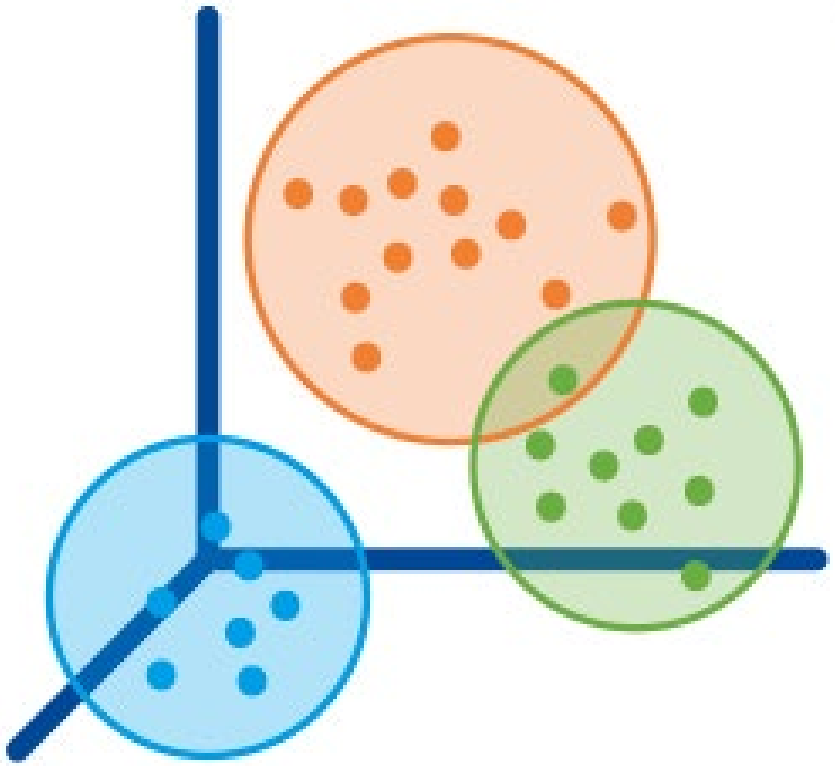
Unsupervised ML part II : k-Mean



อ.ดร.ปัญญนัท อ้นพงษ์

ภาควิชาคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร

aonpong_p@su.ac.th

Outline

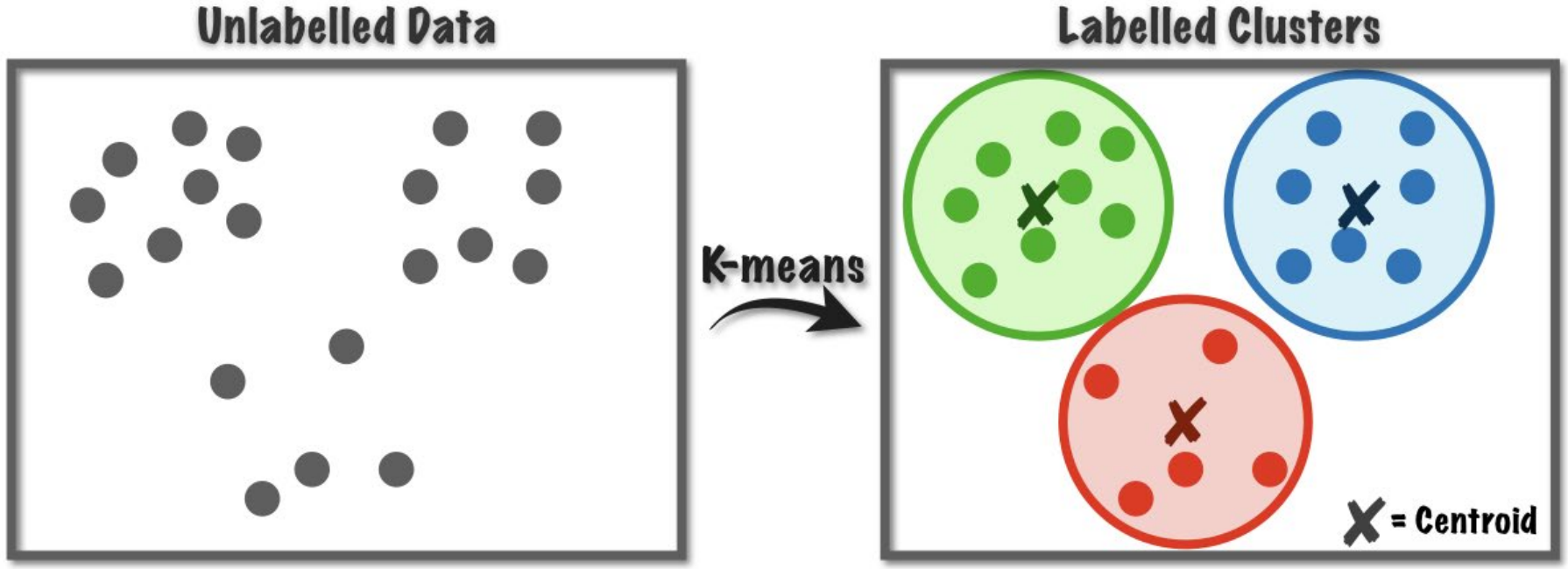
- **ทบทวน Clustering**
- Proximity measurement (Ordinal attribute; distance)
- K-mean Clustering
  - แนวคิดของ K-mean
  - กระบวนการ
  - ความจริงเกี่ยวกับ k-Means
  - Variant ของ k-Means

แนวคิดของ K-mean

- Clustering หรือการจัดกลุ่ม เป็นการเรียนรู้ของเครื่องแบบ Unsupervised หรือไม่ต้องมีผู้สอน
- ต้องมีการ Fit ข้อมูลก่อนเหมือน Supervised Learning
- แต่สิ่งที่การเรียนรู้แบบไม่มีผู้สอนไม่จำเป็นต้องใช้ คือ Ground Truth (Target)

Clustering

- Clustering เป็นการจัดกลุ่มข้อมูลโดยการจัดให้สิ่งที่อยู่ใกล้เคียงกันเป็นข้อมูลกลุ่มเดียวกัน โดยไม่สนใจว่ามันคืออะไร (แค่ดูแล้วมันคล้ายกันกว่าตัวอื่น)



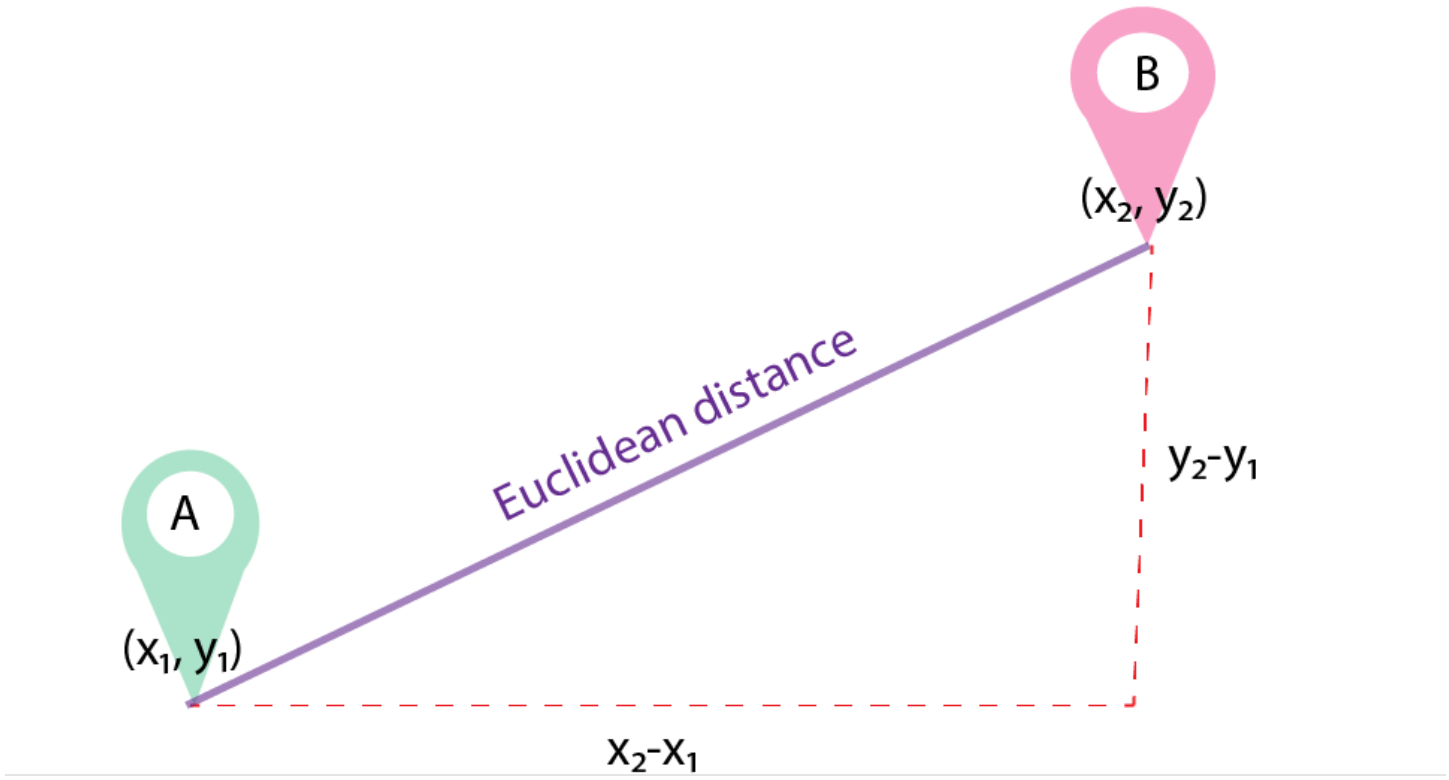
Outline

- ทบทวน Clustering
- **Proximity measurement (Ordinal attribute; Distance)**
- K-mean Clustering
  - แนวคิดของ K-mean
  - กระบวนการ
  - ความจริงเกี่ยวกับ k-Means
  - Variant ของ k-Means

Distance

- การวัดระยะทางระหว่างจุด มีวิธีการวัดหลายวิธี วิธีการที่เป็นที่นิยมมีดังนี้
  - Euclidean Distance
  - Manhattan Distance
  - Minkowski Distance
- นอกจากนี้ยังมี
  - Chebyshev
  - Cityblock
  - l1, l2, p
  - Etc.

