

# Machine Learning for Smarter Innovation

## Week 1: Foundations & Clustering

Discovering Innovation Patterns with ML

BSc Course in AI-Enhanced Innovation

# Prerequisites & What You Need

Setting You Up for Success

## What You Need to Know

- Basic Python (variables, loops, functions)
- High school math (averages, distances)
- How to use Jupyter notebooks
- Basic data concepts (tables, rows, columns)

## What We'll Provide

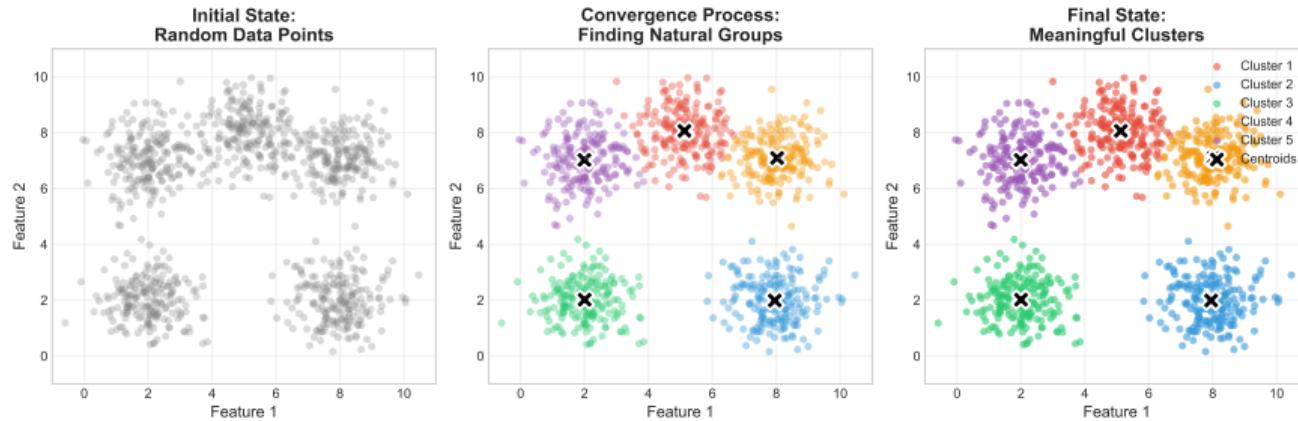
- All code templates
- Step-by-step instructions
- Visual explanations
- Practice datasets

No machine learning experience required!

# Machine Learning + Innovation + Design Thinking

The Power of Convergent Methodologies

The Convergence Flow: From Chaos to Clarity



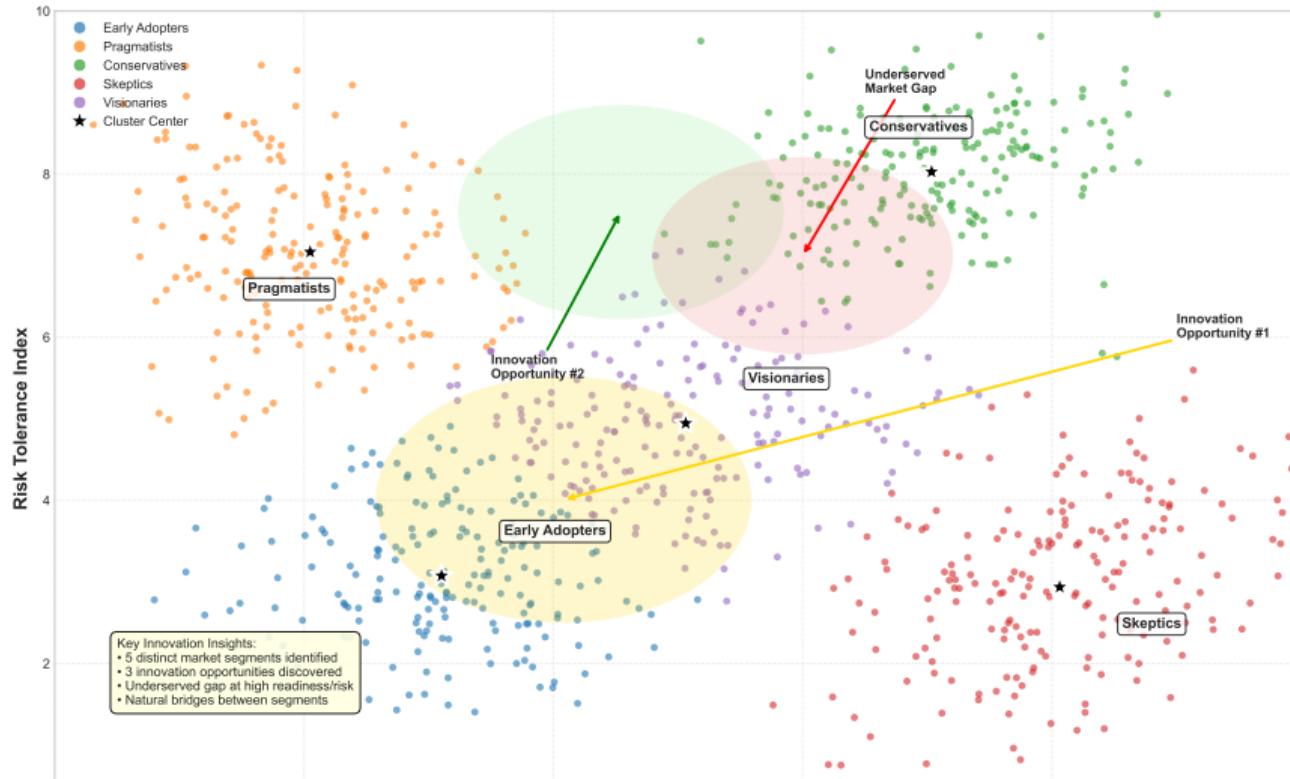
## Where Data Science Meets Human Creativity

Three powerful forces converge to amplify innovation impact

# From Data Points to Innovation Insights

Bridging the Technical-Human Gap

Innovation Pattern Discovery Through Clustering  
Revealing Hidden Market Opportunities



## PART 1

### Foundation & Context

What we'll explore:

- Why traditional design hits limits
- How ML amplifies human insight
- The dual pipeline approach
- Your learning journey ahead

Setting the stage for transformation

# Part 1: Learning Objectives

What You'll Learn in This Section

By the end of Part 1, you will be able to:

- **Understand** the limitations of traditional innovation approaches
- **Recognize** how ML enhances human creativity
- **Explain** the dual pipeline methodology
- **Navigate** the 10-week learning journey
- **Identify** Week 1's role in the overall course

Success Criteria

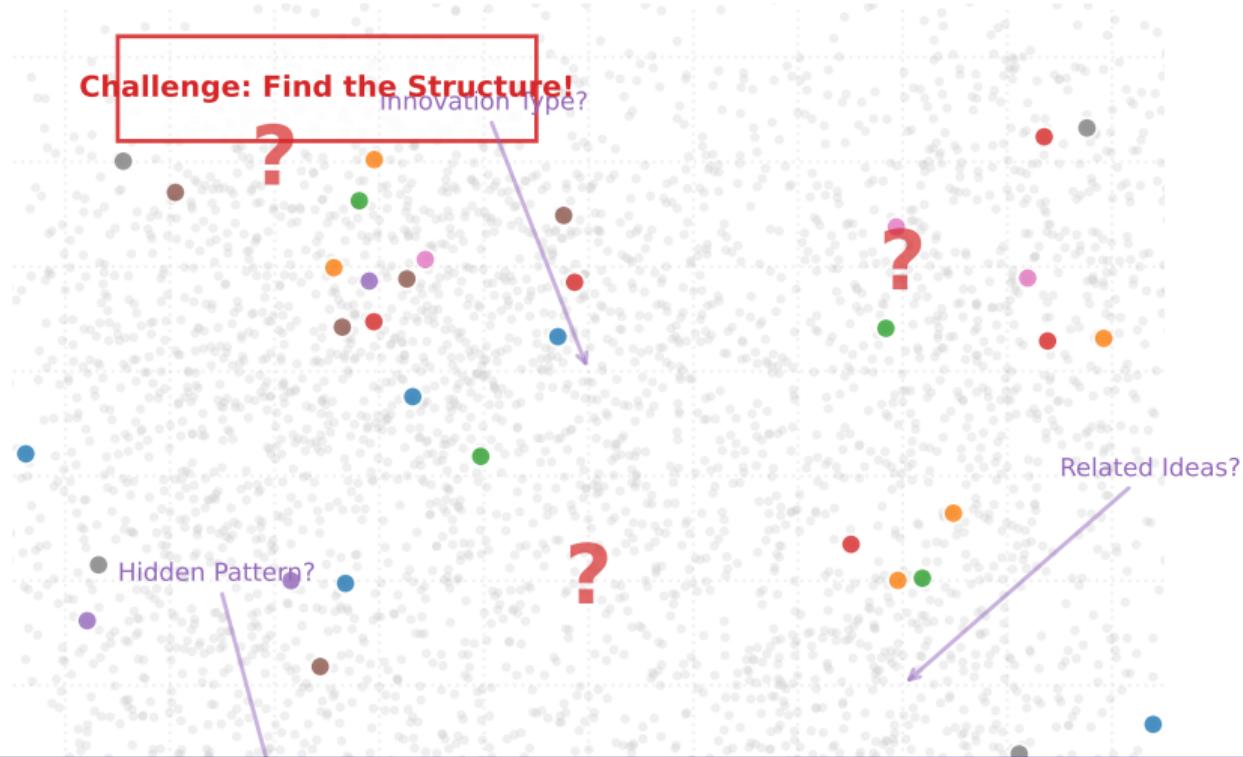
- Can articulate 3+ traditional design limitations
- Can describe ML's value proposition
- Can map ML pipeline to design pipeline
- Understand clustering's role in innovation

# PART 1

## Foundation & Context

Understanding the Innovation Challenge

## 5000+ Innovation Ideas: Where Do We Start?



# The Hidden Complexity

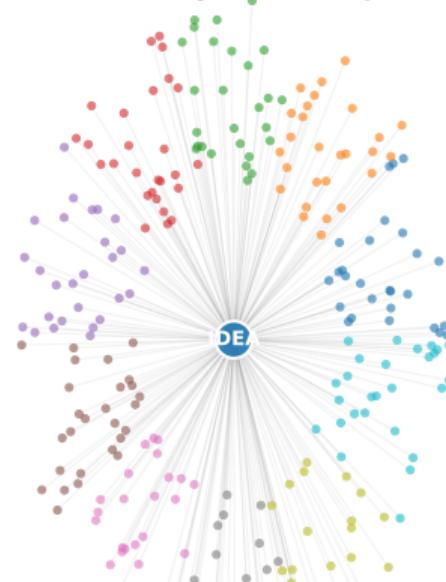
Each Innovation Depends on Hundreds of Features

## The Hidden Complexity: Each Innovation Depends on Hundreds of Features

What looks simple...



...is actually THIS complex!



Feature Dimensions

Technical	~27
Market	~27
User	~27
Financial	~27
Legal	~27
Environmental	~27
Social	~27
Competitive	~27
Supply Chain	~27
Cloud	~27

# The Innovation Challenge

Why Traditional Design Needs AI Enhancement

## Traditional Design Limits

- **Scale:** Can analyze 50 ideas, not 50,000
- **Speed:** Months for insights
- **Bias:** Designer's perspective dominates
- **Patterns:** Miss hidden connections
- **Iteration:** Slow feedback loops

## AI-Enhanced Innovation

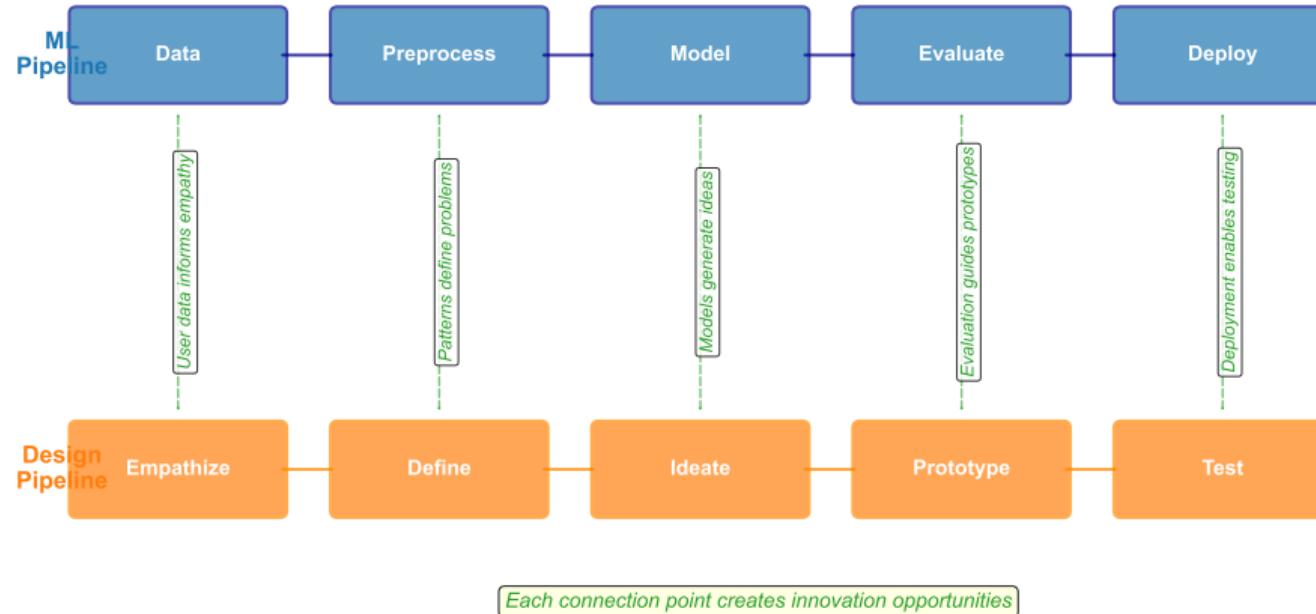
- **Scale:** Analyze millions of data points
- **Speed:** Real-time insights
- **Objectivity:** Data-driven discovery
- **Patterns:** Find non-obvious relationships
- **Iteration:** Continuous learning

