AI VIET NAM
@aivietnam.edu.vn

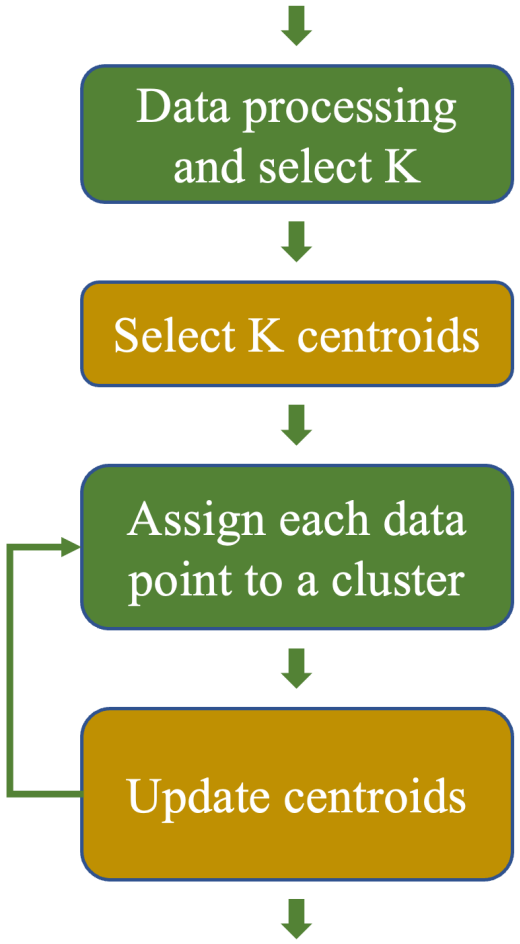
K-Means

Quang-Vinh Dinh
Ph.D. in Computer Science

Year 2024

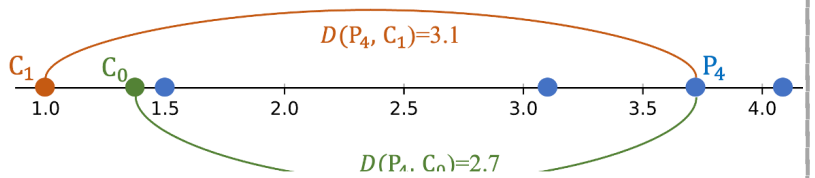
Objectives

Introduction



Tabular Data

	Index	Length	Distance to C0	Distance to C1	Centroid
C_0	0	1.4	0.0	0.4	0
C_1	1	1	0.4	0.0	1
	2	1.5	0.1	0.5	-
	3	3.1	1.7	2.1	-
	4	3.7	2.4	2.8	-
	5	4.1	2.7	3.1	-



Image&Text Data

Outline

SECTION 1

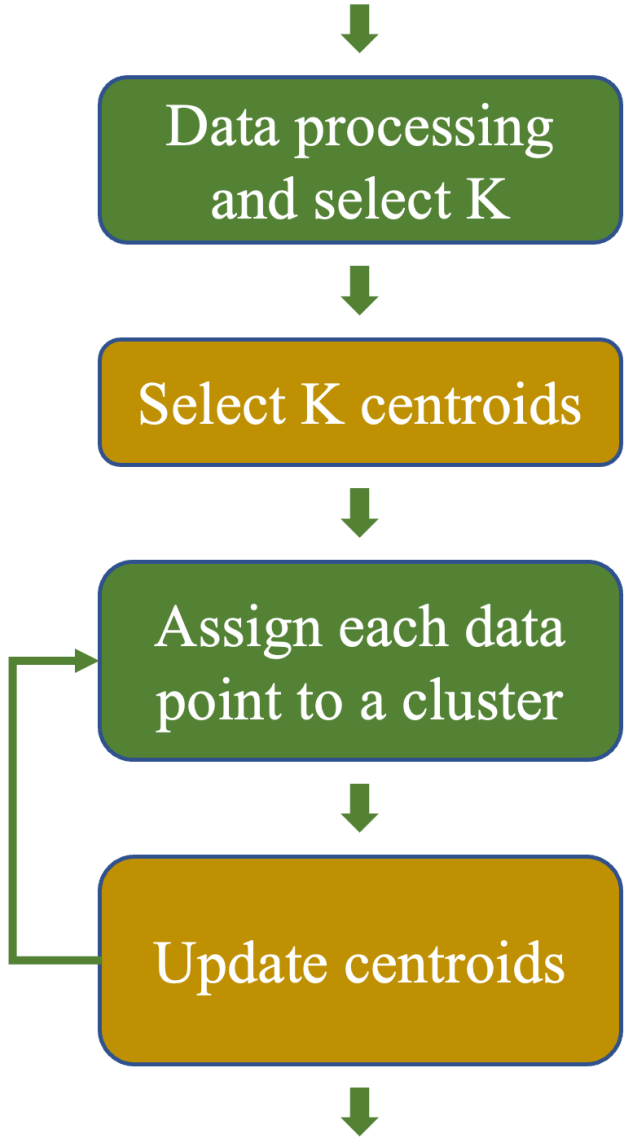
Introduction

SECTION 2

Tabular Data

SECTION 3

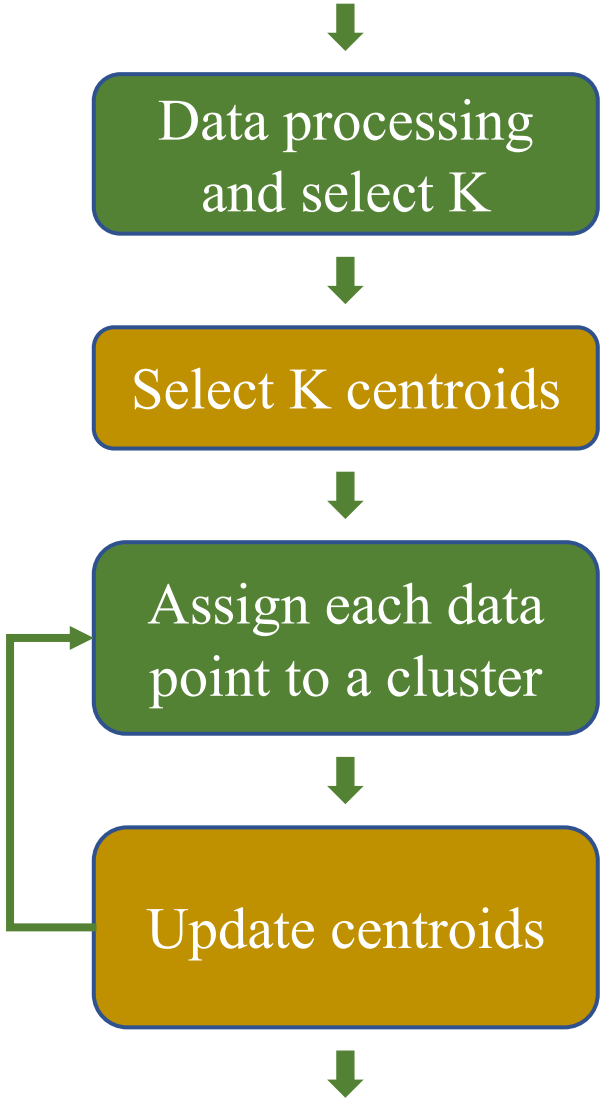
Image&Text Data



K-Means

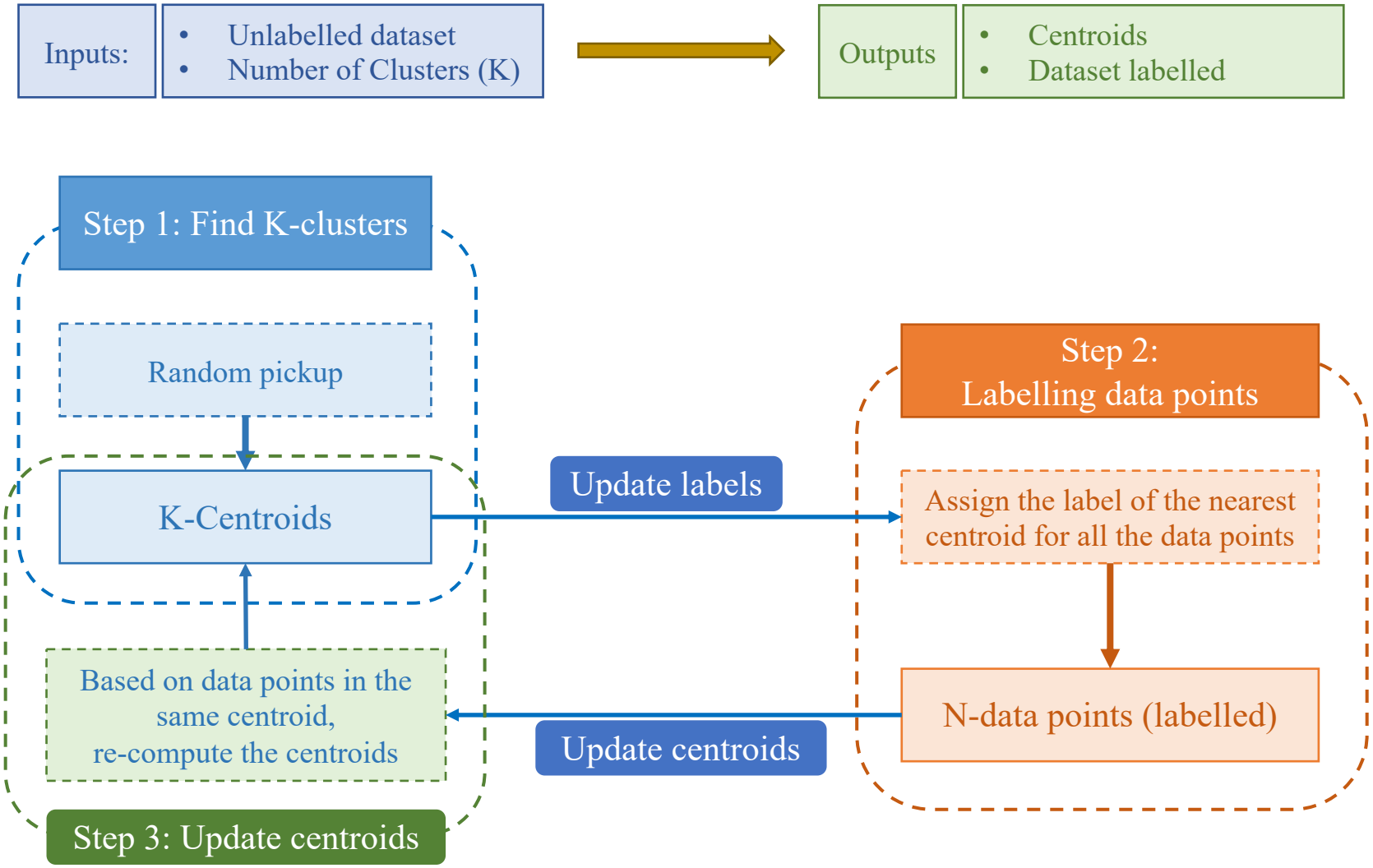
❖ Procedure

1. Initialize the value of k
2. Select random K centroids.
3. Assign each data point to the closest centroid
4. Calculate a new centroid of each cluster
5. If not convergent, go to step 3. If yes, stop.



K-Means

❖ K-means algorithm



K-Means

❖ Formulae

Given $(x_1, \dots, x_n)$ where $x_i \in \mathbb{R}^d$

Given an initial set of k centroids $c_0^{(1)}, c_1^{(1)}, \dots$

Let $S = (S_0, S_1, \dots)$ be k clusters

$$\underset{S}{\operatorname{argmin}} \sum_{i=0}^{k-1} \sum_{x \in S_i} \|x - c_i\|^2$$

Norm

$$\|\vec{u}\| = \sqrt{u_1^2 + \cdots + u_n^2}$$

Squared norm

$$\|\vec{u}\|^2 = u_1^2 + \cdots + u_n^2$$

K-Means

❖ Formulae

Given $(x_1, \dots, x_n)$ where $x_i \in \mathbb{R}^d$

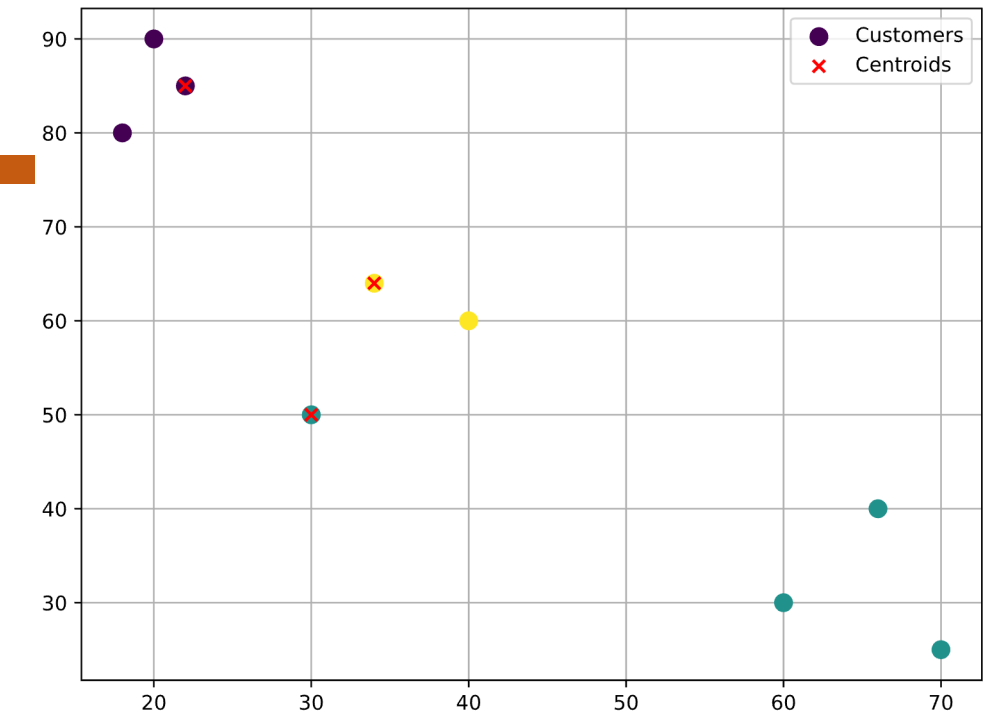
Given an initial set of k centroids $c_0^{(1)}, c_1^{(1)}, \dots$

