Week - 1: Unsupervised Learning

What is Clustering?

Clustering is an **unsupervised learning** technique used to find **patterns or groups** (called **clusters**) in a dataset **without labels**.

Supervised vs. Unsupervised Learning

Aspect	Supervised Learning	Unsupervised Learning
Data includes labels?	✅ Yes (features **x** and labels **y**)	❌ No (only features **x**)
Goal	Learn to predict labels	Find patterns/structure
Example	Binary classification	Clustering

What Does Clustering Do?

- Groups similar data points together.
- Identifies **structure** in the data.
- Finds **natural groupings** without knowing any labels.

Example:

Imagine a dataset of dots plotted in 2D space.

- In **supervised learning**, you know which dot is which class (e.g., red vs. blue).
- In **clustering**, you don't know that — you ask the algorithm to group the dots based on similarity.

Applications of Clustering:

- 📰 **News article grouping** (e.g., similar topics like science or sports)
- 📈 **Market segmentation** (e.g., learners with different goals)
- 🧬 **Genetic data analysis** (e.g., finding people with similar traits)
- 🌌 **Astronomy** (e.g., grouping stars or galaxies)

🔹 K-means Clustering

K-means is an unsupervised algorithm that groups data into clusters based on similarity.

🔹 K-means Algorithm Summary

1. **Initialize Centroids**:
   Randomly choose K cluster centers (centroids).

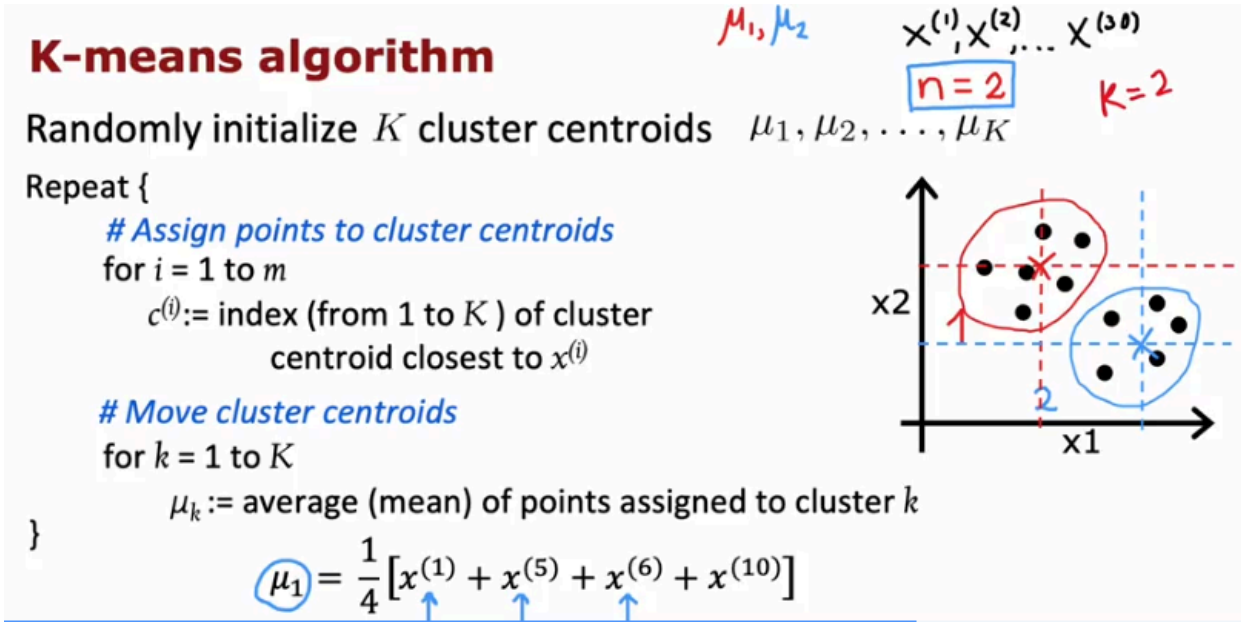
2. **Repeat Until Convergence**:

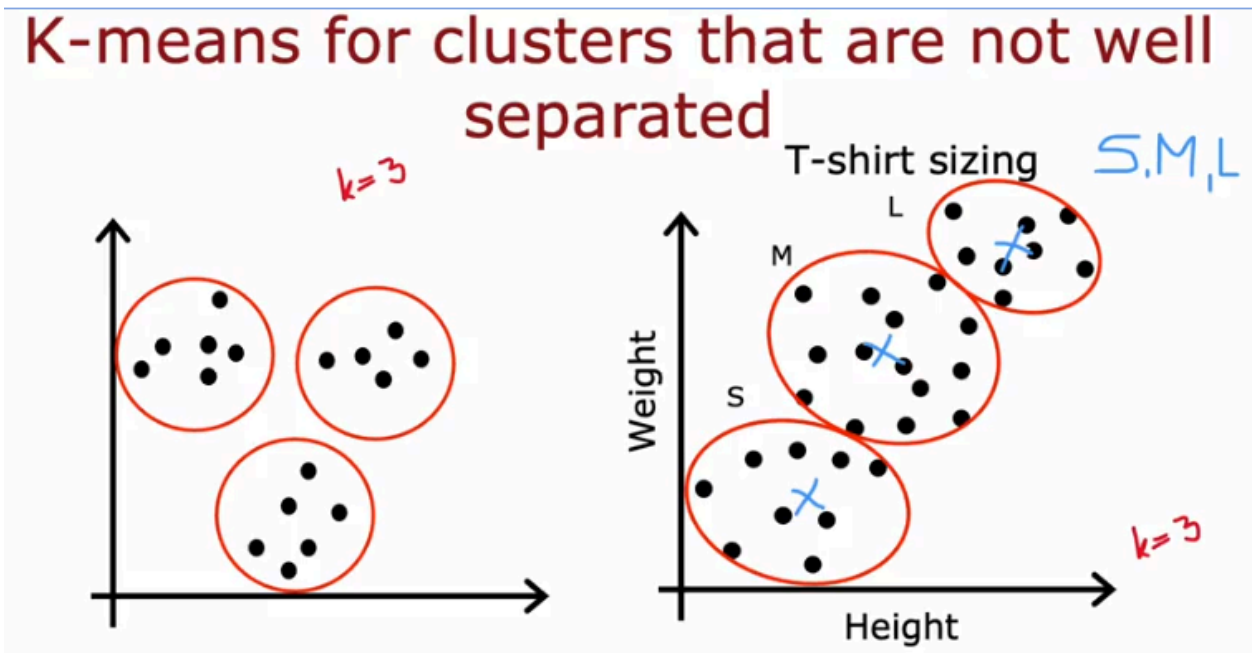
   - **Assign Points**: Each data point is assigned to the nearest centroid.
   - **Update Centroids**: Move each centroid to the average of the points assigned to it.

3. **Special Case**:
   If a centroid has no points, either remove it or reinitialize it randomly.

4. **Convergence**:
   The algorithm stops when point assignments and centroid positions no longer change.

5. **Use Case**:
   Works well even without clear clusters—like grouping people into t-shirt sizes using height and weight.





✅ **Final Result:**

- Points are grouped into k clusters.
- The algorithm stops when the clusters stabilize (no more changes).

K-means optimization objective

$c^{(i)}$ = index of cluster $(1, 2, \ldots, K)$ to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid k

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned

$x^{(10)}$ $c^{(10)}$ $\mu_c^{(10)}$

Cost function

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - \mu_{c^{(i)}}\|^2$$

$$\min_{\substack{c^{(1)}, \ldots, c^{(m)} \\ \mu_1, \ldots, \mu_K}} J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

🔹 K-means is Also an Optimization Algorithm

In supervised learning (like linear regression), you define a **cost function** and then **optimize** it (e.g., with gradient descent). Similarly, **K-means** also **minimizes a cost function**, but it uses a different method than gradient descent.

