DM,
Autumn, 2026

Clustering — k-Means & Customer Segmentation

Faculty of DS & AI
Autumn semester, 2026

Trong-Nghia Nguyen

Business AI Lab

Content

- **Clustering**
- Customer Segmentation
- Example

Clustering



DESCRIBE

Goal: Summarize and visualize data.

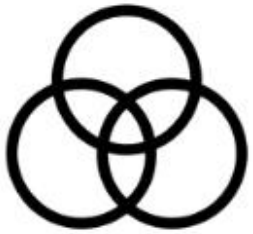
Key Question: What is the data saying?



RELATE

Goal: Identify correlations and associations.

Key Question: What usually goes together?



GROUP

Goal: Cluster objects with shared, natural traits.

Key Question: How many groups can these objects be divided into?



PREDICT

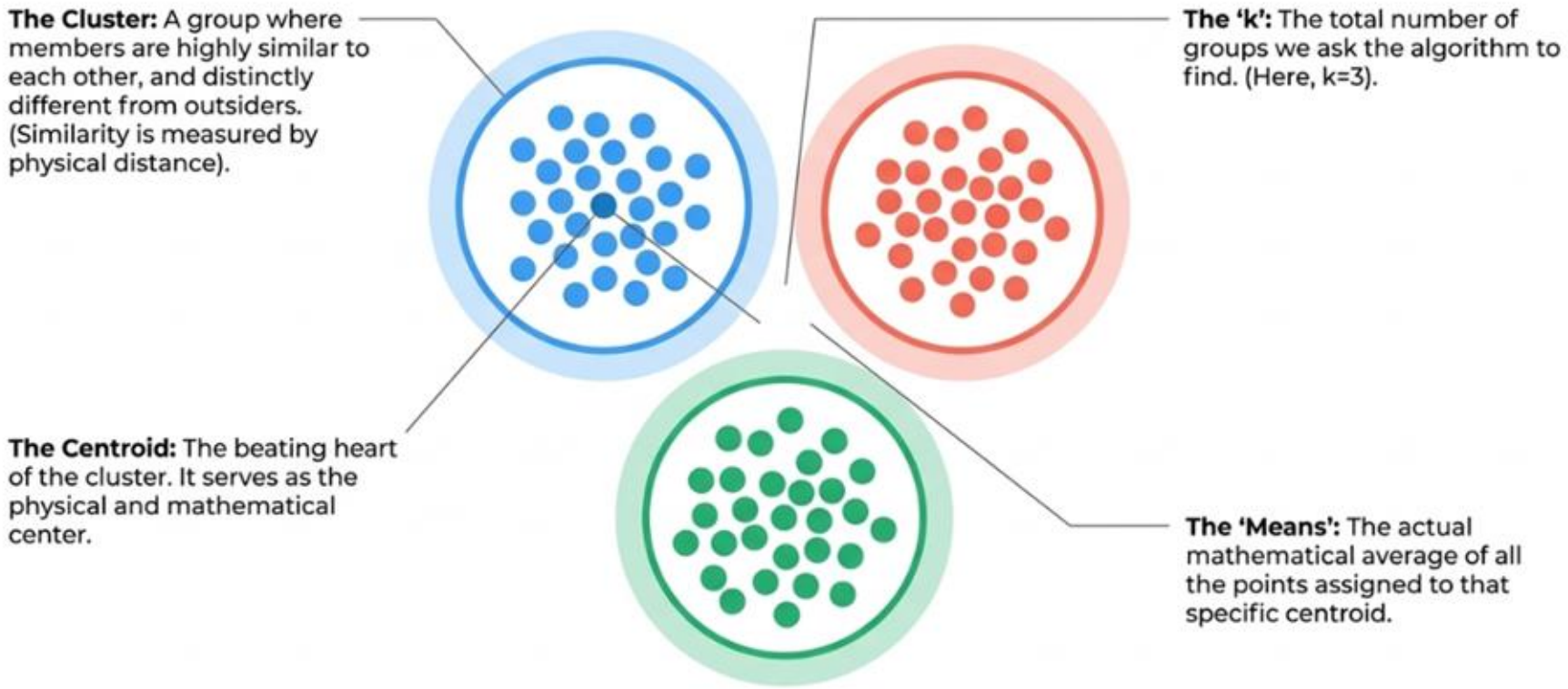
Goal: Utilize classification and regression.

Key Question: What will happen next?

Clustering

Definition

- **Clustering**: dividing a set of objects into clusters such that objects within the same cluster are similar, while distinct clusters are clearly different.
- "Similarity" is measured by distance—the closer two points are, the more similar they are.
- **K-Means**: The idea is:
  - Pre-select **k** clusters, each with a centroid;
  - Assign each point to the cluster with the nearest centroid;
  - Shift the centroid to the mean of the points within that cluster;
  - Repeat the process until stability is reached.



"Means" = averages: because the center of each cluster is the average of its members.

Clustering

Definition

- **Euclidean distance:** For a point $x = (x_1, x_2, ..., x_m)$ and a cluster center $c = (c_1, c_2, ..., c_m)$ (m = number of features/columns):

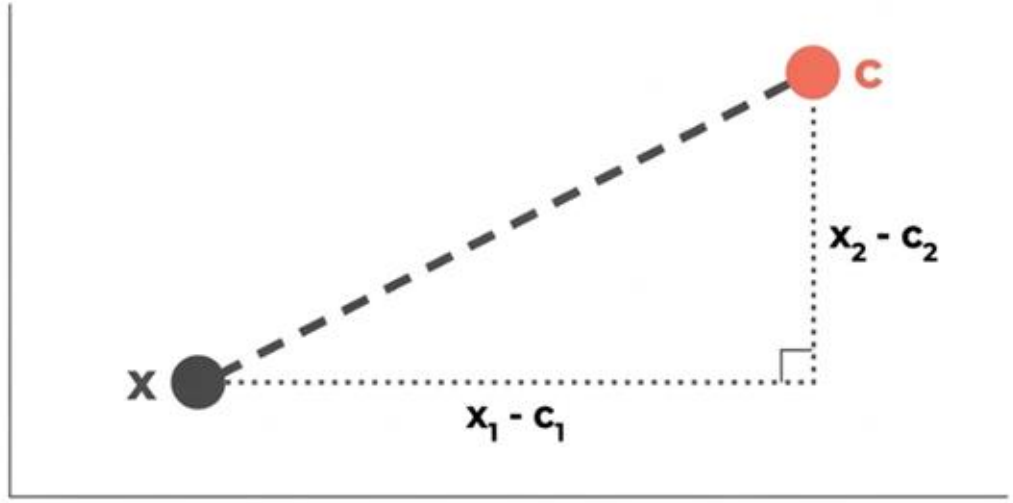
$$d(x, c) = \sqrt{(x_1 - c_1)^2 + (x_2 - c_2)^2 + \cdots + (x_m - c_m)^2} = \sqrt{\sum_{i=1}^{m} (x_i - c_i)^2}$$

- **Cluster center = average (centroid update):** The center of cluster S_j is the average of the coordinates of all points in the cluster:

$$c_j = \frac{1}{|S_j|} \sum_{x \in S_j} x$$

(e.g., the **centroid's income** coordinate is the **average income** of **customers** in the cluster; similarly for Spend).