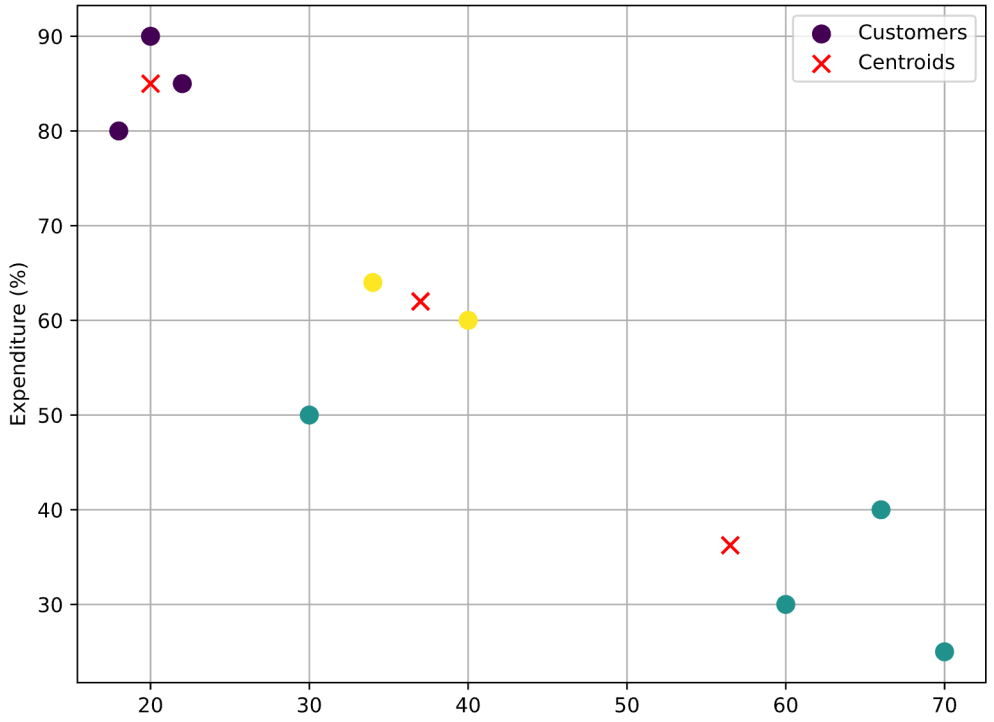
Assignment step

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - c_i^{(t)} \right\|^2 \leq \left\| x_p - c_j^{(t)} \right\|^2 \forall j, 0 \leq j < k \right\}$$

Update step

$$c_i^{(t+1)} = \frac{1}{\left| S_i^{(t)} \right|} \sum_{x \in S_i^{(t)}} x$$





https://en.wikipedia.org/wiki/K-means_clustering

Outline

SECTION 1

Introduction

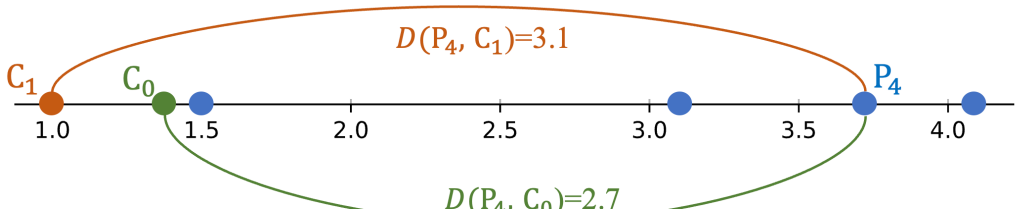
SECTION 2

Tabular Data

SECTION 3

Image&Text Data

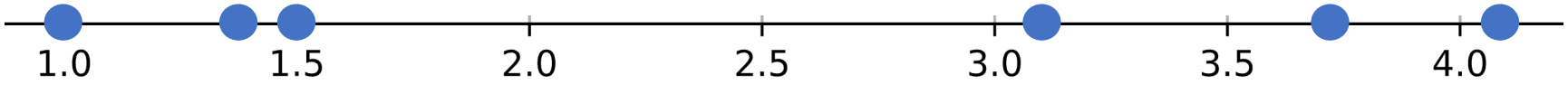
	Index	Length	Distance to C0	Distance to C1	Centroid
C_0	0	1.4	0.0	0.4	0
C_1	1	1	0.4	0.0	1
	2	1.5	0.1	0.5	-
	3	3.1	1.7	2.1	-
	4	3.7	2.4	2.8	-
	5	4.1	2.7	3.1	-



K-Means

❖ Simple Example

Index	Length
0	1.4
1	1.0
2	1.5
3	3.1
4	3.7
5	4.1



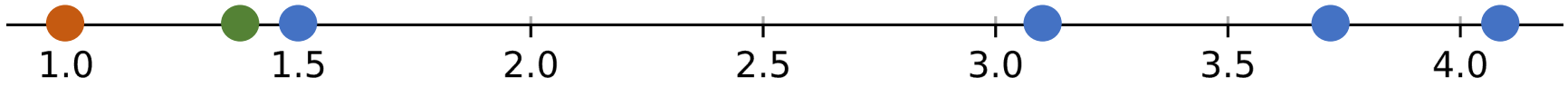
$k = 2$

$C_0 = 1.4$

$C_1 = 1.0$

Index	Length	
1	1.4	centroid 0
2	1	centroid 1
3	1.5	
4	3.1	
5	3.7	
6	4.1	

1. Initialize k centroids (randomly)



K-Means

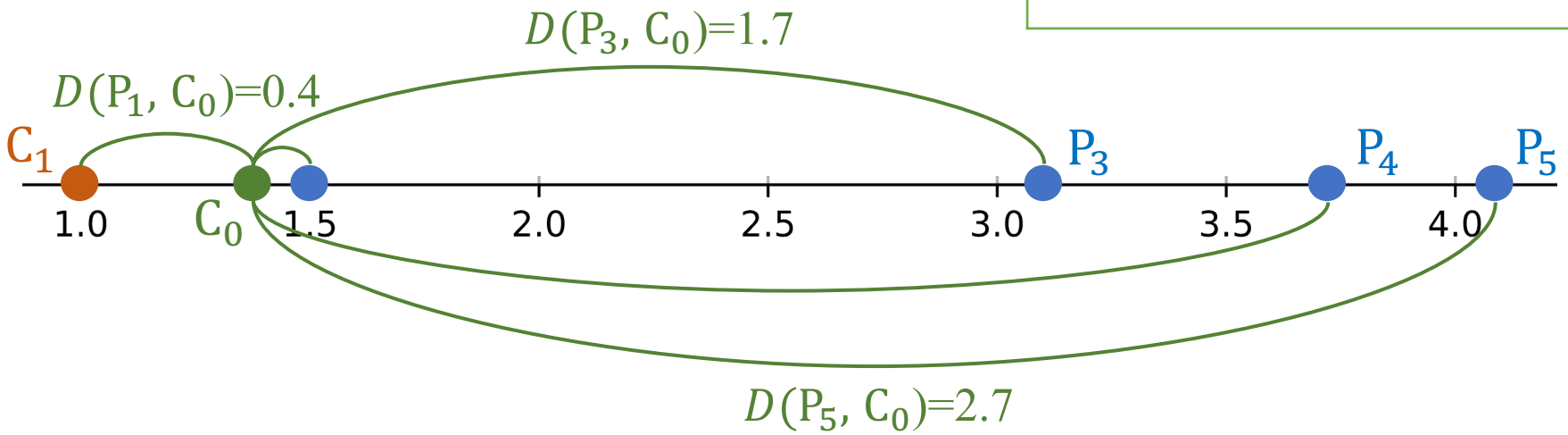
❖ Simple Example

1. Initialize k centroids (randomly)

2. Compute distances to each centroids

$k = 2$ $\quad C_0 = 1.4$ $\quad C_1 = 1.0$

	Index	Length	Distance to C0	Distance to C1	Centroid
C_0	0	1.4	0.0	0.4	0
C_1	1	1	0.4	0.0	1
	2	1.5	0.1	0.5	-
	3	3.1	1.7	2.1	-
	4	3.7	2.4	2.8	-
	5	4.1	2.7	3.1	-



K-Means

❖ Simple Example

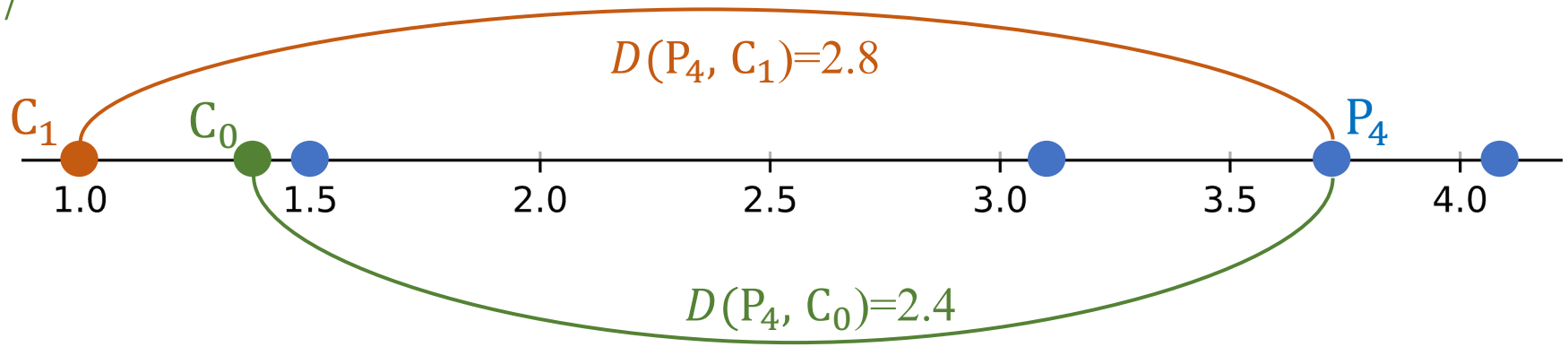
1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point

$k = 2$ $\quad C_0 = 1.4$ $\quad C_1 = 1.0$

	Index	Length	Distance to C0	Distance to C1	Centroid	Label
C_0	0	1.4	0.0	0.4	0	0
C_1	1	1	0.4	0.0	1	1
	2	1.5	0.1	0.5	-	
	3	3.1	1.7	2.1	-	
	4	3.7	2.4	2.8	-	
	5	4.1	2.7	3.1	-	

$D(P_4, C_1)=3.1$

$D(P_4, C_0)=2.7$



K-Means

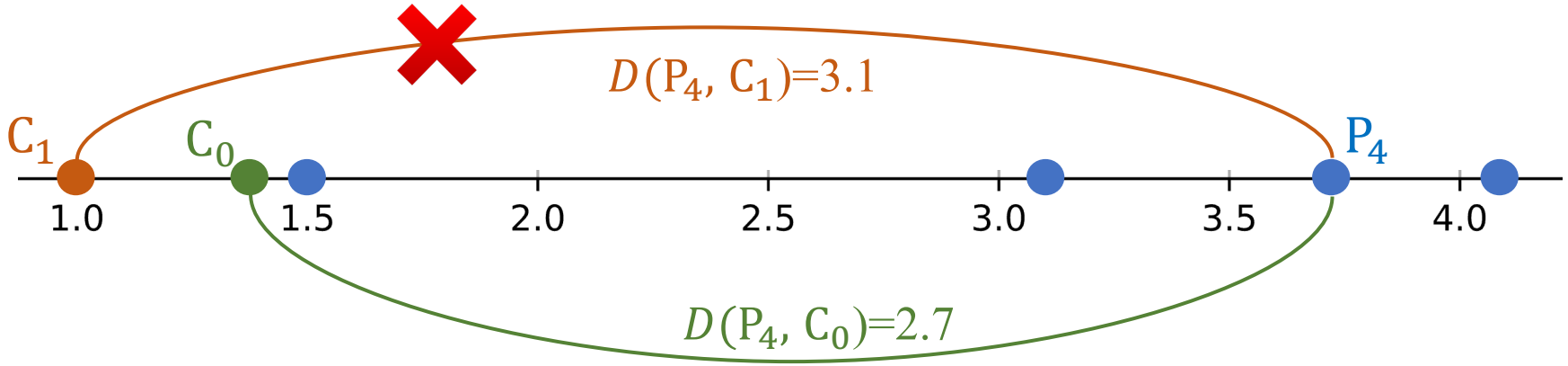
❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point

$k = 2$ $\quad C_0 = 1.4$ $\quad C_1 = 1.0$

	Index	Length	Distance to C0	Distance to C1	Centroid	Label
C_0	0	1.4	0.0	0.4	0	0
C_1	1	1	0.4	0.0	1	1
	2	1.5	0.1	0.5	-	
	3	3.1	1.7	2.1	-	
	4	3.7	2.4	2.8	-	0
	5	4.1	2.7	3.1	-	

