงาน เราใช้ sklearn เพื่อคำนวณ linear regression และค่า R^2, MSE
2.	สร้าง numpy array เก็บค่า x_train และ y_train คือข้อมูลทั้งหมด ตัวอย่างนี้มี 10 คน กรณีของ x_train ต้องการ array ขนาด (10,1) หรือ 10 row, 1 column ดังนั้น reshape(-1,1) จะได้ x_train ที่มี shape คือ [[40] [12] .... [31]]
3.	**การเทรน** สร้าง model แบบ linear regression และ เทรนข้อมูล x_train, y_train เมื่อเทรนแล้ว จะได้ model หรือ regression equation ในการวิเคราะห์ข้อมูลชุดนี้
4.	เมื่อเทรนเสร็จ จะได้โมเดล ซึ่งบอก regression equation นำโมเดลมาหาค่า R^2 และ MSE เพื่อวิเคราะห์ความเหมาะสมของโมเดล
5.	**ทำนายค่า unseen data** หรือพยากรณ์ค่า y จากค่า x ที่ต้องการ กรณีนี้เราต้องการเช็คว่าจะวิ่งได้ระยะทางเท่าไร ถ้าใช้เวลา 80, 100 และ 150 วินาที
6.	**สร้างกราฟ prediction** แสดงจุด x, y และเส้น linear regression equation เราทำตัวอย่างให้ดูทั้ง 2 แบบ แบบแรก ถ้าต้องการวาดเส้น regression จากค่า x_train, y_pred เลย x_train.flatten() เพื่อทำ x_train ให้กลับเป็น array 1มิติ แบบที่สอง บรรทัด 46 กำหนดค่า x ในช่วง [0, 50] ถ้าต้องการเส้น regression จาก 0 – 50 x1 = np.linspace(0,50,2) ทำให้ได้ค่า x1 คือ [0, 50] ดังนั้นต้อง reshape x1 เพื่อนำไป predict หาค่า y หมายเหตุ ถ้าต้องการวาดเส้นกราฟ prediction ที่จุดอื่น ๆ สามารถเปลี่ยนค่า x1 ได้ เช่น x1 = np.linspace(0,100,2) เพื่อวาดเส้น prediction ที่ x1 = 0 ถึง 100

```
Coefficient(b)    : 6.05
Intercept(a)      : -0.31
Linear eqauation:     Y = 6.05X + -0.31
R2    : 0.85
MSE   : 515.37

Predicted response of X:
80   483.89
100      604.94
150      907.57
```



ฝึกทำ lab python

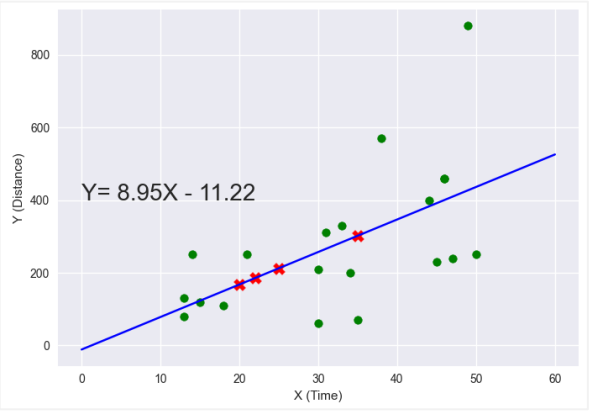
ตัวอย่าง 4.4 ให้เขียน python เพื่อวิเคราะห์ข้อมูลค่า X, Y ที่มี โดย

1. output: สมการ linear regression
2. output: ค่า R^2 และ MSE ของ Training data
3. ทำนายว่า ถ้า X เป็น 20, 22, 25 และ 35 ค่า Y เป็นเท่าไร
4. วาดกราฟแสดง
    a. ข้อมูลการวิ่งทั้งหมดเป็นจุดสีเขียว,
    b. เส้น linear regression สีน้ำเงิน โดยแสดงจากจุด x = 0 ถึง 20
    c. จุดที่ได้จากการทำนายค่า Y ในข้อ 6. เป็นกากบาทสีแดง (จุดทุกจุดนี้ควรอยู่บนเส้นสีน้ำเงิน)
    d. สมการ linear regression

X	34	44	18	31	13	45	15	46	13	47
Y	200	400	110	310	130	230	120	460	80	240

X	46	30	30	50	38	14	21	35	33	49
Y	460	210	60	250	570	250	250	70	330	880

```
Coefficient(b)    : 8.95
Intercept(a)      : -11.22
Linear eqauation:     Y = 8.95X + -11.22
R2    : 0.34
MSE   : 24715.26

Predicted response of X:
20    167.75
22    185.65
25    212.49
35    301.98
```