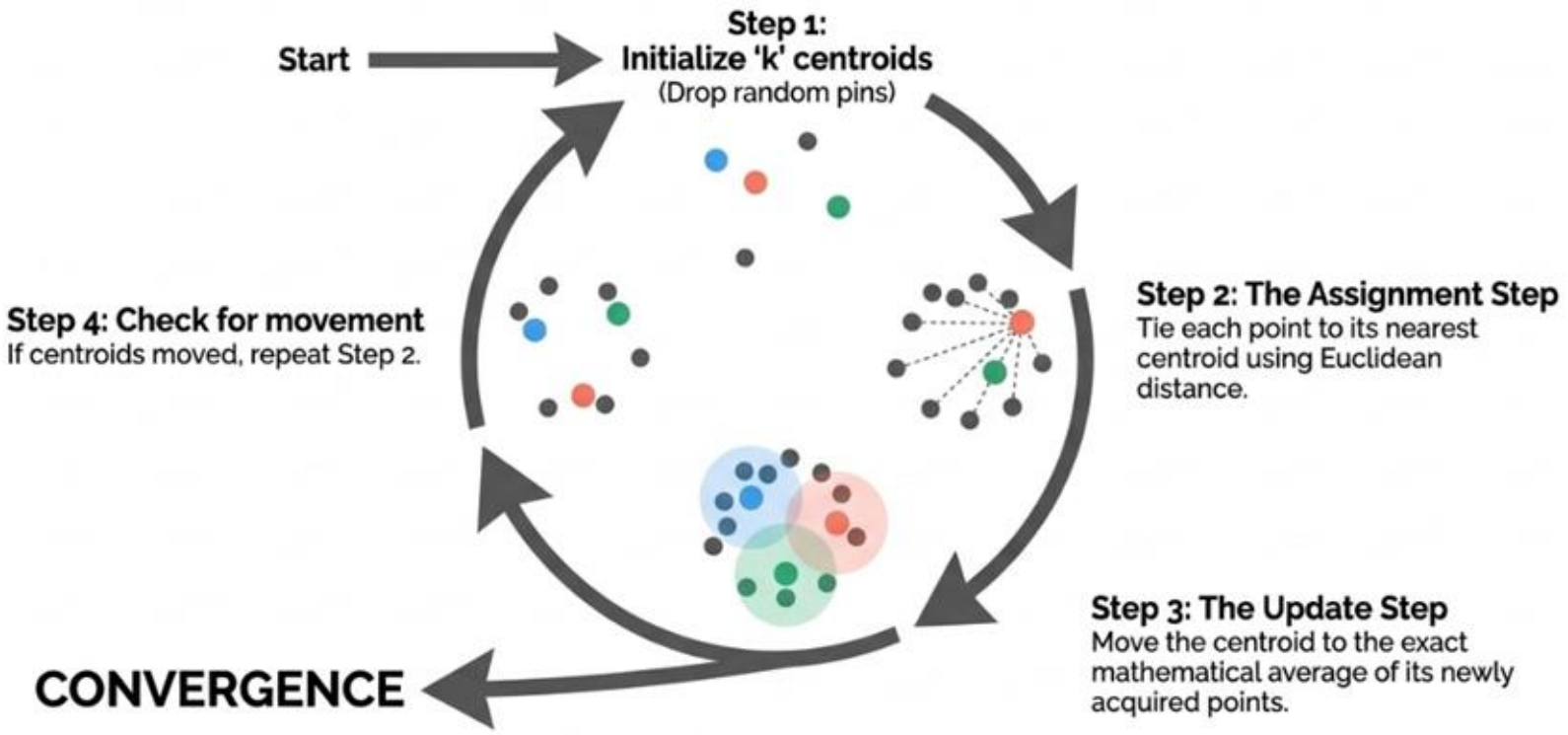
$$d(x, c) = \sqrt{(x_1-c_1)^2 + (x_2-c_2)^2 + \ldots + (x_m-c_m)^2} = \sqrt{\sum (x_i - c_i)^2}$$



Clustering

K-Means algorithm (Lloyd's)

- **Step 1:** Choose k and initialize k initial centroids (randomly or by selecting k points).
- **Step 2 - Assignment step:** assign each point to the cluster with the nearest center (using Euclidean distance).
- **Step 3 - Update step:** recalculate each centroid as the average of the points assigned to that cluster.
- **Step 4 - Repeat steps 2–3** until the centers (or assignments) remain unchanged → convergence.



Clustering

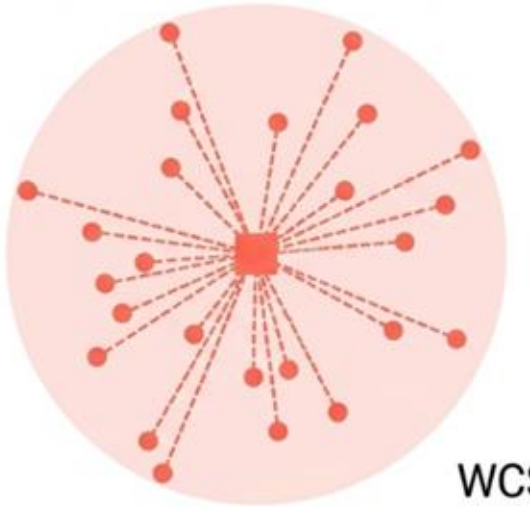
K-Means algorithm (Lloyd's)

- **Evaluation:** To measure the "quality" of the clusters, WCSS (inertia) is used.
- WCSS (Within-Cluster Sum of Squares) = the sum of the squares of the distances from each point to its cluster center

$$\text{WCSS} = \sum_{j=1}^{k} \sum_{x \in S_j} d(x, c_j)^2$$

*The **smaller** the WCSS, the more **closely** the points cluster around the center, and the "tighter" the cluster becomes.*
k-Means is essentially the process of progressively reducing WCSS through iterations.