$D(P_4, C_1) > D(P_4, C_0)$



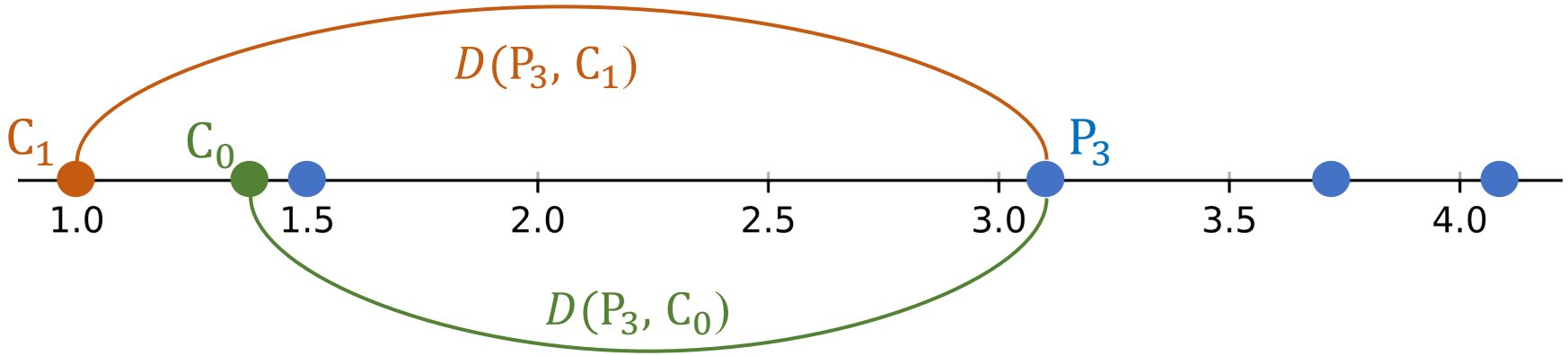
K-Means

❖ Simple Example: Quiz

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point

$k = 2$ $\quad C_0 = 1.4$ $\quad C_1 = 1.0$

	Index	Length	Distance to C0	Distance to C1	Centroid	Label
C_0	0	1.4	0.0	0.4	0	
C_1	1	1	0.4	0.0	1	
	2	1.5	0.1	0.5	-	
	3	3.1	1.7	2.1	-	?
	4	3.7	2.4	2.8	0	
	5	4.1	2.7	3.1	-	



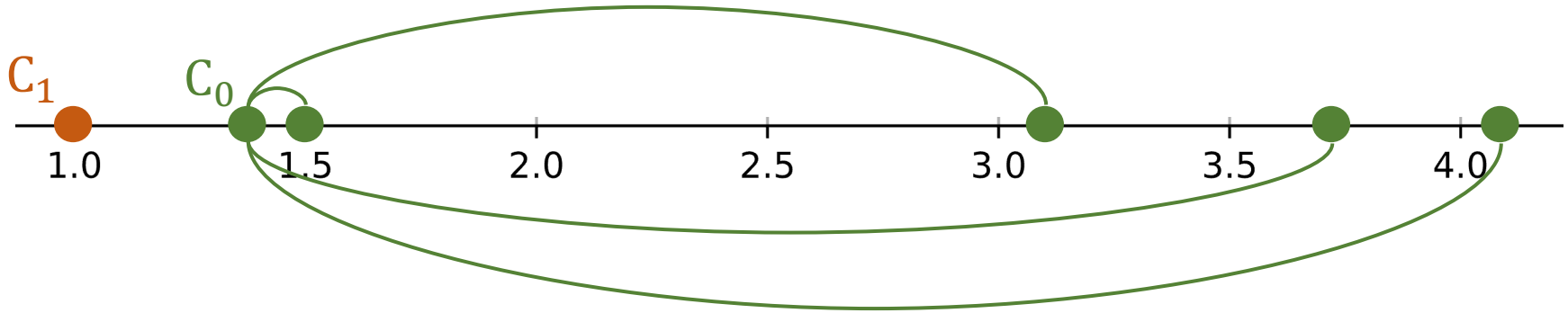
K-Means

❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point

$k = 2$ $\quad C_0 = 1.4$ $\quad C_1 = 1.0$

Index	Length	Distance to C0	Distance to C1	Centroid	Label
0	1.4	0.0	0.4	0	0
1	1	0.4	0.0	1	1
2	1.5	0.1	0.5	-	0
3	3.1	1.7	2.1	-	0
4	3.7	2.4	2.8	-	0
5	4.1	2.7	3.1	-	0



K-Means

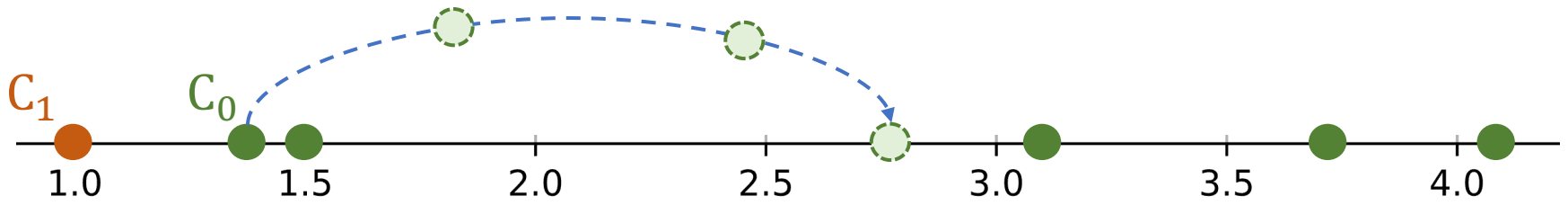
❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point
4. Update centroids

	Index	Length	Distance to C0	Distance to C1	Centroid	Label
C_0	0	1.4	0.0	0.4	0	0
C_1	1	1	0.4	0.0	1	1
	2	1.5	0.1	0.5	-	0
	3	3.1	1.7	2.1	-	0
	4	3.7	2.4	2.8	-	0
	5	4.1	2.7	3.1	-	0

$$C_0 = \frac{1.4 + 1.5 + 3.1 + 3.7 + 4.1}{5} = 2.78$$

$$C_1 = 1.0$$



K-Means

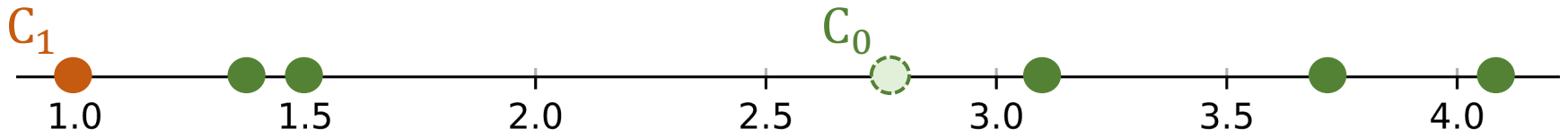
❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point
4. Update centroids

$C_0 = 2.78$

$C_1 = 1.0$

Index	Length	Distance to C0	Distance to C1	Label
0	1.4	-	-	0
1	1	-	-	1
2	1.5	-	-	0
3	3.1	-	-	0
4	3.7	-	-	0
5	4.1	-	-	0



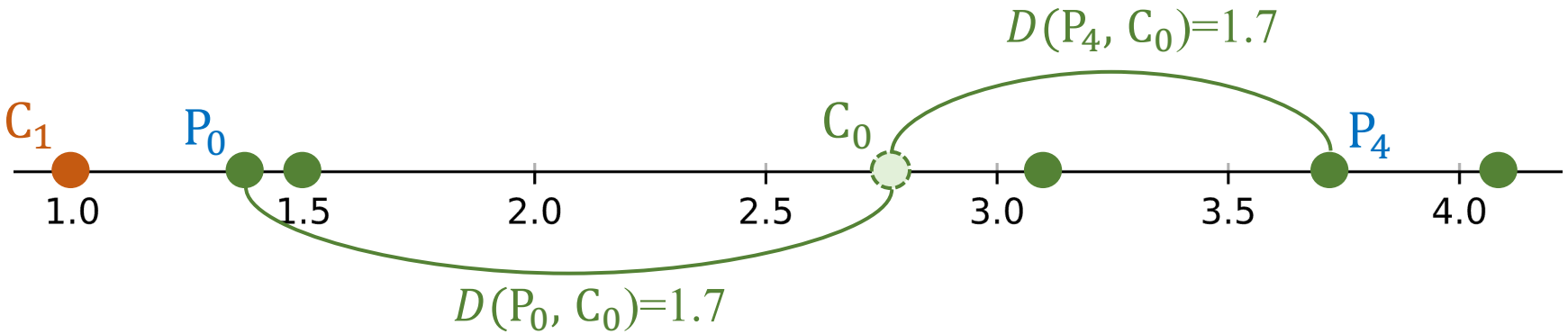
K-Means

❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point
4. Update centroids

Index	Length	Distance to C0	Distance to C1	Label
0	1.4	1.38	0.4	-
1	1	1.78	0.0	-
2	1.5	1.28	0.5	-
3	3.1	0.32	2.1	-
4	3.7	1.02	2.8	-
5	4.1	1.32	3.1	-

$C_0 = 2.78$ $C_1 = 1.0$



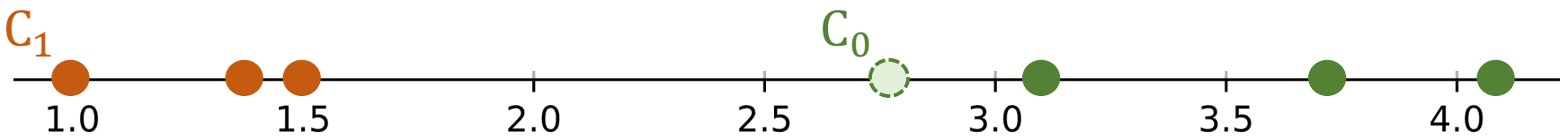
K-Means

❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. ➡ Assign a label to each data point
4. Update centroids

Index	Length	Distance to C0	Distance to C1	Label
0	1.4	1.38	0.4	1
1	1	1.78	0.0	1
2	1.5	1.28	0.5	1
3	3.1	0.32	2.1	0
4	3.7	1.02	2.8	0
5	4.1	1.32	3.1	0

$C_0 = 2.78$ $C_1 = 1.0$



K-Means

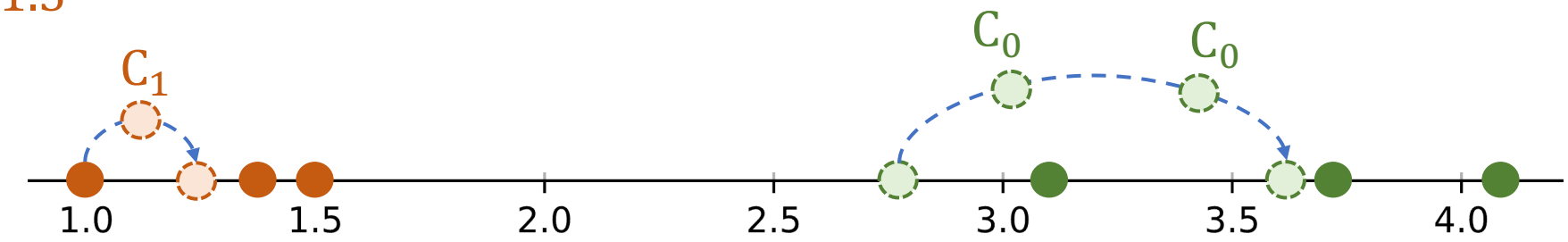
❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point
4. Update centroids

Index	Length	Distance to C0	Distance to C1	Label
0	1.4	1.38	0.4	1
1	1	1.78	0.0	1
2	1.5	1.28	0.5	1
3	**3.1**	0.32	2.1	0
4	**3.7**	1.02	2.8	0
5	**4.1**	1.32	3.1	0

$$C_0 = \frac{3.1 + 3.7 + 4.1}{3} = 3.67$$

$$C_1 = \frac{1.4 + 1.0 + 1.5}{3} = 1.3$$



K-Means

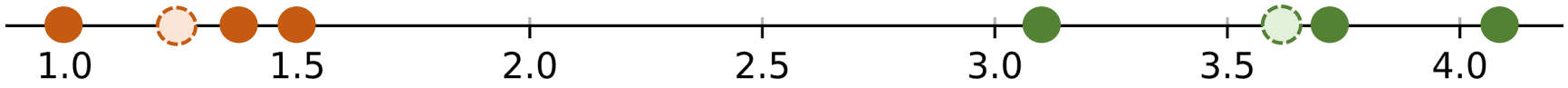
❖ Simple Example

1. Initialize k centroids (randomly)
2. Compute distances to each centroids
3. Assign a label to each data point
4. Update centroids

$$C_0 = \frac{3.1 + 3.7 + 4.1}{3} = 3.67$$

$$C_1 = \frac{1.4 + 1.0 + 1.5}{3} = 1.3$$

Index	Length	Distance to C0	Distance to C1	Label
0	1.4			1
1	1			1
2	1.5			1
3	**3.1**			0
4	**3.7**			0
5	**4.1**			0



K-Means

❖ Implementation using Numpy

X = | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

C = | 1.4 | 1.0 |

C_0 C_1

Distance to C0	Distance to C1
0.0	0.4
0.4	0.0
0.1	0.5
1.7	2.1
2.4	2.8
2.7	3.1

np.abs(1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 - 1.4)

0.0 | 0.4 | 0.1 | 1.7 | 2.4 | 2.7

np.abs(1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 - 1.0)

0.4 | 0.0 | 0.5 | 2.1 | 2.8 | 3.1

```
np.abs(X-C[0])
✓ 0.0s
array([0. , 0.4, 0.1, 1.7, 2.4, 2.7])

np.abs(X-C[1])
✓ 0.0s
array([0.4, 0. , 0.5, 2.1, 2.8, 3.1])
```

K-Means

❖ Implementation using Numpy

$X =$ | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

$C =$ | 1.4 | 1.0 |

C_0 C_1

Distance to C0	Distance to C1
0.0	0.4
0.4	0.0
0.1	0.5
1.7	2.1
2.4	2.8
2.7	3.1

```
X = np.array([1.4])
print(X.shape)
print(X)
```
✓ 0.0s
```
(1,)
[1.4]
```

```
print(C.shape)
```
✓ 0.0s
```
(2,)
```

```
np.abs(X-C)
```
✓ 0.0s
```
array([0. , 0.4])
```

```
print(X.shape)
print(X)
```
✓ 0.0s
```
(6,)
[1.4 1.  1.5 3.1 3.8 4.1]
```

```
print(C.shape)
```
✓ 0.0s
```
(2,)
```

```
np.abs(X-C)
```
⊗ 0.0s
```
---------------------------------
ValueError
Cell In[21], line 1
----> 1 np.abs(X-C)

ValueError: operands could not
```