Distance: Euclidean Distance



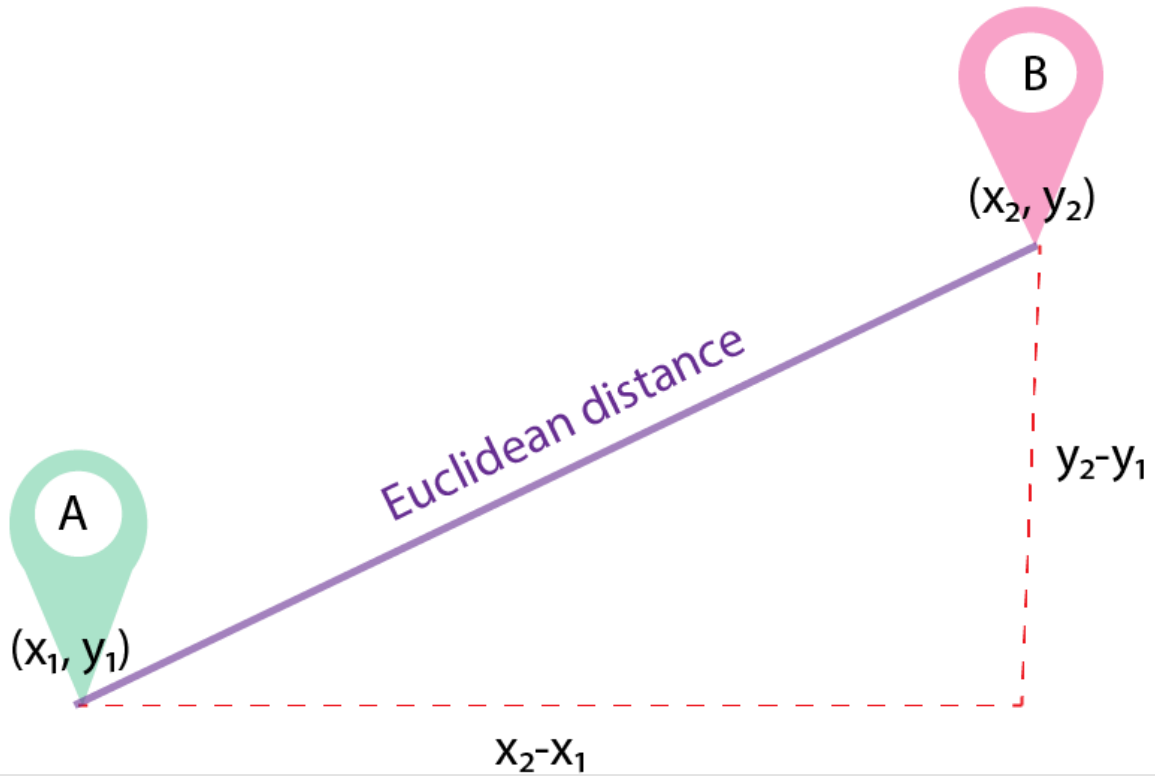
กรณีข้อมูลมี 2 มิติ

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

กรณีข้อมูลมี n มิติ (p --> q)

$$d = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Distance: Manhattan Distance



กรณีข้อมูลมี 2 มิติ

$$d = |x_2 - x_1| + |y_2 - y_1|$$

กรณีข้อมูลมี n มิติ (p --> q)

$$d = \sum_{i=1}^{n} |p_i - q_i|$$

Distance: Minowski Distance

- Minowski Distance สามารถปรับความซับซ้อนได้โดยการปรับค่า q
  - ถ้า q=1 จะเป็น Manhattan (L1)
  - ถ้า q=2 จะเป็น Euclidean (L2)

กรณีข้อมูลมี 2 มิติ $$d = ((x_2 - x_1)^q + (y_2 - y_1)^q)^{\frac{1}{q}}$$

กรณีข้อมูลมี n มิติ (p --> q) $$d = (\sum_{i=1}^{n} (p_i - q_i)^q)^{\frac{1}{q}}$$

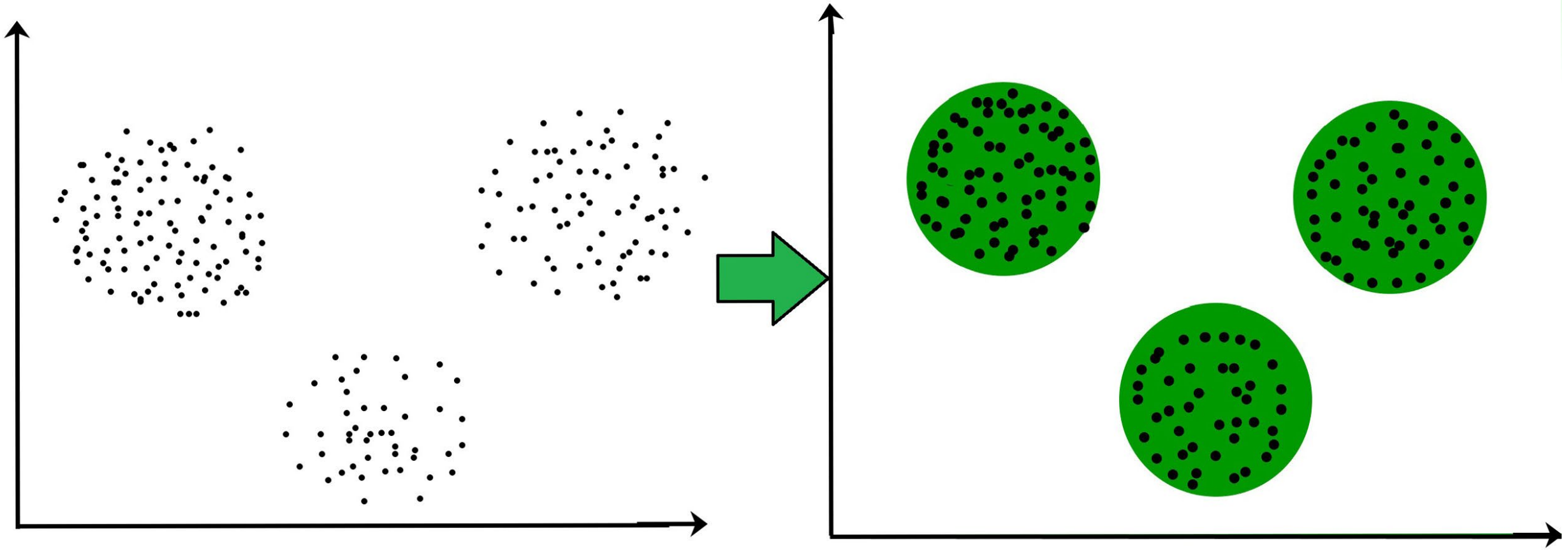
Outline

- ทบทวน Clustering
- Proximity measurement (Ordinal attribute; Distance)
- **K-mean Clustering**
  - **แนวคิดของ K-mean**
  - กระบวนการ
  - ความจริงเกี่ยวกับ k-Means
  - Variant ของ k-Means

แนวคิดของ k-mean

- K-mean เป็นวิธีการ clustering แบบ centroid-based หรือใช้จุดกลางของกลุ่มเป็นแนวทางหรือจุดอ้างอิงการจัดกลุ่มข้อมูล
- เพราะเป็น clustering จึงไม่ต้องใช้ GT
- จำแนกได้ว่าข้อมูลที่เข้าสู่ระบบเหมือนหรือต่างกัน บอกได้ว่าอยู่กลุ่มข้อมูลไหน แต่ k-mean ไม่สามารถระบุได้ว่าชื่อคลาสคืออะไร

แนวคิดของ k-mean



Outline

- ทบทวน Clustering
- Proximity measurement (Ordinal attribute; Distance)
- **K-mean Clustering**
  - แนวคิดของ K-mean
  - **กระบวนการ**
  - ความจริงเกี่ยวกับ k-Means
  - Variant ของ k-Means

กระบวนการ k-mean

- สมมติข้อมูลตั้งต้นมี 1 มิติ ดังนี้

- ข้อมูลนี้ เราไม่รู้คลาสของมันมาก่อนเลย แต่เราต้องการจัดกลุ่มมัน

กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4

กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4
2. วางจุด centroid ลงบนค่า k ค่าแบบสุ่ม จุดเหล่านี้เรียกว่า initial centroid

กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4
2. วางจุด centroid ลงบนค่า k ค่าแบบสุ่ม
3. จากข้อมูลแต่ละตัว ให้พิจารณาว่าข้อมูลตัวนั้นอยู่ใกล้ centroid ใดมากที่สุด พิจารณาแบบนี้ทุกตัว (คำนวณ distance ในกรณีเป็นกราฟ > 2 มิติ)

กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4
2. วางจุด centroid ลงบนค่า k ค่าแบบสุ่ม
3. จากข้อมูลแต่ละตัว ให้พิจารณาว่าข้อมูลตัวนั้นอยู่ใกล้ centroid ใดมากที่สุด พิจารณาแบบนี้ทุกตัว (คำนวณ distance ในกรณีเป็นกราฟ > 2 มิติ)
4. เลื่อน centroid ไปไว้ตรงกลางของข้อมูลกลุ่มของตัวเอง (ในครั้งนี้ centroid อาจไม่ได้อยู่บนข้อมูลก็ได้)

กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4
2. วางจุด centroid ลงบนค่า k ค่าแบบสุ่ม
3. จากข้อมูลแต่ละตัว ให้พิจารณาว่าข้อมูลตัวนั้นอยู่ใกล้ centroid ใดมากที่สุด พิจารณาแบบนี้ทุกตัว (คำนวณ distance ในกรณีเป็นกราฟ > 2 มิติ)
4. เลื่อน centroid ไปไว้ตรงกลางของข้อมูลกลุ่มของตัวเอง (ในครั้งนี้ centroid อาจไม่ได้อยู่บนข้อมูลก็ได้)
5. ทำข้อ 3 และ 4 ซ้ำไปเรื่อย ๆ จนแต่ละจุดมี centroid คงที่
   - ทำข้อ 3 ซ้ำอีกครั้งในครั้งแรกๆ จะพบว่าแต่ละจุดมี centroid ของตัวเองเปลี่ยนไปจากเดิม

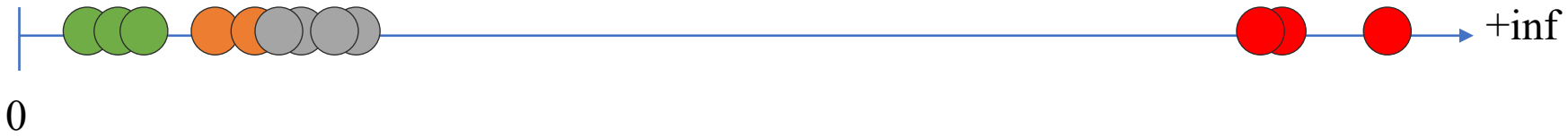
กระบวนการ k-mean

K-mean clustering

1. กำหนดค่า k หรือจำนวนกลุ่มที่เราต้องการจำแนก (จะพูดถึงหลักการการเลือกค่า k ทีหลัง) สมมติกำหนดให้ k=4
2. วางจุด centroid ลงบนค่า k ค่าแบบสุ่ม
3. จากข้อมูลแต่ละตัว ให้พิจารณาว่าข้อมูลตัวนั้นอยู่ใกล้ centroid ใดมากที่สุด พิจารณาแบบนี้ทุกตัว (คำนวณ distance ในกรณีเป็นกราฟ > 2 มิติ)
4. เลื่อน centroid ไปไว้ตรงกลางของข้อมูลกลุ่มของตัวเอง (ในครั้งนี้ centroid อาจไม่ได้อยู่บนข้อมูลก็ได้)
5. ทำข้อ 3 และ 4 ซ้ำไปเรื่อย ๆ จนแต่ละจุดมี centroid คงที่
   - ทำข้อ 3 ซ้ำในครั้งแรกๆ จะพบว่าแต่ละจุดมี centroid ของตัวเองเปลี่ยนไปจากเดิม
   - ทำข้อ 4 ซ้ำในครั้งแรกๆ จะพบว่า centroid เคลื่อนไหว

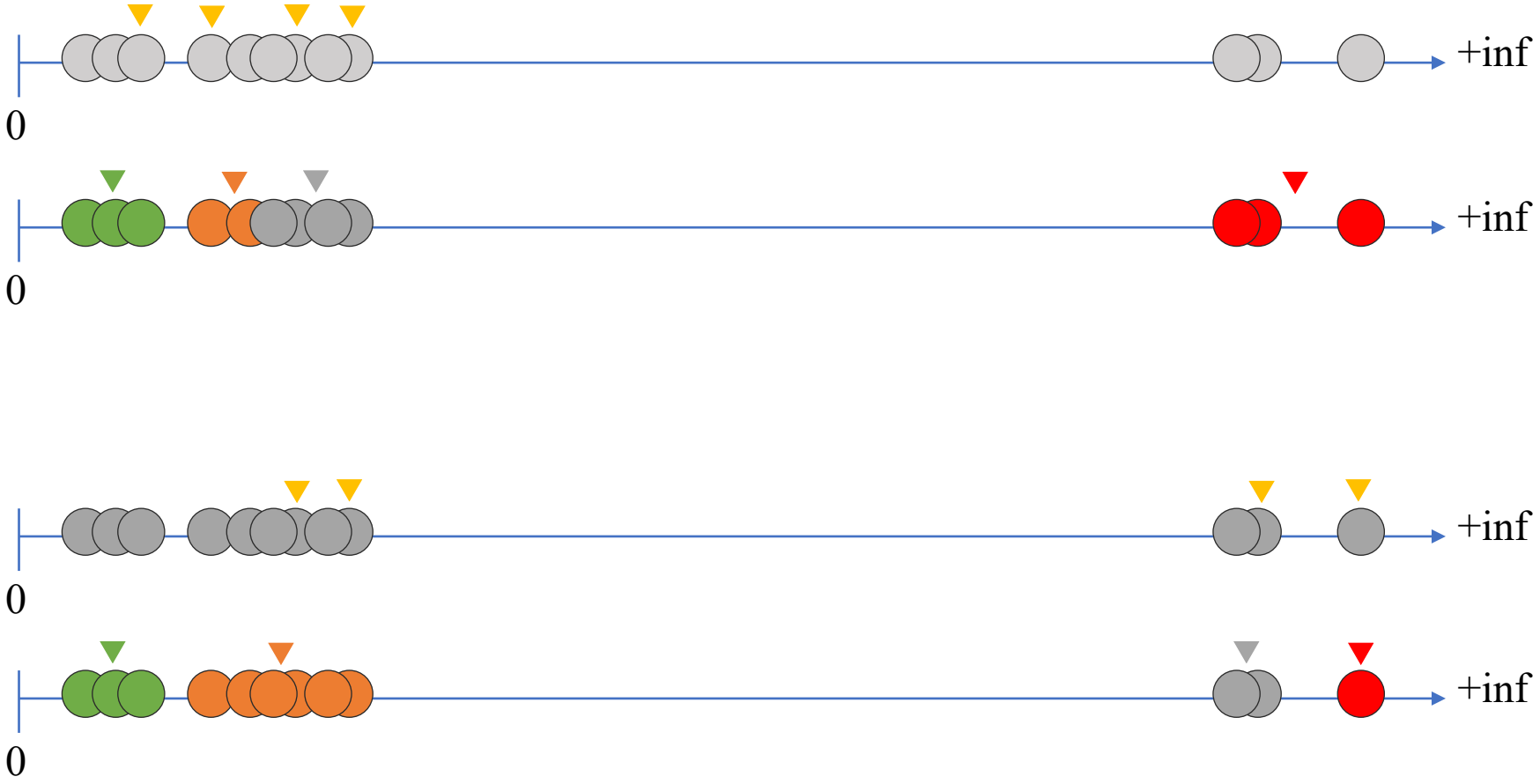
กระบวนการ k-mean

ผลลัพธ์ที่ได้



กระบวนการ k-mean

อย่างไรก็ดี การวาง centroid ในครั้งแรกส่งผลต่อผลลัพธ์ของการจัดกลุ่มด้วย



กระบวนการ k-mean

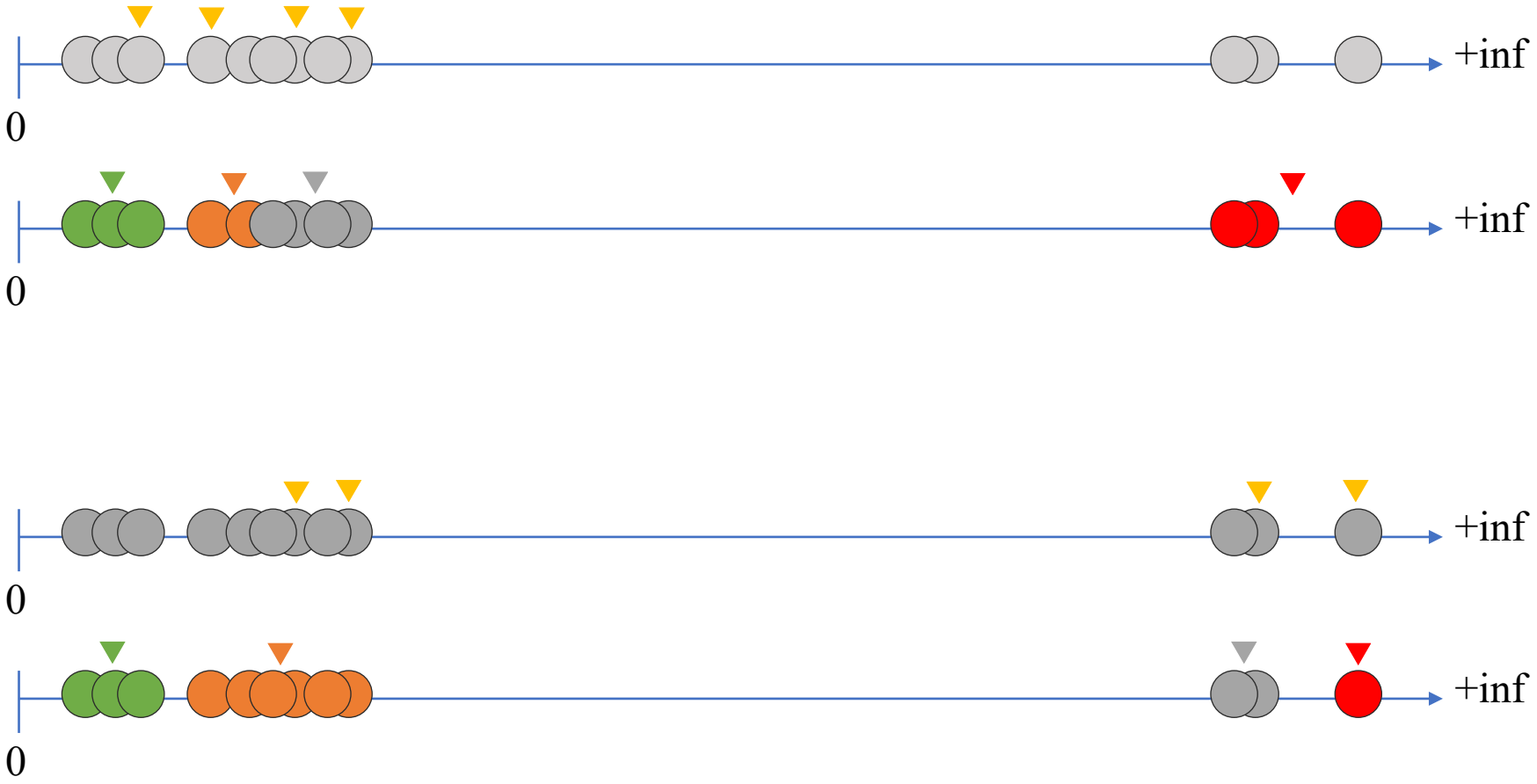
- การเลือกค่า k ที่เหมาะสม
  - อัลกอริทึม k-means จะให้คำตอบของชุดคลัสเตอร์เพียงชุดเดียว ซึ่งผู้ใช้ต้องระบุจำนวนทางเลือกของคลัสเตอร์ที่ต้องการทั้งหมด k คลัสเตอร์
  - ด้วยเหตุนี้ k จึงควรเป็นจำนวนคลัสเตอร์ที่ผู้ใช้คาดหวังให้ปรากฏ
  - การเลือกค่า k ที่เหมาะสมสำหรับชุดข้อมูลใด ๆ เป็นปัญหาที่ใหญ่ที่สุดของ K-mean เพราะเรามักไม่รู้จำนวนคลัสเตอร์ที่แท้จริงที่ปรากฏในข้อมูล
  - บ่อยครั้ง จำนวนคลัสเตอร์ที่ปรากฏก็ไม่ชัดเจน เนื่องจากความใกล้เคียงกันของตัวอย่าง