**The Promise: 100x more insights, 10x faster innovation**

# The Dual Pipeline

Where ML Meets Design Thinking

## The Convergence: ML Meets Design Thinking



# The Dual Pipeline (Continued)

Understanding Both Worlds

## ML Pipeline

**Data → Preprocess → Model → Evaluate → Deploy**

- Collect innovation data
- Clean and transform
- Train algorithms
- Validate accuracy
- Scale to production

## Design Pipeline

**Empathize → Define → Ideate → Prototype → Test**

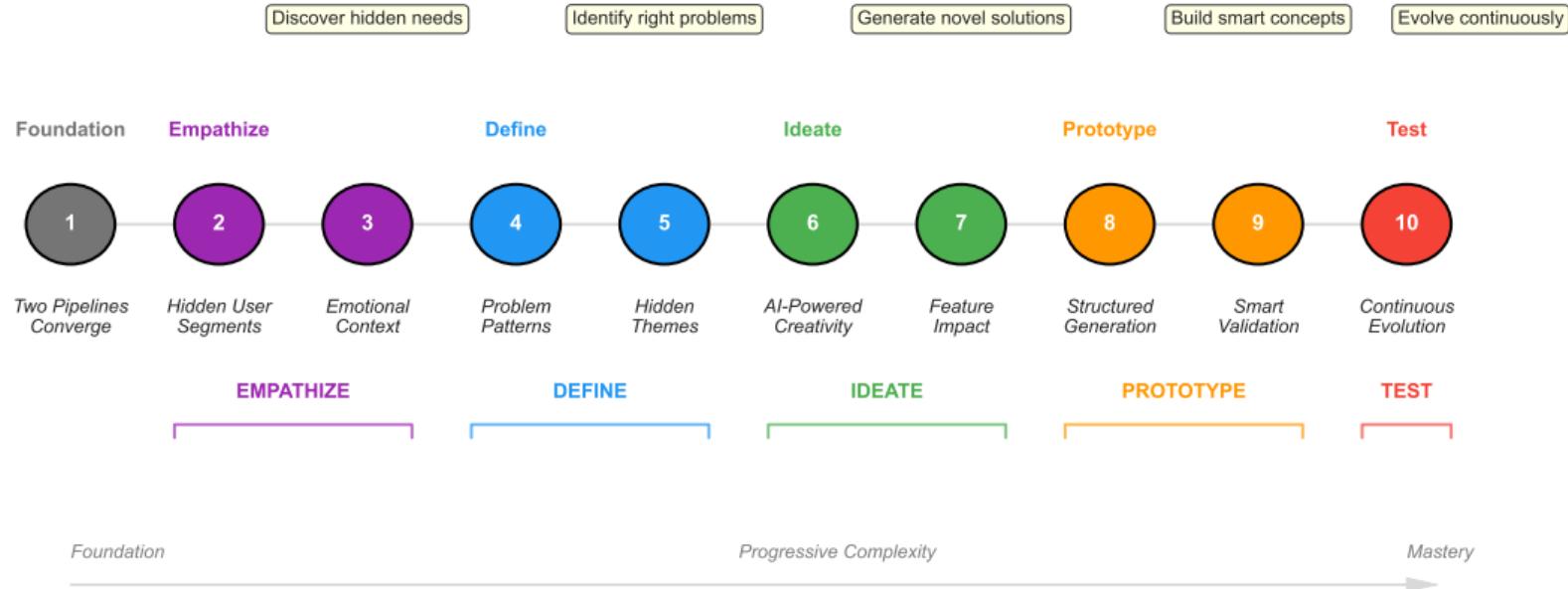
- Understand innovation needs
- Frame problems
- Generate solutions
- Build concepts
- Validate innovation impact

**Integration = Innovation at Scale**

# Your Innovation Journey

10 Weeks to Understanding AI-Powered Design

## 10-Week Innovation Journey



# Your Innovation Journey (Continued)

What You'll Learn in Each Stage

## Innovation Stages

### Discover (Weeks 1-2)

Find hidden innovation opportunities

### Define (Weeks 3-4)

Identify the right problems to solve

### Ideate (Weeks 5-6)

Generate novel solutions with AI

## Building Innovation Skills

### Prototype (Weeks 7-8)

Build smart, adaptive concepts

### Test (Weeks 9-10)

Evolve through continuous learning

### This Week:

Clustering for Innovation Pattern Discovery

# Week 1: Clustering for Innovation

From Scattered Ideas to Innovation Patterns

## What We'll Learn:

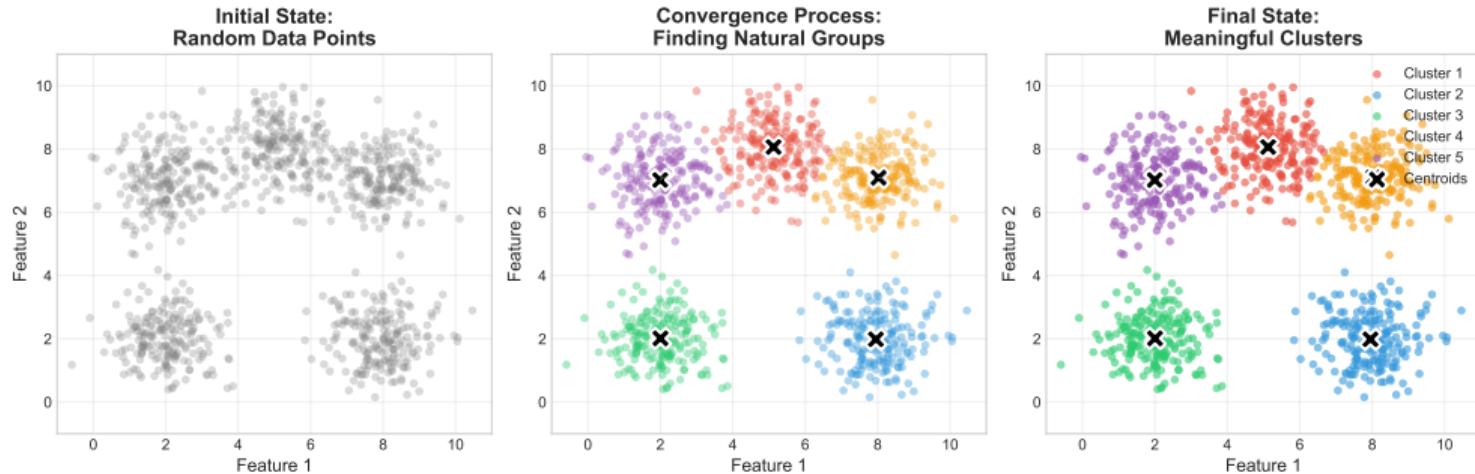
- How clustering reveals innovation categories
- K-means algorithm fundamentals
- Finding the optimal number of clusters
- Quality metrics for validation
- Advanced clustering techniques

## Design Applications:

- Create innovation archetypes
- Map innovation evolution paths
- Identify opportunities systematically
- Prioritize design efforts
- Scale analysis to thousands of ideas

**Goal: Transform scattered ideas into innovation patterns**

## The Convergence Flow: From Chaos to Clarity



## The Convergence Flow: Order from Chaos

*Watch 5000 innovation ideas self-organize into meaningful patterns*

# Check Your Understanding - Part 1

Quick Knowledge Check

Progress: 1/3

## True or False?

- ① Clustering requires labeled data (F)
- ② ML can process more data than humans (T)
- ③ Design thinking has 5 stages (T)
- ④ Clustering finds hidden patterns (T)

## Can You Explain?

- What is the dual pipeline approach?
- Why combine ML with design thinking?
- What problem does clustering solve?

**Ready for Part 2? Let's dive into the technical details!**

*Next: Clustering algorithms, evaluation metrics, and implementation*

# Now Let's Get Technical

From Understanding the Problem to Finding Solutions

## We've seen the challenge:

Thousands of innovation ideas with hidden connections

## Traditional approach:

Manual segmentation based on demographics

## The ML solution:

Let the data reveal its own natural groups

Enter: Clustering Algorithms

## PART 2

### Technical Core

What we'll learn:

- K-means clustering algorithm
- Finding optimal K with elbow method
- Distance metrics and quality measures
- Advanced techniques (DBSCAN, Hierarchical)
- Feature importance analysis

Learning the basics step by step

## Part 2: Learning Objectives

Technical Skills You'll Develop

By the end of Part 2, you will understand:

- **How** K-means clustering works
- **What** the elbow method shows us
- **Why** we measure distances
- **How to check** if clusters are good
- **Differences** between algorithms
- **When to use** each method

Practical Skills

- Use K-means step by step
- Understand quality scores
- Pick the right algorithm
- Adjust settings properly
- Work with different patterns
- Prepare data for analysis

# PART 2

## Technical Core

Machine Learning Algorithms & Implementation

# The Innovation Classification Problem

5000 Ideas - How Do They Connect?

## The Pain

### Current Reality:

- One-size-fits-all solutions
- Generic innovation categories
- Missed opportunities
- Unhappy edge cases

### The Cost:

- Most innovations get misclassified
- Features with low adoption rates
- Inefficient resource allocation

## The Question

### What if we could...

- Find natural innovation clusters?
- Discover innovation patterns?
- Innovate at scale?
- Identify opportunity gaps?

We can!

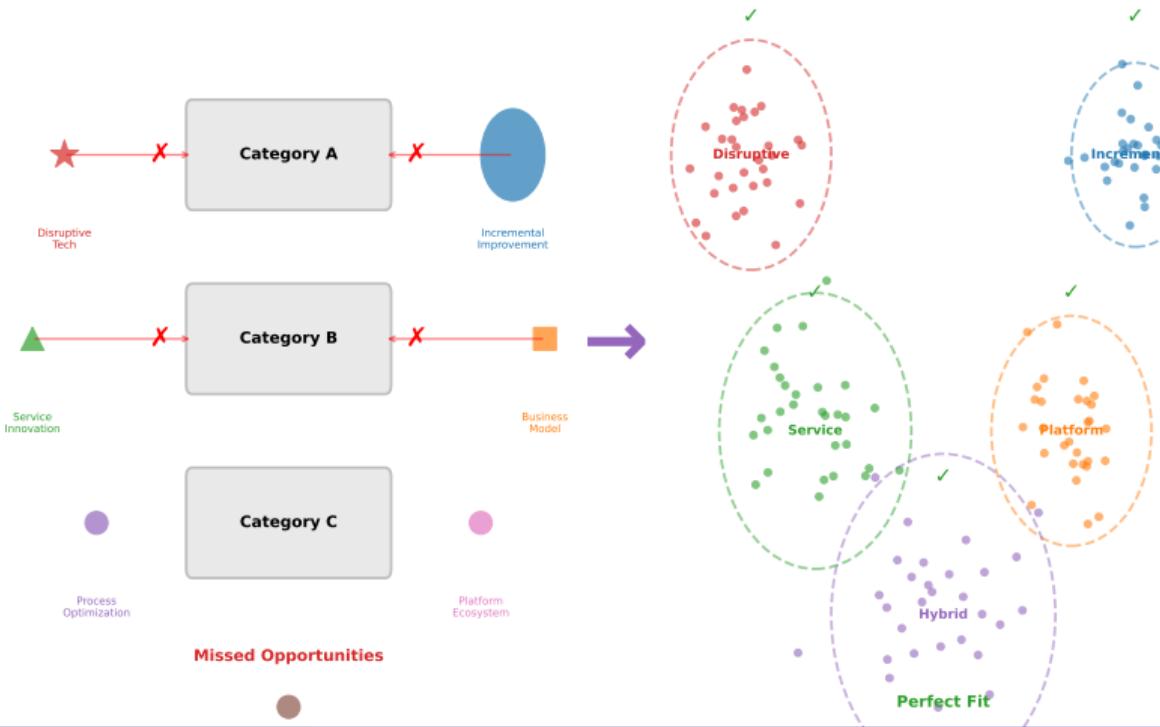
Solution: Clustering

# Current Reality: The Problem

Why One-Size-Fits-All Doesn't Work

## The Problem with One-Size-Fits-All Innovation Categories

Current Reality: Generic Categories      ML Solution: Natural Clusters



# Innovation Archetypes

Common Patterns We'll Discover

## Core Innovation Types

### Disruptive Innovation

Completely new approaches that reshape markets

### Incremental Innovation

Step-by-step improvements to existing solutions

### Service Innovation

New ways to deliver value to customers

## Emerging Patterns

### Business Model Innovation

New ways to create and capture value

### Process Innovation

Better ways to produce and deliver

### Platform Innovation

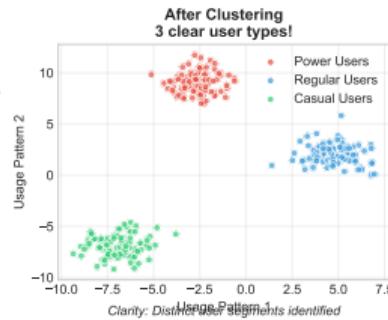
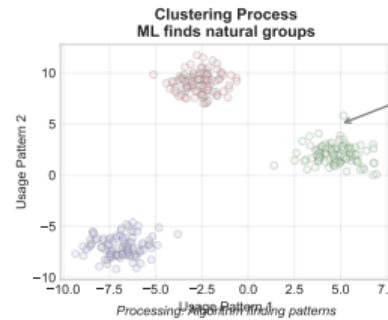
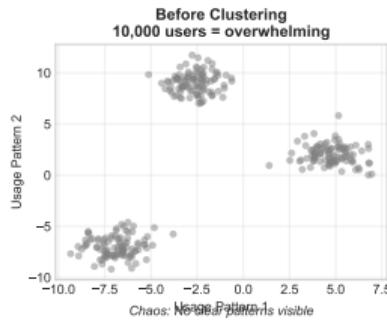
Creating ecosystems for innovation