K-Means

❖ Implementation using Numpy

X = | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

C = | 1.4 | 1.0 |

C_0 C_1

Distance to C0	Distance to C1
0.0	0.4
0.4	0.0
0.1	0.5
1.7	2.1
2.4	2.8
2.7	3.1

np.abs((6,1) | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 | - (2,) | 1.4 | 1.0 |)

np.abs(| 1.4 | 1.4 | 1.0 | 1.0 | 1.5 | 1.5 | 3.1 | 3.1 | 3.8 | 3.8 | 4.1 | 4.1 | - | 1.4 | 1.0 |)

```
print(C.shape)
print(X.shape)
```

✓ 0.0s

```
(2,)
(6,)
```

```
np.abs(X.reshape(-1, 1) - C)
```

✓ 0.0s

```
array([[0. , 0.4],
       [0.4, 0. ],
       [0.1, 0.5],
       [1.7, 2.1],
       [2.4, 2.8],
       [2.7, 3.1]])
```

K-Means

❖ Implementation using Numpy

X = | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

C = | 1.4 | 1.0 |

C_0 C_1

Distance to C0	Distance to C1
0.0	0.4
0.4	0.0
0.1	0.5
1.7	2.1
2.4	2.8
2.7	3.1

➡

Label
0
1
0
0
0
0

(1) np.argmin(D, axis=0)

(2) np.argmin(D, axis=1)

0	1	0	0	0	0

(a)

0	1

(b)

```
D
✓ 0.0s
array([[0. , 0.4],
       [0.4, 0. ],
       [0.1, 0.5],
       [1.7, 2.1],
       [2.4, 2.8],
       [2.7, 3.1]])

np.argmin(D, axis=0)
✓ 0.0s
array([0, 1])

np.argmin(D, axis=1)
✓ 0.0s
array([0, 1, 0, 0, 0, 0])
```

K-Means

❖ Implementation using Numpy

$C =$ | 1.4 | 1.0 |

C_0 C_1

$X =$ | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

$L =$ | 0 | 1 | 0 | 0 | 0 | 0 |

$C_{new} =$ | 2.78 | 1.0 |

C_0 C_1

$X =$ | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

$L =$ | 0 | 1 | 0 | 0 | 0 | 0 |

$X[L == 0] =$ | 1.4 | 1.5 | 3.1 | 3.8 | 4.1 | → mean() → 2.78

$X =$ | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

$L =$ | 0 | 1 | 0 | 0 | 0 | 0 |

$X[L == 1] =$ | 1.0 | → mean() → 1.0

```
np.array([X[L == i].mean() for i in range(k)])
```

K-Means

❖ Implementation using Numpy

$C =$ | 1.4 | 1.0 |

C_0 C_1

$X =$ | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

$L =$ | 0 | 1 | 0 | 0 | 0 | 0 |

$C_{new} =$ | 2.78 | 1.0 |

C_0 C_1

$(C == C_{new}) =$ | F | T |

```
C = np.array([1.4, 1.0])
C_new = np.array([2.78, 1.0])
print(C==C_new)
```

✓ 0.0s

```
[False  True]
```

```
print( np.all([True, True]) )
print( np.all([False, True]) )
print( np.all([True, False]) )
print( np.all([False, False]) )
```

✓ 0.0s

```
True
False
False
False
```

```
if np.all(C == C_new):
    break
```

np.all(| 1.4 | 1.0 | - | 2.78 | 1.0 |)

⬇

np.all(| F | T |)

⬇

| F |

K-Means

Norm $\|\vec{u}\| = \sqrt{u_1^2 + \cdots + u_n^2}$

Squared norm $\|\vec{u}\|^2 = u_1^2 + \cdots + u_n^2$

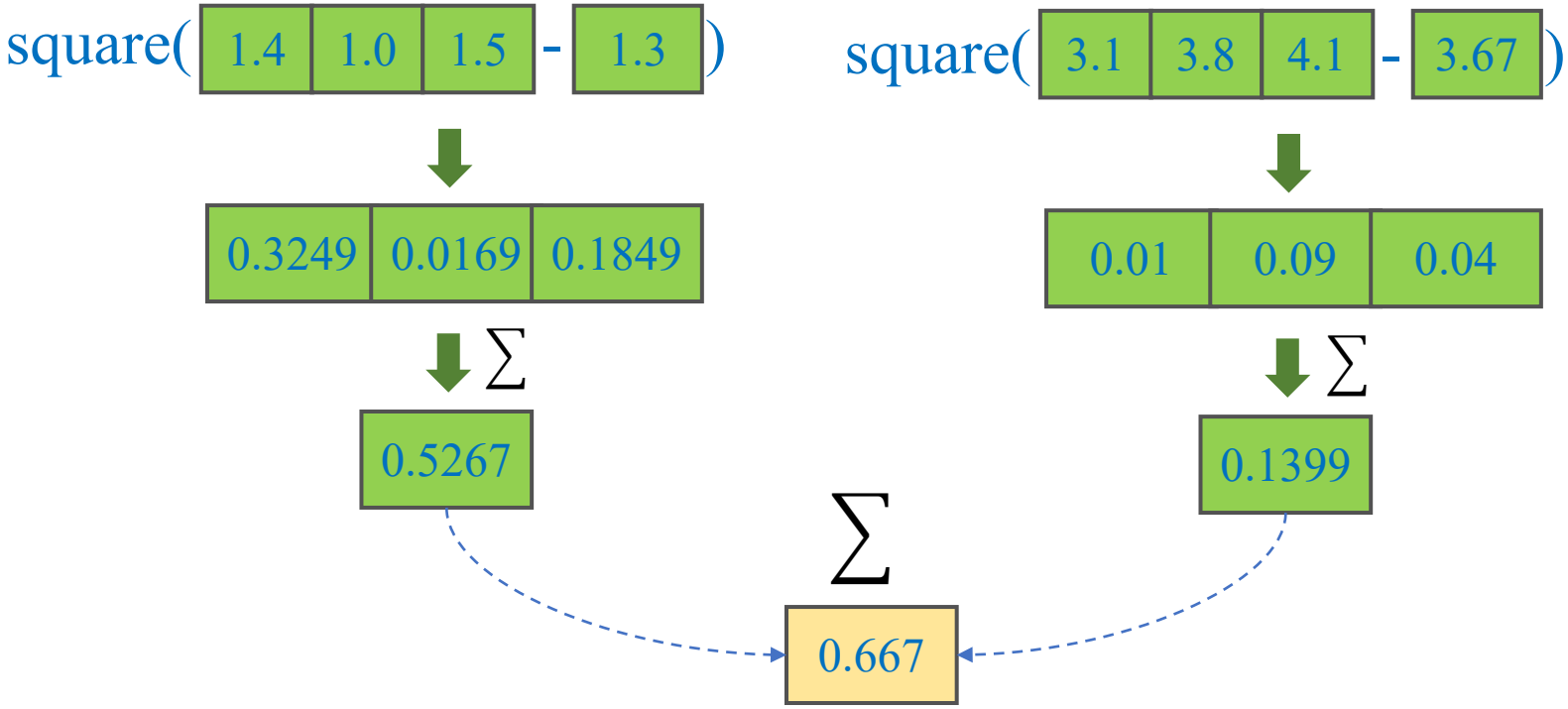
❖ Implementation using Numpy

Given $(x_1, \ldots, x_n)$ where $x_i \in \mathbb{R}^d$

$S = (S_0, S_1, \ldots)$

$$\text{WCSS} = \sum_{i=0}^{k-1} \sum_{x \in S_i} \|x - c_i\|^2$$

$$\text{WCSS} = \sum_{x \in S_0} \|x - c_0\|^2 + \sum_{x \in S_1} \|x - c_1\|^2$$



X = | 1.4 | 1.0 | 1.5 | 3.1 | 3.8 | 4.1 |

L = | 1 | 1 | 1 | 0 | 0 | 0 |

C = | 3.67 | 1.3 |

C_0 C_1

```
np.sum([np.sum(np.square(X[L == i] - C[i])) for i in range(k)])
```


Demo

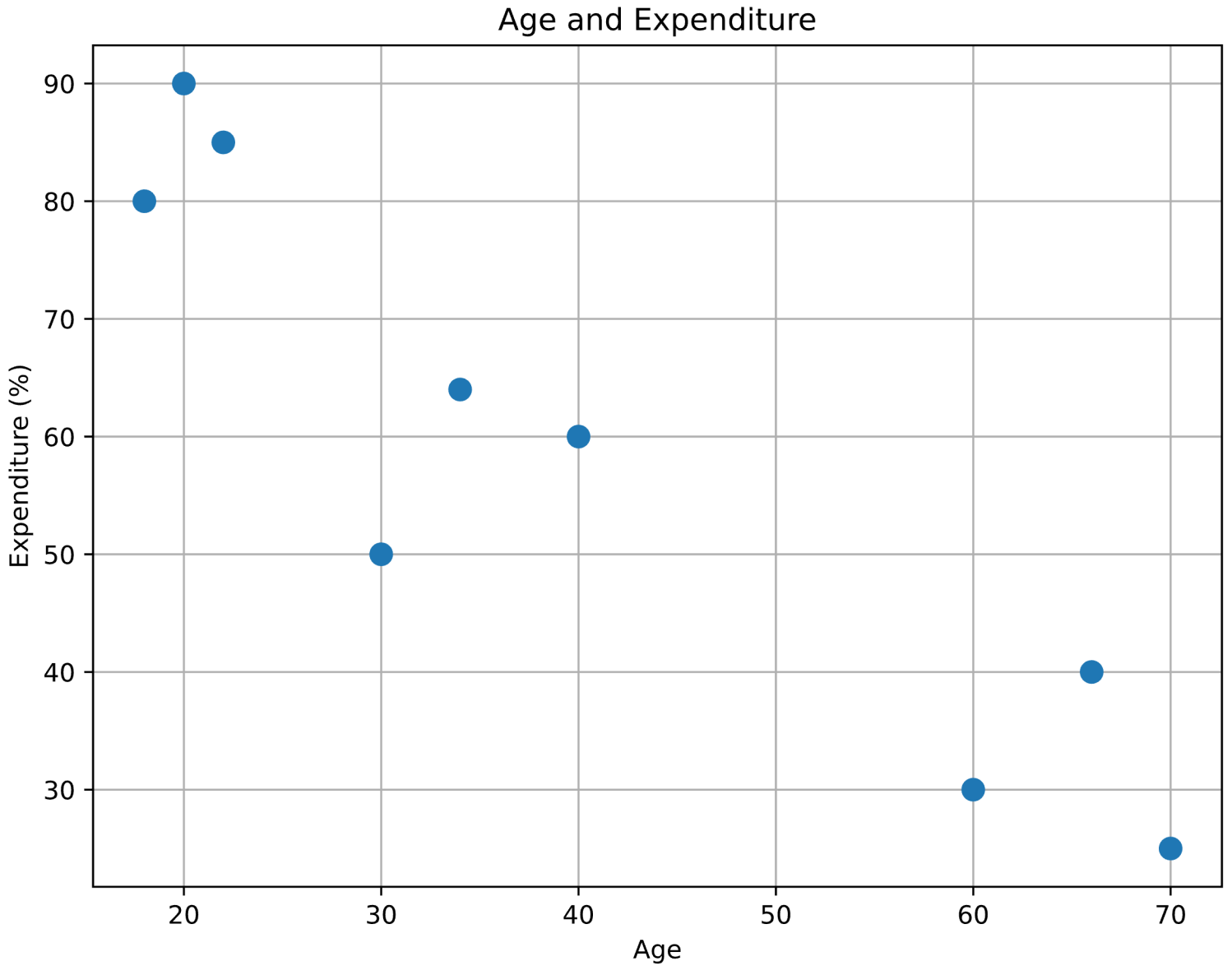
#Features=2
#clusters=3

K-Means

❖ Dataset

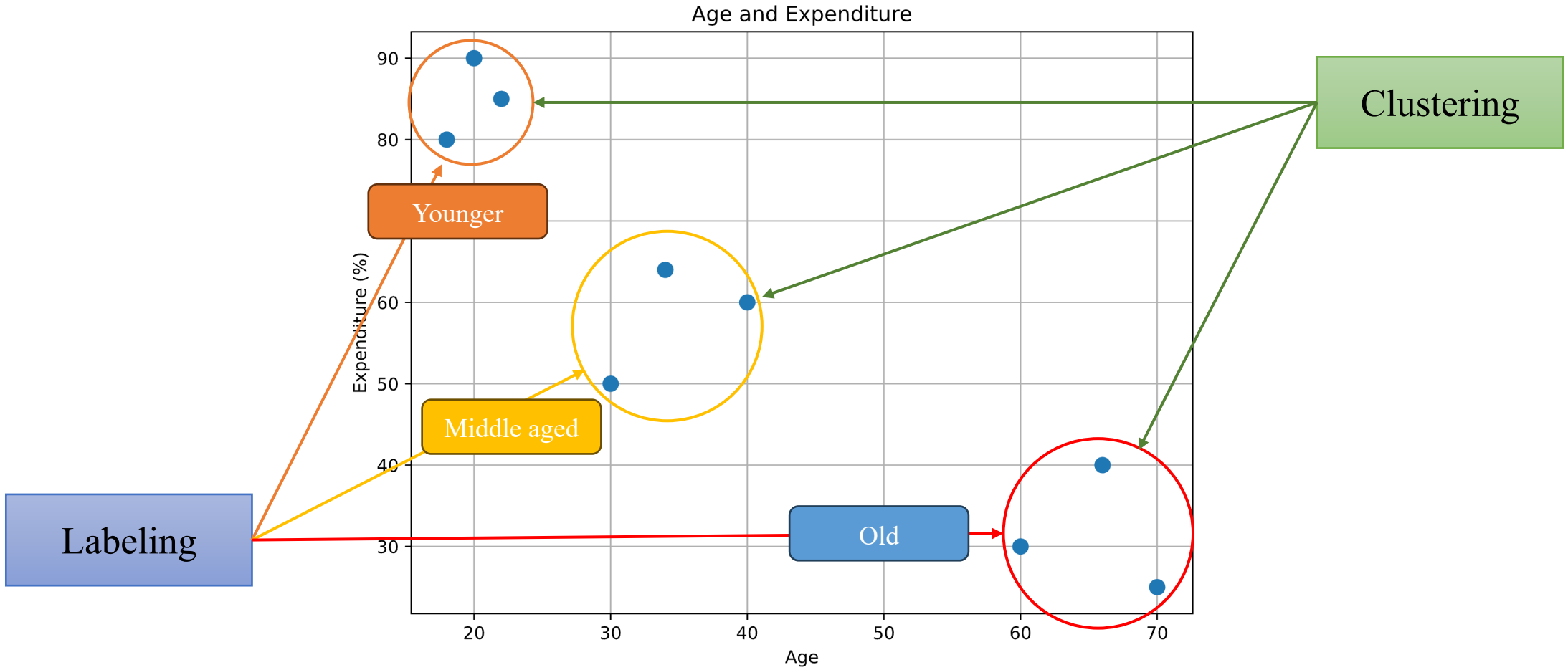
The dataset describes the relationship of monthly expenditure to age.

