K-Means Algorithm

1 Initialize
Randomly select k centroids

↓

2 Assignment Step
Assign each point to nearest centroid

↓

3 Update Step
Recompute centroids as cluster means

↓

4 Check Convergence
Stop when centroids don't move

Time Complexity

`O(n · k · i · d)`

linear in samples and features

Strengths
- ✓ Simple to understand
- ✓ Fast computation
- ✓ Scalable to large datasets

Weaknesses
- ✗ Assumes spherical clusters
- ✗ Sensitive to initialization
- ✗ Requires pre-specified k

Finding optimal k: Use elbow method or silhouette analysis

📊 Detailed Computational Example (k=3)

0 Initial Dataset

🔢 Given 6 points in 2D space:

$P_1 = (2, 3)$ $P_2 = (3, 3)$ $P_3 = (8, 7)$ $P_4 = (9, 6)$ $P_5 = (5, 2)$ $P_6 = (6, 3)$

🎯 **Goal:** Partition these 6 points into k=3 clusters

1 Initialize Centroids

🎲 Randomly selected initial centroids:

$C_1 = (2, 3)$ $C_2 = (8, 7)$ $C_3 = (5, 2)$

💡 In practice, we randomly select k points from the dataset as initial centroids

2 Iteration 1 - Assignment Step

📏 Calculate Euclidean distances from each point to all centroids:

For $P_1 = (2, 3)$:

$d(P_1, C_1) = \sqrt{[(2-2)^2 + (3-3)^2]} = \sqrt{[0 + 0]} = 0.00$

$d(P_1, C_2) = \sqrt{[(2-8)^2 + (3-7)^2]} = \sqrt{[36 + 16]} = 7.21$

$d(P_1, C_3) = \sqrt{[(2-5)^2 + (3-2)^2]} = \sqrt{[9 + 1]} = 3.16$

→ Assign P_1 to Cluster 1 (minimum distance: 0.00)

For $P_2 = (3, 3)$:

$d(P_2, C_1) = \sqrt{[(3-2)^2 + (3-3)^2]} = \sqrt{[1 + 0]} = 1.00$

$d(P_2, C_2) = \sqrt{[(3-8)^2 + (3-7)^2]} = \sqrt{[25 + 16]} = 6.40$

$d(P_2, C_3) = \sqrt{[(3-5)^2 + (3-2)^2]} = \sqrt{[4 + 1]} = 2.24$

→ Assign P_2 to Cluster 1 (minimum distance: 1.00)

For $P_3 = (8, 7)$:

$d(P_3, C_1) = \sqrt{[(8-2)^2 + (7-3)^2]} = \sqrt{[36 + 16]} = 7.21$

$d(P_3, C_2) = \sqrt{[(8-8)^2 + (7-7)^2]} = \sqrt{[0 + 0]} = 0.00$

$d(P_3, C_3) = \sqrt{[(8-5)^2 + (7-2)^2]} = \sqrt{[9 + 25]} = 5.83$

→ Assign P_3 to Cluster 2 (minimum distance: 0.00)

For $P_4 = (9, 6)$:

$d(P_4, C_1) = \sqrt{[(9-2)^2 + (6-3)^2]} = \sqrt{[49 + 9]} = 7.62$

$d(P_4, C_2) = \sqrt{[(9-8)^2 + (6-7)^2]} = \sqrt{[1 + 1]} = 1.41$

$d(P_4, C_3) = \sqrt{[(9-5)^2 + (6-2)^2]} = \sqrt{[16 + 16]} = 5.66$

→ Assign P_4 to Cluster 2 (minimum distance: 1.41)

For $P_5 = (5, 2)$:

$d(P_5, C_1) = \sqrt{[(5-2)^2 + (2-3)^2]} = \sqrt{[9 + 1]} = 3.16$

$d(P_5, C_2) = \sqrt{[(5-8)^2 + (2-7)^2]} = \sqrt{[9 + 25]} = 5.83$

$d(P_5, C_3) = \sqrt{[(5-5)^2 + (2-2)^2]} = \sqrt{[0 + 0]} = 0.00$

→ Assign P_5 to Cluster 3 (minimum distance: 0.00)

For $P_6 = (6, 3)$:

$d(P_6, C_1) = \sqrt{[(6-2)^2 + (3-3)^2]} = \sqrt{[16 + 0]} = 4.00$

$d(P_6, C_2) = \sqrt{[(6-8)^2 + (3-7)^2]} = \sqrt{[4 + 16]} = 4.47$

$d(P_6, C_3) = \sqrt{[(6-5)^2 + (3-2)^2]} = \sqrt{[1 + 1]} = 1.41$

→ Assign P_6 to Cluster 3 (minimum distance: 1.41)

Cluster 1

$P_1 = (2, 3)$

$P_2 = (3, 3)$

Cluster 2

$P_3 = (8, 7)$

$P_4 = (9, 6)$

Cluster 3

$P_5 = (5, 2)$

$P_6 = (6, 3)$

3 Iteration 1 - Update Centroids

🔄 Calculate new centroids as the mean of assigned points:

New C_1 :

Points in Cluster 1: P_1 (2,3), P_2 (3,3)

C_1_new = ((2+3)/2, (3+3)/2) = (2.5, 3.0)

C_1 : (2, 3) → (2.5, 3.0) ✓ Changed

New C_2 :

Points in Cluster 2: P_3 (8,7), P_4 (9,6)

C_2_new = ((8+9)/2, (7+6)/2) = (8.5, 6.5)

C_2 : (8, 7) → (8.5, 6.5) ✓ Changed

New C_3 :

Points in Cluster 3: P_5(5,2), P_6(6,3)

```
C₃ _new = ((5+6)/2, (2+3)/2) = (5.5, 2.5)
```

C_3 : (5, 2) → (5.5, 2.5) ✓ Changed

📍 Updated centroids for Iteration 2:

$C_1 = (2.5, 3.0)$ | $C_2 = (8.5, 6.5)$ | $C_3 = (5.5, 2.5)$

🔁 Centroids have moved! Continue to Iteration 2...

2 Iteration 2 - Assignment Step

📏 Recalculate distances with new centroids:

For $P_1 = (2, 3)$:

$d(P_1, C_1) = \sqrt{[(2-2.5)^2 + (3-3)^2]} = 0.50$ ✓ min

$d(P_1, C_2) = \sqrt{[(2-8.5)^2 + (3-6.5)^2]} = 7.27$

$d(P_1, C_3) = \sqrt{[(2-5.5)^2 + (3-2.5)^2]} = 3.54$

→ Remains in Cluster 1

... (similar calculations for P_2, P_3, P_4, P_5, P_6) ...

Cluster 1

$P_1 = (2, 3)$

$P_2 = (3, 3)$

Cluster 2

$P_3 = (8, 7)$

$P_4 = (9, 6)$

Cluster 3

$P_5 = (5, 2)$

$P_6 = (6, 3)$

📌 **Result:** No points changed clusters! (same as Iteration 1)

3 Iteration 2 - Update Centroids

🔄 Recalculate centroids:

```
C₁ _new = ((2+3)/2, (3+3)/2) = (2.5, 3.0)
C₂ _new = ((8+9)/2, (7+6)/2) = (8.5, 6.5)
C₃ _new = ((5+6)/2, (2+3)/2) = (5.5, 2.5)
```

📍 Centroids after Iteration 2:

$C_1 = (2.5, 3.0)$ - No change

$C_2 = (8.5, 6.5)$ - No change

$C_3 = (5.5, 2.5)$ - No change

4 Check Convergence

✅ Algorithm Converged!

Condition met: Centroids did not move between iterations

- C_1: (2.5, 3.0) → (2.5, 3.0) - Movement: 0.00
- C_2: (8.5, 6.5) → (8.5, 6.5) - Movement: 0.00
- C_3: (5.5, 2.5) → (5.5, 2.5) - Movement: 0.00

🎉 Final Clustering Result:

Cluster 1	Cluster 2	Cluster 3
P_1 = (2, 3)	P_3 = (8, 7)	P_5 = (5, 2)
P_2 = (3, 3)	P_4 = (9, 6)	P_6 = (6, 3)
Centroid: (2.5, 3.0)	Centroid: (8.5, 6.5)	Centroid: (5.5, 2.5)

📊 **Summary:** The algorithm converged in 2 iterations. Total distance calculations: 6 points × 3 centroids × 2 iterations = 36 distance computations