ตัวอย่าง 4.5 การสร้าง linear regression โดยการอ่านข้อมูลจากไฟล์ประเภท .csv

จากตัวอย่างที่ผ่านมา มีข้อมูลเพียง 20 records ทำให้เราสร้าง numpy array ได้เลย แต่ส่วนมากข้อมูลที่นำมาวิเคราะห์มีจำนวนมาก อาจเป็น 1000 record ดังนั้นเราจะเก็บข้อมูลไว้ในไฟล์ประเภท .csv เราจึงต้องอ่านข้อมูลจากไฟล์ ข้อมูลประสบการณ์ทำงานและเงินเดือนของพนักงานจำนวนหนึ่งเก็บไว้ในไฟล์ **data/salary.csv**

คำสั่ง

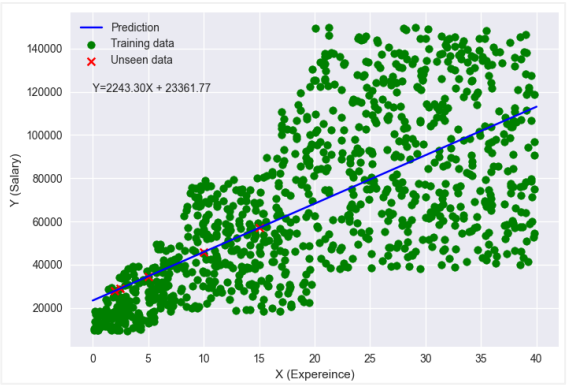
a. เขียน python code ส่วนที่เหลือให้สมบูรณ์เพื่อวิเคราะห์ข้อมูลจากไฟล์ salary.csv
b. แสดงสมการ linear regression
c. แสดงค่า R^2 และ MSE วิเคราะห์ว่าข้อมูลชุดนี้ควรทำ linear regression หรือไม่ พยากรณ์ว่า ถ้าประสบการณ์ทำงาน {2, 2.5, 5, 10, 15} ปี คาดหวังว่าจะได้เงินเดือนเท่าไร
d. สร้างกราฟโดยแสดงข้อมูลทั้งหมด, เส้น linear regression และ สมการบนกราฟ
e. ถ้าพนักงานคนหนึ่งได้เงินเดือน 50000 บาท คิดว่าพนักงานคนนี้มีประสบการณ์ทำงานกี่ปี

ตัวอย่างข้อมูลจากไฟล์ salary.csv

experience	salary
11	46576
16.2	82888
24.6	145595
29.6	121111
10.6	42415
11	46576
16.2	82888

```python
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import pandas as pd
np.set_printoptions(precision=2)
plt.style.use('seaborn')

#อ่านข้อมูลจากไฟล์ salary.csv ข้อมูลเก็บเป็น DataFrame
# head() แสดงข้อมูล 5 records แรก
df = pd.read_csv("data/salary.csv")
print(df.head())

# x คือ experience, y คือ salary
x_train = df[['experience']]
y_train = df['salary']

#จำนวน record ทั้งหมด
n = x_train.shape[0]

#print detail of Train/Test
print("There are %i records." % n)
```

```
   experience  salary
0        11.0   46576
1        16.2   82888
2        24.6  145595
3        29.6  121111
4        10.6   42415
There are 1000 records.
LinearRegression equation: Y = 2243.30X + 23361.77
R2: 0.46
MSE: 788047186.88

Predicted response of some experience years

2.0      27848.37
2.5      28970.02
5.0      34578.27
10.0     45794.77
15.0     57011.27
Person with salary 50000 B. have experience 11.87 years
```



* สังเกตว่าค่า MSE สูงมาก เพราะเราไม่ได้ทำ data normalization ... *

**** ทำแบบฝึกหัดข้อ 1. , 2. และ 3.**

4.4 Multiple linear regression

Multiple linear regression ใช้เมื่อมีตัวแปรมากกว่า 1 ตัวที่ส่งผลกับข้อมูลหลักที่เราสนใจ เช่น ข้อมูลการเดิน ถ้า simple linear regression คือมีตัวแปรเดียวที่ส่งผลกับระยะทางที่เดินได้ คือ เวลาที่ใช้เดิน แต่ในความเป็นจริงตัวแปรอื่น ๆ ก็ส่งผลกับการเดิน เช่น อุณหภูมิ อายุ น้ำหนัก เป็นต้น กรณีนี้ เราต้องใช้ multiple linear regression ในการสร้างสมการ

แนวคิดของการทำ multiple linear regression คล้ายกับ simple linear regression เพียงแต่ในเมื่อมีตัวแปรมากขึ้น ค่าที่คำนวณก็มากขึ้นด้วย

