Today's Journey

Total Duration: 120 minutes

Section	Time	Focus
1. Motivation	10 min	Why unsupervised + clustering intuition
2. K-Means Deep Dive	55 min	Derive + visualize + algorithm
3. Practical Issues	40 min	Init, scaling, complexity, variants
4. Hierarchical	15 min	Alternative approach + recap

Today's Journey

Total Duration: 120 minutes

Section	Time	Focus
1. Motivation	10 min	Why unsupervised + clustering intuition
2. K-Means Deep Dive	55 min	Derive + visualize + algorithm
3. Practical Issues	40 min	Init, scaling, complexity, variants
4. Hierarchical	15 min	Alternative approach + recap

Key Philosophy: Intuition ⟶ Visualization ⟶ Mathematics ⟶ Code

Supervised vs Unsupervised Learning

Supervised Learning

- **Have**: Features X + Labels Y
- **Goal**: Learn $f : X \to Y$
- **Example**: Spam detection

Clear success metric (accuracy)

Unsupervised Learning

- **Have**: Features X only (no labels!)
- **Goal**: Find **structure/patterns**
- **Example**: Customer segmentation

No "ground truth", success is subjective

Supervised vs Unsupervised Learning

Supervised Learning

- **Have**: Features X + Labels Y
- **Goal**: Learn $f : X \to Y$
- **Example**: Spam detection

Clear success metric (accuracy)

Unsupervised Learning

- **Have**: Features X only (no labels!)
- **Goal**: Find **structure/patterns**
- **Example**: Customer segmentation

No "ground truth", success is subjective

Key Difference: We discover patterns, not predict labels!

Why Unsupervised Learning?

Three Main Reasons:

1. Labels are expensive or impossible to obtain

- Medical images: Need expert radiologists
- Customer behavior: No "true" groupings exist
- Exploratory analysis: Don't know what to look for yet

Why Unsupervised Learning?

Three Main Reasons:

1. Labels are expensive or impossible to obtain

- Medical images: Need expert radiologists
- Customer behavior: No “true” groupings exist
- Exploratory analysis: Don’t know what to look for yet

2. Discover hidden patterns

- Find new disease subtypes from patient data
- Identify market segments you didn’t know existed
- Detect anomalies (fraud, network intrusion)

Why Unsupervised Learning?

Three Main Reasons:

1. Labels are expensive or impossible to obtain

- Medical images: Need expert radiologists
- Customer behavior: No "true" groupings exist
- Exploratory analysis: Don't know what to look for yet

2. Discover hidden patterns

- Find new disease subtypes from patient data
- Identify market segments you didn't know existed
- Detect anomalies (fraud, network intrusion)

3. Data preprocessing

- Dimensionality reduction before supervised learning
- Feature extraction, data compression

Clustering: The Core Task

AIM: Find groups/subgroups in a dataset

REQUIREMENTS: A notion of similarity/dissimilarity

Clustering: The Core Task

AIM: Find groups/subgroups in a dataset

REQUIREMENTS: A notion of similarity/dissimilarity

Central Question: What makes two data points "similar"?

- **Euclidean distance**: $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 = \sqrt{\sum_d \left(x_{id} - x_{jd}\right)^2}$
- **Cosine similarity**: $\cos(\theta) = \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|}$
- **Manhattan distance**: $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1 = \sum_d |x_{id} - x_{jd}|$

Clustering: The Core Task

AIM: Find groups/subgroups in a dataset

REQUIREMENTS: A notion of similarity/dissimilarity

Central Question: What makes two data points "similar"?

- **Euclidean distance**: $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 = \sqrt{\sum_d \left(x_{id} - x_{jd}\right)^2}$
- **Cosine similarity**: $\cos(\theta) = \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|}$
- **Manhattan distance**: $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1 = \sum_d |x_{id} - x_{jd}|$

Today we'll focus on **Euclidean distance** (most common)

K-Means: The Workhorse Algorithm

K-Means Intuition: Choosing K

Question: How many clusters should we find?

Different values of K: K=6 (over), K=5 (optimal), K=4 (under), K=3 (very under)

K-Means Intuition: Choosing K

Question: How many clusters should we find?

Different values of K: K=6 (over), K=5 (optimal), K=4 (under), K=3 (very under)

Key Observation: We need a **quantitative** way to measure cluster quality!

