5.1 - Clustering Methods

Giovanni Della Lunga
giovanni.dellalunga@unibo.it

Introduction to Machine Learning for Finance

Bologna - February-March, 2026

Lecture 5.1 — What This Lesson Is About

Unsupervised learning is the branch of machine learning where we:

- do **not** have labels/targets (y),
- do **not** have a clear notion of "accuracy" with respect to ground truth,
- try to extract **structure** from data by making explicit modeling choices.

Key question of the lecture:

What structure, if any, is hidden in financial data when no target is provided?

Positioning of This Lecture in the Course

This lecture shifts the focus:

- from **prediction** $\rightarrow$ to **structure discovery**,
- from **loss minimization** $\rightarrow$ to **similarity & representation**,
- from **out-of-sample accuracy** $\rightarrow$ to **interpretability and robustness**.

In finance, unsupervised methods are often used to:

- build **data-driven groupings** (assets, customers, countries),
- reveal **market structure** and **risk concentrations**,
- support **downstream supervised models** (segmented scoring, features, regimes).

Learning Objectives

By the end of this lecture, you should be able to:

- define **unsupervised learning** and distinguish it from supervised learning,
- explain what **clustering** is and why it is central in unsupervised learning,
- choose and justify a **distance/similarity metric** for a given financial problem,
- describe and compare three core clustering families:
  - **centroid-based** (K-Means),
  - **connectivity-based** (Hierarchical),
  - **density-based** (DBSCAN),
- discuss **validation without ground truth** (internal metrics, stability, domain review),
- connect clustering outputs to concrete **financial use cases**.

Unsupervised Learning

Unsupervised Learning: The Big Picture

Unsupervised learning aims to learn patterns from X alone:

$$X = \{x_1, \ldots, x_n\}, \quad x_i \in \mathbb{R}^p$$

Typical goals:

- **Grouping** observations (clustering),
- **Reducing** dimensionality (representation learning),
- **Estimating** structure/distributions (density estimation),
- **Detecting** unusual behavior (anomaly/novelty detection).

Important: without labels, results are **model- and choice-dependent**.

Clustering Within Unsupervised Learning

Clustering is one of the most widely used unsupervised tasks:

- It produces **discrete groupings**: a mapping $x_i \mapsto c_i$ where $c_i \in \{1, \dots, k\}$,
- It is driven by a notion of **similarity** (or distance),
- It is useful when we suspect that data are generated by **multiple regimes/types**.

In finance, clustering is often used for:

- asset grouping (diversification, correlation structure),
- customer segmentation (products, pricing, retention),
- country/issuer risk buckets,
- anomaly screening (transactions, behaviors).

Why Unsupervised Learning Is Tricky in Finance

Finance is a difficult environment for unsupervised learning because:

- data are **noisy** and often **non-stationary**,
- different scalings/transformations can change results dramatically,
- there is rarely a unique "true" clustering (multiple answers can be useful),
- interpretability and governance matter: clusters must be **economically meaningful**.

Key message: in unsupervised learning, you must document **assumptions**, **preprocessing**, and **validation strategy**.

A Map of Choices: What You Control

Even before choosing an algorithm, you control:

- **Data representation**: returns vs. prices, horizons, scaling,
- **Feature set**: which variables enter X,
- **Distance/metric**: Euclidean, correlation-based, etc.,
- **Algorithm family**: centroid / hierarchical / density / distribution,
- **Validation**: internal metrics, stability, expert review.

This lecture focuses on **clustering methods**, but the same logic applies to all unsupervised models.

Introduction to Clustering

The Clustering Problem

Definition

In unsupervised learning, clustering groups data points into subsets (clusters) where points within the same cluster are more similar to each other than to those in other clusters.

Key Characteristics:

- **Unlabeled data** - No pre-assigned categories
- **Pattern discovery** - Identify inherent structures
- **No ground truth** - Must define similarity metrics
- **Exploratory analysis** - Reveal hidden patterns

Goal

Partition data such that intra-cluster similarity is maximized and inter-cluster similarity is minimized.