สมการของ multiple linear regression คือ

$$Y = b_1X_1 + b_2X_2 + b_3X_3 + \cdots + b_nX_n + a$$

* ไม่ต้องตกใจ เราจะคำนวณ multiple linear regression โดยใช้ python

แนวคิดการเขียน python เพื่อทำ Multiple linear regression มีขั้นตอนเหมือนกับ simple linear regression สิ่งที่แตกต่างคือ multiple linear regression มีตัวแปรต้นมากกว่า 1 ตัว นั้นคือ array x มีขนาดเป็น (n_samples, n_features) โดย n_features คือจำนวนของตัวแปรต้นนั่นเอง เช่น

ถ้ามีข้อมูลการเดินของคน 10 คน และตัวแปรที่ส่งผลกับการเดินคือ เวลา อุณหภูมิ อายุ และน้ำหนัก array x จะมีขนาดเป็น (10,4)

และเมื่อวิเคราะห์ด้วย LinearRegression model แล้ว จะมีค่า coef_ 4 ค่า คือ (b_1, ..., b_4)

ตัวอย่าง 4.6 ไฟล์ walk.csv เก็บข้อมูลการเดิน ประกอบด้วย ตัวแปรตามคือ ระยะทางที่เดินได้ ตัวแปรต้นคือเวลา อุณหภูมิ อายุ และน้ำหนักของผู้เดิน เขียน python เพื่อสร้าง multiple linear regression

ตัวอย่างข้อมูล

distance (ระยะทาง)	time (เวลา)	temperature (อุณหภูมิ)	age (อายุ)	weight (น้ำหนัก)
100	65	38	25	90
50	35	38	18	100
20	20	26	20	90
50	25	25	30	50
60	30	22	50	50
80	45	25	10	80

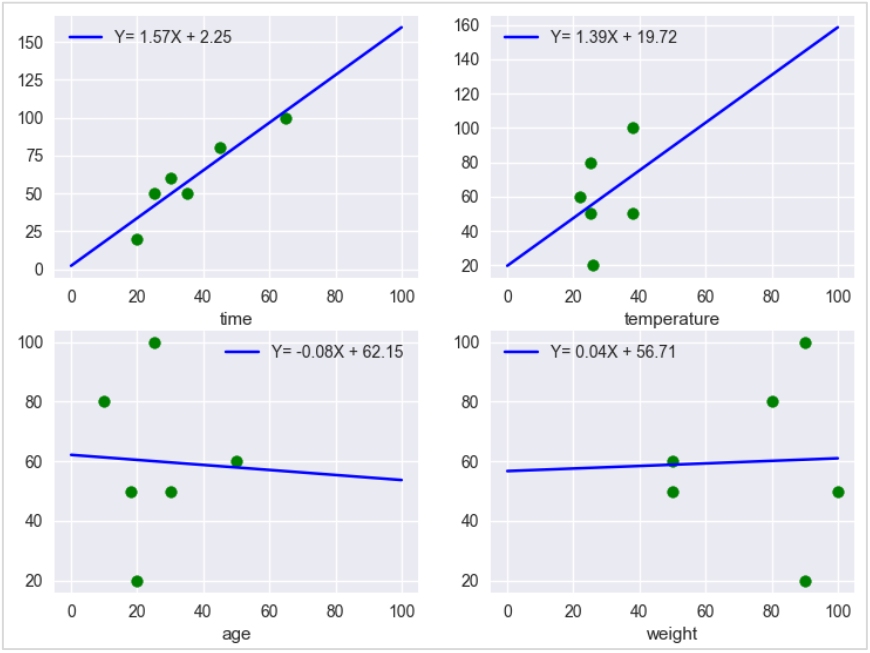
ตอบคำถามต่อไปนี้

1. สมการ multiple linear regression คือ _____
2. ค่า R^2 และ ค่า MSE คือ _____
3. สมการนี้สามารถอธิบายความสัมพันธ์ระหว่างข้อมูลชุดนี้ได้ดีหรือไม่
4. จากค่า b_i ทั้ง 4 ค่า อธิบายความสัมพันธ์ของข้อมูลนี้ได้อย่างไร
5. หากต้องการวิเคราะห์อย่างละเอียดว่าตัวแปรแต่ละค่าส่งผลกับระยะทางจริงหรือไม่ ควรทำอย่างไร ให้หาสมการ linear regression ของแต่ละตัวแปร
6. สร้างกราฟของ simple linear regression ที่ได้จากข้อ 5.
7. ทำนายว่าข้อมูลด้านล่าง วิ่งได้ระยะทางเท่าไร โดยใช้ multiple linear regression

time	temperature	age	weight
200	30	25	53
150	30	20	65
20	25	30	55
50	20	22	85
60	25	15	80

```python
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import pandas as pd
plt.style.use('seaborn')

df = pd.read_csv("data/walk.csv")

y_train = df['distance']
x_train = df[['time','temperature','age','weight']]

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