K-Means: Problem Setup

Given:

- N points: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$
- Number of clusters: K (specified in advance)

K-Means: Problem Setup

Given:

- N points: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$
- Number of clusters: K (specified in advance)

Find: Partition into K clusters $C_1, C_2, \ldots, C_K$ such that:

1. Every point belongs to exactly one cluster:

$$C_1 \cup C_2 \cup \ldots \cup C_K = \{1, 2, \ldots, n\}$$

2. Clusters don't overlap (hard assignment):

$$C_i \cap C_j = \emptyset \text{ for } i \neq j$$

K-Means: Objective Function

Goal: Minimize the **total within-cluster variation**

$$\min_{C_1,\ldots,C_K} \sum_{i=1}^{K} \mathrm{WCV}(C_i)$$

K-Means: Objective Function

Goal: Minimize the **total within-cluster variation**

$$\min_{C_1,\ldots,C_K} \sum_{i=1}^{K} \mathrm{WCV}(C_i)$$

For cluster C_i:

$$\mathrm{WCV}(C_i) = \frac{1}{|C_i|} \sum_{a \in C_i} \sum_{b \in C_i} \|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2^2$$

K-Means: Objective Function

Goal: Minimize the **total within-cluster variation**

$$\min_{C_1,\ldots,C_K} \sum_{i=1}^{K} \mathrm{WCV}(C_i)$$

For cluster C_i:

$$\mathrm{WCV}(C_i) = \frac{1}{|C_i|} \sum_{a \in C_i} \sum_{b \in C_i} \|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2^2$$

This can be simplified to:

$$\mathrm{WCV}(C_i) = 2 \sum_{a \in C_i} \|\boldsymbol{x}_a - \boldsymbol{\mu}_i\|_2^2$$

where $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{a \in C_i} \boldsymbol{x}_a$ is the **centroid**

K-Means: Final Objective

Combining everything, K-Means minimizes:

$$\min_{C_1,\ldots,C_K} \sum_{i=1}^{K} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|_2^2$$

K-Means: Final Objective

Combining everything, K-Means minimizes:

$$\min_{C_1,\ldots,C_K} \sum_{i=1}^{K} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|_2^2$$

Alternative names:

- **Inertia** (scikit-learn terminology)
- **Within-Cluster Sum of Squares (WCSS)**
- **Distortion**

Finding the Optimal Centroid

Question: For fixed cluster assignments C_i, what is the best centroid?

Take derivative and set to zero:

$$\frac{\partial}{\partial \boldsymbol{\mu}_i} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|_2^2 = 0$$

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$$

Finding the Optimal Centroid

Question: For fixed cluster assignments C_i, what is the best centroid?

Take derivative and set to zero:

$$\frac{\partial}{\partial \boldsymbol{\mu}_i} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|_2^2 = 0$$

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$$

Result: The optimal centroid is simply the **mean** of all points in the cluster!

K-Means Algorithm

Key Idea: Alternate between two steps:

K-Means Algorithm

Key Idea: Alternate between two steps:

E-Step (Assignment): Fix centroids, assign points to nearest centroid

M-Step (Update): Fix assignments, recompute centroids as means

K-Means Algorithm

Key Idea: Alternate between two steps:

E-Step (Assignment): Fix centroids, assign points to nearest centroid

M-Step (Update): Fix assignments, recompute centroids as means

Repeat until convergence (assignments don't change)

K-Means Algorithm

Key Idea: Alternate between two steps:

E-Step (Assignment): Fix centroids, assign points to nearest centroid

M-Step (Update): Fix assignments, recompute centroids as means

Repeat until convergence (assignments don't change)

Guarantee: Each step decreases (or maintains) the objective $\longrightarrow$ converges to a **local minimum**

K-Means: Step-by-Step Example

Data: 6 points in 2D

$$\boldsymbol{x}_1 = (1,1), \quad \boldsymbol{x}_2 = (2,1), \quad \boldsymbol{x}_3 = (1,2)$$

$$\boldsymbol{x}_4 = (5,4), \quad \boldsymbol{x}_5 = (5,5), \quad \boldsymbol{x}_6 = (6,5)$$

Goal: Cluster into $K = 2$ groups

K-Means: Step-by-Step Example

Data: 6 points in 2D

$$\boldsymbol{x}_1 = (1,1), \quad \boldsymbol{x}_2 = (2,1), \quad \boldsymbol{x}_3 = (1,2)$$

$$\boldsymbol{x}_4 = (5,4), \quad \boldsymbol{x}_5 = (5,5), \quad \boldsymbol{x}_6 = (6,5)$$

Goal: Cluster into $K = 2$ groups

Iteration 0: Randomly initialize centroids

$$\boldsymbol{\mu}_1^{(0)} = (1,1), \quad \boldsymbol{\mu}_2^{(0)} = (5,4)$$

K-Means: Step-by-Step Example

Data: 6 points in 2D

$$\boldsymbol{x}_1 = (1, 1), \quad \boldsymbol{x}_2 = (2, 1), \quad \boldsymbol{x}_3 = (1, 2)$$

$$\boldsymbol{x}_4 = (5, 4), \quad \boldsymbol{x}_5 = (5, 5), \quad \boldsymbol{x}_6 = (6, 5)$$