High WCSS: Loose, unrefined groupings



Low WCSS: Tight, cohesive groupings



$$\text{WCSS} = \sum \sum d(x, c_j)^2$$

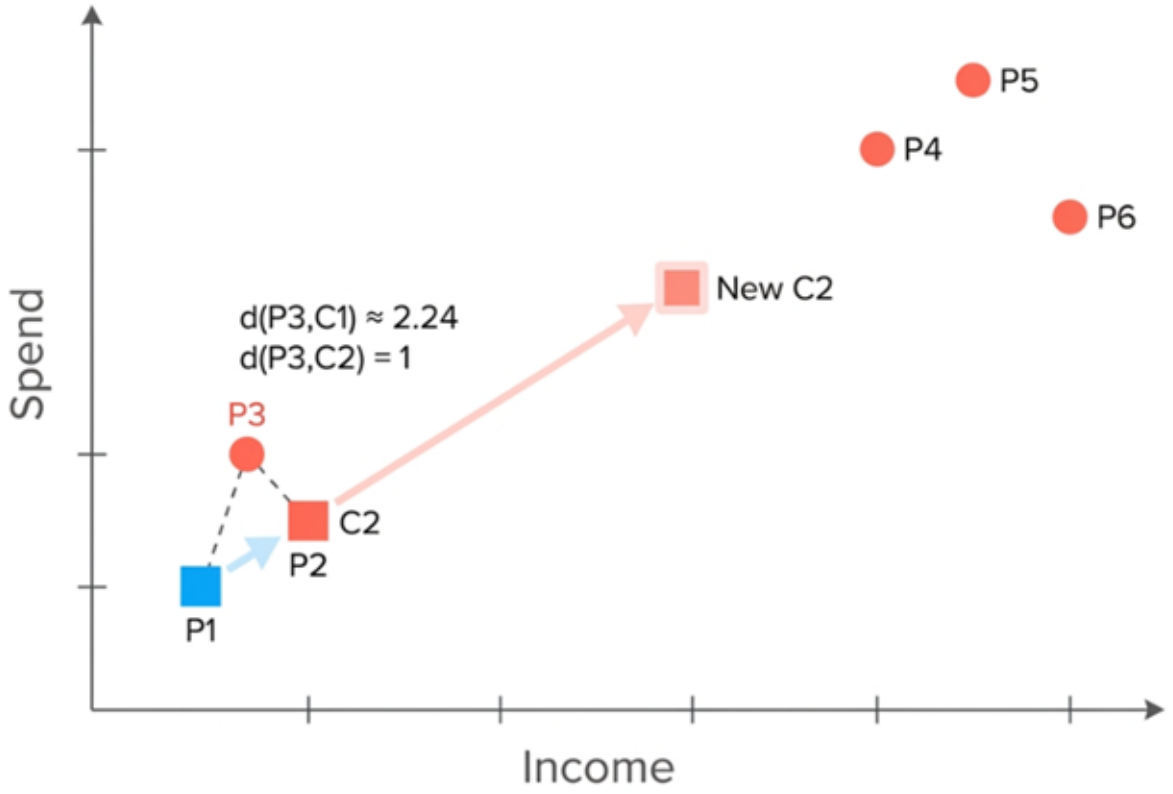
k-Means is simply an optimization machine designed to shrink WCSS.
Our final example achieved a WCSS of 6.67.

Clustering

K-Means algorithm (Lloyd's)

- **Example:** Data: 6 customers, 2 features (income, spend):

Điểm	Income	Spend
P1	2	2
P2	3	3
P3	3	4
P4	10	10
P5	11	11
P6	12	9



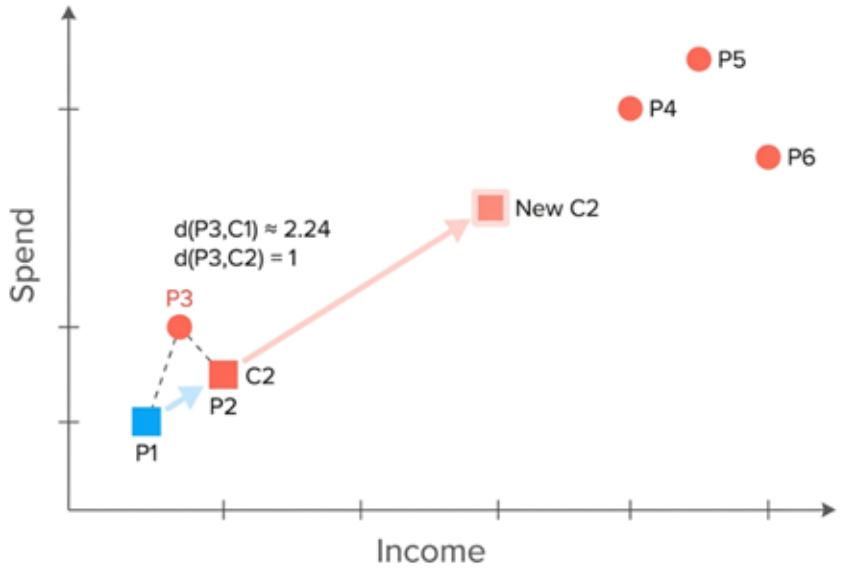
=> Choose k = 2 and initialize (deliberately "poorly") C1 = (2, 2) and C2 = (3, 3).

Clustering

K-Means algorithm (Lloyd's)

- **Example:** Data: 6 customers, 2 features (income, spend):

Điểm	Income	Spend
P1	2	2
P2	3	3
P3	3	4
P4	10	10
P5	11	11
P6	12	9



Iteration 1

- Calculate the distance from each point to the two centroids and assign it to the nearer centroid. For example, with P3 = (3, 4):

$$d(P3, C1) = \sqrt{(3-2)^2 + (4-2)^2} = \sqrt{1+4} = \sqrt{5} \approx 2.24; \quad d(P3, C2) = \sqrt{(3-3)^2 + (4-3)^2} = 1$$

→ P3 is closer to C2 → assign to cluster 2.