```
All bi [ 1.80667986 -0.0887871  -0.21324092 -0.52453805]
Linear eqauation:    Y = 1.81Time - 0.09Temp - 0.21Age  - 0.52Weight + 41.98
R_squared    : 0.98
MSE  : 12.07

Predicted response of X:  [103.5   45.55  24.33  52.31  57.34  76.97]

+++ LinearRegression of each single X +++
Linear eqauation (x0):   Y = 1.57time + 2.25
R2   : 0.87
MSE  : 82.08
Linear eqauation (x1):   Y = 1.39temperature + 19.72
R2   : 0.13
MSE  : 552.31
Linear eqauation (x2):   Y = -0.08age + 62.15
R2   : 0.00
MSE  : 632.21
Linear eqauation (x3):   Y = 0.04weight + 56.71
R2   : 0.00
MSE  : 632.62
```



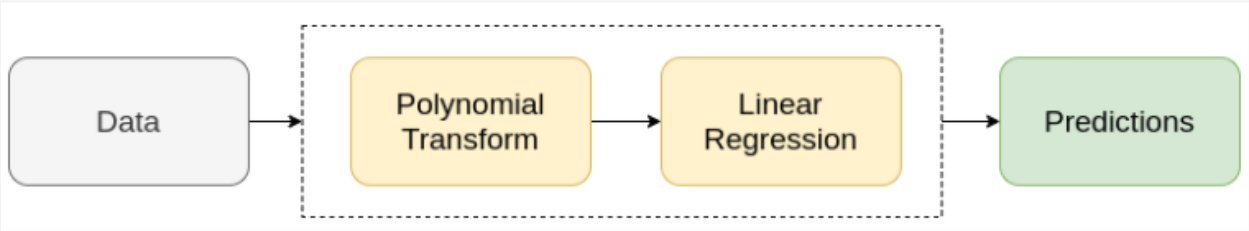
** ทำ Lab 4.

4.5 Polynomial regression

เป็นการทำ regression analysis ที่ความสัมพันธ์ระหว่างตัวแปรต้นและตัวแบบตามเป็นแบบเส้นโค้ง และไม่เหมาะสมจะทำแบบ linear regression สามารถเขียนสมการได้คือ

$$y = b_1 x_1 + b_2 x_2^2 + b_3 x_3^3 + \cdots + a$$

python for Polynomial regression



Pipeline การวิเคราะห์ด้วย Polynomial คือ transform ข้อมูลให้เป็น polynomial โดยกำหนด degree ที่ต้องการ แล้วทำการวิเคราะห์ด้วย Linear regression ตัวอย่าง Polynomial degree ต่าง ๆ เช่น

Polynomial degree	Polynomial equation
2	$y = b_1 x_1 + b_2 x_2^2 + a$
3	$y = b_1 x_1 + b_2 x_2^2 + b_3 x_3^3 + a$

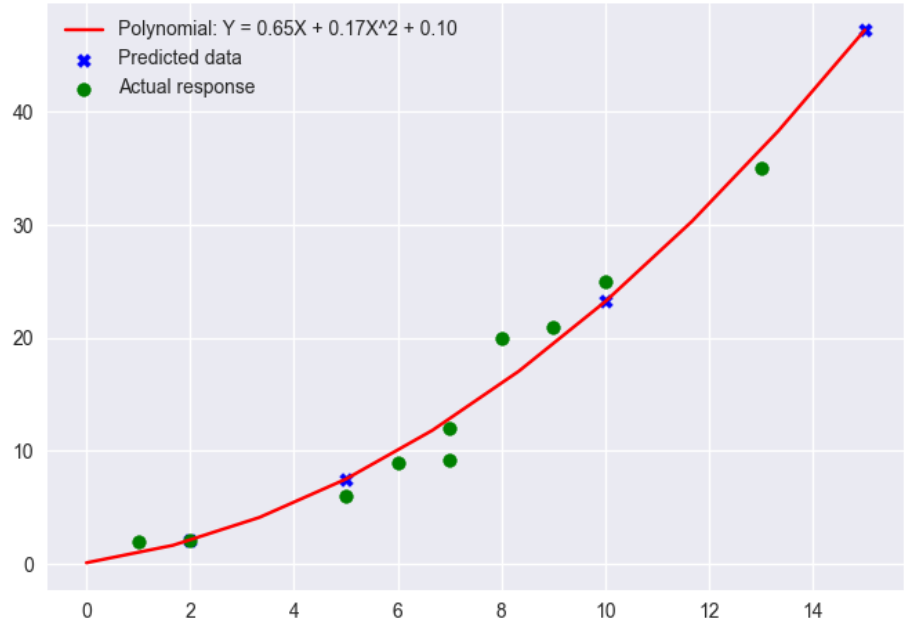
ตัวอย่าง 4.7 ไฟล์ polydata.csv เก็บข้อมูลค่า y และ x อ่านข้อมูลจากไฟล์เพื่อสร้างสมการ polynomial regression

ตัวอย่างข้อมูล

y	x
2	1
2.1	2
6	5
9	6
9.2	7
12	7
20	8