Goal: Cluster into $K = 2$ groups

Iteration 0: Randomly initialize centroids

$$\boldsymbol{\mu}_1^{(0)} = (1, 1), \quad \boldsymbol{\mu}_2^{(0)} = (5, 4)$$

Iteration 1: Assign to nearest $\longrightarrow C_1 = \{1, 2, 3\}, C_2 = \{4, 5, 6\}$

Update centroids $\longrightarrow \boldsymbol{\mu}_1 = (1.33, 1.33), \boldsymbol{\mu}_2 = (5.33, 4.67)$

K-Means: Step-by-Step Example

Data: 6 points in 2D

$$\boldsymbol{x}_1 = (1, 1), \quad \boldsymbol{x}_2 = (2, 1), \quad \boldsymbol{x}_3 = (1, 2)$$

$$\boldsymbol{x}_4 = (5, 4), \quad \boldsymbol{x}_5 = (5, 5), \quad \boldsymbol{x}_6 = (6, 5)$$

Goal: Cluster into $K = 2$ groups

Iteration 0: Randomly initialize centroids

$$\boldsymbol{\mu}_1^{(0)} = (1, 1), \quad \boldsymbol{\mu}_2^{(0)} = (5, 4)$$

Iteration 1: Assign to nearest $\longrightarrow C_1 = \{1, 2, 3\}, C_2 = \{4, 5, 6\}$

Update centroids $\longrightarrow \boldsymbol{\mu}_1 = (1.33, 1.33), \boldsymbol{\mu}_2 = (5.33, 4.67)$

Iteration 2: Assignments don't change $\longrightarrow$ Converged!

Why K-Means Converges

Theorem: K-Means algorithm converges in finite iterations

Why K-Means Converges

Theorem: K-Means algorithm converges in finite iterations

Proof sketch:

1. **Assignment step**: objective decreases
2. **Update step**: objective decreases
3. **Finite states**: Only finitely many partitions
4. Monotonic decrease + finite states $\longrightarrow$ Must converge!

Why K-Means Converges

Theorem: K-Means algorithm converges in finite iterations

Proof sketch:

1. **Assignment step**: objective decreases
2. **Update step**: objective decreases
3. **Finite states**: Only finitely many partitions
4. Monotonic decrease + finite states $\longrightarrow$ Must converge!

Warning: K-Means finds a **local minimum**, not necessarily **global**!

Solution: Run multiple times with different random initializations

Practical Issues & Advanced Variants

Issue #1: Poor Initialization

Problem: Random initialization $\longrightarrow$ bad local minima

Issue #1: Poor Initialization

Problem: Random initialization $\longrightarrow$ bad local minima

Classic solution: Run K-Means 10-100 times, pick best

Issue #1: Poor Initialization

Problem: Random initialization $\longrightarrow$ bad local minima

Classic solution: Run K-Means 10-100 times, pick best

Better solution: K-Means++ (smart initialization)

- Choose first centroid randomly
- For each next centroid, choose with probability $\propto D(x)^2$
- Points far from existing centroids have higher chance

Issue #1: Poor Initialization

Problem: Random initialization $\longrightarrow$ bad local minima

Classic solution: Run K-Means 10-100 times, pick best

Better solution: K-Means++ (smart initialization)

- Choose first centroid randomly
- For each next centroid, choose with probability $\propto D(x)^2$
- Points far from existing centroids have higher chance

Theorem (Arthur & Vassilvitskii, 2007): K-Means++ is $O(\log K)$-competitive

Issue #2: Feature Scaling

Problem: Features with large ranges dominate distance calculations

Example: Age (20-80) vs Income ($20k-$200k)

Issue #2: Feature Scaling

Problem: Features with large ranges dominate distance calculations

Example: Age (20-80) vs Income ($20k-$200k)

Distance heavily influenced by income, age almost irrelevant!

Issue #2: Feature Scaling

Problem: Features with large ranges dominate distance calculations

Example: Age (20-80) vs Income (\$20k-\$200k)

Distance heavily influenced by income, age almost irrelevant!

Solution: Standardize features: $x'_j = \frac{x_j - \mu_j}{\sigma_j}$ (z-score)

Issue #3: Time Complexity

K-Means complexity: $O(n \cdot K \cdot d \cdot T)$

where:

- n = number of points
- K = number of clusters
- d = dimensionality
- T = number of iterations (typically 10-100)

Issue #3: Time Complexity

K-Means complexity: $O(n \cdot K \cdot d \cdot T)$

where:

- n = number of points
- K = number of clusters
- d = dimensionality
- T = number of iterations (typically 10-100)

Mini-Batch K-Means: Use random sample of b points per iteration

10-100x faster, slight accuracy loss

Issue #4: Choosing K (Elbow Method)

Problem: How do we pick K?