Clustering will reveal which type each idea belongs to

# What is Clustering?

Like Organizing a Messy Room - Finding Things That Belong Together

From Chaos to Clarity Through Clustering



## Clustering Finds:

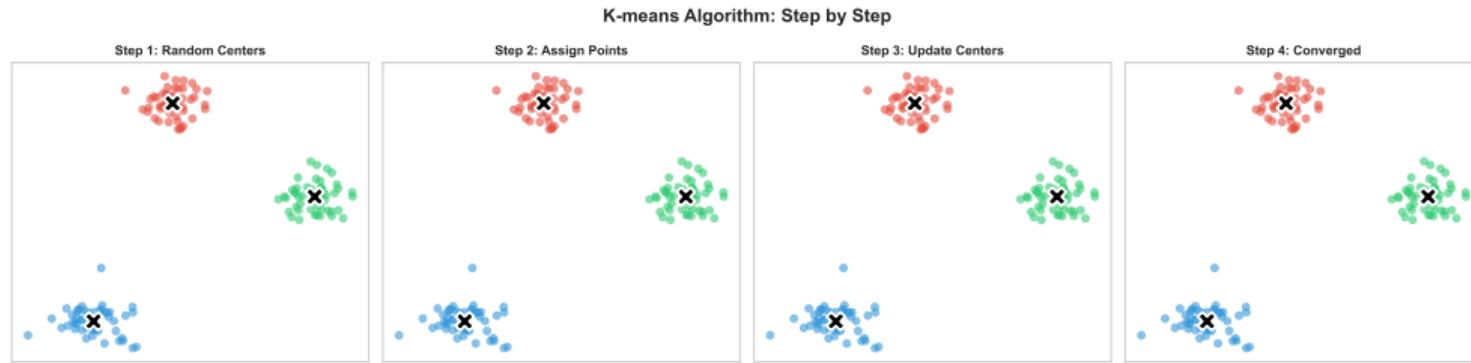
- Natural groupings
- Similar approaches
- Hidden patterns
- Innovation relationships

### Key Insight:

Things that look similar often belong in the same group  
*(Just like organizing books by topic on a shelf)*

# K-Means: The Basic Clustering Method (Part 1)

Initial Setup - Like Choosing City Centers



## Step 1: Choose K

- Decide number of clusters
- Based on domain knowledge
- Or use elbow method

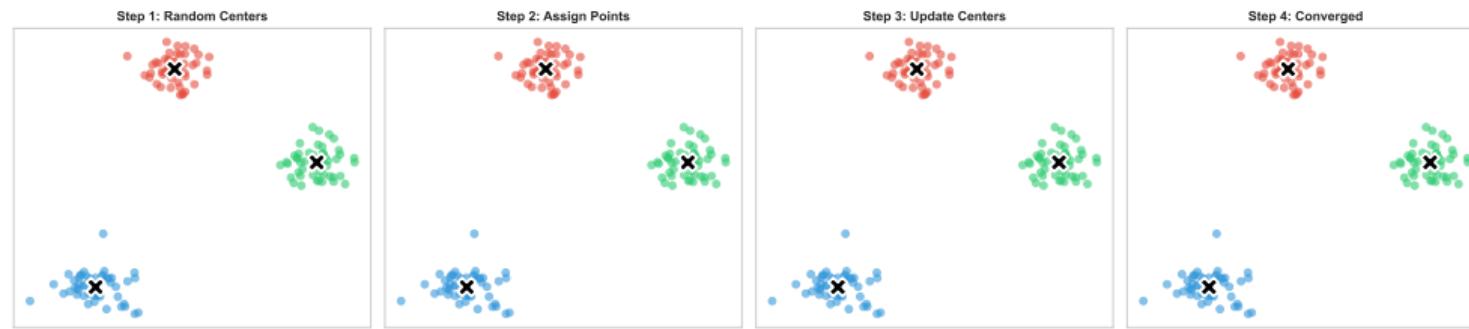
## Step 2: Initialize Centroids

- Place K random points
- These are initial cluster centers
- Random but spread out

# K-Means: The Basic Clustering Method (Part 2)

## Iteration Process - Finding Natural Groups

K-means Algorithm: Step by Step



### Step 3: Assign

- Distance to centroids
- Assign to nearest
- Forms clusters

### Step 4: Update

- Calculate mean
- Move centroids
- Better positions

### Step 5: Converge

- Repeat 3-4
- Stop when stable
- 5-10 iterations

# The Goldilocks Problem

Too Few vs. Too Many Groups

Too Few ( $K=2$ )

## Oversimplification

- Mixed segments
- Lost nuance
- Generic solutions

Just Right ( $K$ )

## Optimal Balance

- Clear segments
- Actionable insights
- Manageable complexity

Too Many ( $K$ )

## Analysis Paralysis

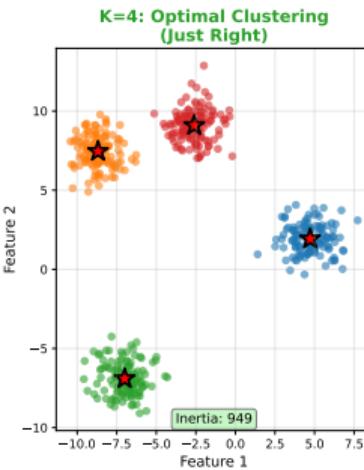
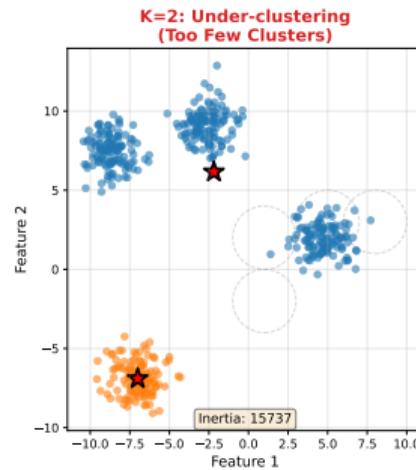
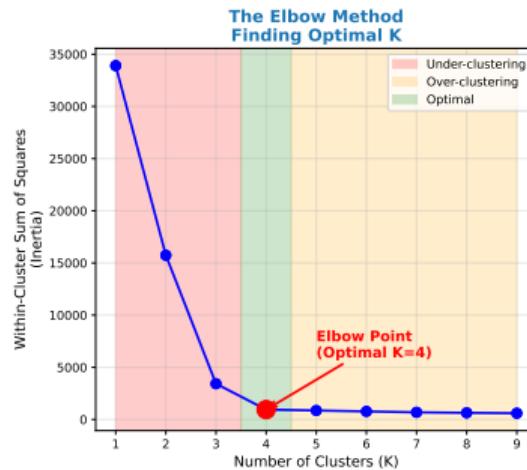
- Overfitting
- Tiny segments
- Impossible to act on

How do we find the sweet spot?

# The Elbow Method

How Many Groups Should We Have? (Like Goldilocks - Not Too Few, Not Too Many)

## The Elbow Method: Finding the Right Number of Clusters



### Finding the Elbow:

- Plot inertia vs K
- Look for the “elbow”
- Balance simplicity vs accuracy

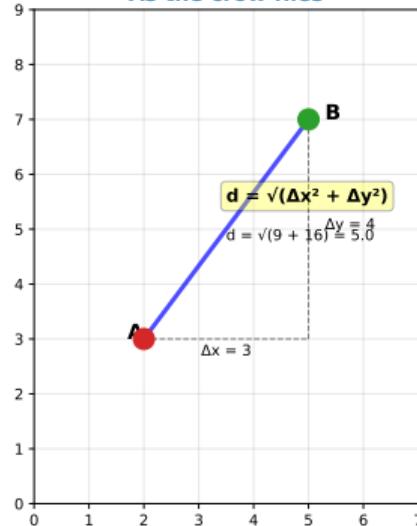
Optimal K = 5  
Best trade-off point

# Distance Metrics

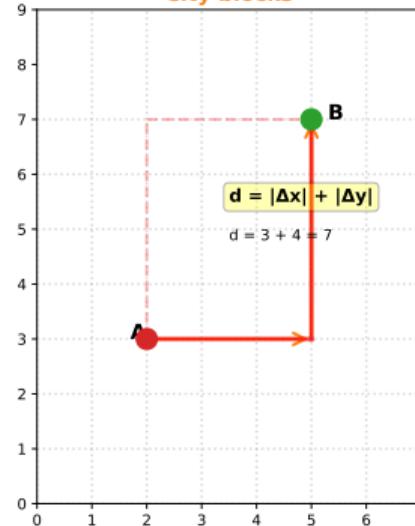
Different Ways to Measure "How Close" Things Are

## Distance Metrics: How We Measure "Closeness"

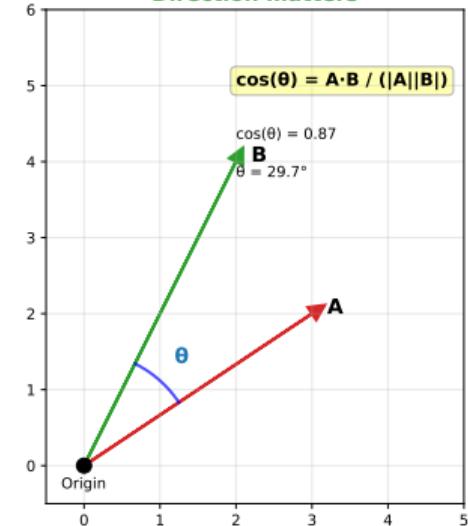
Euclidean Distance  
"As the crow flies"



Manhattan Distance  
"City blocks"



Cosine Similarity  
"Direction matters"

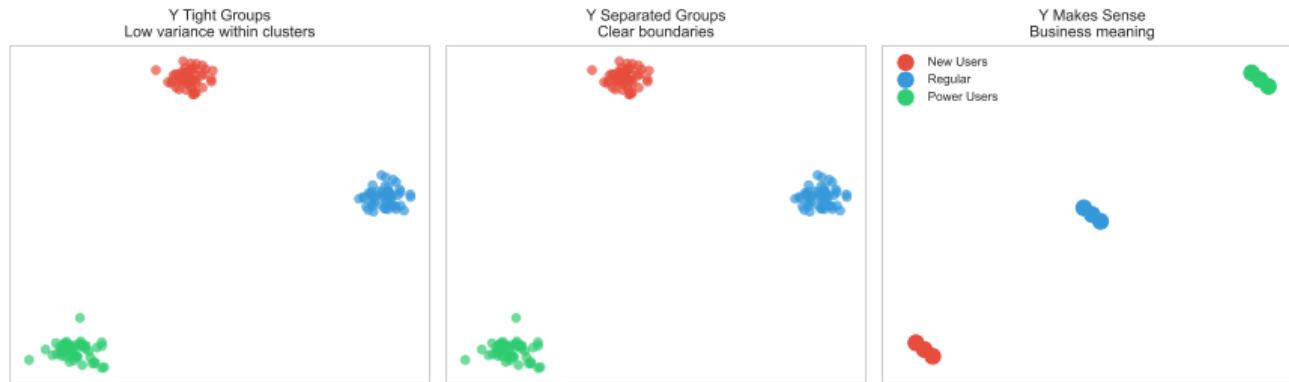


Each metric reveals different patterns in your data

# Cluster Quality Metrics

Are Our Groups Any Good? (Like Checking Your Work)