```python
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error, r2_score
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('seaborn')

df = pd.read_csv("data/polydata.csv")

x_train = df[['x']]
y_train = df['y']

poly = PolynomialFeatures(degree = 2)
x_poly = poly.fit_transform(x_train)
model_poly = LinearRegression()
model_poly.fit(x_poly,y_train)

b  = model_poly.coef_
a  = model_poly.intercept_
y_pred = model_poly.predict(x_poly)
r2 = r2_score(y_train, y_pred)
mse = mean_squared_error(y_train,y_pred)

print('slope (b):', b)
print('intercept (a):',a)
print("R_squared\t : %.2f" % (r2))
print("MSE\t : %.2f" % (mse))

x_new = np.array([2,10,20]).reshape(-1, 1)
y_pred = model_poly.predict(poly.fit_transform(x_new))
print("predict reponses for x_test\n", y_pred)

#plot graph
y_pred = model_poly.predict(poly.fit_transform(x_train))
plt.scatter(x_train, y_train,  color='green')
plt.plot(x_train,y_pred, color = 'red')
plt.xlabel('X ')
plt.ylabel('Y ')
plt.show()
```

```
slope (b): [0.         0.65068811 0.1661962 ]
intercept (a): 0.09918047714442046
R_squared    : 0.96
MSE  : 4.28
predict reponses for x_test
 [ 2.0653415  23.2256816  79.59142268]
```



ตอบคำถามต่อไปนี้

1. สมการ polynomial regression คือ _____
2. ถ้าเปลี่ยน degree = 3 จะได้สมการ polynomial คือ _____
3. ระหว่าง polynomial regression degree 2 และ degree 3 ข้อมูลชุดนี้ควรวิเคราะห์แบบใด เพราะเหตุใด
4. ใช้ข้อมูลนี้สร้าง linear regression และวิเคราะห์ว่าข้อมูลชุดนี้ควรทำ regression แบบใด เพราะเหตุใด

**** ทำ Lab 5., 6.**

4.6 Logistic regression

pip install seaborn: use for showing confusion matrix

รูปแบบการวิเคราะห์ทั้ง 3 แบบที่ผ่านมาทั้ง linear regression หรือ polynomial regression ใช้เมื่อตัวแปรตามเป็นข้อมูลแบบต่อเนื่อง กรณีที่ตัวแปรตาม เป็นข้อมูลแบบไม่ต่อเนื่อง เช่น สอบผ่านหรือไม่ (สอบผ่าน, สอบตก) เพศ (ชาย, หญิง) ประเภทดอกไม้ (กุหลาบ, มะลิ, ทานตะวัน) เราจะใช้การวิเคราะห์ข้อมูลแบบ logistic regression

Logistic regression แบ่งได้เป็น 3 ประเภทคือ

1. Binomial logistic regression ใช้เมื่อตัวแปรตามมีความเป็นไปได้ 2 ค่า เช่น สอบผ่าน/ตก เพศชาย/หญิง
2. Multinomial logistic regression ใช้เมื่อตัวแปรตามเป็นได้หลายค่าและไม่มีลำดับ เช่น ดอกกุหลาบ/มะลิ/ทานตะวัน อาหารที่ต้องการคือข้าวผัด/ผัดไท/ไข่เจียว/ต้มยำ
3. Ordinal Logistic regression ใช้เมื่อตัวแปรตามเป็นได้หลายค่าและมีลำดับ เช่น คะแนนรีวิวของหนังคือ 1 – 5 ดาว นิสิตเรียนอยู่ชั้นปี 1 -4

จะเห็นว่า ตัวแปรตาม (Y) เป็นข้อมูลแบบไม่ต่อเนื่อง และไม่จำเป็นต้องเป็นตัวเลข หรือถึงแม้จะเป็นตัวเลขแต่ตัวเลขเหล่านั้นไม่มีลำดับ เช่น 1 คือตกลง 0 คือปฏิเสธ เป็นต้น ดังนั้นลักษณะการทำ Logistic regression นี้เป็นเหมือนกับการจัดกลุ่มข้อมูล (Classification)

นอกจากนั้น เราจะใช้ค่าทางสถิติอื่นในการรายงานความถูกต้องของ Logistic regression คือ confusion matrix และรายงานผลเป็นค่าความแม่นยำของการทำนายผล (Accuracy)

4.6.2 การแสดง confusion matrix และคำนวณ accuracy

Confusion matrix คือตารางที่อธิบายความถูกต้องในการทำนายผล มีลักษณะตามตัวอย่างเช่น

ตัวอย่าง confusion matrix ขนาด 2x2 กรณีมี 2 คลาส สมมติข้อมูลเป็นคน มี 2 คลาสคือผู้หญิง และผู้ชาย

	Predicted: หญิง	Predicted: ชาย	Total
Actual: หญิง	30	25	55
Actual: ชาย	35	60	95
Total	65	85	**150**

หมายความว่า ข้อมูลจริงเป็นหญิง และโมเดลทำนายถูกว่าเป็นหญิง จำนวน 30 คน

หมายความว่า ข้อมูลจริงเป็นชาย และโมเดลทำนายถูกว่าคนเหล่านี้เป็นชาย จำนวน 60 คน