Issue #4: Choosing K (Elbow Method)

Problem: How do we pick K?

Elbow Method: Plot inertia vs K, look for "elbow"

$$\text{Inertia}(K) = \sum_{i=1}^{K} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

Issue #4: Choosing K (Elbow Method)

Problem: How do we pick K?

Elbow Method: Plot inertia vs K, look for "elbow"

$$\text{Inertia}(K) = \sum_{i=1}^{K} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

Warning: "Elbow" often subjective!

Alternative: Silhouette score, gap statistic

Issue #5: Non-Convex Shapes

K-Means Assumption: Clusters are convex, isotropic (spherical)

Issue #5: Non-Convex Shapes

K-Means Assumption: Clusters are convex, isotropic (spherical)

Limitation: K-Means uses **linear decision boundaries** (Voronoi cells)

Cannot capture complex shapes

Issue #5: Non-Convex Shapes

K-Means Assumption: Clusters are convex, isotropic (spherical)

Limitation: K-Means uses **linear decision boundaries** (Voronoi cells)

Cannot capture complex shapes

Solutions:

- DBSCAN (density-based)
- Spectral clustering (graph-based)
- GMM (elliptical clusters)

Summary: K-Means Techniques

Technique	Problem	When to Use
K-Means++	Poor initialization	Always!
Standardization	Feature scale mismatch	Different units/ranges
Mini-Batch	Large datasets	$n > 10,000$
Elbow/Silhouette	Choosing K	K unknown
DBSCAN/GMM	Non-convex shapes	Arbitrary shapes

Hierarchical Clustering

When K-Means Fails

Problems with K-Means:

- Need to specify K in advance
- Assumes spherical, equal-sized clusters
- No hierarchy

When K-Means Fails

Problems with K-Means:

- Need to specify K in advance
- Assumes spherical, equal-sized clusters
- No hierarchy

Hierarchical Clustering: Builds a **tree** (dendrogram)

No need to specify K in advance!

Linkage Criteria

Problem: What is the distance between two **clusters**?

Linkage Criteria

Problem: What is the distance between two **clusters**?

- **Single**: Distance between **closest** points
- **Complete**: Distance between **farthest** points
- **Average**: Average distance between all pairs
- **Centroid**: Distance between centroids

Linkage Criteria

Problem: What is the distance between two **clusters**?

- **Single**: Distance between **closest** points
- **Complete**: Distance between **farthest** points
- **Average**: Average distance between all pairs
- **Centroid**: Distance between centroids

Choice matters! Different linkages $\longrightarrow$ different dendrograms

Hierarchical vs K-Means

Aspect	K-Means	Hierarchical
K specified?	Yes	No
Shape	Spherical	More flexible
Scalability	Fast: $O(nKdT)$	Slow: $O(n^2 \log n)$
Deterministic?	No	Yes
Best for	Large data, K known	Small data, explore K

Hierarchical vs K-Means

Aspect	K-Means	Hierarchical
K specified?	Yes	No
Shape	Spherical	More flexible
Scalability	Fast: $O(nKdT)$	Slow: $O(n^2 \log n)$
Deterministic?	No	Yes
Best for	Large data, K known	Small data, explore K

Recommendation:

- $n < 10,000$ and need hierarchy $\longrightarrow$ Hierarchical
- $n > 10,000$ or K known $\longrightarrow$ K-Means

Summary: Key Takeaways

1. K-Means is the workhorse:

- Minimize within-cluster sum of squares
- Iterative algorithm (E-step + M-step)
- Converges to local minimum
- Use K-Means++ initialization

Summary: Key Takeaways

1. K-Means is the workhorse:

- Minimize within-cluster sum of squares
- Iterative algorithm (E-step + M-step)
- Converges to local minimum
- Use K-Means++ initialization

2. Practical considerations:

- **Always** standardize features
- Choose K via elbow/silhouette
- Mini-batch for large data

Summary: Key Takeaways

1. K-Means is the workhorse:

- Minimize within-cluster sum of squares
- Iterative algorithm (E-step + M-step)
- Converges to local minimum
- Use K-Means++ initialization

2. Practical considerations:

- **Always** standardize features
- Choose K via elbow/silhouette
- Mini-batch for large data

3. Alternatives:

- **Hierarchical**: No K needed, slow
- **DBSCAN**: Non-convex shapes, finds outliers
- **GMM**: Soft assignments, elliptical clusters

Questions?

Nipun Batra

IIT Gandhinagar

nipun.batra@iitgn.ac.in