## Three Checks for Good Clusters



## Silhouette Score:

- Ranges from -1 to +1
- Higher = better separation
- Our score: **0.73**

**0.73 = Strong clusters!**

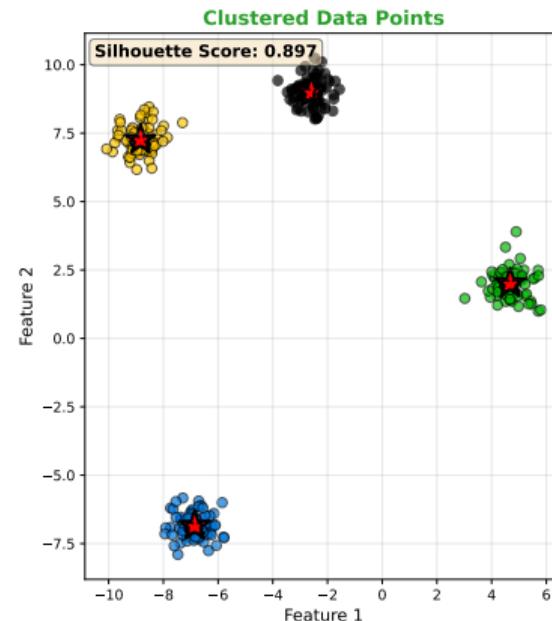
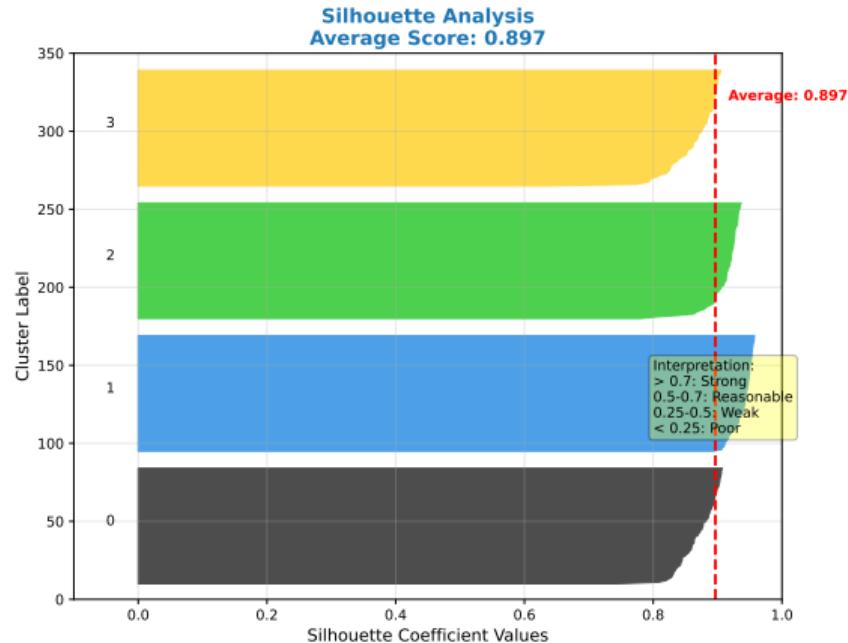
## What it measures:

- Within-cluster cohesion
- Between-cluster separation
- Overall cluster validity

# Evaluation Metric 1: Silhouette Score

Measuring Cluster Cohesion and Separation

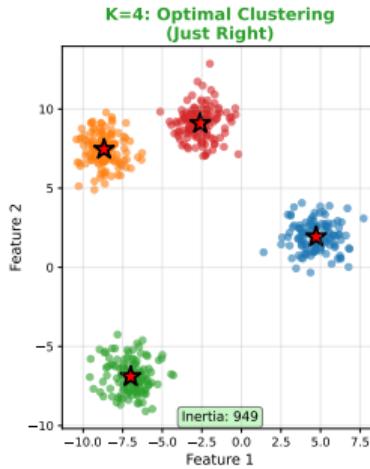
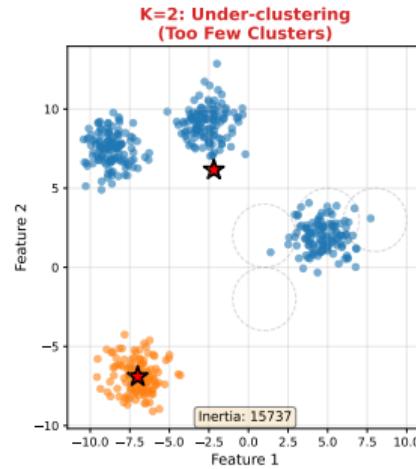
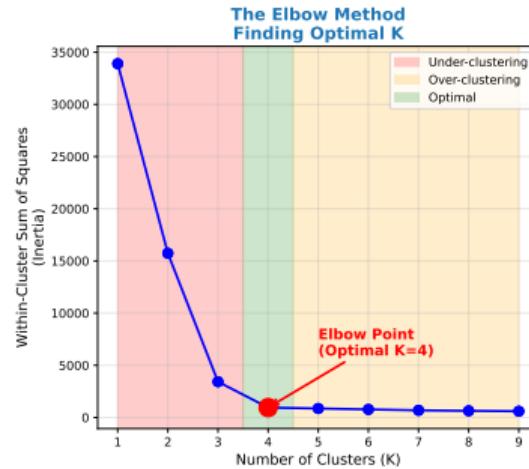
## Silhouette Score: Measuring Cluster Cohesion and Separation



# Evaluation Metric 2: Elbow Method

Finding the Right Number of Clusters

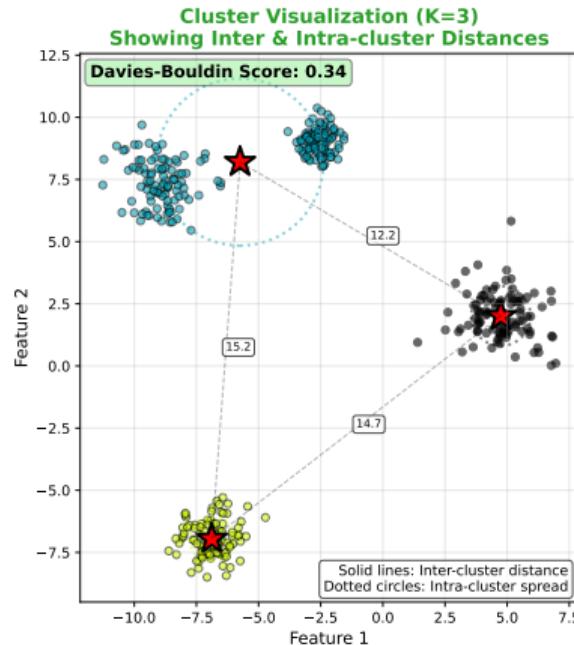
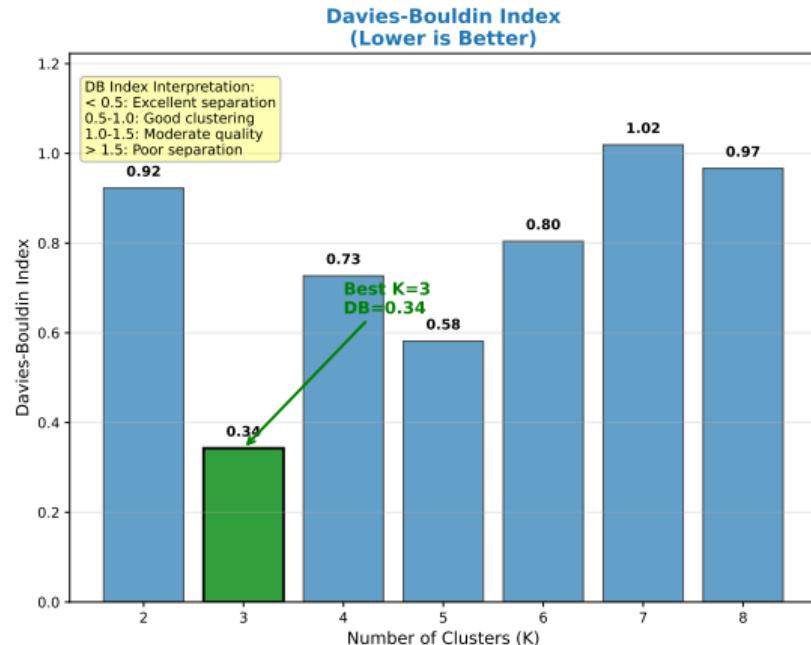
## The Elbow Method: Finding the Right Number of Clusters



# Evaluation Metric 3: Davies-Bouldin Index

Balancing Within and Between Cluster Distances

## Davies-Bouldin Index: Balancing Cluster Separation and Cohesion



# When Circles Don't Work

Real Innovation Clusters Have Complex Shapes

## K-Means Assumes Spherical Clusters

But what about:

- Innovations connected through technology stacks
- Domain-specific innovation clusters
- Evolution patterns (incremental, disruptive)
- Outliers and noise points

## K-Means Forces Round Pegs into Round Holes

**Solution: Density-Based Clustering**

# DBSCAN: Finding Groups Naturally

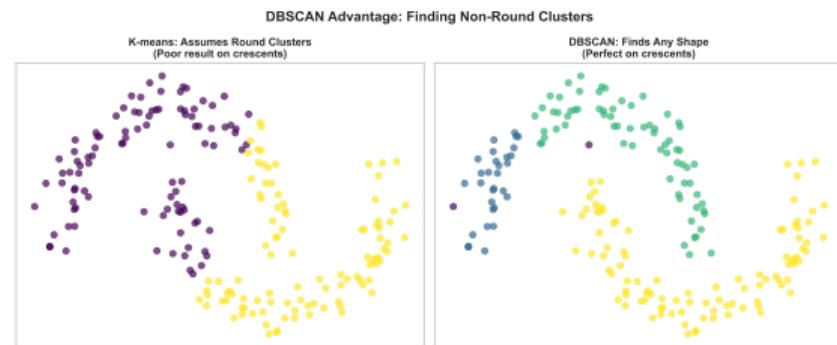
Like Finding Groups of People at a Party - Where Are the Crowds?

## DBSCAN Advantages:

- No need to specify K (*finds groups automatically*)
- Finds arbitrary shapes (*not just circles*)
- Identifies outliers (*points that don't belong*)
- Handles noise well (*robust to random points*)

## Perfect for:

- Non-spherical patterns
- Varying densities
- Outlier detection
- Exploratory analysis



# DBSCAN: Understanding Parameters

Two Simple Settings Control Everything

## Epsilon (Distance)

### **What it does:**

Sets the maximum distance to consider points as neighbors

### **Think of it as:**

How far can points be apart and still be friends?

**Too small:** Many tiny clusters

**Too large:** Everything merges

## MinPts (Density)

### **What it does:**

Minimum neighbors needed to form a dense region

### **Think of it as:**

How many friends make a group?

**Too small:** Noise becomes clusters

**Too large:** Small clusters vanish

**Rule of thumb:  $\text{MinPts} = 2 \times \text{dimensions}$**

# Clustering Algorithm Comparison

## Technical Characteristics at a Glance

Algorithm	Speed	Shape	Outliers	Params	Best For
K-Means	Fast $O(nkt)$	Spherical clusters	Sensitive	K only	Quick segments
DBSCAN	Medium $O(n \log n)$	Any shape	Robust (detects)	eps, MinPts	Complex shapes
Hierarchical	Slow $O(n^2)$	Any shape	Moderate	Distance threshold	Multi-level analysis
GMM	Medium $O(nkt)$	Elliptical clusters	Moderate	K, covariance	Overlapping groups

Each algorithm has its strengths - choose wisely!

# When to Use Each Algorithm

## Practical Decision Guide

### K-Means

#### Perfect when:

- Speed is critical
- Clusters are roughly equal size
- You know K in advance
- Data has spherical patterns

### Hierarchical

#### Perfect when:

- Need multiple granularities
- Want to visualize relationships
- Small to medium datasets
- Exploring data structure

### DBSCAN

#### Perfect when:

- Clusters have irregular shapes
- Outliers need identification
- Density varies across data
- You don't know K

### GMM

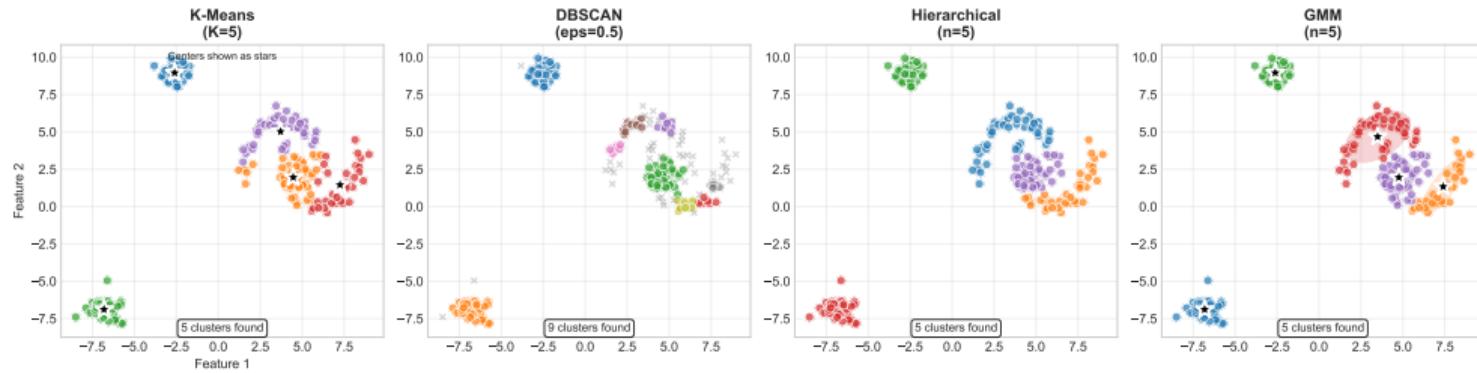
#### Perfect when:

- Groups overlap
- Need probability scores
- Elliptical cluster shapes
- Soft assignments needed

# Algorithm Visual Comparison

Same Data, Different Approaches

Clustering Algorithms Visual Comparison  
Same Data, Different Approaches



K-Means (K=5)

- Fast and scalable
- Spherical clusters
- Fixed K required
- Sensitive to outliers

Best for: Quick segmentation with known cluster count

DBSCAN (eps=0.5)

- Finds arbitrary shapes
- Identifies outliers
- No K needed
- Sensitive to parameters

Best for: Anomaly detection and irregular patterns

Hierarchical (n=5)

- Dendrogram output
- No K needed initially
- Interpretable
- Computationally expensive

Best for: Taxonomies and exploring relationships

GMM (n=5)