จากตัวอย่างนี้แสดงว่ามีข้อมูลทั้งหมด (n) = 30 + 25 + 35 +60 = 150 คน

Actual คือค่าจริงของข้อมูล กรณีคือ - เป็นผู้หญิง 30 +25 = 55 คน

- เป็นผู้ชาย 35 + 60 = 95 คน

Predicted คือการทำนายผลจากโมเดล ถ้า Actual และ Predict ตรงกันแสดงว่าทำนายถูก กรณีนี้คือ ทายผู้หญิงถูก 30 คน และทายผู้ชายถูก 60 คน นอกนั้นคือทายผิดหมด

25 คือ จริง ๆ เป็นผู้หญิง แต่ทำนายว่าเป็นผู้ชาย

35 คือ จริง ๆ เป็นผู้ชาย แต่ทำนายว่าเป็นผู้หญิง

Accuracy คือการบอกความแม่นยำของโมเดล ว่าสามารถทำนายผลถูกต้องกี่เปอร์เซ็นต์ คำนวณจาก

(จำนวนข้อมูลที่ทายถูก) / จำนวนข้อมูลทั้งหมด

กรณีนี้ accuracy = (30+60) / 150

= 0.6

= 60 %

ตัวอย่าง confusion matrix ขนาด 3x3 กรณีมี 3 คลาส

	Predicted: A	Predicted: B	Predicted: C	Total
Actual: A	**10**	3	1	14
Actual: B	0	**30**	**2**	32
Actual: C	2	**0**	**20**	22
Total	12	33	23	**68**

ตัวเลข 10 จาก Actual A, Predicted A หมายความว่า _____

ตัวเลข 3 จาก Actual A, Predicted B หมายความว่า _____

ตัวเลข 1 จาก Actual A, Predicted C หมายความว่า _____

ตัวเลข 0 จาก Actual B, Predicted A หมายความว่า _____

มีข้อมูล C กี่อันที่ถูกทำนายผิดว่าเป็น A _____

มีข้อมูล C กี่อันที่ถูกทำนายผิดว่าเป็น B _____

การทำนายข้อมูลชุดนี้มี accuracy = _____

ตัวอย่าง 4.8 บริษัทขายรถแห่งหนึ่งมีการเก็บข้อมูลการซื้อรถของลูกค้า ต้องการวิเคราะห์ว่า อายุ และเงินเดือนของลูกค้า ส่งผลต่อการซื้อรถหรือไม่ ข้อมูลเก็บไว้ในไฟล์ logic_purchasedCar.csv
ทำการวิเคราะห์ข้อมูลชุดนี้ และทำนายว่า คนเหล่านี้มีแนวโน้มจะซื้อรถหรือไม่

- อายุ 30 เงินเดือน 42000 บาท
- อายุ 25 เงินเดือน 15000 บาท

ตัวอย่างข้อมูล ข้อมูลใน column Purchased เป็น 0 หมายถึงไม่ซื้อ และ 1 หมายถึงซื้อรถ

Purchased	Age	Salary
0	65	4000
0	38	5700
1	23	30500
1	37	30800
0	55	30800
0	59	7400
0	42	7800
1	34	45200
1	52	45400
1	59	65100

ข้อมูลชุดนี้มีตัวแปรอิสระ (Independent variables – x) คือ อายุ และเงินเดือน
ตัวแปรตาม (Dependent variable – y) คือการซื้อหรือไม่ซื้อรถยนต์
จะเห็นว่าตัวแปรตามเป็นข้อมูลแบบไม่ต่อเนื่อง ถึงแม้จะมีค่าเป็น 0 หรือ 1 ก็ตาม แต่ค่า 0 หรือ 1 นี้ไม่ได้บอกปริมาณ (1 ไม่ได้หมายถึงมากกว่า 0 แบบการนับจำนวนทั่วไป) แต่เป็นเพียงการแทนค่าว่า ซื้อ หรือ ไม่ซื้อเท่านั้น ดังนั้นข้อมูลชุดนี้ต้องทำการวิเคราะห์แบบ logistic regression และเพราะตัวแปรตามเป็นไปได้เพียง 2 ค่า ดังนั้นคือการวิเคราะห์แบบ Binomial logistic regression

ตัวอย่างนี้เราแสดงการเขียน python เพื่อวิเคราะห์และแสดงผลหลายรูปแบบแตกต่างกัน

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
import pandas as pd

df = pd.read_csv('data/logic_purchasedCar.csv')
y_train = df['Purchased']
x_train = df[['Age','Salary']]

n_data = y_train.shape[0]              #total number
n_0 = np.count_nonzero(y_train==0)  #not purchase
n_1 = np.count_nonzero(y_train==1)  #purchased
print("There are %i people."%(n_data))
print("%i people did not purchase car."%(n_0))
print("%i people purchased car."%(n_1))

model = LogisticRegression(random_state=0)
model.fit(x_train, y_train)

y_pred = model.predict(x_train)
cm = confusion_matrix(y_train, y_pred)
print("\nConfusion matrix\n",cm)

accuracy = (cm[0][0] + cm[1][1])/n_data
print("Accuracy: %.2f"%(accuracy))

from sklearn import metrics
print("Accuracy: %.2f"%(metrics.accuracy_score(y_train, y_pred)))

x_new = np.array([[30,42000],[25,15000]])
y_pred_new = model.predict(x_new)
print("\nPredicted response of X:")
print(y_pred_new)
```