กระบวนการ k-mean

- การเลือกค่า k ที่เหมาะสมอย่างง่าย
  - ปกติจะใช้ความรู้เกี่ยวกับข้อมูลที่มีอยู่ก่อนหน้า ว่าในชุดข้อมูลมีข้อมูลกี่กลุ่ม เช่น ชายหรือหญิง แต่ถ้าเราไม่รู้ ก็ต้องใช้วิธีถัดไป
  - ทดลองตั้งแต่ k=1, 2, 3, ... ไล่ไปเรื่อยๆ และคำนวณ “**variation**”

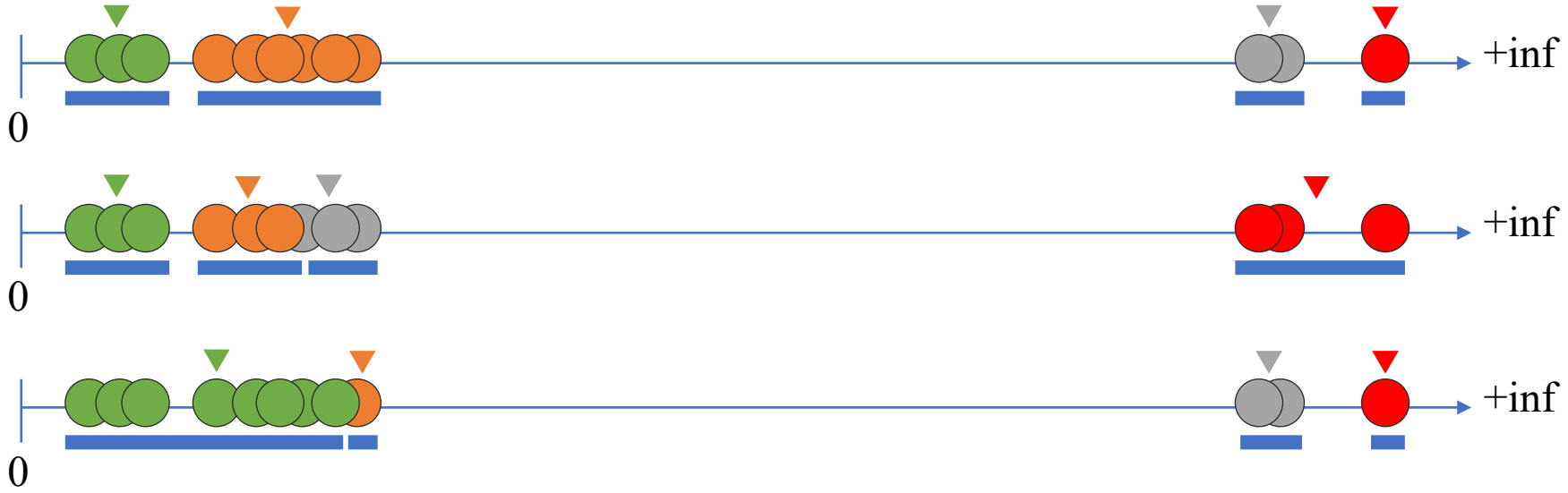
กระบวนการ k-mean

เนื่องจากการวาง centroid ในครั้งแรกส่งผลต่อผลลัพธ์ของการจัดกลุ่มด้วย



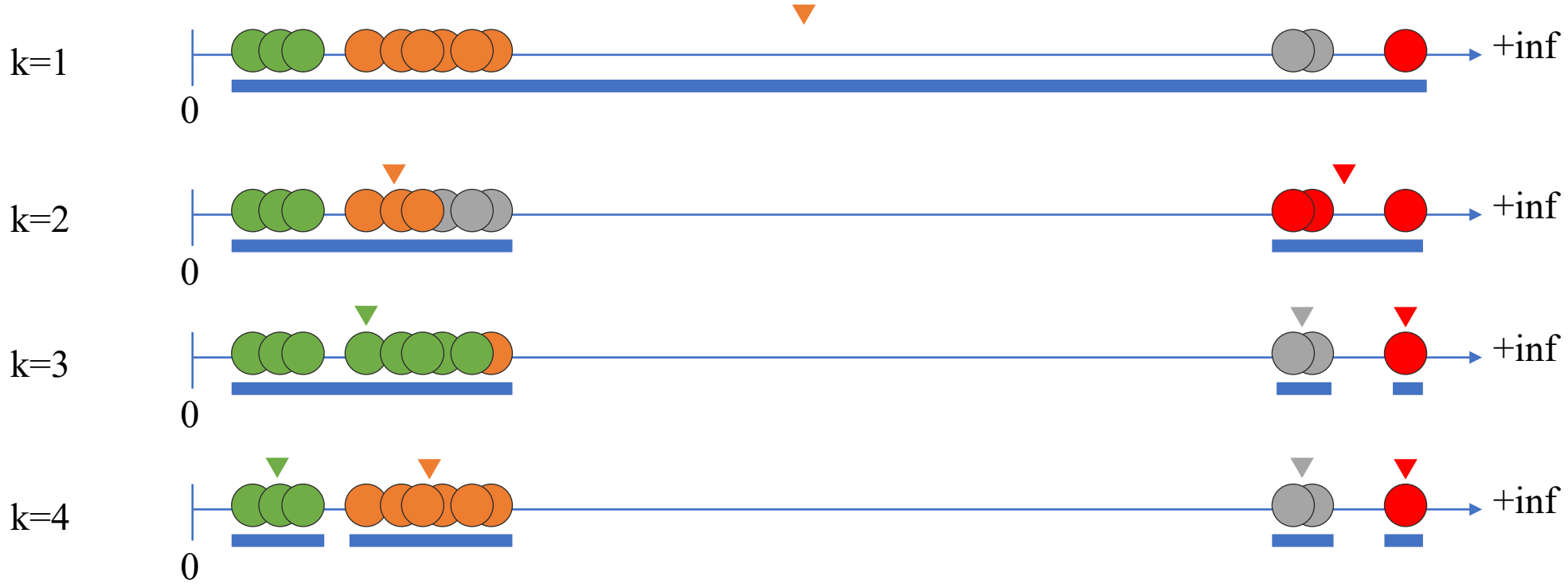
กระบวนการ k-mean

- การใช้ k-mean จึงมักถูกแนะนำให้ทำหลายๆ ครั้ง และหาครั้งที่มี Variation ต่ำที่สุด
- การหา Variation ใน k-mean คือการตรวจสอบว่าระยะห่างของข้อมูลตัวแรกและตัวสุดท้าย คือเท่าไหร่ ปกติแล้วตัวที่ถูกเลือกจะเป็นคำตอบที่เกิดบ่อยครั้งและ/หรือมีช่องว่างน้อย



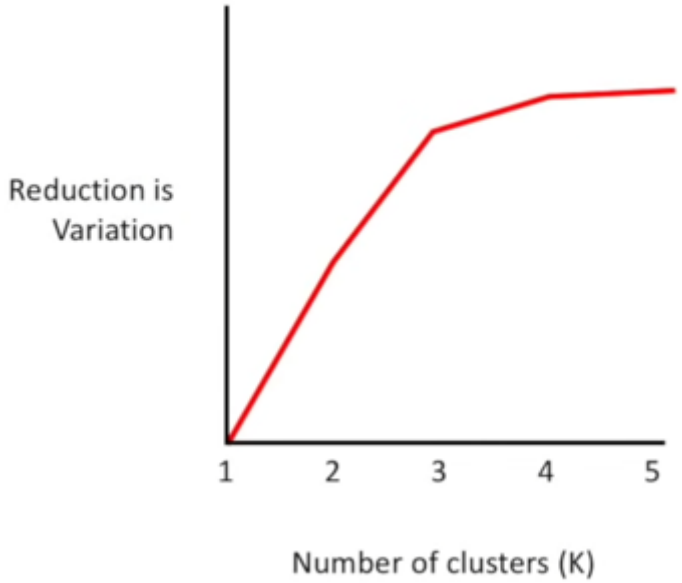
กระบวนการ k-mean

- การเลือกค่า k ที่เหมาะสม
  - ปกติจะใช้ความรู้เกี่ยวกับข้อมูลที่มีอยู่ก่อนหน้า แต่ถ้าเราไม่รู้ ก็ต้องใช้วิธีถัดไป
  - ทดลองตั้งแต่ k=1, 2, 3, ... ไล่ไปเรื่อยๆ และคำนวณ variation