- Soft assignments
- Elliptical clusters
- Probabilistic
- Assumes Gaussian distribution

Best for: Overlapping groups and uncertainty modeling

Complexity:  $O(nkt)$

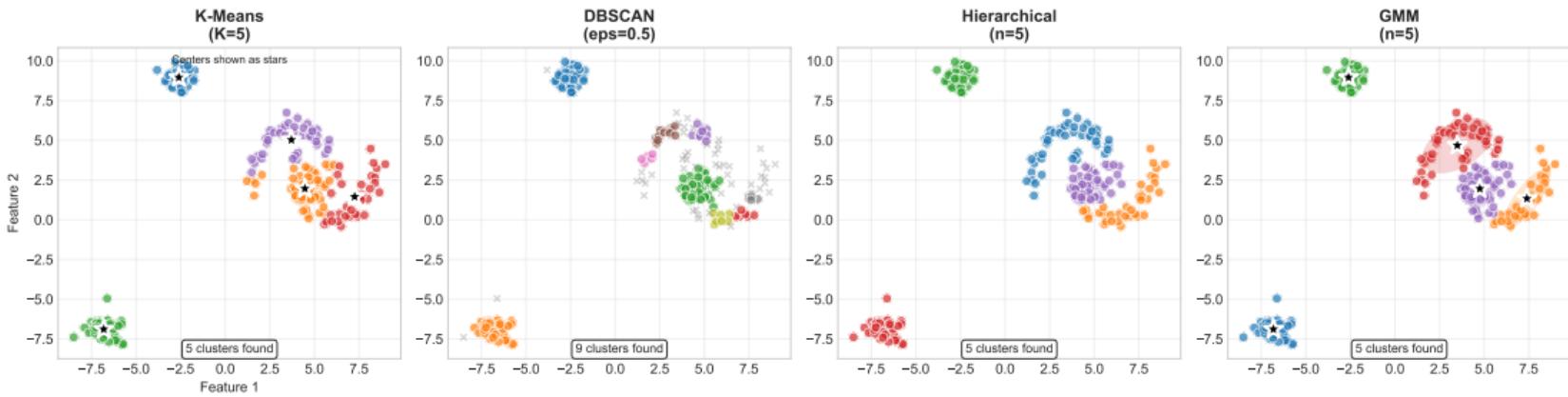
Complexity:  $O(n \log n)$

Complexity:  $O(n^2)$

Complexity:  $O(nd)$

Dataset: Mix of 3 Gaussian blobs and 2 moon-shaped clusters (250 points total)

## Clustering Algorithms Visual Comparison Same Data, Different Approaches



**K-Means (K=5)**

- Fast and scalable
- Spherical clusters
- Fixed K required
- Sensitive to outliers

Best for: Quick segmentation with known cluster count

**DBSCAN (eps=0.5)**

- Finds arbitrary shapes
- Identifies outliers
- No K needed
- Sensitive to parameters

Best for: Anomaly detection and irregular patterns

**Hierarchical (n=5)**

- Dendrogram output
- No K needed initially
- Interpretable
- Computationally expensive

Best for: Taxonomies and exploring relationships

**GMM (n=5)**

- Soft assignments
- Elliptical clusters
- Probabilistic
- Assumes Gaussian distribution

Best for: Overlapping groups and uncertainty modeling

Complexity:  $O(nkl)$

Complexity:  $O(n \log n)$

Complexity:  $O(n^2)$

Complexity:  $O(nkl)$

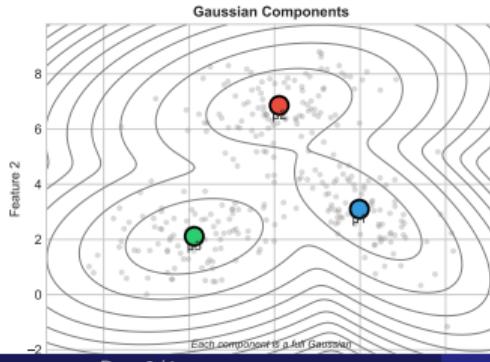
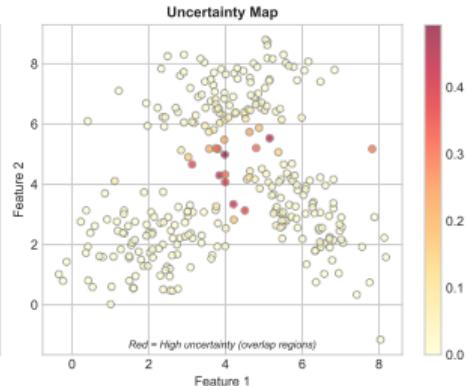
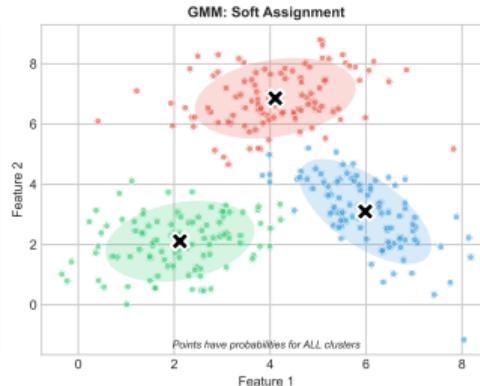
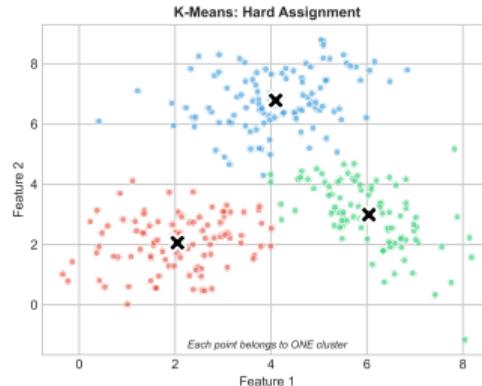
Dataset: Mix of 3 Gaussian blobs and 2 moon-shaped clusters (250 points total)

# Gaussian Mixture Models (GMM)

Soft Clustering for Overlapping Innovation Categories

## Gaussian Mixture Models (GMM): Soft Clustering for Innovation

Beyond Hard Boundaries: Probabilistic Innovation Classification



### GMM vs K-means

- GMM Advantages:
- Soft assignments (probabilities)
  - Captures cluster shape (elliptical)
  - Handles overlapping clusters
  - Provides uncertainty estimates
  - Models data generation process

- K-means Advantages:
- Faster computation
  - Simpler interpretation
  - Less parameters
  - More stable results
  - Works well for spherical clusters

- When to use GMM:
- Overlapping innovation categories
  - Need probability scores
  - Non-spherical clusters
  - Uncertainty quantification needed

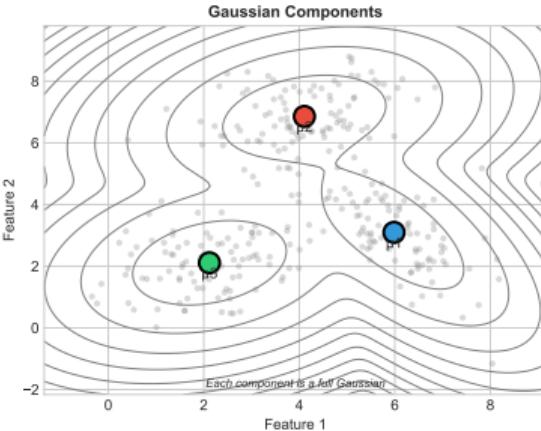
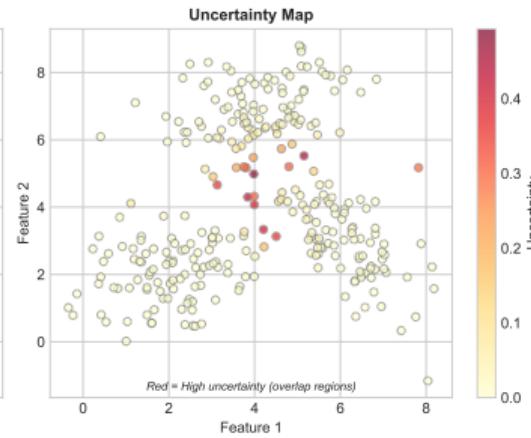
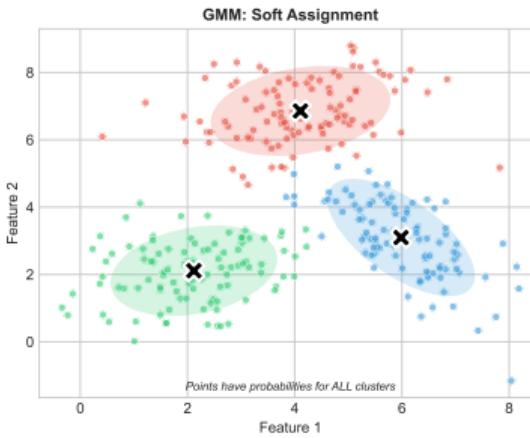
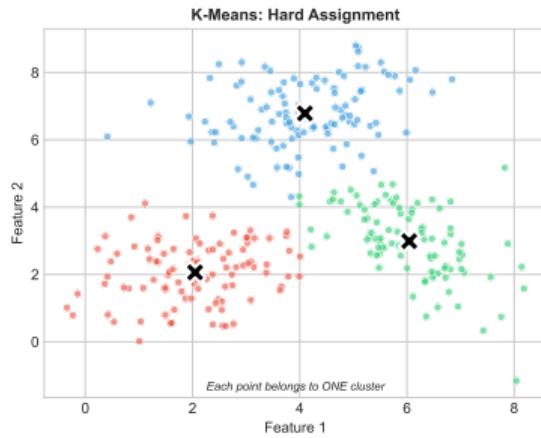
### Innovation Category Probabilities

Innovation	Tech	Service	Social
AI Assistant	0.85	0.10	0.05
Sharing Platform	0.30	0.45	0.25
Green Energy	0.60	0.15	0.25
Digital Health	0.40	0.50	0.10

GMM provides probability of belonging to each category

# Gaussian Mixture Models (GMM): Soft Clustering for Innovation

Beyond Hard Boundaries: Probabilistic Innovation Classification



## GMM vs K-means

- GMM Advantages:**
- Soft assignments (probabilities)
  - Captures cluster shape (elliptical)
  - Handles overlapping clusters
  - Provides uncertainty estimates
  - Models data generation process

- K-means Advantages:**
- Faster computation
  - Simpler interpretation
  - Less parameters
  - More stable results
  - Works well for spherical clusters

- When to use GMM:**
- Overlapping innovation categories
  - Need probability scores
  - Non-spherical clusters
  - Uncertainty quantification needed

## Innovation Category Probabilities

Innovation	Tech	Service	Social
AI Assistant	0.85	0.10	0.05
Sharing Platform	0.30	0.45	0.25
Green Energy	0.60	0.15	0.25
Digital Health	0.40	0.50	0.10

GMM provides probability of belonging to each category

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \Sigma_k)$$

# The Granularity Challenge

When You Need Multiple Levels of Detail

## Fixed K Gives One View

But real relationships are hierarchical:

- Organization: Company → Department → Team → Individual
- Geography: Country → Region → City → Neighborhood
- Products: Category → Subcategory → Brand → SKU
- Innovations: All → Categories → Sub-types → Specific solutions

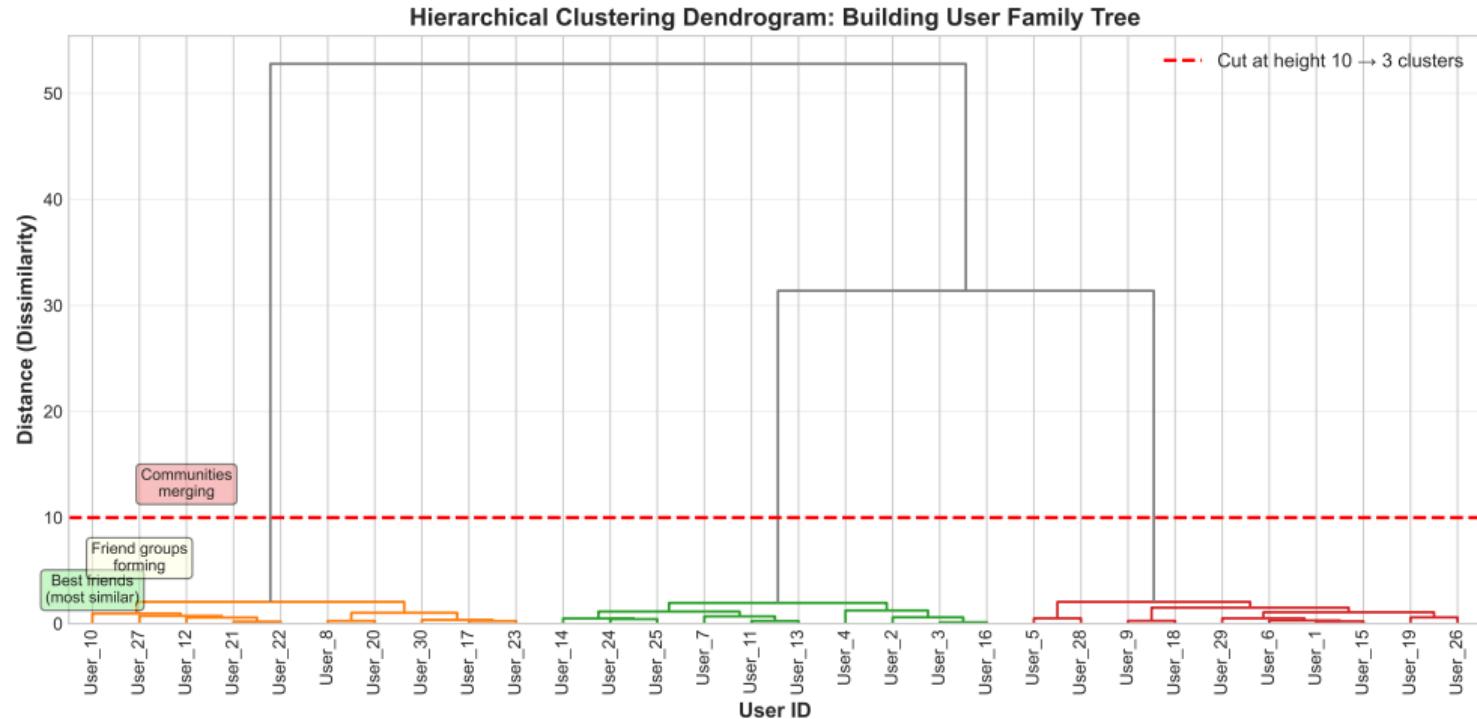
**K-means: Pick 5 groups and that's it**