Key Aspects of Clustering

1. **Objective:**
   - Group similar data points together
   - Definition of "similarity" varies by algorithm and data nature
2. **Unlabeled Data:**
   - No labeled examples to learn from
   - Discover patterns based on inherent properties
3. **Algorithm Selection:**
   - K-means, Hierarchical, DBSCAN, GMM, etc.
   - Choice depends on data nature, desired clusters, computation
4. **Evaluation:**
   - Challenging without ground truth labels
   - Metrics: Silhouette score, Davies-Bouldin index
   - Visual inspection often necessary

Similarity and Distance Metrics

Fundamental Concept

Clustering efficacy depends on the chosen similarity or distance metric.

Common Distance Measures:

Euclidean Distance (2D):

$$\text{Distance} = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

Multidimensional (m features):

$$d = \sqrt{\sum_{j=1}^{m} (\nu_{pj} - \nu_{qj})^2}$$

where ν_{pj} and ν_{qj} are the values of the j-th feature for P and Q

Cluster Morphology and Challenges

Cluster Diversity:

- Varying shapes, sizes, and distributions
- Complex structures with overlapping boundaries
- Determination of optimal number of clusters

Major Challenges:

1. **No universal definition** - Context-dependent
2. **Algorithm selection** - Data characteristics matter
3. **High dimensionality** - Curse of dimensionality
4. **Scalability** - Large dataset performance
5. **Interpretation** - Domain expertise needed

Important

Clustering is inherently **subjective** - different algorithms may produce different valid results!

Applications in Finance

Clustering reveals hidden patterns and informs decision-making:

1. **Portfolio Management**
   - Categorize assets by returns, volatility
   - Diversification and correlation analysis
2. **Fraud Detection**
   - Identify unusual patterns and anomalies
   - Detect money laundering activities
3. **Customer Segmentation**
   - Group by spending habits, risk tolerance
   - Personalized marketing and products
4. **Credit Scoring**
   - Group borrowers by credit characteristics
   - Improve risk assessment models

More Financial Applications

5. **Market Structure Analysis**
   - Identify market segments
   - Understand participant behavior
   - Inform trading strategies
6. **Risk Management**
   - Group by risk characteristics
   - Understand systemic risks
   - Develop mitigation strategies
7. **Operational Efficiency**
   - Identify similar processes
   - Optimize resource allocation
   - Improve customer service

Key Benefit

Clustering provides powerful tools for navigating financial market complexities!

Four Types of Clustering Methods

1. **Centroid-Based Clustering**
   - Clusters represented by centroids
   - Based on distance to cluster center
   - Example: **K-Means**
2. **Connectivity-Based Clustering**
   - Groups nearest neighbors by distance
   - Examples: **Hierarchical**, Spectral
3. **Density-Based Clustering**
   - Defined by areas of concentrated density
   - Example: **DBSCAN**
4. **Distribution-Based Clustering**
   - Points belong to probability distributions
   - Example: **Gaussian Mixture Models**

K-Means Clustering

K-Means: Introduction

Definition

K-Means is one of the simplest and most popular unsupervised ML algorithms for grouping similar data points.

Objective:

- Find a fixed number (k) of clusters
- Minimize within-cluster sum of squares
- Each data point assigned to nearest centroid

Key Concepts:

- **Centroid** - Center point of a cluster (real or imaginary)
- k - Number of clusters (predefined)
- **Assignment** - Allocate points to nearest cluster

The "Means" in K-Means

How K-Means Works

Algorithm Steps:

1. **Initialize** - Randomly select k centroids
2. **Assign** - Allocate each point to nearest centroid
3. **Update** - Recalculate centroid positions (mean of assigned points)
4. **Repeat** - Iterate steps 2-3 until convergence

Stopping Criteria:

- Centroids stabilize (no change in values)
- Maximum number of iterations reached

Local vs Global Optima

Important Issue

K-Means is guaranteed to converge, but may converge to a **local optimum** depending on initial centroid positions!