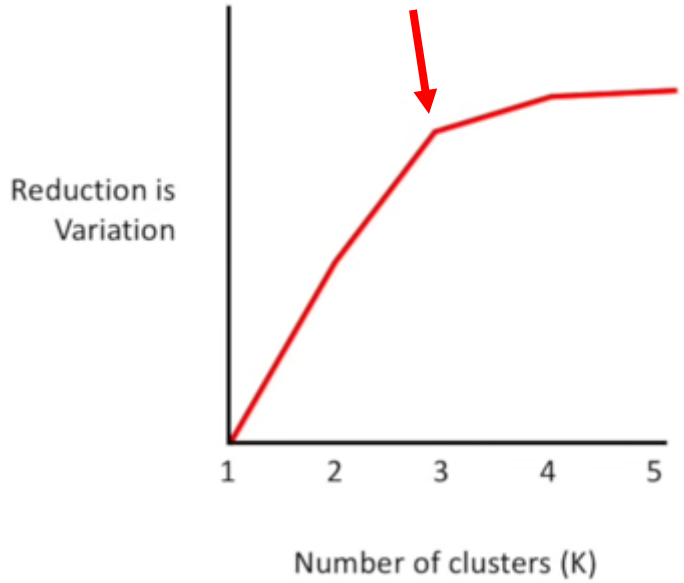
กระบวนการ k-mean

- การเลือกค่า k ที่เหมาะสม
  - ปกติจะใช้ความรู้เกี่ยวกับข้อมูลที่มีอยู่ก่อนหน้า แต่ถ้าเราไม่รู้ ก็ต้องใช้วิธีถัดไป
  - ทดลองตั้งแต่ k=1, 2, 3, ... ไล่ไปเรื่อยๆ และคำนวณ variation
    - วาดกราฟการลดลงของ Variation (Elbow Plot)
    - เริ่มต้นที่ k=1, variation ที่ลดลงคือ 0 (จุดเริ่มต้น)
    - ที่ k=2, นำ variation ที่ k=2 ลบ variation ที่ k=1
    - ที่ k=3, นำ variation ที่ k=3 ลบ variation ที่ k=2
    - ที่ k=4, นำ variation ที่ k=4 ลบ variation ที่ k=3
    - ที่ k=5, นำ variation ที่ k=5 ลบ variation ที่ k=4
    - ...



กระบวนการ k-mean

- การเลือกค่า k ที่เหมาะสม
  - ปกติจะใช้ความรู้เกี่ยวกับข้อมูลที่มีอยู่ก่อนหน้า แต่ถ้าเราไม่รู้ ก็ต้องใช้วิธีถัดไป
  - ทดลองตั้งแต่ k=1, 2, 3, ... ไล่ไปเรื่อยๆ และคำนวณ variation
    - วาดกราฟการลดลงของ Variation (Elbow Plot)
    - เราจะสนใจค่า k ที่ทำให้กราฟหักลงอย่างรวดเร็ว



กระบวนการ k-mean

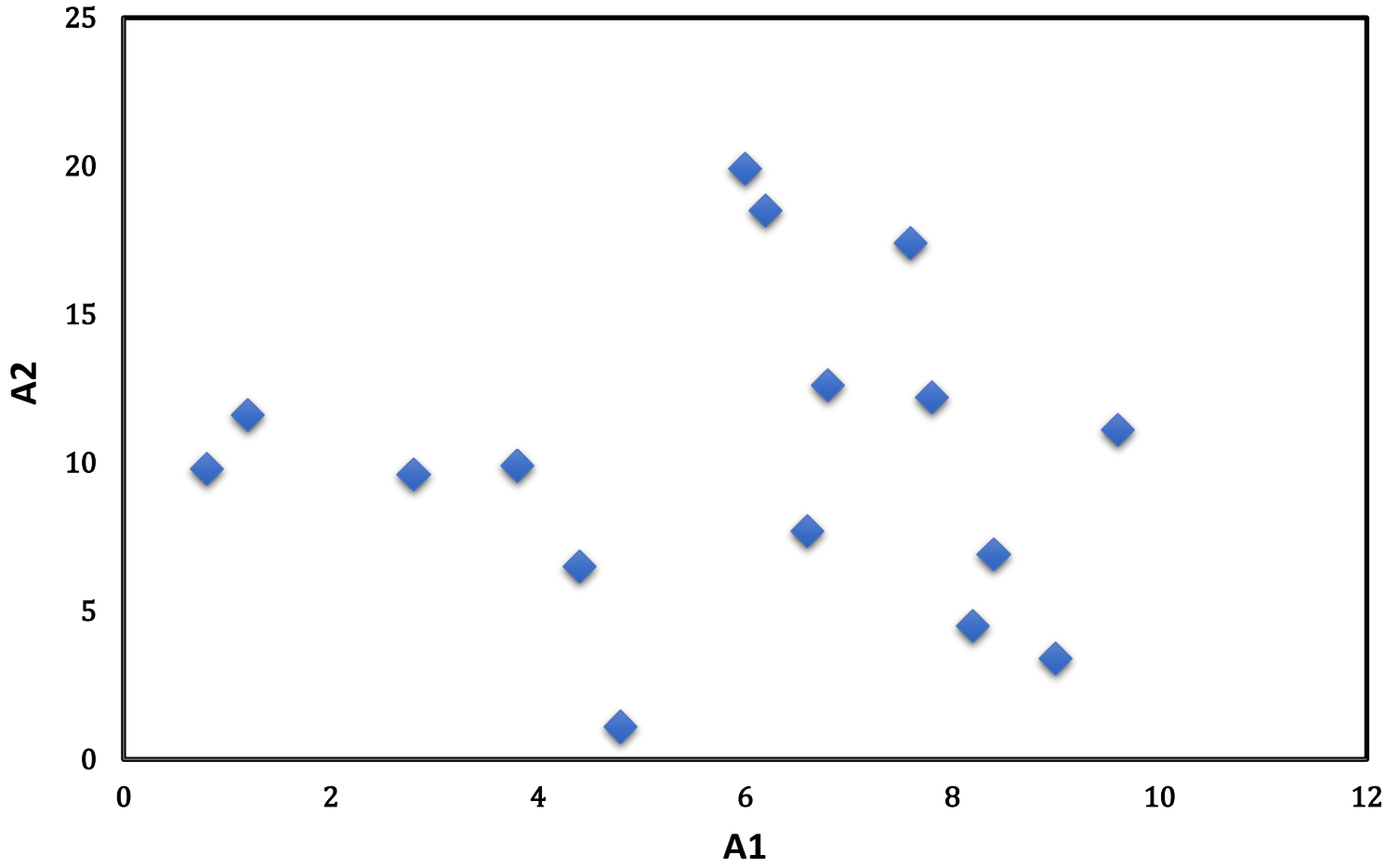
- การเลือกค่า k ที่เหมาะสม
  - ปกติจะใช้ความรู้เกี่ยวกับข้อมูลที่มีอยู่ก่อนหน้า แต่ถ้าเราไม่รู้ ก็ต้องใช้วิธีถัดไป
  - ทดลองตั้งแต่ k=1, 2, 3, ... ไล่ไปเรื่อยๆ และคำนวณ variation
  - ใช้ Magic Number หรือ k=7
    - ไม่ใช่ว่าผลจะดีทุกครั้งไป
    - ไม่เหมาะกับการนำไปรายงานที่ไหน เหมาะกับการทดลองเบื้องต้นอย่างเดียว
    - ประมาณ 60% จะใช้การได้

กระบวนการ k-mean

- ถ้าข้อมูลมีมิติมากขึ้น วิธีการก็ยังคงเหมือนเดิม
  1. Centroid ก็ยังคงถูกสุ่มจุดตอนเริ่ม
  2. ต้องพิจารณาทุกข้อมูลเหมือนเดิม แต่ต้องคำนวณระยะระหว่างข้อมูลไปยัง centroid ด้วย distance
  3. เลื่อน centroid ไปตรงจุดกึ่งกลางของข้อมูลของกลุ่มตัวเอง
  4. ทำข้อ 2-3 ใหม่