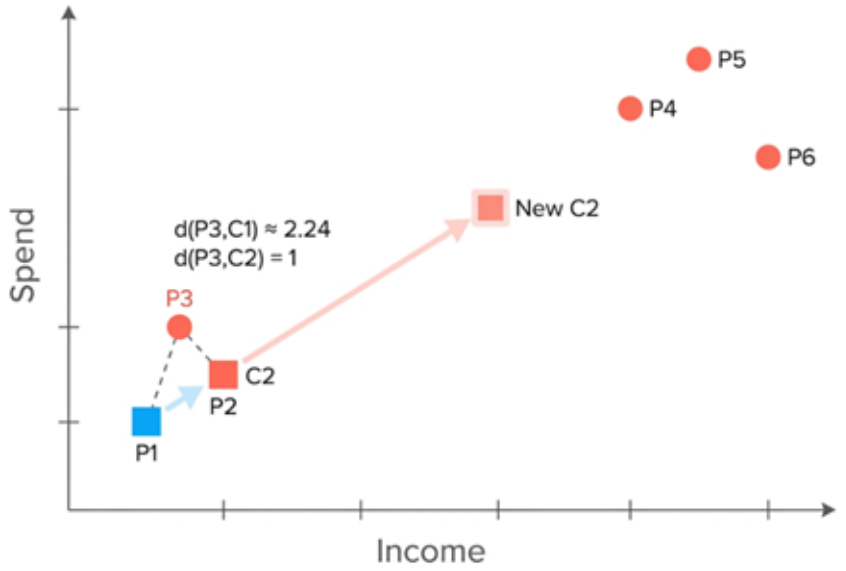
- Repeat the process: P1 → cluster 1;
- P2, P3, P4, P5, P6 → cluster 2.
- **Update centroids**: C1 = (2, 2); C2 = average of {P2, P3, P4, P5, P6} = (7.8, 7.4).

Clustering

K-Means algorithm (Lloyd's)

- **Example:** Data: 6 customers, 2 features (income, spend):

Điểm	Income	Spend
P1	2	2
P2	3	3
P3	3	4
P4	10	10
P5	11	11
P6	12	9



Iteration 2

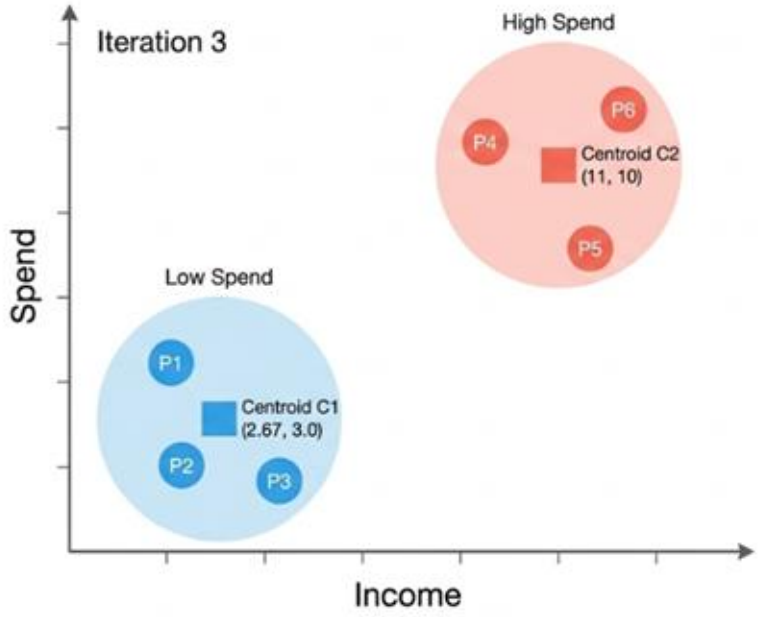
- **Reassign** to new centroids → P1, P2, P3 → Cluster 1;
- P4, P5, P6 → Cluster 2.
- **Update centroids**:
  - C1 = average of {P1, P2, P3} = (2.67, 3.0);
  - C2 = average of {P4, P5, P6} = (11, 10).

Clustering

K-Means algorithm (Lloyd's)

- **Example:** Data: 6 customers, 2 features (income, spend):

Điểm	Income	Spend
P1	2	2
P2	3	3
P3	3	4
P4	10	10
P5	11	11
P6	12	9



The centroids no longer move. The algorithm has found the natural structure of the data.

Iteration 3

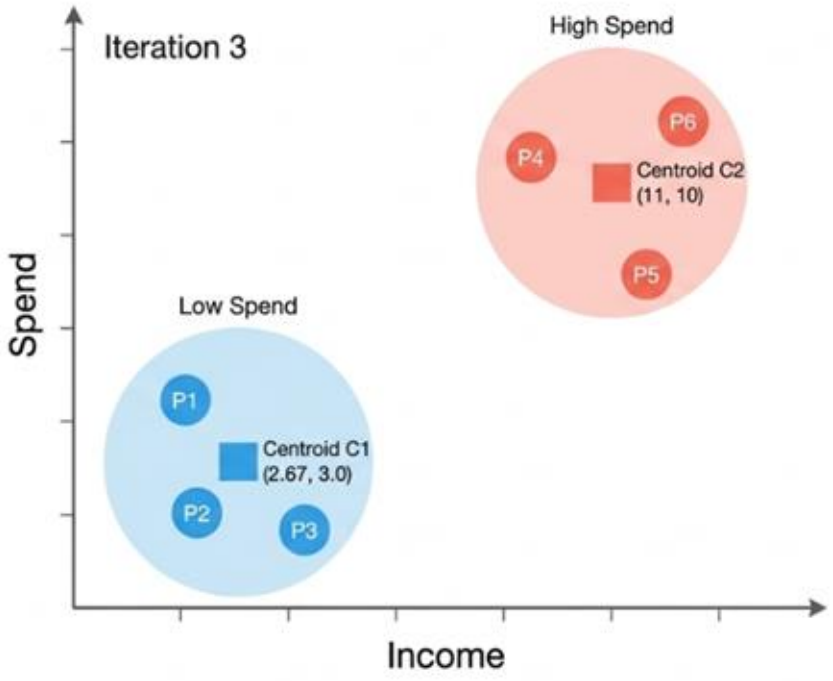
- **Reassignment → unchanged** (P1, P2, P3 ↔ Cluster 1; P4, P5, P6 ↔ Cluster 2).
- **Centroids unchanged → CONVERGENCE.**

Clustering

K-Means algorithm (Lloyd's)

- **Example:** Data: 6 customers, 2 features (income, spend):

Điểm	Income	Spend
P1	2	2
P2	3	3
P3	3	4
P4	10	10
P5	11	11
P6	12	9



The centroids no longer move. The algorithm has found the natural structure of the data.