**What if we need flexibility?**

Solution: See the full hierarchy, cut where needed

# Hierarchical Clustering

Building a Tree of Relationships



## Dendrogram Benefits:

- Shows cluster hierarchy

Part 0/4

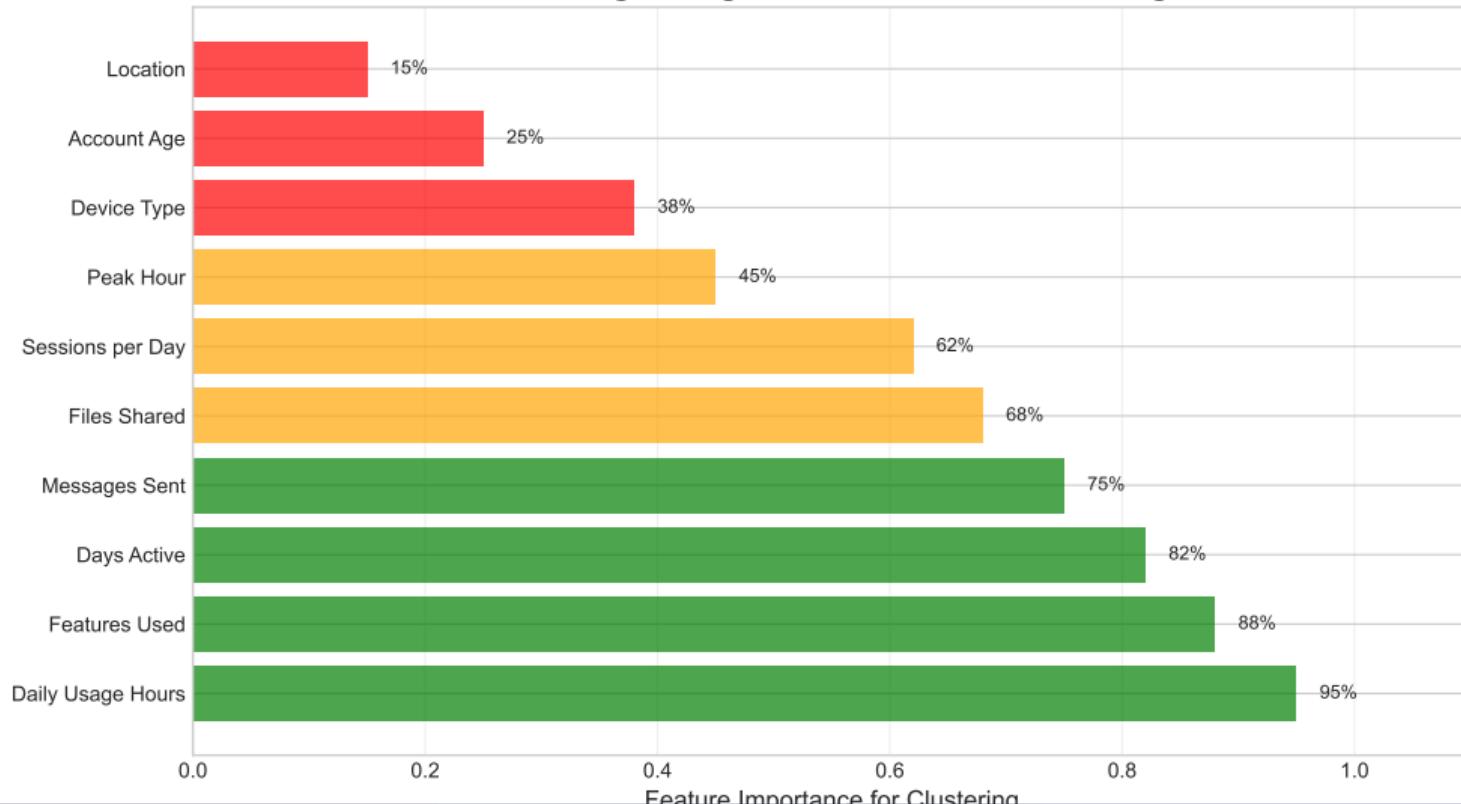
## Cut the tree at any level:

- High cut = Few clusters

# What Drives the Clusters?

## Feature Importance Analysis

Choosing the Right Features for User Clustering

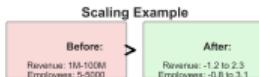
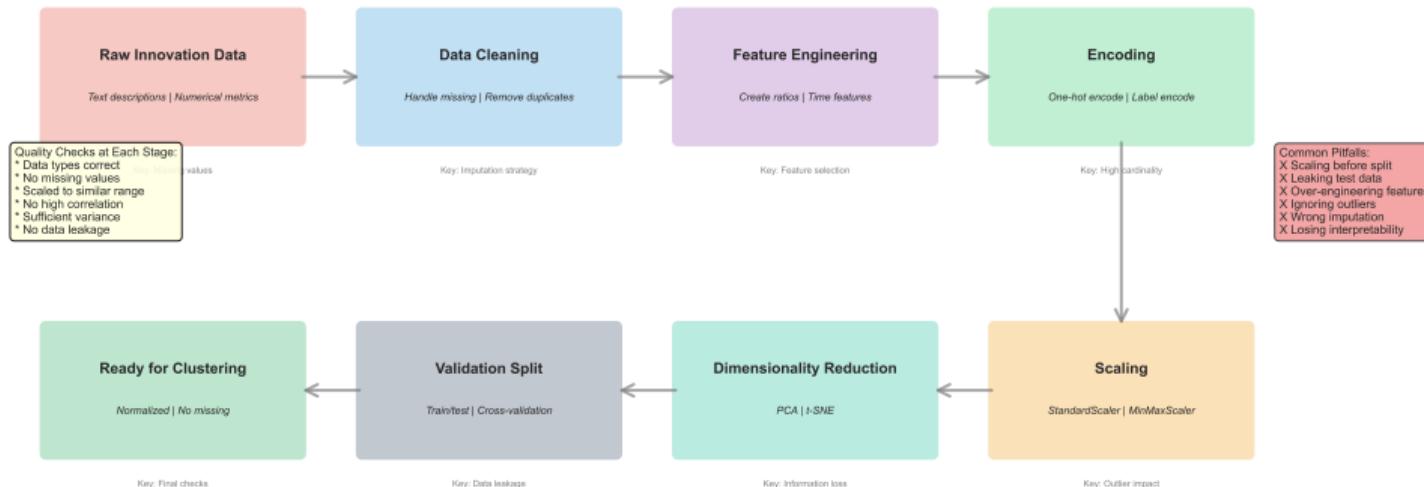


# Data Preprocessing Pipeline

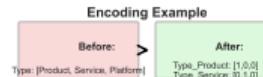
From Raw Data to Clustering-Ready Features

## Data Preprocessing Pipeline for Innovation Clustering

From Raw Data to Clustering-Ready Features



(StandardScaler)



(One-Hot Encoding)



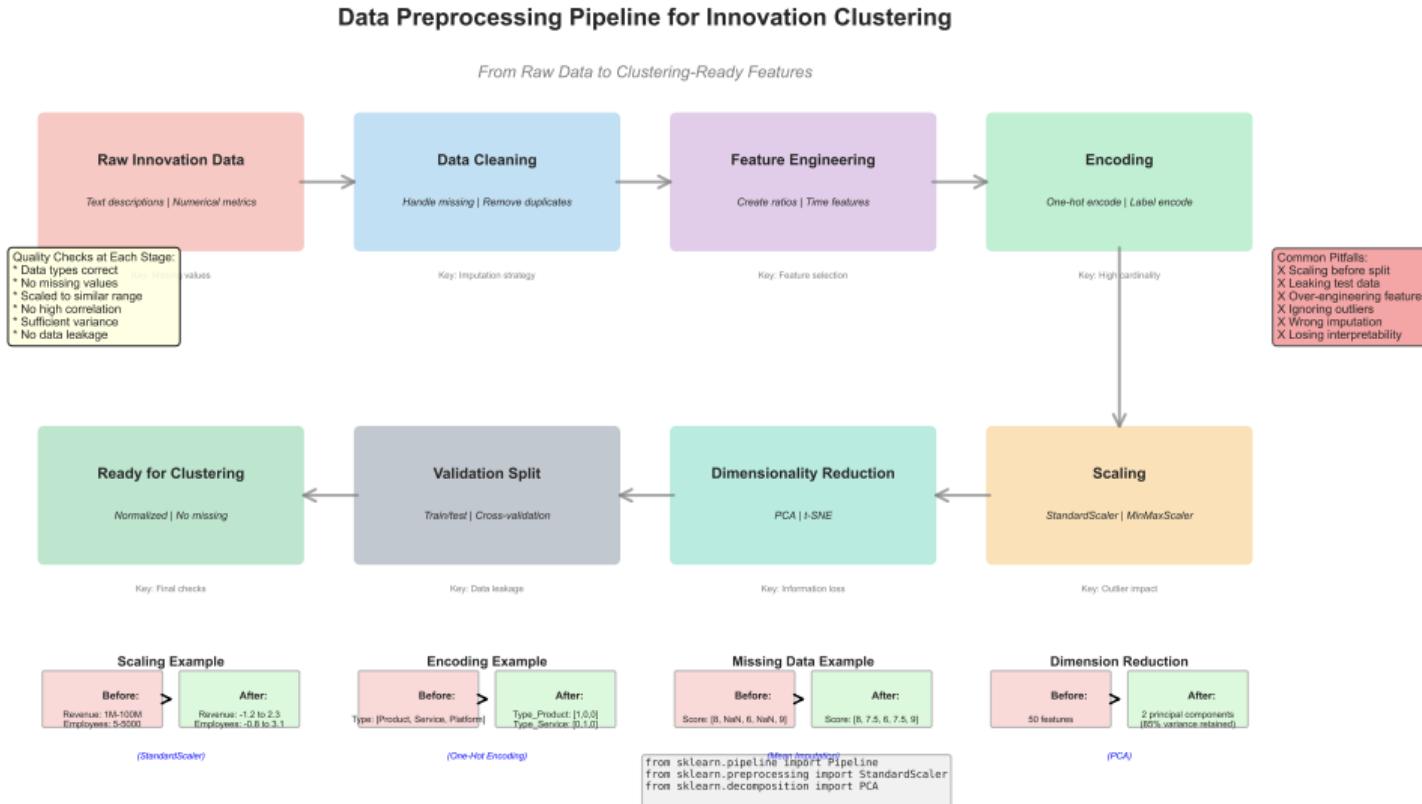
(Mean Imputation)  
from sklearn.pipeline import Pipeline  
from sklearn.preprocessing import StandardScaler  
from sklearn.decomposition import PCA



(PCA)

# Data Preprocessing Pipeline - Example

## Real Innovation Data Transformation



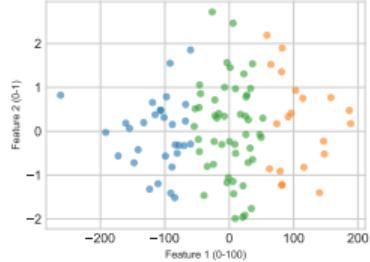
# Common Mistakes & Troubleshooting

Learn from These Pitfalls

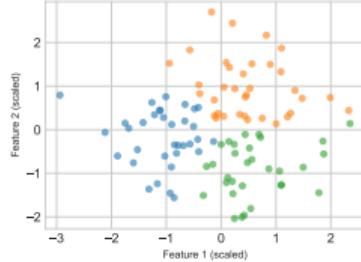
## Common Clustering Mistakes & Troubleshooting Guide

Learn from These Mistakes to Master Clustering

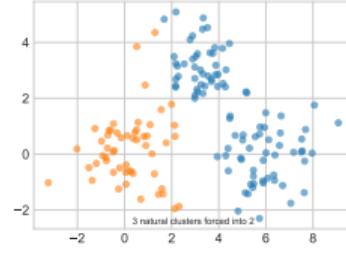
Visual Examples of Common Mistakes:



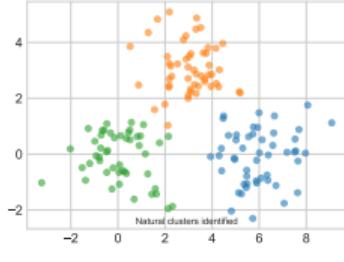
CORRECT: Scaled Features



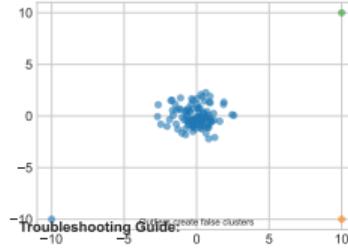
MISTAKE: K=2 (too few)