Index	Age	Expenditure
1	18	80
2	20	90
3	22	85
4	30	50
5	34	64
6	40	60
7	60	30
8	66	40
9	70	25



K-Means

❖ Targets



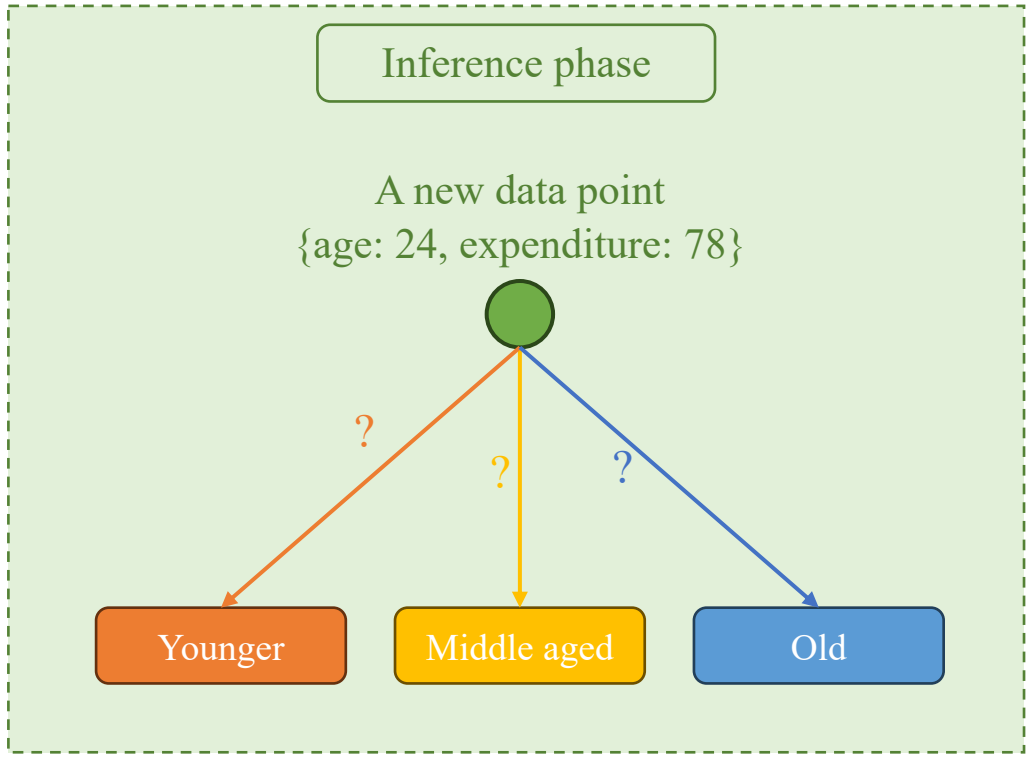
K-Means

❖ Targets

Index	Age	Expenditure
1	18	80
2	20	90
3	22	85
4	30	50
5	34	64
6	40	60
7	60	30
8	66	40
9	70	25

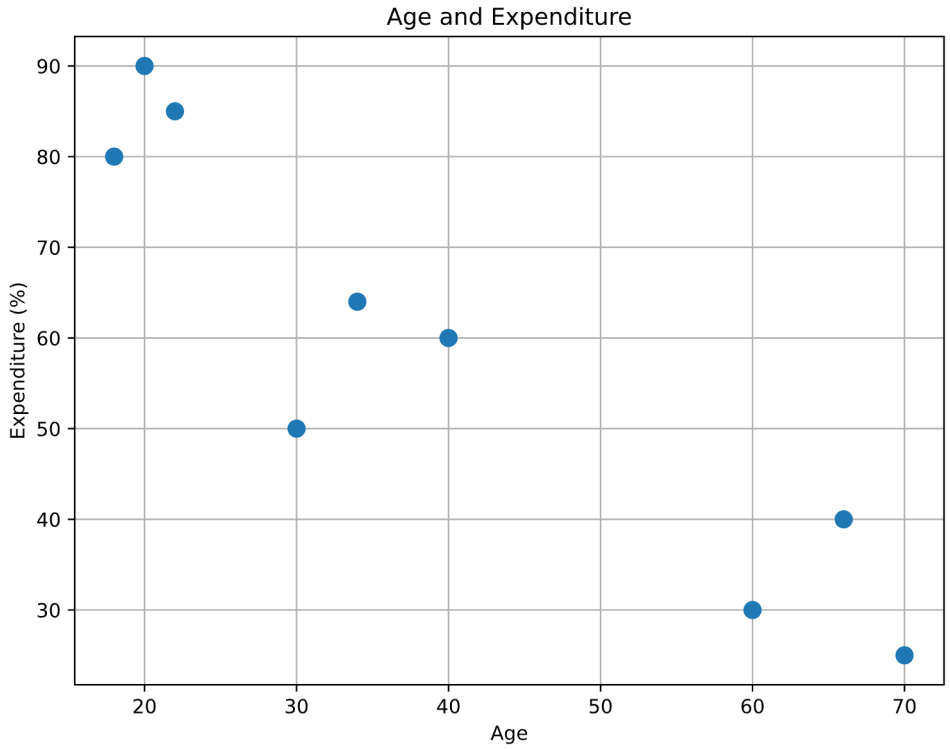
Training phase

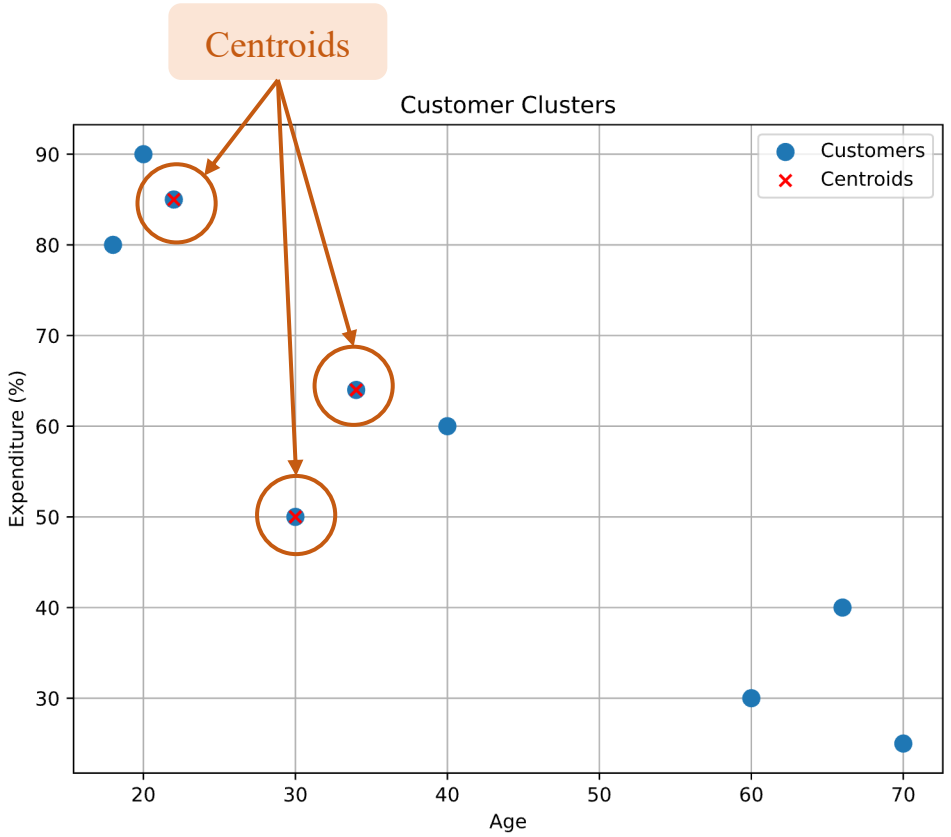
Cluster	Label
1	younger
1	younger
1	younger
2	middle-aged
2	middle-aged
2	middle-aged
3	old
3	old
3	old



K-Means

❖ Step 1: Init Centroids by random pickup



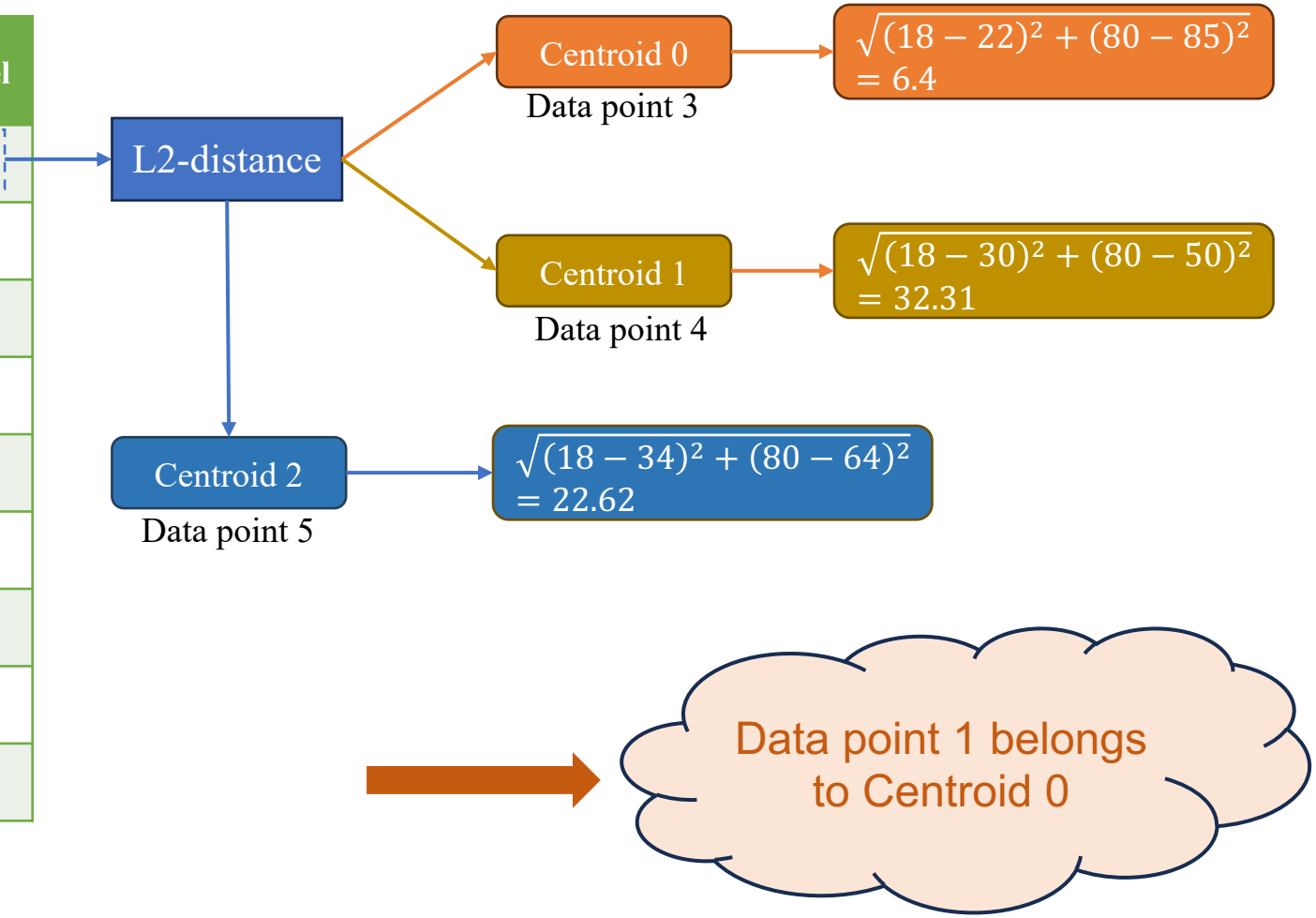


Index	Age	Expenditure
1	18	80
2	20	90
3	22	85
4	30	50
5	34	64
6	40	60
7	60	30
8	66	40
9	70	25