- **Result:** two distinct clusters:
  - The "low-spending" group {P1, P2, P3}
  - The "high-spending" group {P4, P5, P6}.
  - WCSS = 6.67.

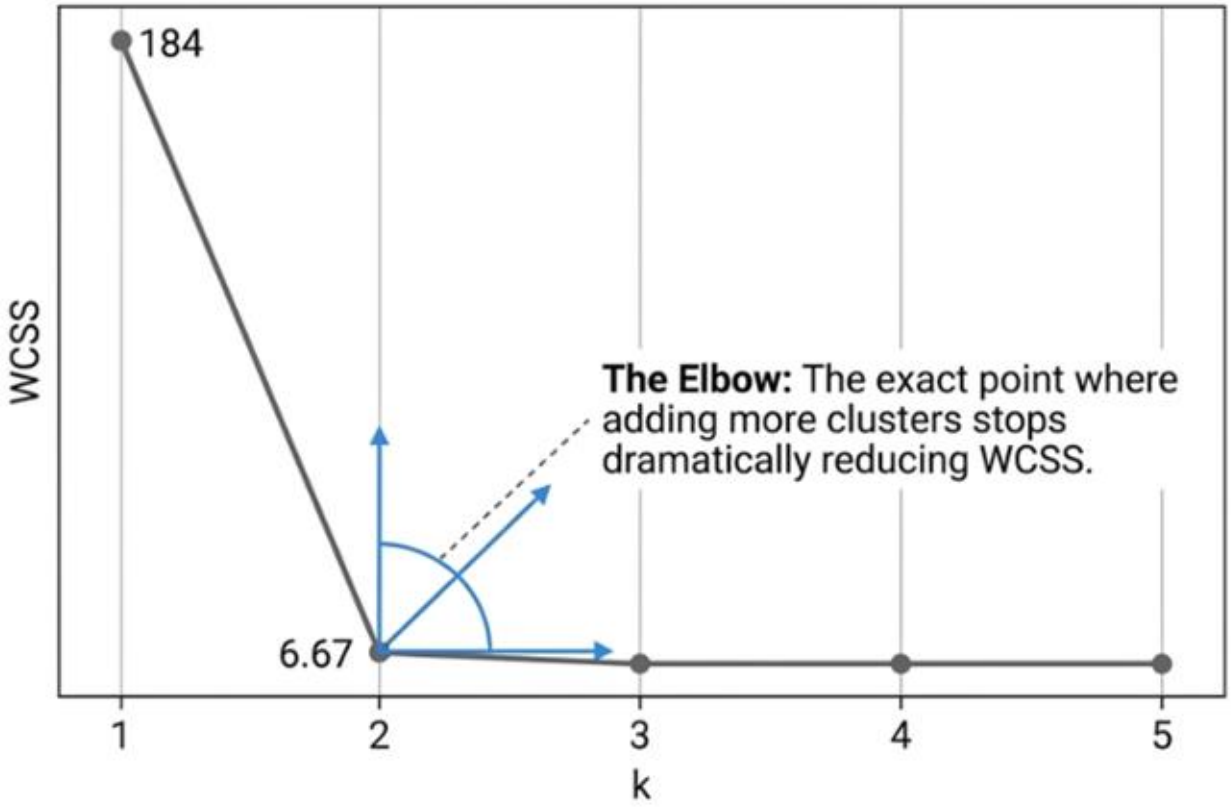
Note on initialization: since initial centers are chosen randomly, k-Means can yield different results across runs. In practice, the tool runs multiple times and retains the result with the lowest WCSS (controlled by the `n_init` or "replications" parameter).

Clustering

K selection

K-Means does not automatically know how many clusters to form—we have to choose k.

- **Method 1:** The Elbow Method
  - Run k-Means with various values of k (1, 2, 3, ...) and plot WCSS against k.
  - WCSS always decreases as k increases, but at a certain point, the rate of decrease slows down significantly—forming an "elbow." Select the value of k at that elbow.
  - In the example figure: WCSS drops from 184 to 6.67 as k goes from 1 to 2, then levels off. There is an "elbow" at k = 2, so we choose 2 clusters (consistent with intuition).