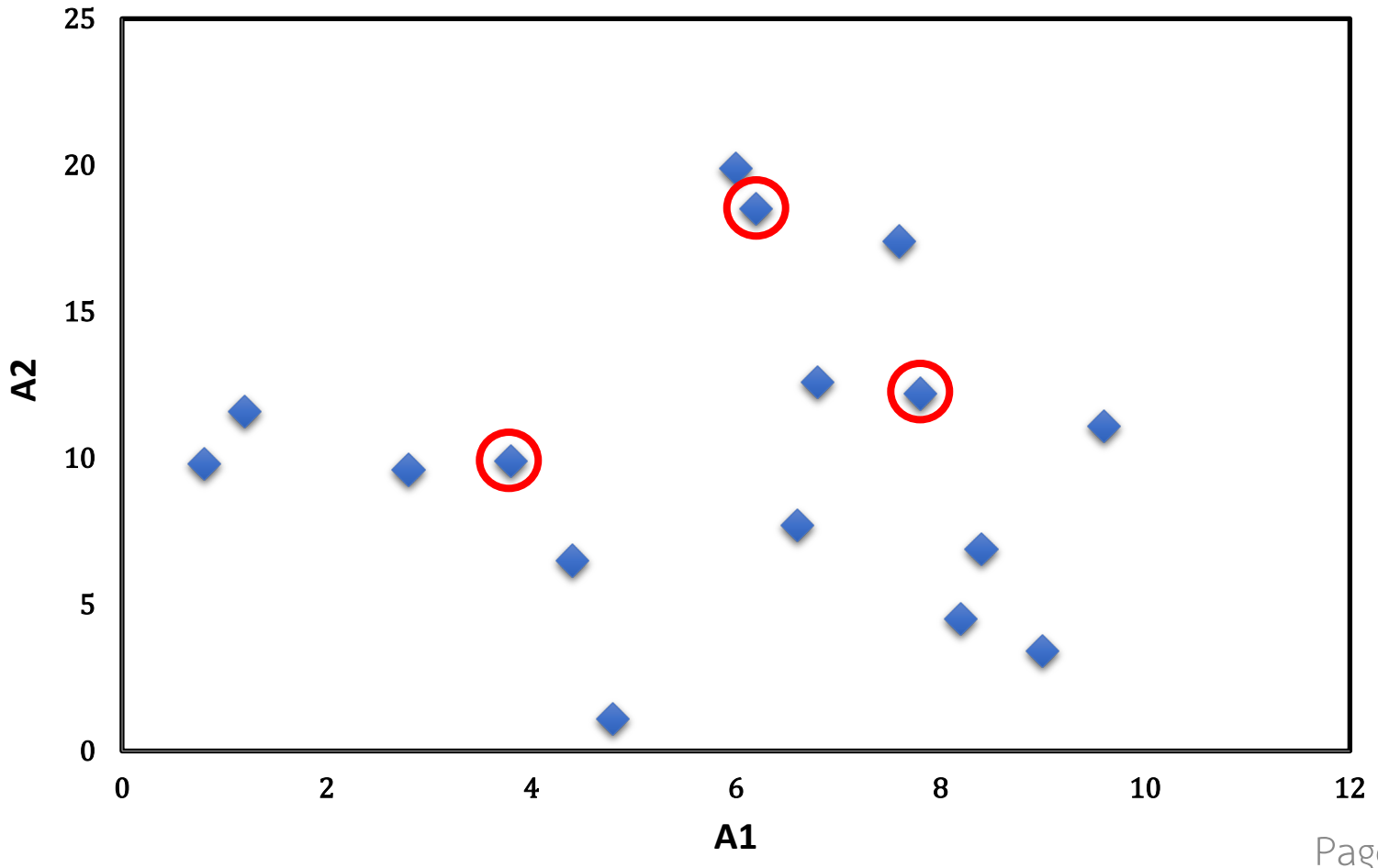
ลองคำนวณจากตัวอย่างโจทย์: มีข้อมูลดังนี้

A_1	A_2
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1



ลองคำนวณจากตัวอย่างโจทย์: กำหนด C ดังนี้

A_1	A_2
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1



ลองคำนวณจากตัวอย่างโจทย์: จงเติมข้อมูลที่หายไป

Centroid	Objects	
	A1	A2
c_1	3.8	9.9
c_2	7.8	12.2
c_3	6.2	18.5

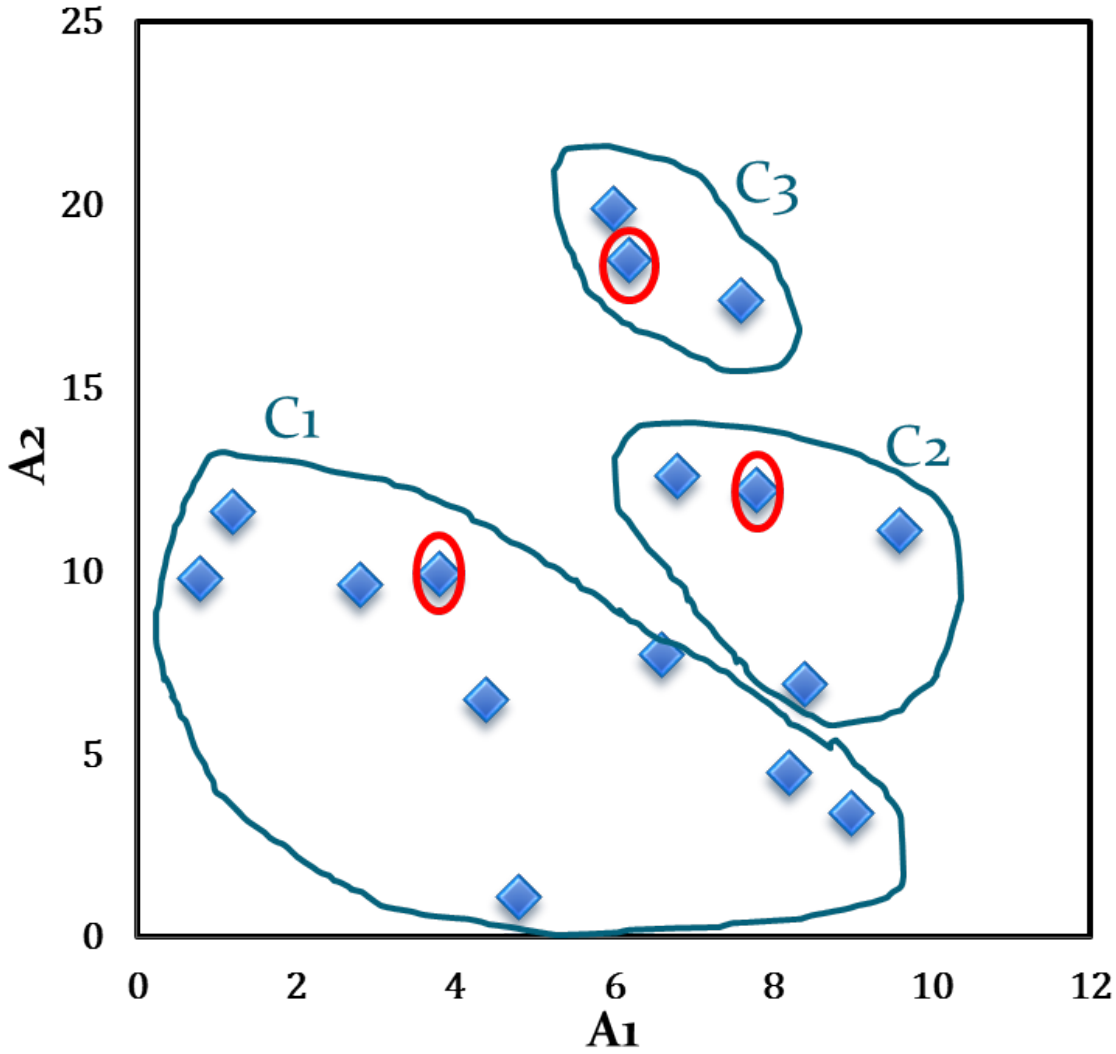
A_1	A_2	d_1	d_2	d_3	cluster
6.8	12.6				
0.8	9.8				
1.2	11.6				
2.8	9.6				
3.8	9.9				1
4.4	6.5				
4.8	1.1				
6.0	19.9				
6.2	18.5				3
7.6	17.4				
7.8	12.2				2
6.6	7.7				
8.2	4.5				
8.4	6.9				
9.0	3.4				
9.6	11.1				

ลองคำนวณจากตัวอย่างโจทย์

A_1	A_2	d_1	d_2	d_3	cluster
6.8	12.6	4.0	1.1	5.9	2
0.8	9.8	3.0	7.4	10.2	1
1.2	11.6	3.1	6.6	8.5	1
2.8	9.6	1.0	5.6	9.5	1
3.8	9.9	0.0	4.6	8.9	1
4.4	6.5	3.5	6.6	12.1	1
4.8	1.1	8.9	11.5	17.5	1
6.0	19.9	10.2	7.9	1.4	3
6.2	18.5	8.9	6.5	0.0	3
7.6	17.4	8.4	5.2	1.8	3
7.8	12.2	4.6	0.0	6.5	2
6.6	7.7	3.6	4.7	10.8	1
8.2	4.5	7.0	7.7	14.1	1
8.4	6.9	5.5	5.3	11.8	2
9.0	3.4	8.3	8.9	15.4	1
9.6	11.1	5.9	2.1	8.1	2

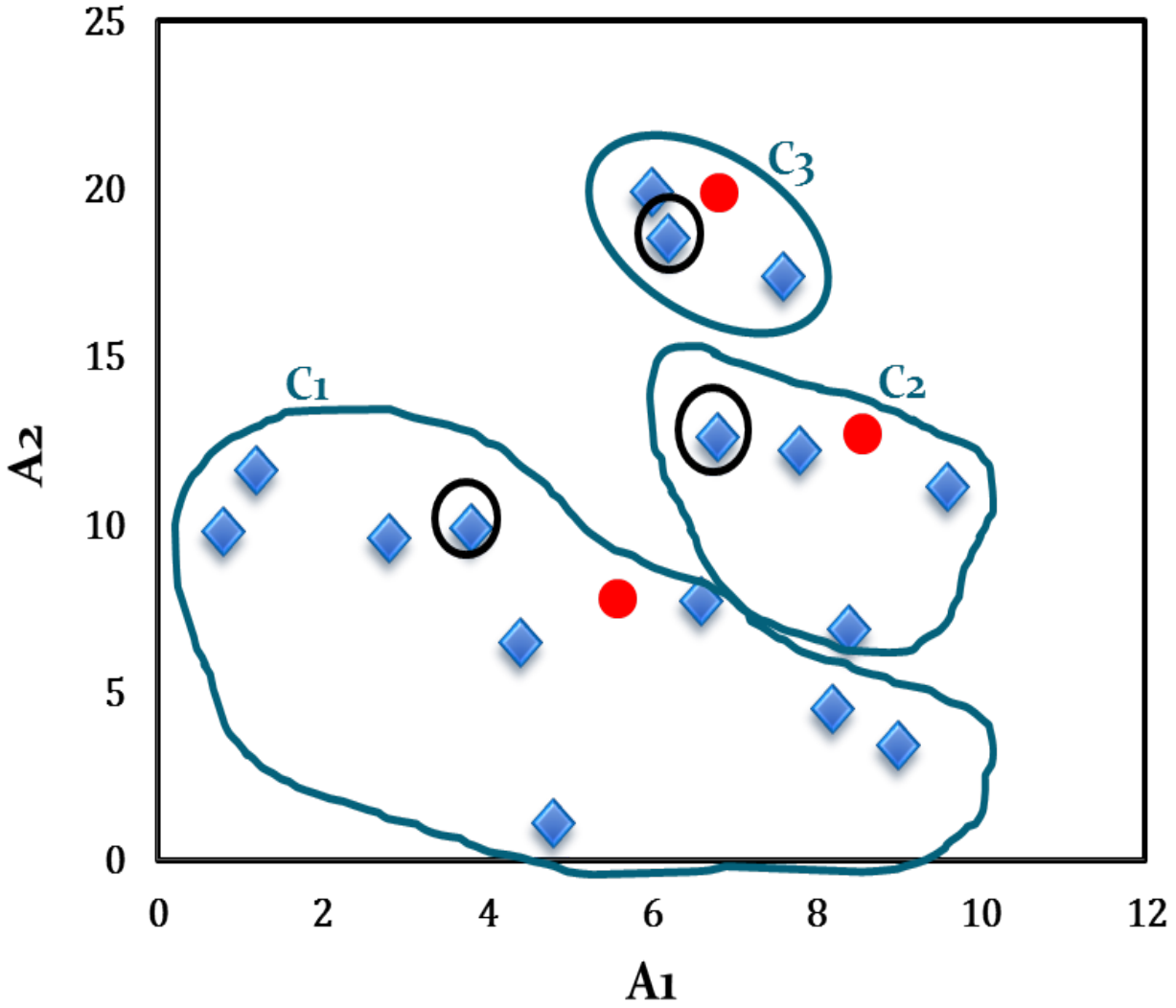
ลองคำนวณจากตัวอย่างโจทย์: จงเติมค่า C ใหม่