Solution:

- Run algorithm multiple times with different random initializations
- Keep the best solution (lowest inertia)

In Scikit-Learn:

- Controlled by `n_init` hyperparameter
- Default: `n_init=10` (runs 10 times)
- Keeps best solution automatically

K-Means Implementation Example

Basic Python implementation:

```
from sklearn.cluster import KMeans
import numpy as np

# Generate random data
X = np.random.rand(300, 2)

# Create KMeans object
kmeans = KMeans(n_clusters=3, n_init='auto')

# Fit the model
kmeans.fit(X)

# Get cluster centers
centers = kmeans.cluster_centers_
```

Inertia (Within-Cluster Sum of Squares)

Definition

Inertia measures the performance of K-Means clustering.

Formula:

$$\text{Inertia} = \sum_{i=1}^{n} d_i^2$$

where d_i is the distance of observation i from its cluster center

Objective:

- Minimize inertia for any given k
- Lower inertia = tighter clusters

Caution

Inertia alone is NOT a good metric for choosing k - it always decreases as k increases!

Finding the Right Number of Clusters

Challenge

Determining optimal k is crucial but not straightforward.

Approaches:

1. **Elbow Method**
   - Plot inertia vs. number of clusters
   - Look for "elbow" where decrease slows
   - Balance between fit and complexity
2. **Silhouette Analysis**
   - More precise but computationally expensive
   - Measures how well points fit their clusters
   - Values range from -1 to $+1$
3. **Domain Expertise**
   - Use knowledge of the problem domain
   - Consider business requirements

Elbow Method

Procedure:

1. Plot Within-Cluster Sum of Squares (WCSS) vs. k
2. WCSS measures cluster compactness
3. Identify "elbow" point where decrease slows significantly
4. This point suggests good balance

Interpretation:

- Elbow at $k = 3$ suggests 3 clusters
- Before elbow: dramatic improvement
- After elbow: diminishing returns
- May split good clusters unnecessarily

Note

Elbow method is somewhat subjective - the "elbow" may not always be clear!

Silhouette Score

Definition

Silhouette score measures how similar a point is to its own cluster compared to other clusters.

Formula:

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

where:

- $a(i)$ = mean intra-cluster distance
- $b(i)$ = mean nearest-cluster distance

Interpretation:

- $s \approx +1$ - Well inside own cluster, far from others (**good**)
- $s \approx 0$ - Close to cluster boundary (**uncertain**)
- $s \approx -1$ - May be assigned to wrong cluster (**bad**)

Silhouette Analysis Implementation

```
from sklearn.metrics import silhouette_score

# Test different numbers of clusters
range_n_clusters = [2, 3, 4, 5, 6, 7, 8, 9, 10]
silhouette_scores = []

for n_clusters in range_n_clusters:
    # Fit KMeans
    clusterer = KMeans(n_clusters=n_clusters, random_state=0)
    cluster_labels = clusterer.fit_predict(X)

    # Calculate silhouette score
    silhouette_avg = silhouette_score(X, cluster_labels)
    silhouette_scores.append(silhouette_avg)

    print(f"For n_clusters = {n_clusters}, ...")
```

K-Means: Country Risk Example

Problem: Understanding country risk for foreign investment

Features:

- GDP growth rate (IMF)
- Corruption index (Transparency International)
- Peace index (Institute for Economics and Peace)
- Legal Risk Index (Property Rights Association)

Dataset:

- 122 countries
- Different scales for each feature
- Need for normalization

Key Insight

Features have different variances - normalization is essential!