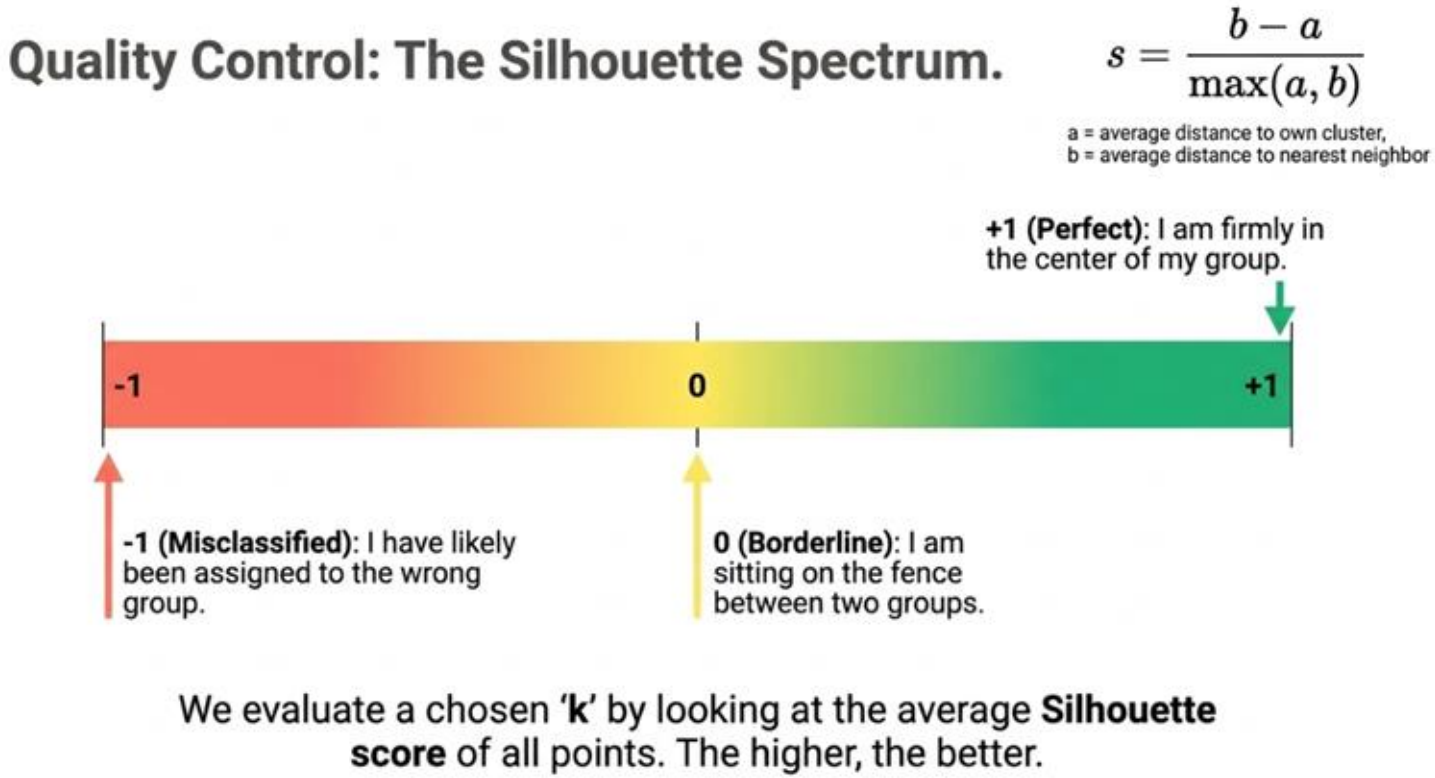
Clustering

K selection

K-Means does not automatically know how many clusters to form—we have to choose k.

- **Method 2:** Silhouette coefficient
  - Measure how well a point fits into its cluster. For a single point:
    - *a* = average distance to points within the same cluster (the smaller, the better).
    - *b* = average distance to the points of the nearest neighboring cluster.

$$s = \frac{b - a}{\max(a, b)}, \qquad s \in [-1, 1]$$



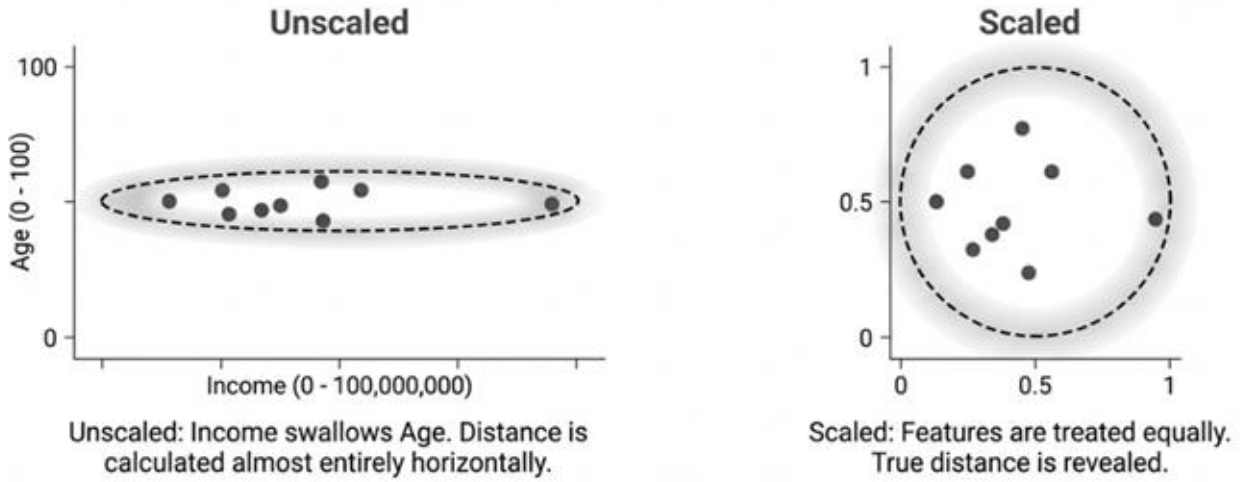
Clustering

Normalization

- k-Means is based on distance, so the scale of the variables has a direct impact.
- For example, **Income** (0–100 million) and **Age** (0–100 years) operate on different scales: if income is measured in millions while age is in the tens, the distance between points **depends almost entirely on income**—effectively "swamping" the age variable.
- Solution: Normalize or standardize all variables to the same scale (e.g., 0–1, or mean 0 and standard deviation 1) before clustering.

The Golden Rule: You must normalize distance-based data.