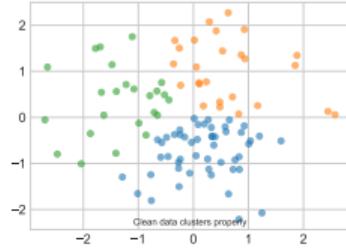
CORRECT: K=3 (optimal)



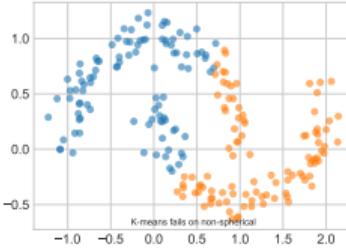
MISTAKE: Outliers distort



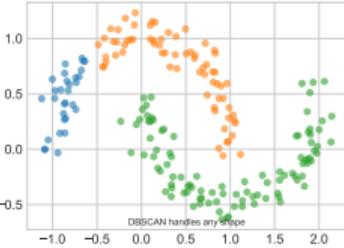
CORRECT: Outliers removed



MISTAKE: K-means on moons



CORRECT: DBSCAN for shapes



### Problem

### Symptoms

### Solution

### Prevention

#### Poor separation

##### GOLDEN RULES:

1. Always scale your features
2. Visualize before clustering
3. Try multiple algorithms
4. Validate with domain knowledge
5. Check cluster stability

#### Low silhouette score

Different runs = diff

Takes too long to co

- WARNING SIGNS:**
- Silhouette < 0.3
  - Clusters change each run
  - Single point clusters
  - All points in one cluster

#### Silhouette score

Set random\_state, increase n\_int

Reduce features, subsample data

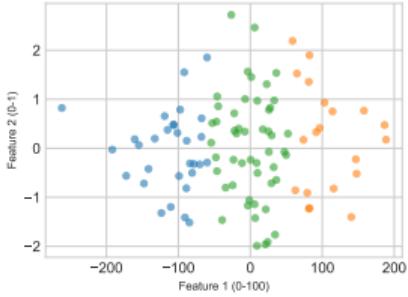
#### Use elbow method

- SUCCESS INDICATORS:**
- Silhouette > 0.5
  - Stable across runs
  - Balanced cluster sizes
  - Makes business sense

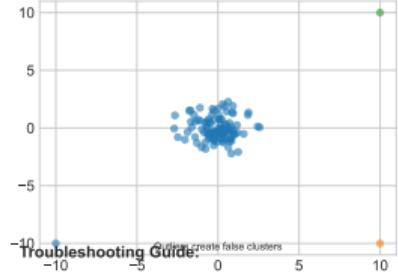
# Common Clustering Mistakes & Troubleshooting Guide

Learn from These Mistakes to Master Clustering

## Visual Examples of Common Mistakes:



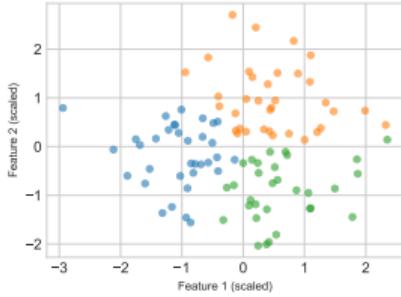
**MISTAKE:** Outliers distort



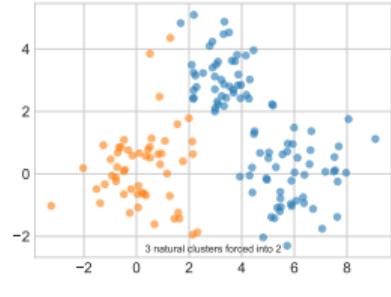
## Troubleshooting Guide:

Outliers create false clusters

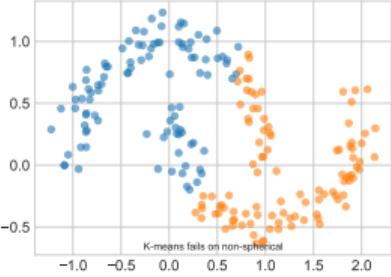
**CORRECT:** Scaled Features



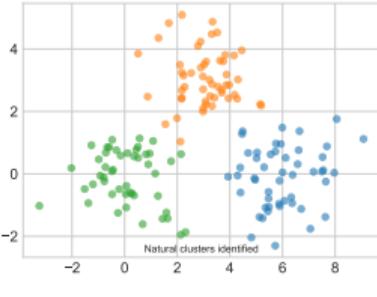
**MISTAKE:** K=2 (too few)



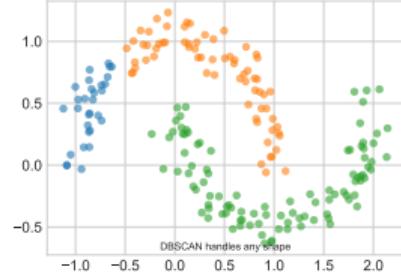
**MISTAKE:** K-means on moons



**CORRECT:** K=3 (optimal)



**CORRECT:** DBSCAN for shapes



## Problem

## Symptoms

## Solution

## Prevention

### Poor separation

- GOLDEN RULES:

  - Always scale your features
  - Visualize before clustering
  - Try multiple algorithms
  - Validate with domain knowledge
  - Check cluster stability

### Unbalanced clusters

Low silhouette score

Different runs = different results

Takes too long to converge

One huge, others tiny

- WARNING SIGNS:**
- \* Silhouette < 0.3
  - \* Clusters change each run
  - \* Single point clusters
  - \* All points in one cluster

Try different K or algorithm

Set random\_state, increase n\_init

Reduce features, subsample data

Check for outliers, try GMM

Use elbow method

Batch K-means

Success indicators:

- \* Silhouette > 0.5
- \* Stable across runs
- \* Balanced cluster sizes
- \* Makes business sense

### No convergence

Algorithm doesn't stop

Increase max\_iter, check data

Inspect data distribution

Normalize features first

# Parameter Tuning Guidelines

## Recommended Ranges and Best Practices

### Clustering Parameter Tuning Guidelines

Recommended Ranges, Methods, and Best Practices

K-Means			
Parameter	Range	Default	Tuning Method
n_clusters (K)	2-10	3-5	Elbow/Silhouette
init	['k-means++', 'random']	Always k-means++	
n_init	10-100	10	More for stability
max_iter	100-1000	300	Increase if no convergence
tol	1e-6 to 1e-2	1e-4	Smaller for precision

#### Tuning Strategies

##### Grid Search

Pros: Exhaustive, Reproducible, Simple

Cons: Slow, Curse of dimensionality

Use when: Small parameter space

##### Random Search

Pros: Faster, Better for many params, Parallelizable

Cons: May miss optimum, Not reproducible

Use when: Large parameter space

##### Bayesian Opt

Pros: Efficient, Learns from history, Fewer iterations

Cons: Complex, Overhead for simple problems

Use when: Expensive evaluations

DBSCAN			
Parameter	Range	Default	Tuning Method
eps	0.01-2.0	0.5	k-distance plot
min_samples	3-20	2*dim	Domain knowledge
metric	['euclidean', 'manhattan']	Data dependent	
algorithm	['auto', 'ball_tree']	Auto is fine	
leaf_size	10-50	30	Memory vs speed

#### Validation Metrics

Metric	Range	Interpretation	Use For
Silhouette	[-1, 1]	Higher is better	General quality
Davies-Bouldin	(0, ∞)	Lower is better	Cluster separation
Calinski-Harabasz	(0, ∞)	Higher is better	Dense clusters
Inertia	(0, ∞)	Lower is better	K-means only
BIC/AIC	(-∞, ∞)	Lower is better	GMM selection

GMM			
Parameter	Range	Default	Tuning Method
n_components	2-10	3-5	BIC/AIC
covariance_type	['full', 'diag', 'spherical']	Start full, simplify	
max_iter	50-500	100	Monitor convergence
n_init	1-10	1	More for stability
init_params	['kmeans', 'random']	Kmeans faster	

#### Tuning Best Practices

1. Start with defaults, then tune

2. Use cross-validation when possible

3. Consider computational budget

4. Log all experiments

5. Visualize parameter effects

6. Use domain knowledge

7. Check stability across runs

8. Don't overfit to metrics

##### IMPORTANT:

No metric is perfect!

Always validate:

- Visual inspection

- Domain expertise

- Business goals

# Check Your Understanding - Part 2

Technical Concepts Review

Progress: 2/3

## Quick Quiz

① K in K-means stands for:

- Kernel
- Number of clusters
- Constant

② DBSCAN finds:

- Only circles
- Any shape clusters
- Exactly K groups

## Can You Calculate?

If Silhouette Score = 0.75:

- Is this good? Yes!
- Range is [-1, 1]
- Higher = better separation

## Remember:

- Elbow method finds optimal K
- Scale your data first!

Great job! Now let's apply these concepts!

Next: *Design integration, innovation patterns, and real-world applications*

# From Algorithms to Innovation Insights

What Does This Mean for Innovation Opportunities?

**We've learned the technical tools:**

Clustering, metrics, quality measures

**But clusters are just numbers...**

Until we connect them to innovation opportunities

**Let's transform data into innovation insights**

Each cluster represents innovation opportunities and patterns

## PART 3

### Innovation Pattern Analysis

What we'll create:

- Data-driven innovation archetypes
- Innovation pattern maps per category
- Cluster-specific journeys
- Opportunity heat maps
- Design priority matrices

Where ML reveals innovation patterns

## Part 3: Learning Objectives

Innovation Applications You'll Explore

By the end of Part 3, you will be able to:

- **Create** innovation archetypes
- **Map** innovation patterns
- **Design** opportunity matrices
- **Analyze** innovation lifecycles
- **Build** ecosystem maps
- **Prioritize** innovation efforts

Design Outcomes

- Innovation taxonomy framework
- Cluster-based strategies
- Data-driven prioritization
- Opportunity identification
- Pattern recognition skills
- Ecosystem understanding