Feature Selection and Normalization

Observations:

- Corruption and Legal indices highly correlated (0.92)
- GDP Growth has skewed distribution
- Different scales affect distance calculations

Solutions:

1. **Drop Corruption** - Keep only Peace, Legal, GDP Growth
2. **Normalize** - Make features equally weighted

Normalization Formula:

$$X_{\text{normalized}} = \frac{X - \mu}{\sigma}$$

Results in mean $= 0$, standard deviation $= 1$ for all features

Country Risk: Results

Optimal Clusters:

- Elbow method suggests $k = 3$ or $k = 4$
- Silhouette analysis confirms $k = 3$ as best
- Maximum average silhouette score at $k = 3$

Three Risk Clusters Identified:

1. **Low Risk** - High Legal, Low Peace index, Stable GDP
   - Examples: US, UK, Switzerland, Japan
2. **Medium Risk** - Moderate indicators
   - Examples: Most developing countries
3. **High Risk** - Low Legal, High Peace index, Volatile GDP
   - Examples: Venezuela, Ukraine, Yemen

Limitations of K-Means

Despite many merits (fast, scalable), K-Means has limitations:

1. **Multiple runs needed** - Avoid suboptimal solutions
2. **Must specify** k - Number of clusters not automatic
3. **Assumes spherical clusters** - Struggles with:
   - Varying cluster sizes
   - Different densities
   - Non-spherical shapes
4. **Sensitive to outliers** - Centroids can be pulled away
5. **Scale dependent** - Features must be normalized

When to Use

K-Means works best with roughly spherical, similarly-sized clusters of comparable density.

Hierarchical Clustering

Hierarchical Clustering: Definition

Overview

Hierarchical clustering builds a hierarchy of clusters either bottom-up (agglomerative) or top-down (divisive).

Two Approaches:

1. **Agglomerative (Bottom-Up):**
   - Start: Each point is its own cluster
   - Iteratively merge closest clusters
   - End: All points in one cluster
   - **Most commonly used**
2. **Divisive (Top-Down):**
   - Start: All points in one cluster
   - Recursively partition into smaller clusters
   - End: Each point is its own cluster

The Dendrogram

Visualization Tool

Both approaches produce a **dendrogram** - a tree-like diagram showing cluster hierarchy.

Key Features:

- **Height** - Represents distance between merged clusters
- **Branches** - Show which clusters merge and when
- **Leaves** - Individual data points
- **Cutting** - Choose number of clusters by cutting at specific height

Advantages:

- Visual representation of cluster structure
- Shows relationships at multiple levels
- Provides insights into data organization

Linkage Criteria

Definition

Linkage criterion determines distance between clusters for merging decisions.

Common Linkage Methods:

1. **Single Linkage (Nearest Point):**
   - Distance = minimum pairwise distance
   - Can create "chaining" effect
   - Long, straggly clusters
2. **Complete Linkage (Farthest Point):**
   - Distance = maximum pairwise distance
   - Produces compact, tight clusters
   - Preferred for well-separated clusters
3. **Average Linkage:**
   - Distance = average of all pairwise distances

More Linkage Methods

4. **Centroid Linkage:**
   - Distance between cluster centroids
   - Mean point of each cluster
5. **Ward's Method (Minimum Variance):**
   - Minimizes total within-cluster variance
   - Merges clusters with minimum between-cluster distance
   - Tends to create roughly equal-sized clusters
   - **Often preferred in practice**

Important

Choice of linkage criterion significantly affects cluster shape and size - choose based on data nature and desired resolution!

The Distance Matrix

Definition

A table showing pairwise distances between all objects in the dataset.

Properties:

- For N objects: $N \times N$ matrix
- **Diagonal** $= 0$ (distance to self)
- **Symmetric** - $d(i, j) = d(j, i)$
- Contains all pairwise dissimilarities

Role in Hierarchical Clustering:

1. Algorithm consults matrix to find closest clusters
2. After each merge, matrix is updated
3. Updated to reflect distances to new clusters
4. Process continues until hierarchy complete

Data Normalization Importance

Problem

Without normalization, variables with larger values dominate distance calculations.

Example Scenario:

- Body weight (kg) and diastolic BP (mmHg) have similar values
- But diastolic and systolic BP have similar **patterns**
- Weight and height also have similar patterns

Solution - Standardization:

- Transform all variables: mean $= 0$, std $= 1$
- Variables with similar patterns have short distances
- Pattern-based clustering becomes possible

Result

After standardization: BP variables cluster together, body measurements cluster together!