The Un-normalized Distortion Lens



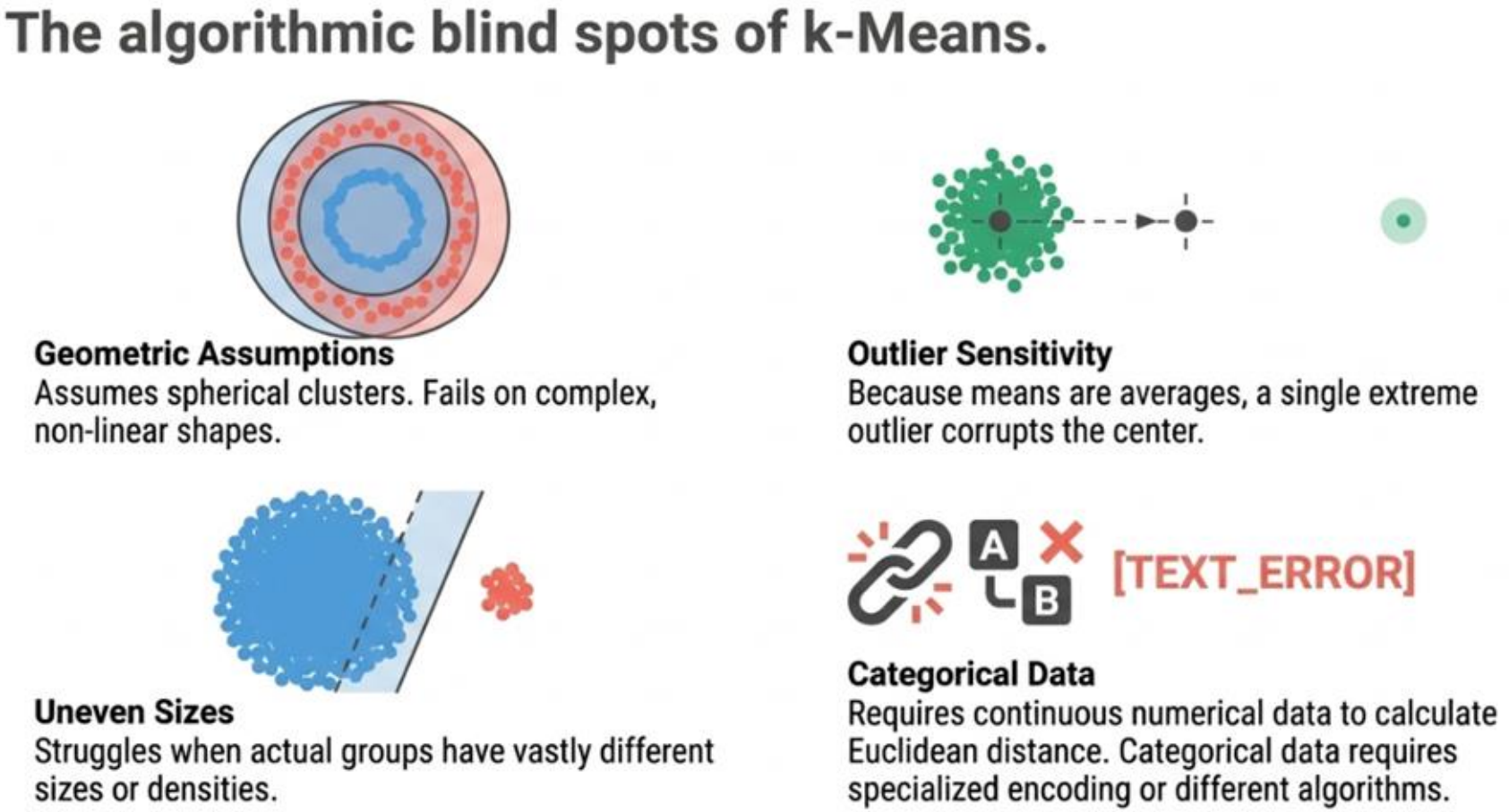
Unscaled: Income swallows Age. Distance is calculated almost entirely horizontally.

Scaled: Features are treated equally. True distance is revealed.

WARNING: Always normalize/standardize variables before running k-Means or KNN.