🔸 The Cost Function of K-means (Also Called the "Distortion Function")

The **cost function (J)** in K-means is:

$$J = \frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - \mu_{c^{(i)}}\|^2$$

Where:

- $x^{(i)}$: the i-th data point
- $c^{(i)}$: the index of the cluster assigned to $x^{(i)}$
- $\mu_{c^{(i)}}$: the centroid of the cluster that $x^{(i)}$ is assigned to

👉 This is the **average squared distance** between each point and its assigned cluster centroid.

🔸 Why the Two K-means Steps Minimize This Cost

1. **Assign Points to Closest Centroids (Step 1)**
   For each point x(i) we assign it to the **closest** centroid.
   ✅ This **minimizes** the squared distance from that point to a centroid.

2. **Move Centroids to the Mean of Assigned Points (Step 2)**
   After assigning points, update each centroid to be the **average of the points** assigned to it.
   ✅ This choice **minimizes** the average squared distance for that cluster.

🔸 Why K-means Converges

- Each step of K-means **reduces or keeps the cost function the same**
- Since the cost function can't go below 0, eventually the algorithm **must stop**
- Once the cost stops decreasing → **K-means has converged**

🔸 Practical Use of Cost Function

- 🧠 **Use it to detect convergence**: When the cost stops changing, stop the algorithm.
- 🎲 **Try multiple random initializations**: Run K-means several times with different random centroids and pick the run with the **lowest final cost** for better results.

🔹 Final Takeaway

K-means works by:

1. Assigning each point to its **closest centroid**
2. Moving centroids to the **average of assigned points**
3. Repeating until the **cost function (distortion) stops decreasing**

This process **optimizes** the clustering and guarantees convergence (though not always to the global best).

🔹 K-means Initialization (Short Summary)

- K-means starts by **randomly selecting K training examples** as initial centroids.
- **Different random starts** can lead to **different clustering results** — some good, some bad.
- To get better results:
  - **Run K-means multiple times** (e.g., 100 times) with different initializations.
  - **Pick the best run** — the one with the **lowest cost (distortion J)**.
- This reduces the chance of getting stuck in a **bad local minimum** and gives **better clusters**.

🔹 Practical Tip

- Doing multiple random initializations helps **avoid bad local minima** and usually gives **much better clusters**.
- 100 initializations is common; going beyond 1000 may give little improvement and take more time.

🔹 How to Choose the Number of Clusters in K-means

The **K-means algorithm** requires you to **choose K (the number of clusters)** beforehand. But how do you know what the "right" K is?

✅ 1. No One Correct Answer

- In many real-life datasets, there's no clear or “correct” number of clusters.
- For the same data, some people might see **2 clusters**, others might see **4**, and both can be valid.
- This is because **clustering is unsupervised** — there are no labels to compare with.

✅ 2. Elbow Method

- One method used in research is the **elbow method**:
  - Run K-means with various values of K (e.g., 1 to 10).

- Plot the **cost function (J)** against K. This function shows the average squared distance between points and their assigned centroids.
- Initially, as K increases, the cost reduces sharply.
- At some point, the cost starts decreasing more slowly — this point is called the **"elbow"**, and it's a suggested value for K.

🔸 But this method doesn't always work. In practice, many plots **don't show a clear elbow**, so it can still be ambiguous.

❌ 3. What Not to Do

- Don't pick K just by **minimizing the cost function**.
- More clusters **always reduce the cost**, but it can **overfit** or **make things unnecessarily complex**.

✅ 4. Better Strategy: Choose K Based on Your Goal

- Think about **how you'll use the clusters**:
  - Example: **T-shirt sizing**
    - K=3 → Small, Medium, Large
    - K=5 → XS, S, M, L, XL
  - Both options are valid. But more clusters mean **better fit**, while fewer clusters mean **lower cost and easier production**.
  - So you pick the K that makes the most **business or practical sense**.
- Another example: **image compression**
  - More clusters = better image quality, but larger file size.
  - Fewer clusters = smaller file, but lower image quality.
  - Choose K based on your **trade-off between quality and size**.

🟢 Final Thoughts

- There is **no universal rule** to pick the perfect K.
- It depends on the **use case**, **trade-offs**, and **what you want to do** with the clusters.
- Try different values and choose what works **best for your specific goal**.