Hierarchical Clustering Implementation

```
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering
import matplotlib.pyplot as plt

# Generate linkage matrix
Z = linkage(data, method='ward')

# Plot dendrogram
plt.figure(figsize=(10, 5))
dendrogram(Z, labels=labels, leaf_rotation=90)
plt.title("Dendrogram")
plt.xlabel("Data Points")
plt.ylabel("Distance")
plt.show()

# Alternative: Use scikit-learn
```

Advantages of Hierarchical Clustering

Key Benefits:

1. **No need to specify k**
   - Unlike K-means
   - Can choose later by cutting dendrogram
2. **Flexibility and richness**
   - Captures relationships at different levels
   - More nuanced view of data structure
3. **Easy to interpret**
   - Dendrogram provides intuitive visualization
   - Clear representation of clustering process
4. **Deterministic**
   - Same result every time
   - No random initialization

Limitations of Hierarchical Clustering

Challenges:

1. **Computational complexity**
   - $O(n^2 \log n)$ or $O(n^3)$ depending on method
   - Less scalable than K-means for large datasets
2. **Sensitive to noise and outliers**
   - Can lead to misinterpretations
   - Affects cluster structure
3. **Irreversibility**
   - Once merged/split, cannot be undone
   - May lead to suboptimal clustering
4. **Memory intensive**
   - Must store distance matrix
   - Challenging for very large datasets

DBSCAN

DBSCAN: Introduction

Full Name

Density-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise

Core Concept:

- Forms clusters based on **dense regions** of points
- Does not require specifying number of clusters
- Explicitly handles outliers as "noise"

Key Parameters:

1. ϵ **(eps)** - Radius of neighborhood
2. **MinPts** - Minimum points to form dense region

Unique Feature

DBSCAN can find arbitrarily shaped clusters and identify outliers!

DBSCAN: How It Works

Point Classifications:

1. **Core Points:**
   - Have at least MinPts within ϵ radius
   - Form the "backbone" of clusters
2. **Border Points:**
   - Fewer than MinPts in neighborhood
   - But within ϵ of a core point
   - Included in cluster but not core
3. **Noise Points:**
   - Neither core nor border points
   - Not included in any cluster
   - Identified as outliers

Cluster Formation:

DBSCAN: Advantages

Key Benefits:

1. **No need to specify number of clusters**
   - Discovers clusters automatically
   - Unlike K-means
2. **Finds arbitrarily shaped clusters**
   - Not limited to spherical shapes
   - Handles complex geometries
3. **Robust to outliers**
   - Explicitly identifies noise
   - Outliers don't affect cluster formation
4. **Only two parameters**
   - ϵ and MinPts
   - Relatively intuitive

DBSCAN: Disadvantages

Limitations:

1. **Struggles with varying densities**
   - Single ϵ may not work for all clusters
   - Clusters with different densities problematic
2. **Parameter selection challenging**
   - Choosing right ϵ and MinPts difficult
   - Requires domain knowledge or tuning
3. **Sensitive to parameter values**
   - Small changes can significantly affect results
4. **High-dimensional challenges**
   - Distance metrics less meaningful
   - Curse of dimensionality

DBSCAN Implementation

```
from sklearn.cluster import DBSCAN
from sklearn.datasets import make_moons
import matplotlib.pyplot as plt

# Generate moon-shaped data
X, y = make_moons(n_samples=200, noise=0.05, random_state=0)

# Apply DBSCAN
db = DBSCAN(eps=0.2, min_samples=5, metric='euclidean')
labels = db.fit_predict(X)

# Plot results
plt.scatter(X[labels==0, 0], X[labels==0, 1],
            c='lightblue', label='Cluster 1')
plt.scatter(X[labels==1, 0], X[labels==1, 1],
```

Comparison and Best Practices

Clustering Methods Comparison

Method	Advantages	Disadvantages	Best For
K-Means	Fast, scalable, simple	Must specify k, assumes spherical	Large datasets, spherical clusters
Hierarchical	No k needed, interpretable	Slow, memory intensive	Small datasets, hierarchy exploration
DBSCAN	Arbitrary shapes, handles outliers	Parameter selection hard	Non-spherical clusters, noisy data

Key Insight

No single "best" algorithm - choice depends on data characteristics and objectives!