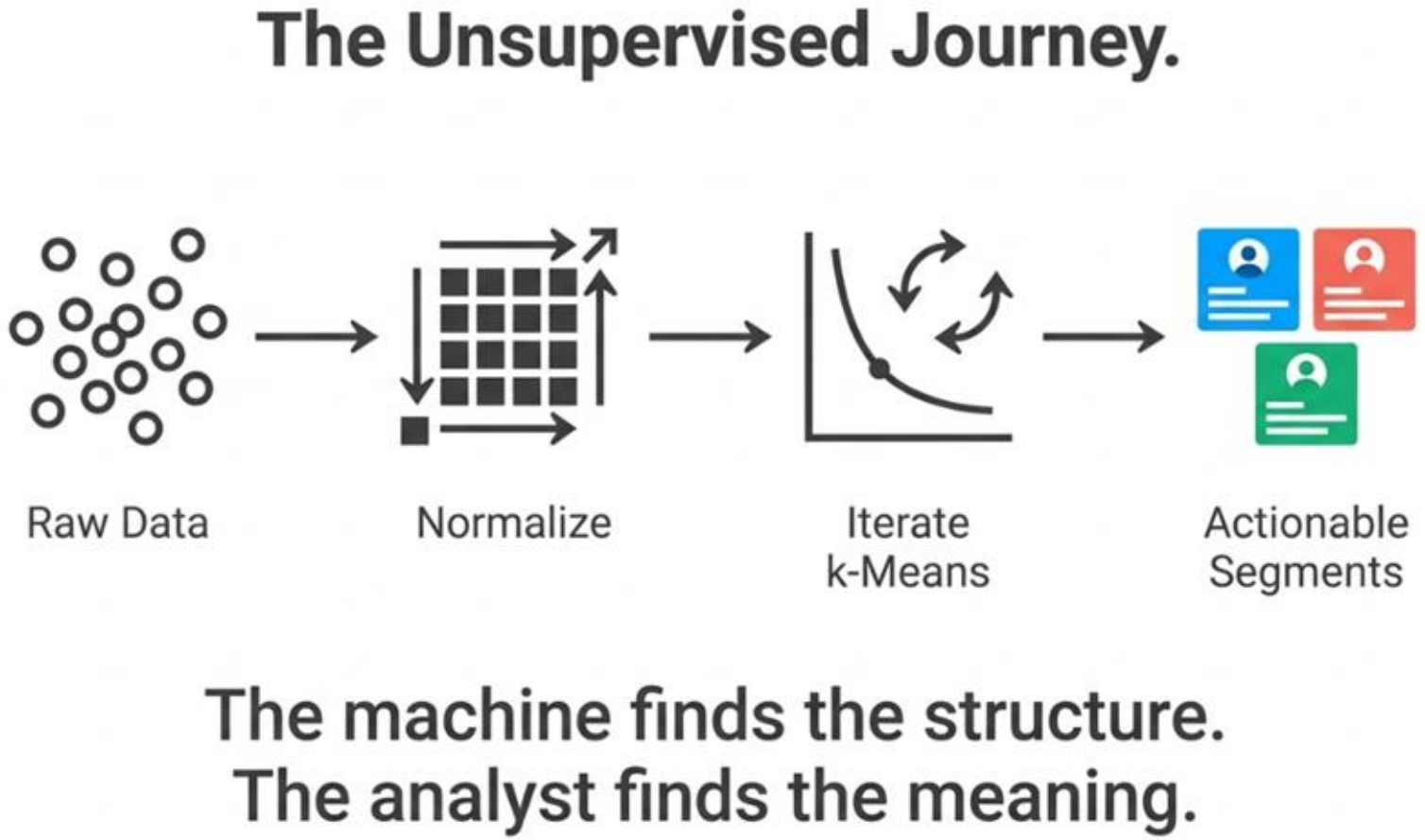
Clustering

Limitation



Clustering

Pipeline



Content

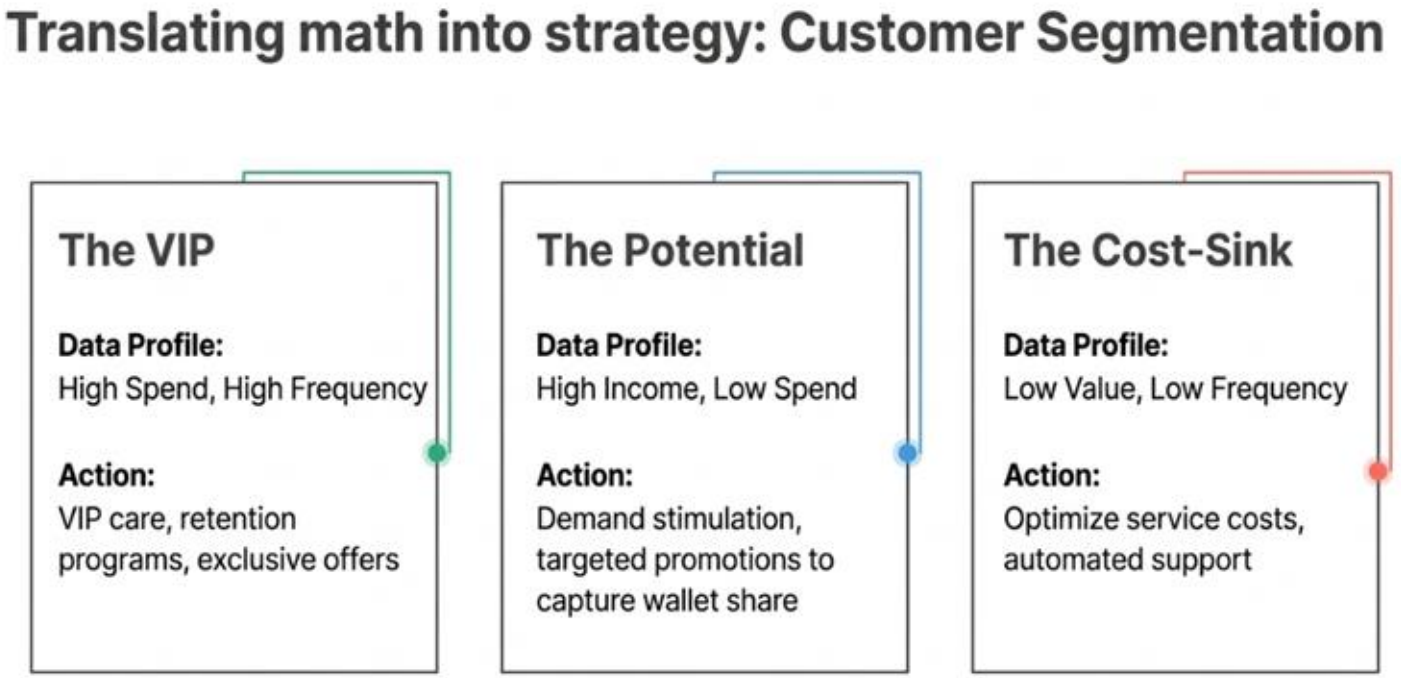
- Clustering
- **Customer Segmentation**
- Example

Customer Segmentation

Definition

Customer segmentation is a classic application of clustering: **grouping customers** with **similar behaviors** or **characteristics** so they can be **treated** differently. For instance, based on spending, visit frequency, and income, k-Means can reveal:

- High-value customers (high spending, frequent visits) → VIP care, retention.
- Potential customers (high income, low spending) → demand stimulation and promotions.
- Low-value customers → optimize service costs



Customer Segmentation

Tools

Step	Orange (No-code)	Python (Scikit-Learn)
Normalize	Preprocess → Normalize widget	StandardScaler
Cluster	k-Means widget	sklearn.cluster.KMeans
Visualize	Scatter Plot (color by cluster)	plt.scatter(..., c=labels)
Evaluate	Silhouette Plot widget	silhouette_score

Content

- Clustering
- Customer Segmentation
- **Example**

Example

Example 1

Context (customer segmentation): 6 customers described by **(Visits/month, Spend)**. Run k-Means with **k = 2**, initial centroids $\mathbf{m_1 = A = (2, 2)}$, $\mathbf{m_2 = B = (3, 2)}$.

Customer	A	B	C	D	E	F
Visits	2	3	2	8	9	7
Spend	2	2	3	8	7	9

Tasks:

1. Compute the Euclidean distance from each customer to m_1 and m_2; assign to the nearer centroid.
2. Update the two centroids (mean of assigned points).
3. Repeat until convergence.
4. Name the two segments in business terms.

Example

Example 2

A store ran k-Means for several k on its customer data and recorded:

k	1	2	3	4	5
WCSS	359.5	164.4	9.0	6.6	5.1
Avg Silhouette	–	0.598	0.876	0.759	0.745

Tasks:

1. Using the **Elbow method**, which k would you choose? Why?
2. Using the **Silhouette** score, which k is best?
3. Do the two methods agree? How many customer segments should the store use?

Example

Example 3

Problem. *StreamBox*, a video-streaming service, wants to group **six subscribers** by how they engage, then act on each group. Last month's usage:

> Subscriber #1 watched **10 hours** over **20 logins**; #2 watched **18 hours** over **24 logins**; #3 watched **12 hours** over **22 logins**; #4 watched **82 hours** but only **6 logins**; #5 watched **90 hours** over **10 logins**; #6 watched **88 hours** over just **4 logins**.

Tasks (reason it out yourself):

1. **Draw the data table** (two features: Hours, Logins).
2. The two features are on **different scales** — explain why this is a problem for k-Means, then **normalize** each to [0, 1] (min–max).
3. Run k-Means with **k = 2**, starting from centroids at **#1 and #3**. Iterate until convergence.
4. **Name** each segment and recommend **one action** StreamBox should take for it.

Example

Example 4

Problem. *ShopVN* (e-commerce) ran k-Means on normalized customer data (spend, frequency, recency) for several values of k. The analyst reported:

> "With **2** clusters, WCSS was **310** and average silhouette **0.49**. With **3** clusters, WCSS fell to **190**, silhouette **0.60**. With **4** clusters, WCSS was **160**, silhouette **0.63**. With **5** clusters, WCSS **150**, silhouette **0.45**. (With 1 cluster, WCSS was **520**.)"
> Marketing adds: *"We can run **at most 4** distinct campaigns, and each segment must be large enough to matter."*

Tasks (reason it out yourself):

1. **Draw the table** (k, WCSS, Silhouette).
2. What does the **Elbow method** suggest? What does the **Silhouette** suggest?
3. The two methods **don't fully agree** — decide **how many segments** ShopVN should use and **justify** your choice (combine the statistics *and* the business constraint).

Thank you!