New Centroid	Objects	
	A1	A2
c_1		
c_2		
c_3		



ลองคำนวณจากตัวอย่างโจทย์

New Centroid	Objects	
	A1	A2
c_1	4.6	7.1
c_2	8.2	10.7
c_3	6.6	18.6



ลองคำนวณจากตัวอย่างโจทย์: จาก C จงเติมข้อมูล

New Centroid	Objects	
	A1	A2
c_1	4.6	7.1
c_2	8.2	10.7
c_3	6.6	18.6

A_1	A_2	d_1	d_2	d_3	cluster
6.8	12.6				
0.8	9.8				
1.2	11.6				
2.8	9.6				
3.8	9.9				
4.4	6.5				
4.8	1.1				
6.0	19.9				
6.2	18.5				
7.6	17.4				
7.8	12.2				
6.6	7.7				
8.2	4.5				
8.4	6.9				
9.0	3.4				
9.6	11.1				

Outline

- ทบทวน Clustering
- Proximity measurement (Ordinal attribute; Distance)
- **K-mean Clustering**
  - แนวคิดของ K-mean
  - กระบวนการ
  - **ความจริงเกี่ยวกับ k-Means**
  - Variant ของ k-Means

ความจริงเกี่ยวกับ k-Means

ในทางปฏิบัติ ไม่ใช่ข้อมูลทุกรูปแบบและทุกสถานการณ์จะเหมาะกับการเลือกใช้ k-Means ต่อจากนี้จะกล่าวถึงจุดเด่นและจุดด้อยของกระบวนการ k-Means ในการทำ Clustering ก่อนอื่นใด มาทำความรู้จักกับตัวแปรที่จะใช้ในการอธิบายกันก่อน

- Notations:
  - x : an object under clustering
  - n : number of objects under clustering
  - $\boldsymbol{C_i}$: the *i-th* cluster
  - c_i : the centroid of cluster $\boldsymbol{C_i}$
  - n_i : number of objects in the cluster $\boldsymbol{C_i}$
  - c : denotes the centroid of all objects
  - k : number of clusters

ความจริงเกี่ยวกับ k-Means

1. ปัญหาค่า k ที่เหมาะสม

- ก่อนหน้านี้เราได้พูดเรื่องการหาค่า k อย่างคร่าวๆ ไปแล้ว
- เราใช้ variation ในการทดสอบอย่างง่ายๆ
- แต่ต่อจากนี้เราจะพูดถึงเมตริกอื่น ๆ ในทางสถิติที่ใช้ทดสอบว่าค่า k เหมาะสมหรือไม่
- จริงๆแล้วอาจเลือกใช้ variation ก็ได้ เพราะง่ายกว่า แต่หลายงานต้องการความละเอียดของการกระจุกตัวมากกว่า ด้วยเหตุนี้ การมองเพียงจุดที่ห่างกันมากที่สุดอาจไม่เพียงพอ
- ขอแนะนำให้รู้จักกับค่า sum square error (SSE)

ความจริงเกี่ยวกับ k-Means

1. ปัญหาค่า k ที่เหมาะสม

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} (x - c_i)^2$$

กำหนดให้ $x - c_i$ แทน error เมื่อ x เป็นสมาชิกของคลัสเตอร์ C_i ที่มีจุด Centroid ที่ c_i

* ค่านี้ยิ่งน้อยยิ่งดี แต่ปกติจะไม่ได้เลือก k ที่ทำให้ค่านี้น้อยที่สุด แต่จะเลือก k ที่ทำให้ค่านี้ลดลงจาก k-1 อย่างมีนัยสำคัญ ให้นึกถึง elbow plot

ความจริงเกี่ยวกับ k-Means

1. ปัญหาค่า k ที่เหมาะสม

จากตาราง แสดงค่า SSE ที่คำนวณมาจากข้อมูลที่มี เมื่อมีค่า k ต่างกัน

จงเลือกค่า k ที่เหมาะสมที่จะใช้ในกระบวนการ k-Means ของข้อมูลต่อไปนี้

k	SSE
1	62.8
2	12.3
3	9.4
4	9.3
5	9.2
6	9.1
7	9.05
8	9.0

Tips: ถ้า k มีค่าเท่ากับจำนวนข้อมูลที่มี ค่า SSE จะเป็น 0 ซึ่งน้อยที่สุดอย่างแน่นอน แต่เราจะไม่มีทางเลือกค่านี้มาใช้ใน k-Means เพราะไม่มีประโยชน์ที่จะทำการ Clustering ถ้าจะให้ข้อมูลอยู่กลุ่มละตัว ดังนั้นค่า k ที่เหมาะสมไม่ใช่ค่าที่ทำให้ SSE น้อยที่สุด

ความจริงเกี่ยวกับ k-Means

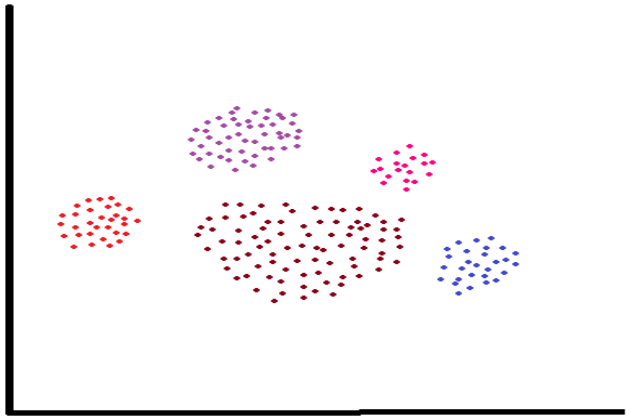
2. ปัญหาการเลือกจุด centroid เริ่มต้นที่เหมาะสม

- ในแง่มุมของการใช้งาน จะพบว่า การวางจุด centroid เริ่มต้นส่งผลต่อผลลัพธ์ในขั้นสุดท้าย
- เราจะรู้ได้อย่างไรว่าจุด centroid ที่เราเลือก (หรือสุ่ม) ในตอนเริ่มแรกนั้นเหมาะสมหรือไม่
- คำตอบคือ ไม่รู้
- แต่เรามักทำการทดลองโดยวางจุด centroid เริ่มต้นให้แตกต่างกันหลายๆครั้ง เพื่อดูแนวโน้มของการจัดกลุ่ม เราจะสมมติและเชื่อว่า centroid ที่เหมาะสมจะแสดงผลลัพธ์เป็นส่วนใหญ่ แต่จะมีบางกรณียกเว้นเท่านั้นที่ให้ผลลัพธ์ผิดไปจากกรณีอื่นๆ และเราจะไม่เชื่อกรณีส่วนน้อยเหล่านั้น

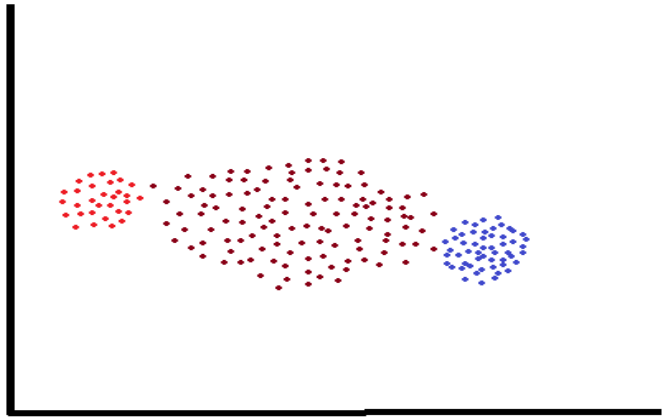
ความจริงเกี่ยวกับ k-Means

3. ข้อมูลที่ไม่เหมาะสมกับกระบวนการวิธี k-Means

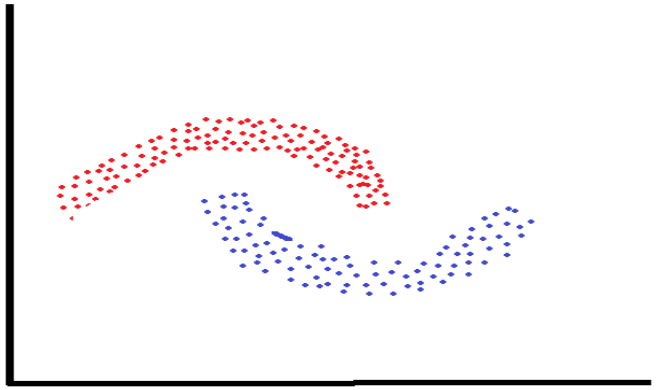
- K-Means ไม่เก่งกับข้อมูลที่มี outliers จำนวนมาก (outlier จะดึงค่าเฉลี่ยให้ผิดเพี้ยนไปจากจุดที่ควรจะเป็น ส่งผลต่อความแม่นยำที่ลดลง)
- k-Means ไม่เก่งกับข้อมูลที่กระจายตัวแบบ convex
- K-Means ไม่เก่งกับข้อมูลที่มีการกระจายตัวหรือมีขนาดในแต่ละคลาสแตกต่างกันมากเกินไป