แสดงจำนวนข้อมูล แยกตามค่าของ y_train

วิเคราะห์ข้อมูลด้วย logistic regression

confusion matrix

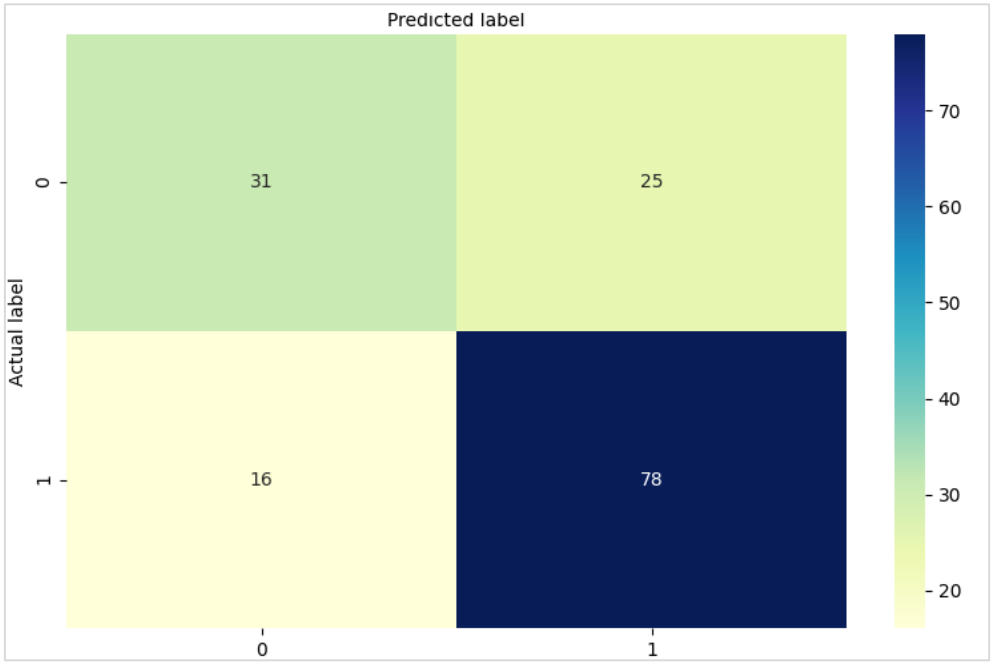
การคำนวณ accuracy ทั้ง 2 วิธี

ทำนายค่าข้อมูลที่ต้องการ

```
There are 150 people.
56 people did not purchase car.
94 people purchased car.

Confusion matrix
 [[31 25]
 [16 78]]
Accuracy: 0.73
Accuracy: 0.73

Predicted response of X:
[1 0]
```