K-Means

❖ Step 2: Assign labels

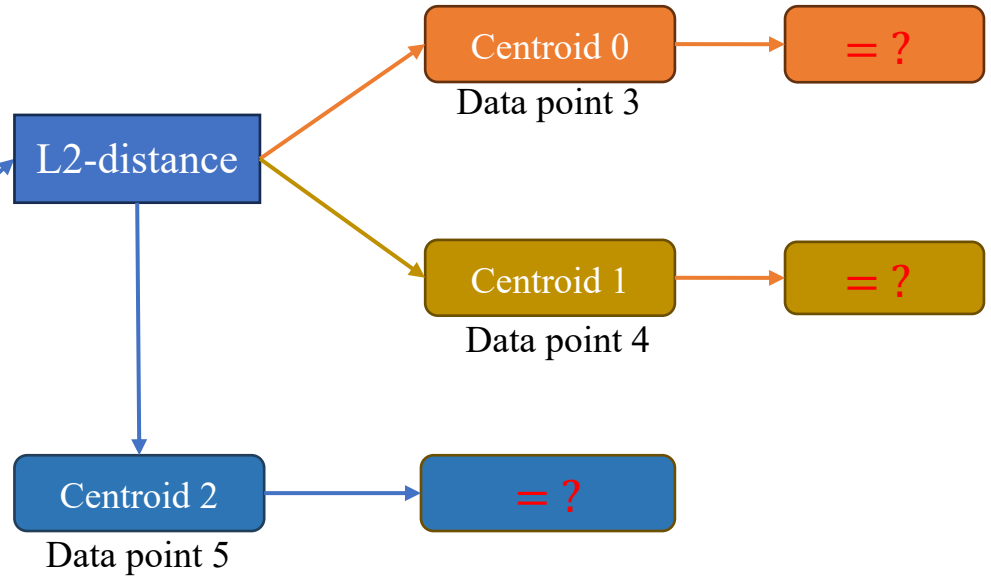
Index	Age	Expenditure	Centroid	Label
1	18	80	-	-
2	20	90	-	-
3	22	85	0	0
4	30	50	1	1
5	34	64	2	2
6	40	60	-	-
7	60	30	-	-
8	66	40	-	-
9	70	25	-	-

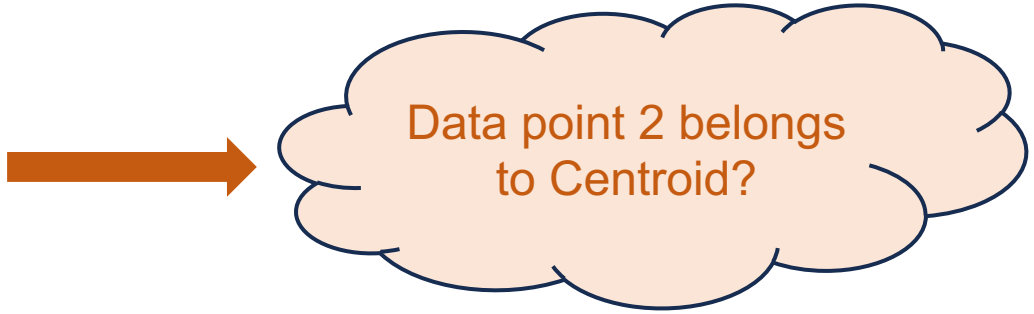


K-Means

❖ Step 2: Assign labels

Index	Age	Expenditure	Centroid	Label
1	18	80	-	-
2	20	90	-	-
3	22	85	0	0
4	30	50	1	1
5	34	64	2	2
6	40	60	-	-
7	60	30	-	-
8	66	40	-	-
9	70	25	-	-





K-Means

❖ Step 2: Assign labels

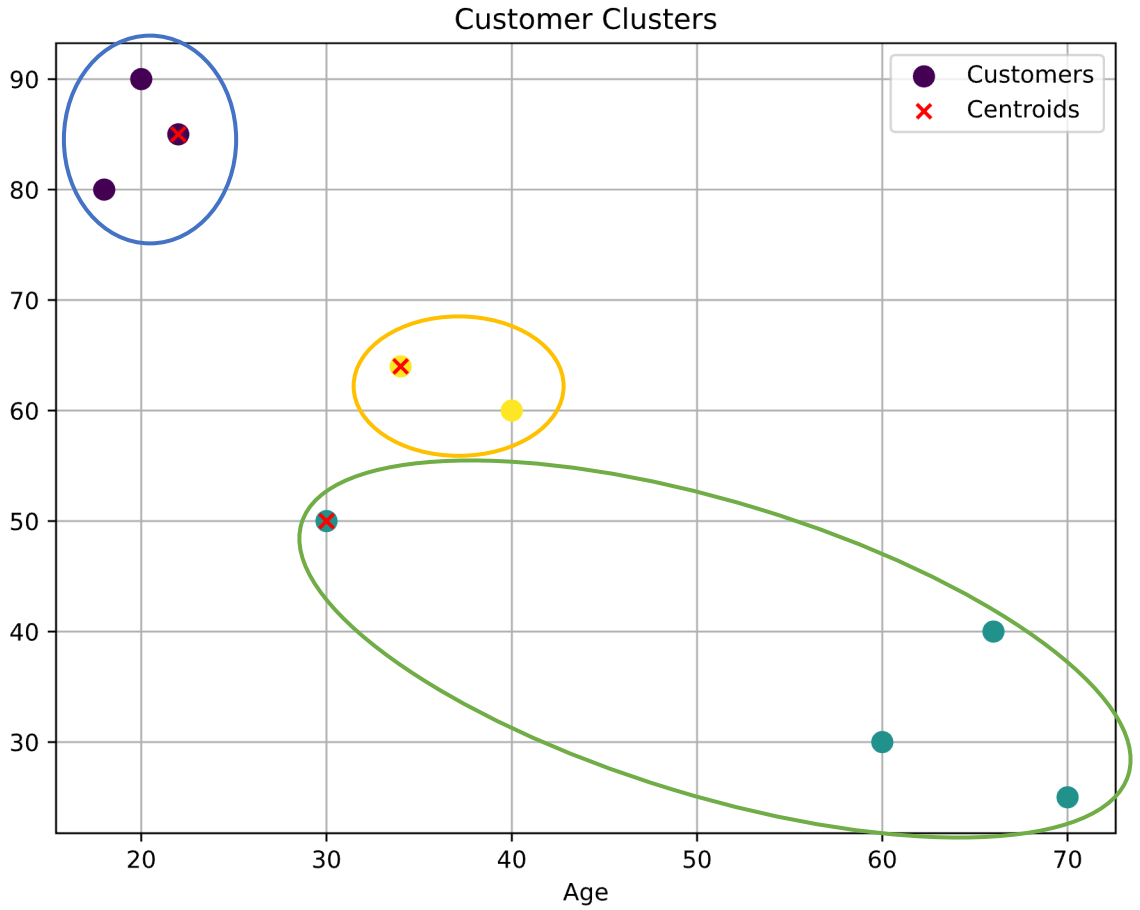
Index	Age	Expenditure	Centroid	Label	
1	18	80	-	-	L2-distance
2	20	90	-	-	L2-distance
3	22	85	0	0	
4	30	50	1	1	
5	34	64	2	2	
6	40	60	-	-	L2-distance
7	60	30	-	-	L2-distance
8	66	40	-	-	L2-distance
9	70	25	-	-	L2-distance

Index	Age	Expenditure	Centroid	Label
1	18	80	-	0
2	20	90	-	0
3	22	85	0	0
4	30	50	1	1
5	34	64	2	2
6	40	60	-	2
7	60	30	-	1
8	66	40	-	1
9	70	25	-	1

K-Means

❖ Step 2: Assign labels

Index	Age	Expenditure	Centroid	Label
1	18	80	-	0
2	20	90	-	0
3	22	85	0	0
4	30	50	1	1
5	34	64	2	2
6	40	60	-	2
7	60	30	-	1
8	66	40	-	1
9	70	25	-	1



K-Means

❖ Step 3: Update Centroids

Index	Age	Expenditure	Centroid	Label
1	18	80	-	0
2	20	90	-	0
3	22	85	0	0
4	30	50	1	1
5	34	64	2	2
6	40	60	-	2
7	60	30	-	1
8	66	40	-	1
9	70	25	-	1

New Centroid 0

$$\left(\frac{18+20+22}{3}, \frac{80+90+85}{3}\right) = (20, 85)$$

New Centroid 2

$$\left(\frac{34+40}{2}, \frac{64+60}{2}\right) = (37, 62)$$

$$\left(\frac{30+60+66+70}{4}, \frac{50+30+40+25}{4}\right) = (56.5, 36.25)$$

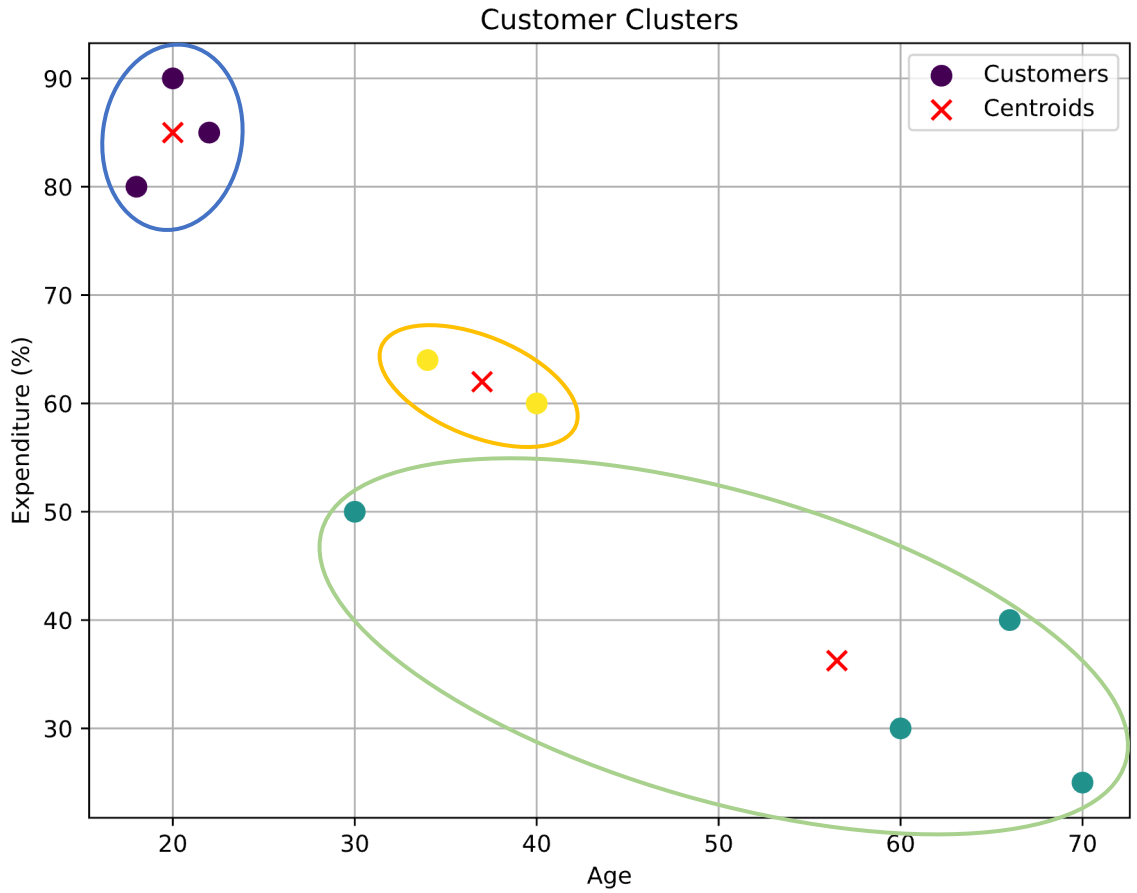
New Centroid 1

Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(56.5,36.25)	1
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1

K-Means

❖ Step 3: Update Centroids

Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(56.5,36.25)	1
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1



K-Means

❖ Repeat step 2: Calculate distance

Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(56.5,36.25)	1
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1

L2-distance

Centroid 0 → $\sqrt{(18-20)^2+(80-85)^2} = 5.38$

Centroid 1 → $\sqrt{(18-56.5)^2+(80-36.25)^2} = 58.27$

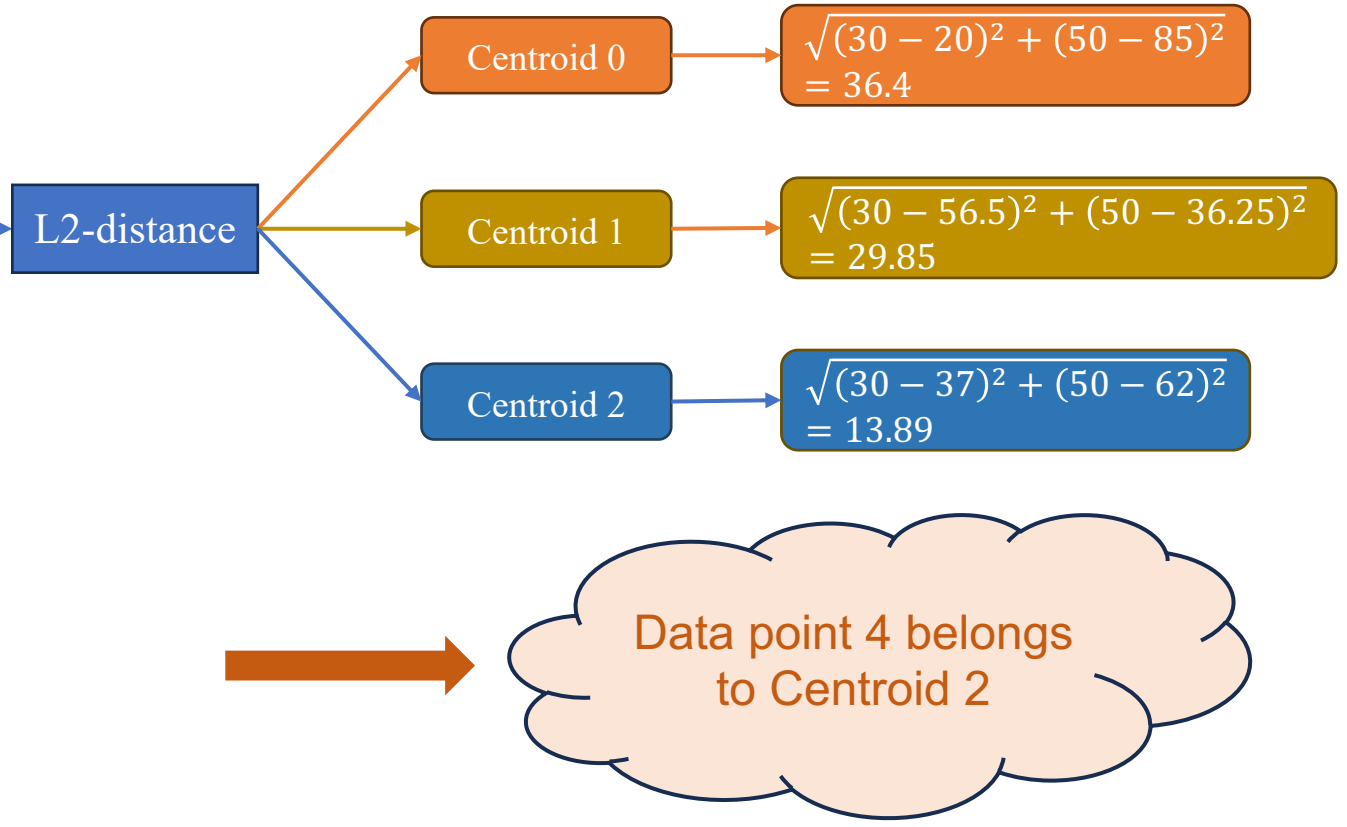
Centroid 2 → $\sqrt{(18-37)^2+(80-62)^2} = 26.17$