Choosing the Right Algorithm

Decision Factors:

1. **Dataset size**
   - Large → K-Means
   - Small/Medium → Hierarchical or DBSCAN
2. **Cluster shape**
   - Spherical → K-Means
   - Arbitrary → DBSCAN
3. **Number of clusters**
   - Known → K-Means
   - Unknown → Hierarchical or DBSCAN
4. **Noise/Outliers**
   - Present → DBSCAN
   - Minimal → K-Means or Hierarchical

Best Practices for Clustering

Data Preparation:

1. **Normalize/Standardize** features
2. **Handle missing values** appropriately
3. **Remove or flag outliers** (except for DBSCAN)
4. **Consider dimensionality reduction** for high-D data

Algorithm Selection:

1. **Try multiple algorithms** and compare
2. **Use appropriate metrics** (Silhouette, Davies-Bouldin)
3. **Visualize results** when possible
4. **Consider domain knowledge**

Validation:

1. **Internal metrics** - Silhouette, inertia
2. **External validation** - If labels available
3. **Stability analysis** - Multiple runs
4. **Domain validation** - Expert review

Evaluation Metrics Summary

Internal Validation (No ground truth):

1. **Silhouette Score:**
   - Range: [-1, 1]
   - Higher is better
   - Measures separation and cohesion
2. **Davies-Bouldin Index:**
   - Lower is better
   - Ratio of within-cluster to between-cluster distances
3. **Calinski-Harabasz Index:**
   - Higher is better
   - Ratio of between-cluster to within-cluster variance

Note

Always combine metrics with visual inspection and domain expertise!

Common Pitfalls to Avoid

1. **Ignoring scale differences**
   - Always normalize features
2. **Not trying multiple algorithms**
   - Different algorithms reveal different patterns
3. **Blindly trusting metrics**
   - Combine with visualization and domain knowledge
4. **Overfitting**
   - Too many clusters can split natural groups
5. **Ignoring interpretability**
   - Clusters should make business sense
6. **Not validating results**
   - Check stability across multiple runs

Financial Applications

Portfolio Management with Clustering

Application: Group assets for diversification

Approach:

1. Collect features: returns, volatility, beta, sector
2. Apply clustering (K-Means or Hierarchical)
3. Identify asset groups with similar characteristics

Benefits:

- Understand correlation patterns
- Improve diversification strategy
- Identify risk concentrations
- Inform asset allocation decisions

Example Features:

- Annualized returns
- Volatility (standard deviation)
- Beta (market sensitivity)

Fraud Detection with Clustering

Application: Identify unusual transaction patterns

Approach:

1. Cluster accounts/transactions by behavior
2. Identify outliers (DBSCAN works well)
3. Investigate anomalies for potential fraud

Features to Consider:

- Transaction amount
- Transaction frequency
- Time of day patterns
- Geographic location
- Merchant categories
- Account age

Why DBSCAN?

Customer Segmentation

Application: Group customers for targeted services

Segmentation Criteria:

- Spending habits and patterns
- Income levels
- Investment preferences
- Risk tolerance
- Age and demographics
- Product holdings

Business Value:

1. **Personalized marketing** - Targeted campaigns
2. **Product development** - Tailored offerings
3. **Customer retention** - Identify at-risk segments
4. **Resource allocation** - Focus on high-value segments

Credit Scoring Enhancement

Traditional Approach: Linear models with fixed criteria

Clustering Enhancement:

1. Cluster borrowers by characteristics
2. Develop segment-specific scoring models
3. More nuanced risk assessment

Benefits:

- Better risk differentiation
- Reduced default rates
- More informed lending decisions
- Fairer assessment across different groups

Features:

- Payment history
- Debt-to-income ratio
- Employment stability

Summary and Conclusions

Key Takeaways

Clustering Fundamentals:

- Unsupervised learning for pattern discovery
- No labeled data required
- Multiple algorithms for different scenarios

Three Main Algorithms:

1. **K-Means** - Fast, scalable, spherical clusters
2. **Hierarchical** - No k needed, interpretable hierarchy
3. **DBSCAN** - Arbitrary shapes, handles noise

Critical Success Factors:

- Proper data preprocessing and normalization
- Appropriate algorithm selection
- Careful parameter tuning
- Validation with multiple metrics
- Domain expertise integration

Financial Applications Recap

Clustering provides powerful tools for:

1. **Portfolio Management**
   - Asset grouping and diversification
2. **Fraud Detection**
   - Anomaly identification
3. **Customer Segmentation**
   - Targeted marketing and services
4. **Credit Scoring**
   - Enhanced risk assessment
5. **Market Analysis**
   - Structure understanding
6. **Risk Management**
   - Systemic risk identification

Best Practices Summary

1. **Always normalize** your features
2. **Try multiple algorithms** - compare results
3. **Use appropriate evaluation metrics**
   - Silhouette score
   - Elbow method
   - Domain-specific metrics
4. **Visualize when possible** - 2D/3D plots
5. **Validate results**
   - Multiple runs
   - Cross-validation
   - Expert review
6. **Consider interpretability** - clusters should make sense
7. **Document assumptions** and parameter choices

Further Learning

Advanced Topics:

- Gaussian Mixture Models (GMM)
- Spectral Clustering
- Affinity Propagation
- OPTICS (extension of DBSCAN)
- Semi-supervised clustering

Recommended Resources:

1. Géron A. - "Hands-On Machine Learning" (O'Reilly)
2. Hull J.C. - "Machine Learning in Business" (2019)
3. Scikit-learn documentation
4. Financial data science papers

Practice:

- Work with real financial datasets
- Kaggle competitions

Conclusions

Key Takeaways (1): What Clustering Really Does

- Clustering does **not** "discover truth"; it discovers **structure under assumptions**.
- The result is only as good as:
  - your **features** (what you measure),
  - your **metric** (what "similar" means),
  - your **preprocessing** (scaling, outliers, missingness),
  - your **algorithm choice** (what shapes you can represent).
- In finance, the main goal is often **robust segmentation**, not maximum compactness.

Key Takeaways (2): When to Use Which Family

A practical mental model:

- **K-Means (centroid-based):** fast and scalable, works best for roughly spherical clusters; requires k.
- **Hierarchical:** interpretable multi-scale structure; can choose k later; less scalable.
- **DBSCAN (density-based):** finds arbitrary shapes and identifies noise/outliers; parameter-sensitive; struggles with varying densities.

No universal best method: pick the method that matches your data geometry and objective.

Key Takeaways (3): Validation Without Ground Truth

Without labels, you validate clustering by combining:

- **Internal metrics** (e.g., silhouette, Davies–Bouldin),
- **Stability analysis** (repeat runs, perturbations, time-slices),
- **Visualization** (when possible),
- **Domain expert review** (economic interpretability).

Rule of thumb: if clusters do not tell a coherent financial story, they are not useful.

Closing: How Clustering Fits the Bigger ML Pipeline

Clustering is often a **building block**:

- **Exploration:** discover structure before modeling,
- **Feature engineering:** cluster assignments as features,
- **Segmentation:** build separate supervised models per cluster,
- **Risk monitoring:** detect changes in cluster composition over time.

Final message: unsupervised learning is not about avoiding assumptions — it is about making them **explicit** and validating them **critically**.

Questions & Discussion

Discussion prompts:

- When would you choose clustering instead of a supervised model?
- What does “similar” mean for assets: returns, correlation, volatility, fundamentals?
- How would you check whether clustering results are stable over time?