Cluster with different sizes



Cluster with different densities



Non-convex shaped clusters

Outline

- ทบทวน Clustering
- Proximity measurement (Ordinal attribute; Distance)
- **K-mean Clustering**
  - แนวคิดของ K-mean
  - กระบวนการ
  - ความจริงเกี่ยวกับ k-Means
  - **Variant ของ k-Means**

Variant ของ k-Means

- การที่ k-Means มีจุดด้อยในการทำงานกับข้อมูลที่มี outlier มาก ทำให้มีแนวคิดพัฒนาวิธีการที่มีโครงสร้างการทำงานเดิมขึ้นมาใหม่
- ตัวอย่าง อย่างที่เราทราบว่า outlier ส่งผลต่อค่าเฉลี่ยมาก แต่พบว่า outlier ส่งผลต่อค่า medoid ไม่มากเท่าค่าเฉลี่ย
- จึงมีแนวคิดการออกแบบวิธีการอัพเดท centroid ที่ใช้ค่า med แทนค่า mean เพื่อใช้กับงานที่มี outlier สูง
- เราเรียกวิธีการนี้ว่า k-Medoids

Variant ของ k-Means : k-Medoid

- ลองเขียนขั้นตอนของ k-Medoids โดยอ้างจากขั้นตอนของ k-Means

Variant ของ k-Means : k-Medoid

- เมื่อเราใช้กระบวนการอัพเดท medoid ที่แตกต่างจาก centroid (จาก mean เป็น med) วิธีการวัดประสิทธิภาพของ medoid (ที่ก่อนหน้านี้ใช้ SSE) จึงต้องเปลี่ยนเป็น SAE (Sum absolute error)

$$SAE = \sum_{i=1}^{k} \sum_{x \in C_i, x \notin M \; and \; c_m \in M} |x - c_m|$$

เมื่อ c_m แทน medoid ที่อัพเดทด้วยค่า med

M แทนเซ็ทของ medoid ทั้งหมดที่มี (อัพเดทด้วยค่า med)

x ข้อมูลทั้งหมดที่มีในชุดข้อมูลที่ไม่ใช่จุดเดียวกับ medoid. i.e. $x \in C_i, x \notin M$

Variant ของ k-Means : k-Medoid

- โดยทั่วไป k-medoids จะทำการอัพเดท medoid และทำกระบวนการซ้ำไปเรื่อย ๆ จนกว่าจุด medoid ทั้งหมดจะไม่มีการเปลี่ยนแปลง หรือเปลี่ยนแปลงน้อยกว่าค่าที่กำหนด
- การอัพเดทอย่างต่อเนื่องและหยุดอัพเดทโดยใช้เงื่อนไขนี้ เรียกว่า PAM (Partitioning around medoid)

Variant ของ k-Means : k-Medoid

- เทียบ PAM กับ k-Means
  - PAM มีความ robust กับข้อมูลที่มี outlier สูงกว่า k-Means
  - ความซับซ้อนของการคำนวณสูง
    - For each iteration, PAM consider $k(n-k)$ pairs of object o_i, o_j for which a cost $cost(o_i, o_j)$ determines. Calculating the cost during each iteration requires that the cost be calculated for all other non-medoids o_j. There are $n-k$ of these. Thus, the total time complexity per iteration is $n(n-k)^2$. The total number of iterations may be quite large.
  - ไม่เหมาะกับชุดข้อมูลที่มีขนาดใหญ่เนื่องจาก time complexity

Variant ของ k-Means : งานอื่น ๆ

There are a quite few variants of the k-Means algorithm. These can differ in the procedure of selecting the initial *k* means, the calculation of proximity and strategy for calculating cluster means. Another variants of k-means to cluster categorical data.

Few variant of k-Means algorithm includes

- Bisecting k-Means (addressing the issue of initial choice of cluster means).
  1. M. Steinbach, G. Karypis and V. Kumar "A comparison of document clustering techniques", *Proceedings of KDD workshop on Text mining,* 2000.
- Mean of clusters (Proposing various strategies to define means and variants of means).
  - B. zhan "Generalised k-Harmonic means – Dynamic weighting of data in unsupervised learning", *Technical report, HP Labs*, 2000.
  - A. D. Chaturvedi, P. E. Green, J. D. Carroll, "k-Modes clustering", *Journal of classification*, Vol. 18, PP. 35-36, 2001.
  - D. Pelleg, A. Moore, "x-Means: Extending k-Means with efficient estimation of the number of clusters", *17th International conference on Machine Learning,* 2000.

Variant ของ k-Means : งานอื่น ๆ

- N. B. Karayiannis, M. M. Randolph, “Non-Euclidean c-Means clustering algorithm”, *Intelligent data analysis journal*, Vol 7(5), PP 405-425, 2003.
- V. J. Olivera, W. Pedrycy, “Advances in Fuzzy clustering and its applications”, Edited book. John Wiley [2007]. (Fuzzy c-Means algorithm).
- A. K. Jain and R. C. Bubes, “Algorithms for clustering Data”, Prentice Hall, 1988.

  Online book at http://www.cse.msu.edu/~jain/clustering_Jain_Dubes.pdf
- A. K. Jain, M. N. Munty and P. J. Flynn, “Data clustering: A Review”, *ACM computing surveys*, 31(3), 264-323 [1999]. Also available online.

ตัวอย่างโค้ด

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8, *, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd'*)

Parameters::

n_clusters : *int, default=8*

The number of clusters to form as well as the number of centroids to generate.

init : *{'k-means++', 'random'}, callable or array-like of shape (n_clusters, n_features), default='k-means++'*

Method for initialization:

'k-means++' : selects initial cluster centroids using sampling based on an empirical probability distribution of the points' contribution to the overall inertia. This technique speeds up convergence, and is theoretically proven to be $\mathcal{O}(\log k)$-optimal. See the description of `n_init` for more details.

'random': choose `n_clusters` observations (rows) at random from data for the initial centroids.

If an array is passed, it should be of shape (n_clusters, n_features) and gives the initial centers.

If a callable is passed, it should take arguments X, n_clusters and a random state and return an initialization.

n_init : *int, default=10*

Number of time the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.

max_iter : *int, default=300*

Maximum number of iterations of the k-means algorithm for a single run.

ตัวอย่างโค้ด

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8, *, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd'*)

Attributes::

cluster_centers_ : *ndarray of shape (n_clusters, n_features)*
Coordinates of cluster centers. If the algorithm stops before fully converging (see `tol` and `max_iter`), these will not be consistent with `labels_`.

labels_ : *ndarray of shape (n_samples,)*
Labels of each point

inertia_ : *float*
Sum of squared distances of samples to their closest cluster center, weighted by the sample weights if provided.

n_iter_ : *int*
Number of iterations run.

n_features_in_ : *int*
Number of features seen during fit.

New in version 0.24.

feature_names_in_ : *ndarray of shape (`n_features_in_`,)*
Names of features seen during fit. Defined only when `X` has feature names that are all strings.

ตัวอย่างโค้ด

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8, *, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd'*)

```
>>> from sklearn.cluster import KMeans
>>> import numpy as np
>>> X = np.array([[1, 2], [1, 4], [1, 0],
...               [10, 2], [10, 4], [10, 0]])
>>> kmeans = KMeans(n_clusters=2, random_state=0).fit(X)
>>> kmeans.labels_
array([1, 1, 1, 0, 0, 0], dtype=int32)
>>> kmeans.predict([[0, 0], [12, 3]])
array([1, 0], dtype=int32)
>>> kmeans.cluster_centers_
array([[10.,  2.],
       [ 1.,  2.]])
```

ตัวอย่างโค้ด

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8, *, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd'*)

`fit`(*X, y=None, sample_weight=None*) [source]

Compute k-means clustering.

Parameters::

X : *{array-like, sparse matrix} of shape (n_samples, n_features)*
Training instances to cluster. It must be noted that the data will be converted to C ordering, which will cause a memory copy if the given data is not C-contiguous. If a sparse matrix is passed, a copy will be made if it's not in CSR format.

y : *Ignored*
Not used, present here for API consistency by convention.

sample_weight : *array-like of shape (n_samples,), default=None*
The weights for each observation in X. If None, all observations are assigned equal weight.

New in version 0.20.

Returns::

self : *object*
Fitted estimator.

ตัวอย่างโค้ด

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8, *, init='kmeans++', n_init=10, max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd'*)

`fit_predict`(*X, y=None, sample_weight=None*) [source]

Compute cluster centers and predict cluster index for each sample.

Convenience method; equivalent to calling fit(X) followed by predict(X).

Parameters::

X : *{array-like, sparse matrix} of shape (n_samples, n_features)*
New data to transform.

y : *Ignored*
Not used, present here for API consistency by convention.

sample_weight : *array-like of shape (n_samples,), default=None*
The weights for each observation in X. If None, all observations are assigned equal weight.

Returns::

labels : *ndarray of shape (n_samples,)*
Index of the cluster each sample belongs to.

สรุป

- ทบทวน Clustering
- Proximity measurement (Ordinal attribute; Distance)
- K-mean Clustering
  - แนวคิดของ K-mean
  - กระบวนการ
  - ความจริงเกี่ยวกับ k-Means
  - Variant ของ k-Means