Data point 1 belongs to Centroid 0

K-Means

❖ Repeat step 2: Calculate distance

Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(56.5,36.25)	1
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1



K-Means

❖ Repeat step 3: Update Centroids

Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(37,62)	2
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1

New Centroid 0

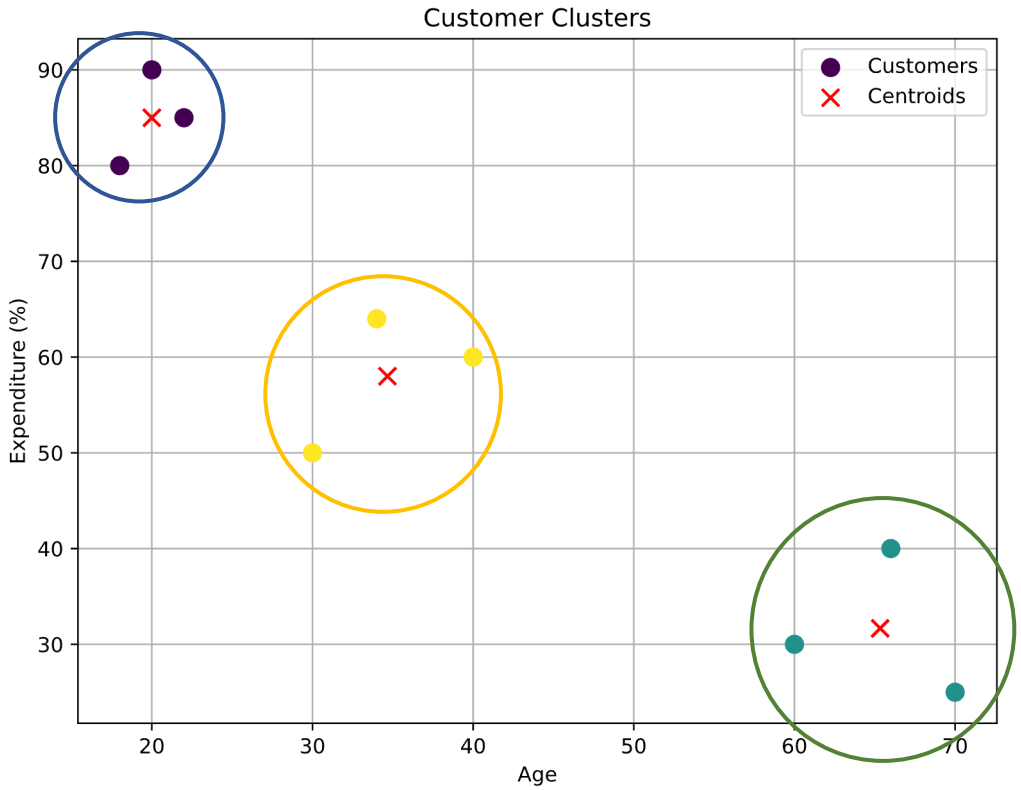
$$\left(\frac{18+20+22}{3}, \frac{80+90+85}{3}\right) = (20, 85)$$

New Centroid 2

$$\left(\frac{30+34+40}{3}, \frac{50+64+60}{3}\right) = (34.66, 58)$$

New Centroid 1

$$\left(\frac{60+66+70}{3}, \frac{30+40+25}{3}\right) = (65.33, 31.66)$$



K-Means

❖ Algorithm convergence

Repeat step 2 and step 3

Util "Centers" are not changed

OR

Limit iterations

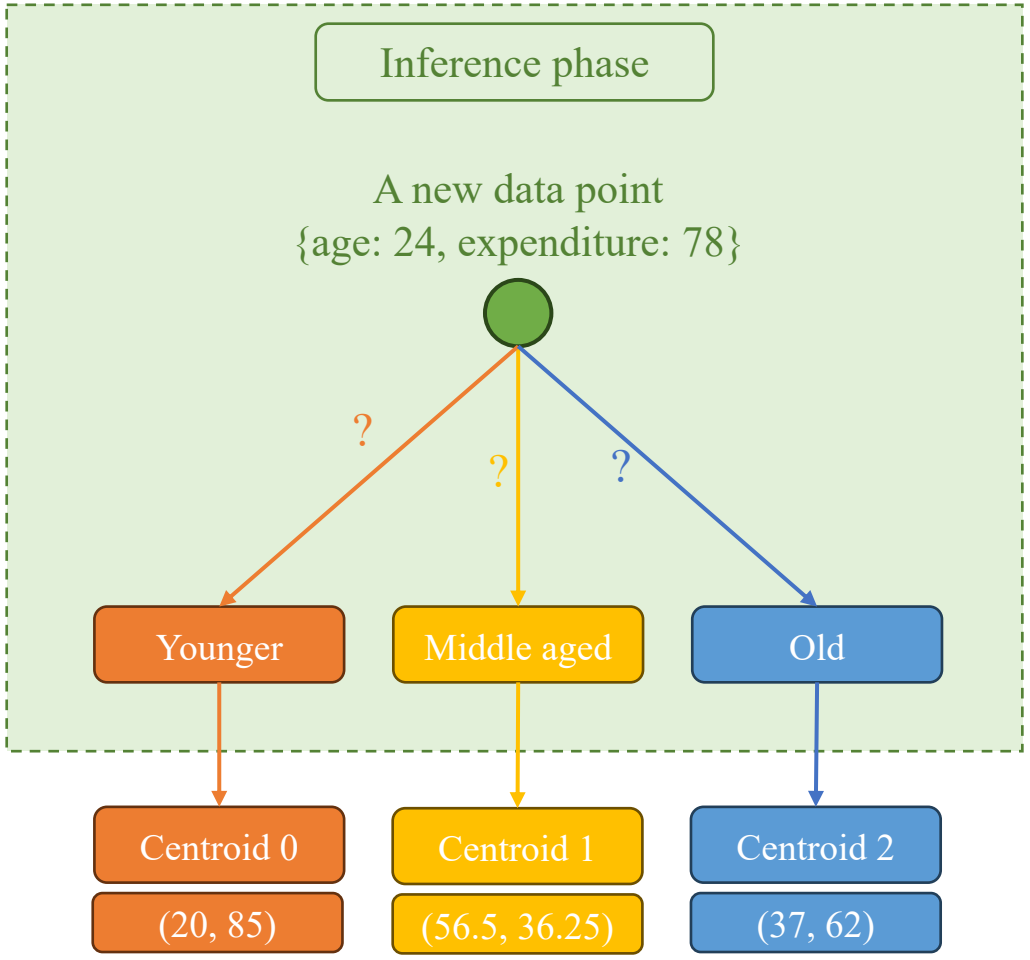
```python
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3, max_iter=10)
kmeans.fit(X)
```

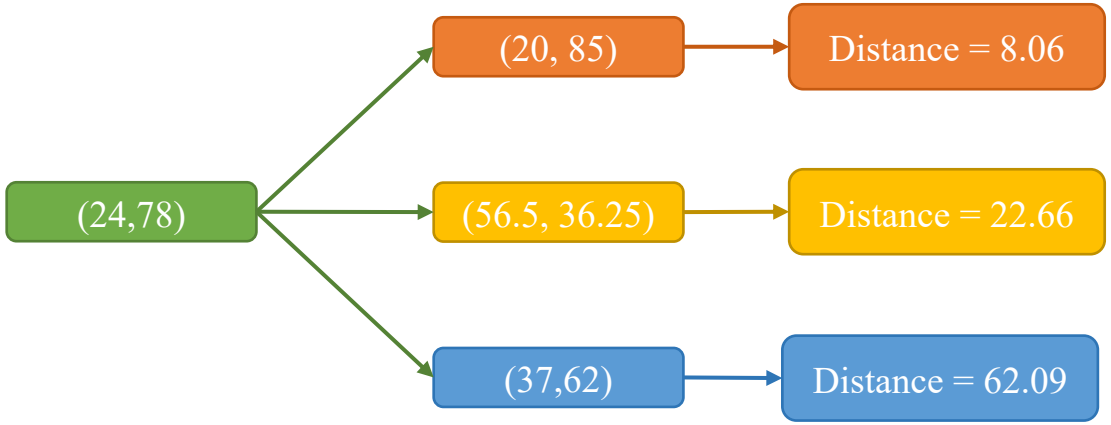
Convergence at iter = 2

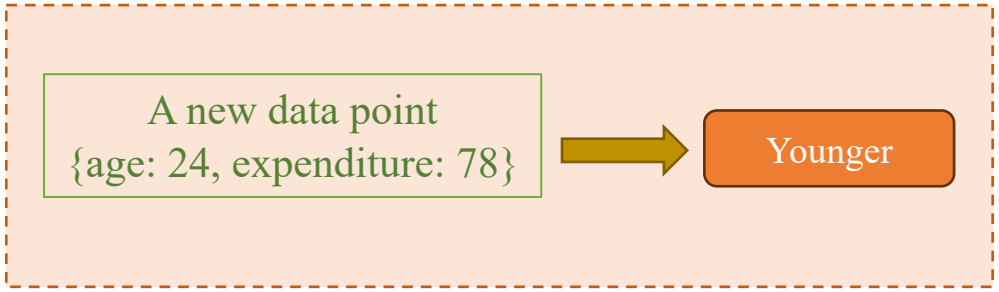
Index	Age	Expenditure	Centroid	Label
1	18	80	(20, 85)	0
2	20	90	(20, 85)	0
3	22	85	(20, 85)	0
4	30	50	(37,62)	2
5	34	64	(37,62)	2
6	40	60	(37,62)	2
7	60	30	(56.5,36.25)	1
8	66	40	(56.5,36.25)	1
9	70	25	(56.5,36.25)	1

K-Means

❖ Inference








QUIZ
QUIZ
TIME

Question 1

❖ Kết quả chương trình sau là gì?

```
data = np.random.rand(64, 2)
print(data.shape)

centroids = np.random.rand(5, 2)
print(centroids.shape)

distances = (data - centroids)**2
print(distances.shape)
```

✓ 0.0s

```
(64, 2)
(5, 2)
```

(a) (64,2)

(b) (64,5)

(c) (5,2)

(d) Khác

Question 2

❖ Kết quả chương trình sau là gì?

```
data = np.random.rand(64, 2)

data1 = data[:, np.newaxis, :]
data2 = data.reshape(1, 1, -1)

print(data1.shape)
print(data2.shape)
```

(a) (64, 1, 2) và (1, 128)

(b) (64, 2, 1) và (1, 1, 128)

(c) (64, 1, 2) và (1, 1, 128)

(d) Khác

Question 3

❖ Kết quả chương trình sau là gì?

```
data = np.random.rand(32, 4)
print(data.shape)

centroids = np.random.rand(3, 4)
print(centroids.shape)

distances = np.sqrt( (data[:, np.newaxis, :] - centroids)**2 )
print(distances.shape)
```

✓ 0.0s

```
(32, 4)
(3, 4)
```

(a) (32, 3)

(b) (32, 3, 4)

(c) (32, 4, 3)

(d) Khác

Question 4

- Cho array chứa **squared norm** giữa các feature của các điểm và của các centroid
- Ý nghĩa của các tham số như hình bên dưới.
- Tính distance giữa các point và các centroid (giải thích thêm)?

```
squared_norm = np.random.rand(32, 3, 4)
# 32 points
# 3 dimensions (3 features)
# 4 centroids
```

(a) np.sqrt(squared_norm.sum(axis=0))

(b) np.sqrt(squared_norm.sum(axis=1))

(c) np.sqrt(squared_norm.sum(axis=2))

(d) Khác

Question 5

- Cho array chứa thông tin **(features-centroids)**
- Ý nghĩa của các tham số như hình bên dưới.
- Tính distance giữa các point và các centroid (giải thích thêm)?

```
distances_detail = np.random.rand(32, 3, 2)
# 32 points
# 3 centroids
# 2 dimensions (2 features)
```

(a) np.linalg.norm(distances_detail, axis=0)

(b) np.linalg.norm(distances_detail, axis=1)

(c) np.linalg.norm(distances_detail, axis=2)

(d) Khác

Question 6

❖ Kết quả chương trình sau là gì?

```
features = np.random.rand(32, 2)
# 32 points
# each point is of 2 dimensions (2 features)

# k=3
# labels contains numbers from 0 to 2
```

(a) np.array([data[labels == i].mean(axis=0) for i in range(k)])

(b) np.array([data[labels == i].mean(axis=1) for i in range(k)])

(c) np.array([data[labels == i].mean(axis=2) for i in range(k)])

(d) np.array([data[labels == i] for i in range(k)])

K-Means

❖ Implementation using Numpy

$$X = \begin{bmatrix} 18 & 80 \\ 20 & 90 \\ 22 & 85 \\ 30 & 50 \\ 34 & 64 \\ 40 & 60 \\ 60 & 30 \\ 66 & 40 \\ 70 & 25 \end{bmatrix} \qquad C = \begin{bmatrix} 22 & 85 \\ 30 & 50 \\ 34 & 64 \end{bmatrix}$$