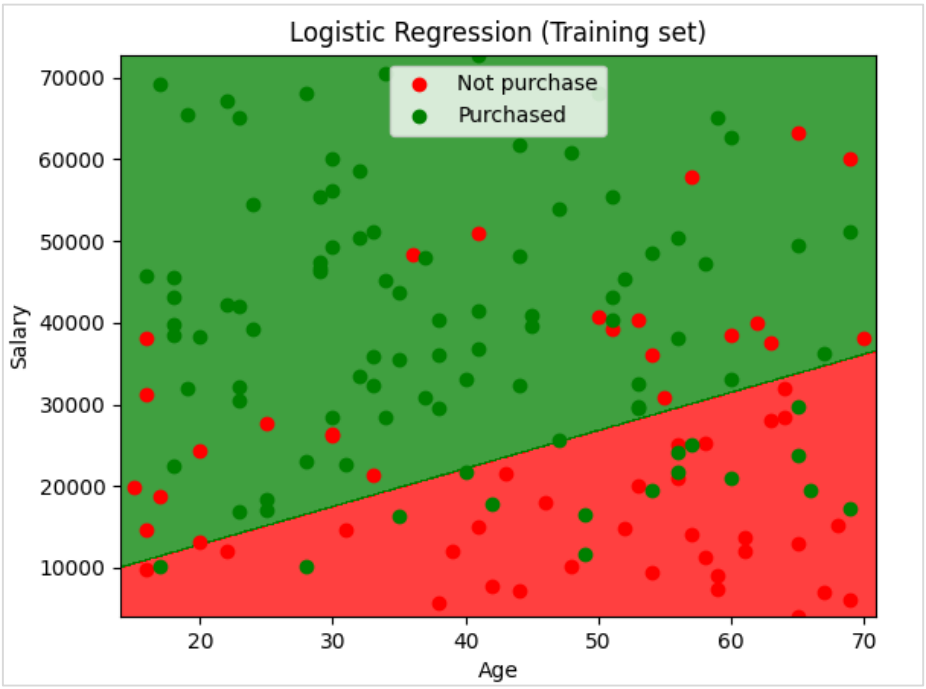
การใช้ Heatmap แสดง confusion matrix

```
#visualize confusion matrix using Heatmap
import seaborn as sns
class_names=[0,1] # name  of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```



การแสดงข้อมูลของ y_train โดยใช้ ListedColormap

```
#visualizing the Training set result
from matplotlib.colors import ListedColormap
X_set, y_set = x_train.values, y_train.values

X1, X2 = np.meshgrid(np.arange(start = X_set[:,0].min()-1, stop = X_set[:,0].max()+1, step = 0.1),
                     np.arange(start = X_set[:,1].min()-1, stop = X_set[:,1].max()+1, step = 100))

plt.contourf(X1, X2, model.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
y_label=["Not purchase","Purchased"]
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label= y_label[i]) #label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.legend()
plt.show()
```



ตัวอย่าง 4.9 จากการเก็บข้อมูลบ้านในประเทศไทย พบว่าบ้านที่ตั้งอยู่แต่ละจังหวัดมีราคา ขนาด และระยะทางจากตัวเมืองต่างกัน ตัวอย่างข้อมูลบ้านจาก 4 จังหวัด จากไฟล์ house.csv ดังนี้

province	price (Million)	area (Square Meter)	distance from center (km.)
Changmai	12	50	0.2
Changmai	5	55	20
Changmai	4	50	20
Changmai	7.8	40	12
Phuket	8	60	10
Phuket	6	55	20
Phuket	2	40	25
Phuket	2.5	45	20
Phuket	4	40	25
Phuket	10	50	15
Mahasarakham	8	120	2
Mahasarakham	3	80	5
Mahasarakham	2	60	20
Mahasarakham	1.2	60	25
Mahasarakham	2.4	60	1
Songkla	5	45	5
Songkla	4.5	50	5
Songkla	3	40	6
Songkla	6	55	3

วิเคราะห์ข้อมูลด้วย logistic regression และตอบคำถามว่า

- โมเดลการวิเคราะห์ข้อมูลชุดนี้ มี accuracy เท่าใด
- ข้อมูลของบ้าน 4 หลังนี้ ทำนายว่าบ้านอยู่จังหวัดใด

price (Million)	area (Square Meter)	distance from center (km.)
10	100	7
6	55	10
2	80	2
3.5	30	20

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
import pandas as pd


df = pd.read_csv("data/house.csv")
print(df.head())

y_train = df['province']
x_train = df[['price','area','distance']]
n_data = y_train.shape[0]           #total number

model = LogisticRegression(random_state=0)
model.fit(x_train, y_train)


y_pred = model.predict(x_train)
cm = confusion_matrix(y_train, y_pred)
print("\nConfusion matrix\n",cm)

from sklearn import metrics
print("Accuracy: %.2f"%(metrics.accuracy_score(y_train, y_pred)))


x_new = np.array([[10,100,7],[6,55,10]])
y_pred_new = model.predict(x_new)
print("\nPredicted response of X:")
print(y_pred_new)
```

```
Confusion matrix
 [[2 0 2 0]
 [0 5 0 0]
 [1 0 5 0]
 [0 0 0 4]]
Accuracy: 0.84

Predicted response of X:
['Mahasarakham' 'Changmai']
```