Index	Age	Expenditure	Distance to C0	Distance to C1	Distance to C2
...	...	...	...	...	...

```
np.linalg.norm(X1 - centroids, axis=1)
✓ 0.0s
array([ 6.40312424, 32.31098884, 22.627417  ])
```

```
( (X1 - centroids)**2 ).sum(axis=1)**0.5
✓ 0.0s
array([ 6.40312424, 32.31098884, 22.627417  ])
```

$$\text{norm}\left(\begin{bmatrix} 18 & 80 \end{bmatrix} - \begin{bmatrix} 22 & 85 \\ 30 & 50 \\ 34 & 64 \end{bmatrix}, \text{axis=1}\right)$$

$$\text{norm}\left(\begin{bmatrix} 18 & 80 \\ 18 & 80 \\ 18 & 80 \end{bmatrix} - \begin{bmatrix} 22 & 85 \\ 30 & 50 \\ 34 & 64 \end{bmatrix}, \text{axis=1}\right)$$

K-Means

❖ Implementation using Numpy

$X =$

18	80
20	90
22	85
30	50
34	64
40	60
60	30
66	40
70	25

$C =$

22	85
30	50
34	64

Index	Age	Expenditure	Distance to C0	Distance to C1	Distance to C2
...	...	...	...	...	...

norm(X[:, np.newaxis, :] - centroids, axis=2)

18	80
20	90
22	85
30	50
34	64
40	60
60	30
66	40
70	25

(9,1,2)

22	85
30	50
34	64

(3,2)

K-Means

❖ Implementation using Numpy

$X =$

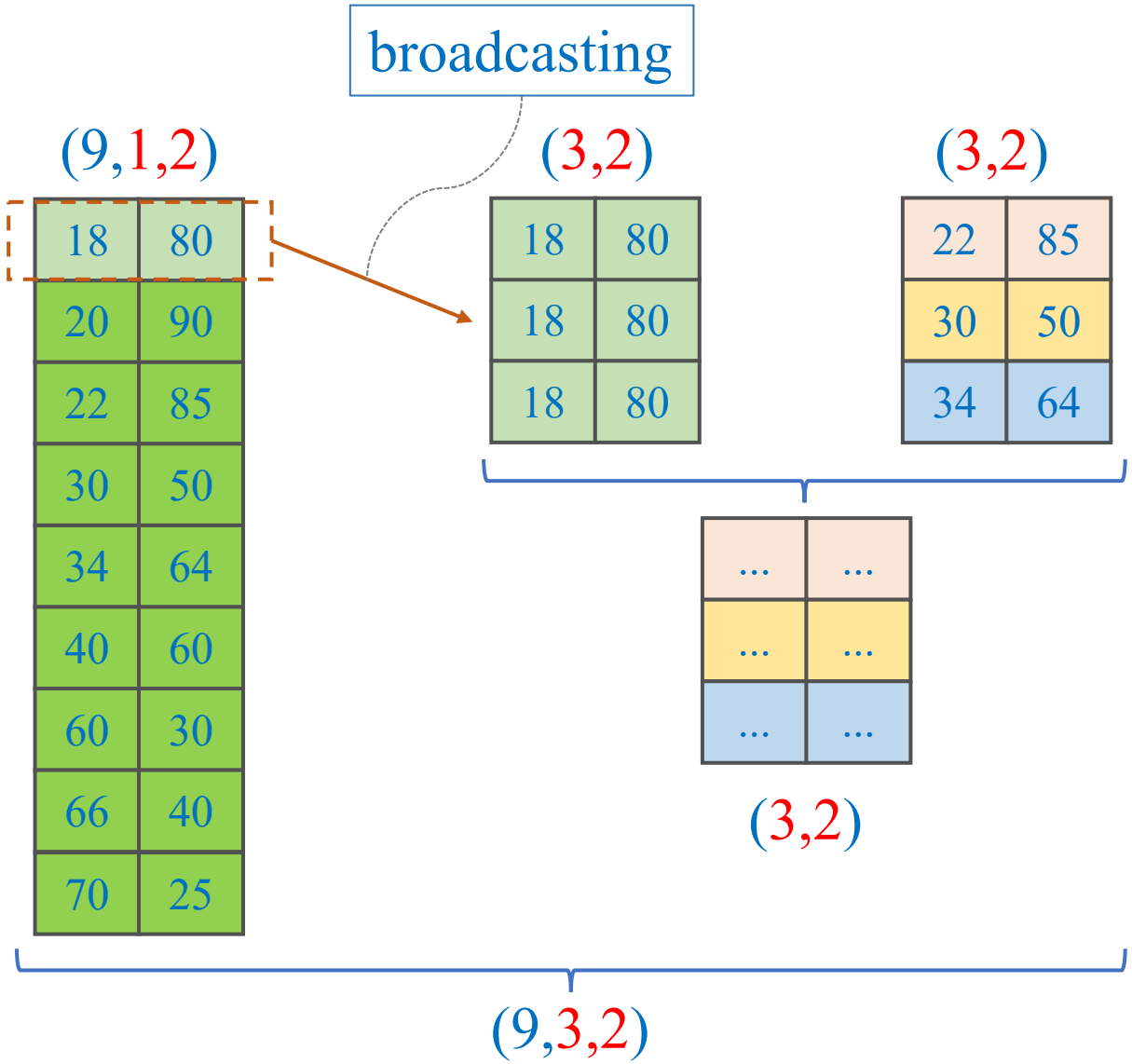
18	80
20	90
22	85
30	50
34	64
40	60
60	30
66	40
70	25

$C =$

22	85
30	50
34	64

Index	Age	Expenditure	Distance to C0	Distance to C1	Distance to C2
...	...	...	...	...	...

X[:, np.newaxis, :] - centroids



K-Means

shape=(9,3)

❖ Implementation using Numpy

$X =$

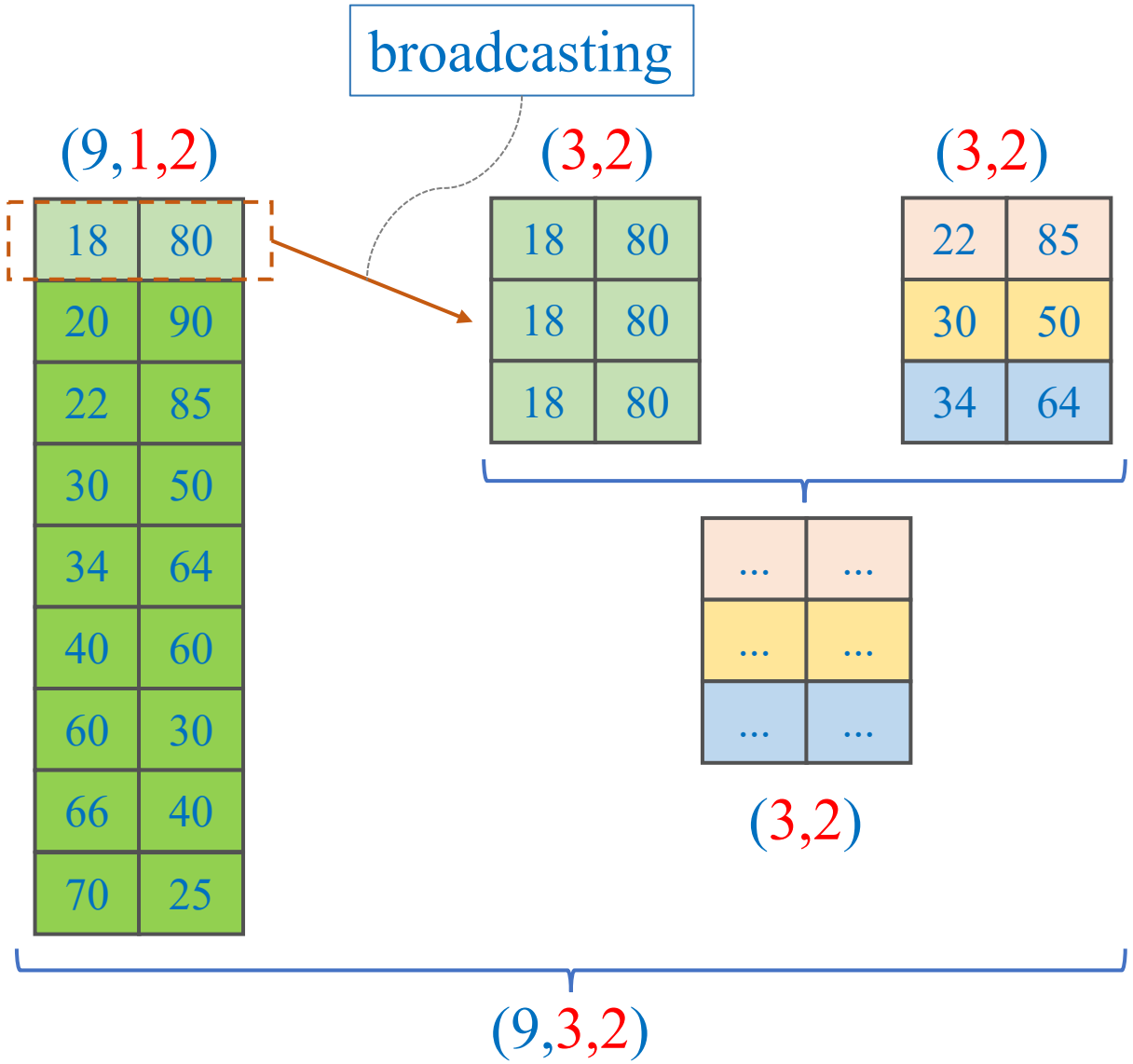
18	80
20	90
22	85
30	50
34	64
40	60
60	30
66	40
70	25

$C =$

22	85
30	50
34	64

Index	Age	Expenditure	Distance to C0	Distance to C1	Distance to C2
...	...	...	...	...	...

norm(X[:, np.newaxis, :] - centroids, axis=2)



K-Means

❖ Implementation using Numpy

$X =$

18	80
20	90
22	85
30	50
34	64
40	60
60	30
66	40
70	25

$C =$

22	85
30	50
34	64

```
L = np.argmin(D, axis=1)
```

Distance to C0	Distance to C1	Distance to C2
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...
...	...	...

(9,3)

Label
...
...
...
...
...
...
...
...
...

(9,)


Demo

Outline

SECTION 1

Introduction

SECTION 2

Tabular Data

SECTION 3

Image&Text Data

K-Means

❖ Image data

Training data

Cats

Dogs

From Cat-Dog dataset

K-Means

❖ Image data

Training data

From Cat-Dog dataset

K-Means

❖ Image data

(224,224,3)

(224*224*3,)

K-Means

cluster 0

cluster 1

Discussion

K-Means

❖ Image data

(224,224,3)

Magic Function

(512,)

K-Means

cluster 0

cluster 1


Demo

Text Data

K-Means

❖ Bag of words

Corpus

doc1 = "deep learning book"

doc2 = "machine learning algorithm"

doc3 = "learning ai from scratch"

doc4 = "ai vietnam"

Tokenization ⇨

['deep', 'learning', 'book']

['machine', 'learning', 'algorithm']

['learning', 'ai', 'from', 'scratch']

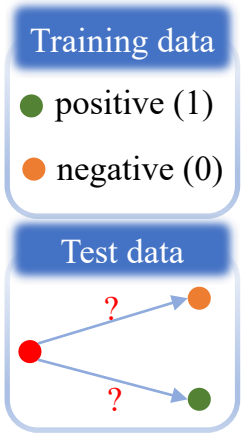
['ai', 'vietnam']

Vocabulary =	deep	learning	book	machine	algorithm	ai	from	scratch	vietnam

👉 **Given a string** = "vietnam machine learning deep learning book"

	deep	learning	book	machine	algorithm	ai	from	scratch	vietnam
BoW	1	2	1	1	0	0	0	0	1
Binary BoW	1	1	1	1	0	0	0	0	1

Doc	Label
góp gió gặt bão	1
có làm mới có ăn	1
đất lành chim đậu	1
ăn cháo đá bát	0
gậy ông đập lưng ông	0
qua cầu rút ván	0



Tokenization

Vocabulary
bát
bão
chim
cháo
có
cầu
gió
góp
gậy
gặt
làm
lành
lưng
mới
qua
rút
ván
ông
ăn
đá
đất
đập
đậu

gậy ông đập lưng ông

	doc_0	doc_1	doc_2	doc_3	doc_4	doc_5
bát	0	0	0	1	0	0
bão	1	0	0	0	0	0
chim	0	0	1	0	0	0
cháo	0	0	0	1	0	0
có	0	2	0	0	0	0
cầu	0	0	0	0	0	1
gió	1	0	0	0	0	0
góp	1	0	0	0	0	0
gậy	0	0	0	0	1	0
gặt	1	0	0	0	0	0
làm	0	1	0	0	0	0
lành	0	0	1	0	0	0
lưng	0	0	0	0	1	0
mới	0	1	0	0	0	0
qua	0	0	0	0	0	1
rút	0	0	0	0	0	1
ván	0	0	0	0	0	1
ông	0	0	0	0	2	0
ăn	0	1	0	1	0	0
đá	0	0	0	1	0	0
đất	0	0	1	0	0	0
đập	0	0	0	0	1	0
đậu	0	0	1	0	0	0

BoW vectors


Demo

Summaary

Introduction

Data processing and select K

⬇

Select K centroids

⬇

assign each data point to a cluster

⬇

update centroids

Tabular Data

	Index	Length	Distance to C0	Distance to C1	Centroid
C_0	0	1.4	0.0	0.4	0
C_1	1	1	0.4	0.0	1
	2	1.5	0.1	0.5	-
	3	3.1	1.7	2.1	-
	4	3.7	2.4	2.8	-
	5	4.1	2.7	3.1	-

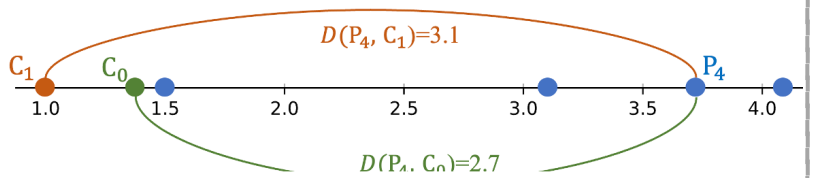


Image Data