# PART 3

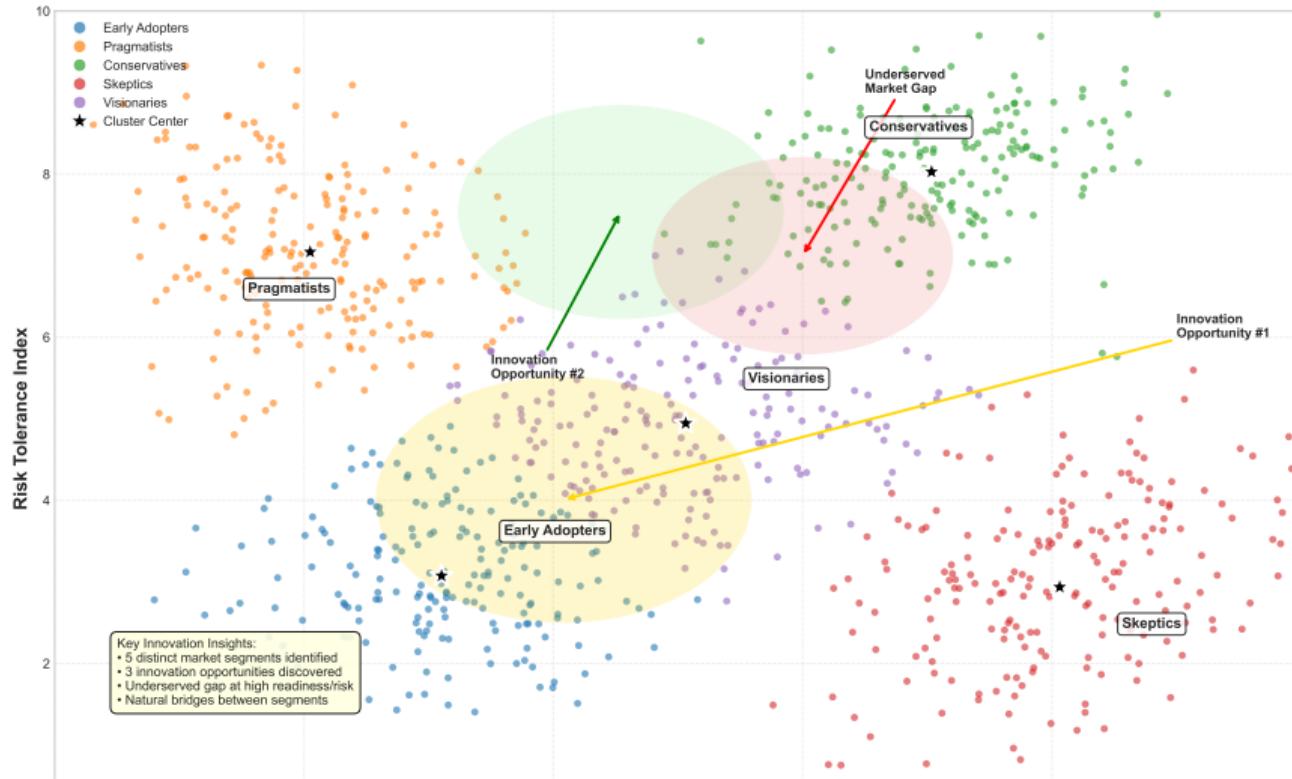
## Design Integration

Bridging Technology & Human Experience

# From Data Points to Innovation Insights

Bridging the Technical-Human Gap

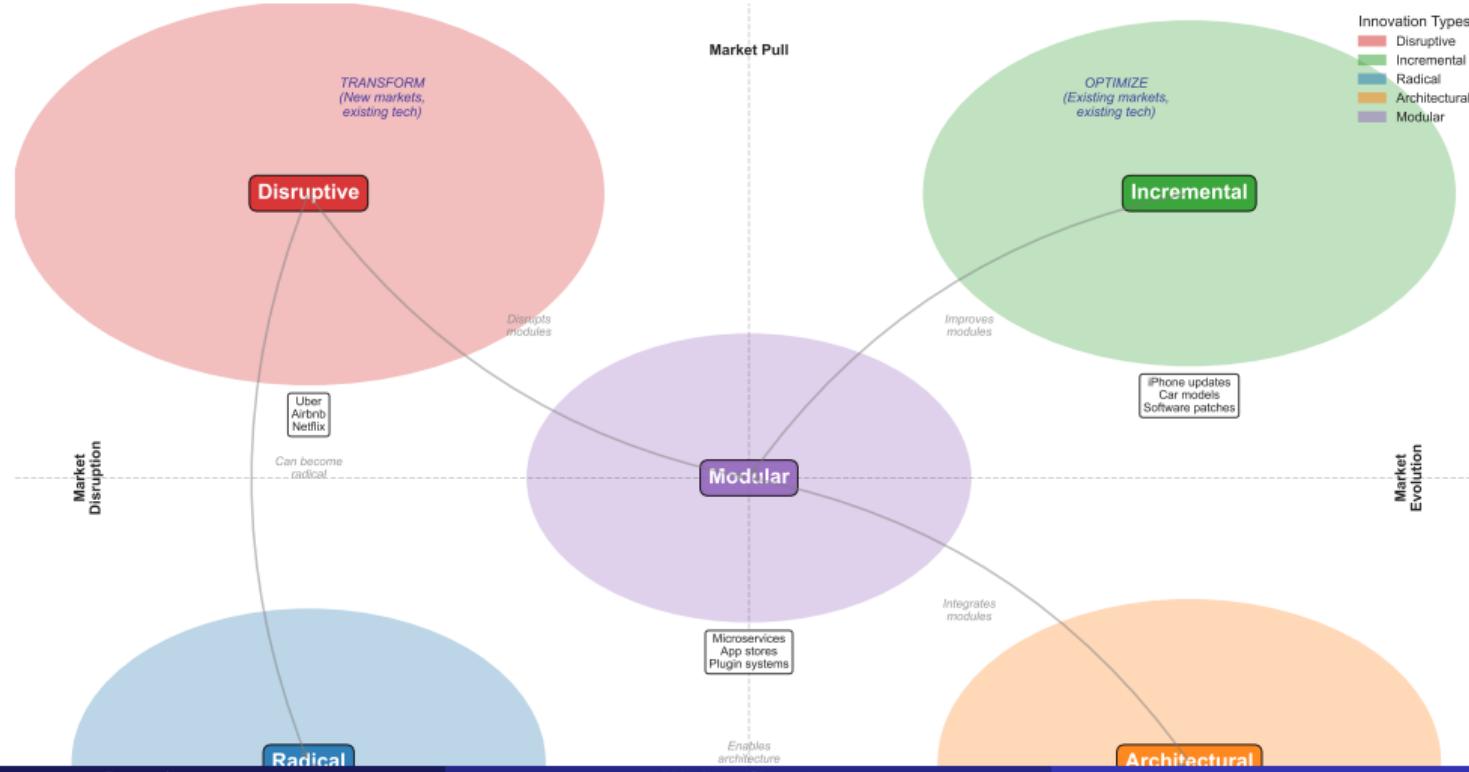
Innovation Pattern Discovery Through Clustering  
Revealing Hidden Market Opportunities



# AI-Generated Innovation Archetypes

Data-Driven Character Development

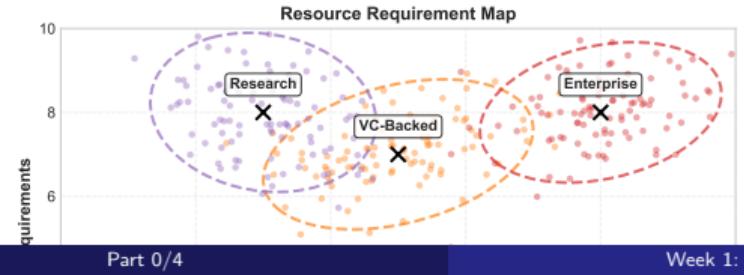
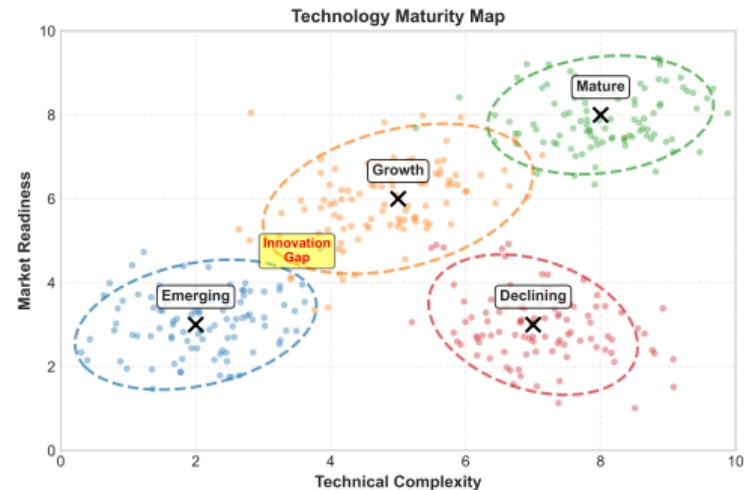
## Innovation Archetypes Discovery Five Distinct Patterns from Clustering Analysis



# Innovation Pattern Maps

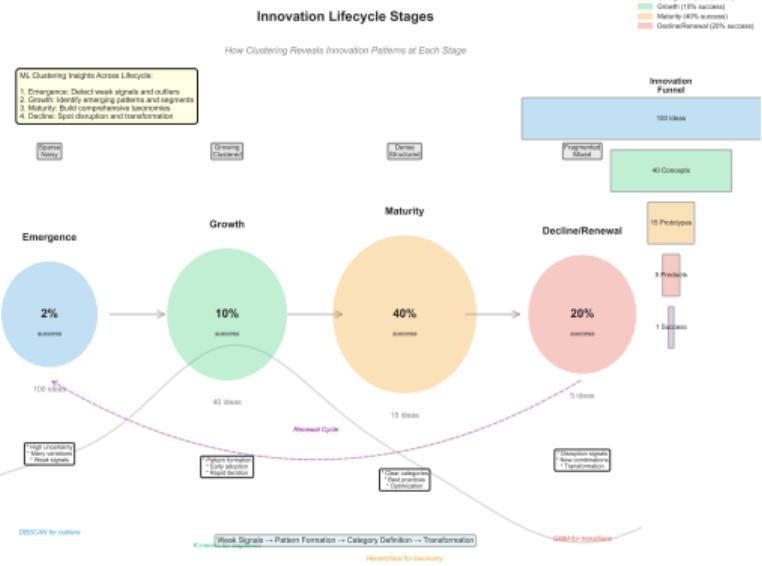
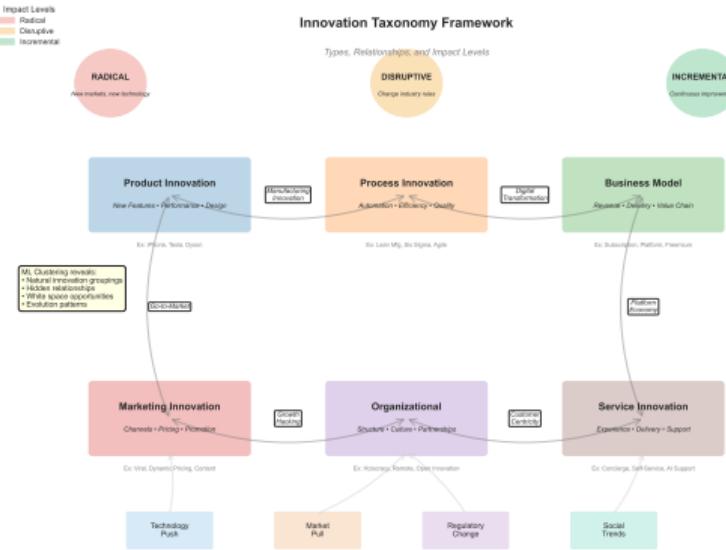
Cluster-Specific Insights

Innovation Pattern Maps  
Four Perspectives on Innovation Categories



# Innovation Framework

## Taxonomy and Lifecycle Stages



## Framework Levels:

- Types & relationships
- Impact measurements
- Strategic positioning

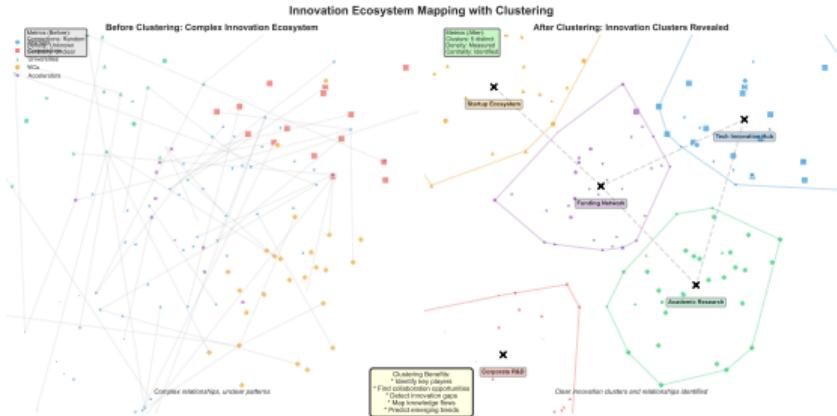
Part 0/4

## Lifecycle Stages:

- Ideation & discovery
- Development & testing
- Launch & scaling

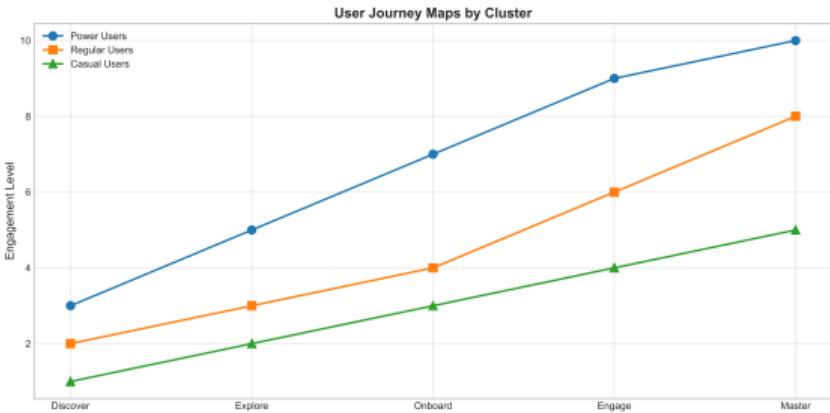
# Innovation Ecosystem & Journey Mapping

From Networks to Evolution Paths



## Ecosystem Elements:

- Network connections
- Stakeholder clusters
- Value flows

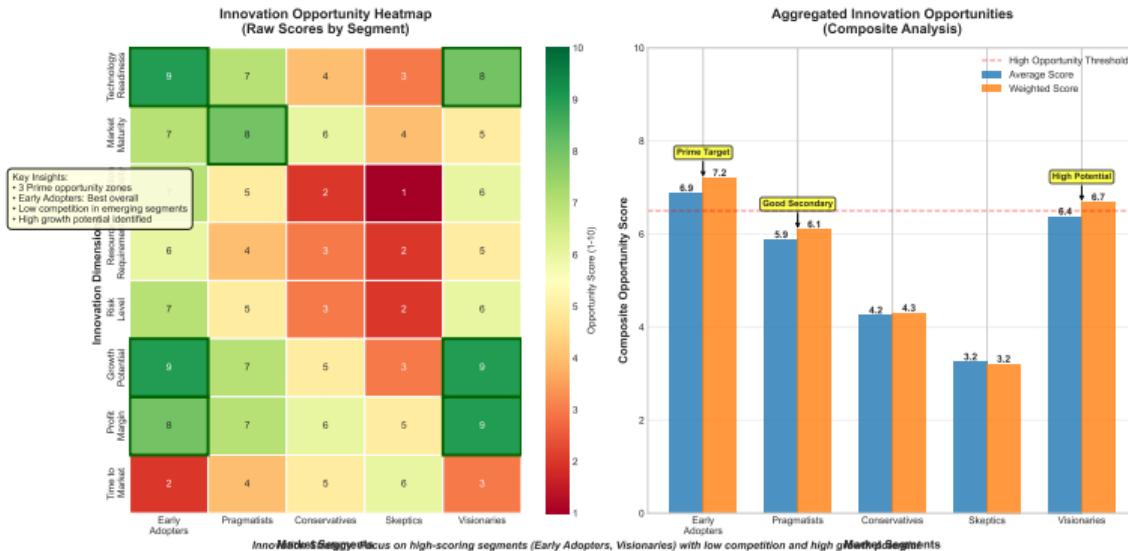


## Evolution Paths:

- Different speeds
- Varying trajectories
- Unique milestones

# Innovation Opportunities by Cluster

Where Each Category Has Potential



## Key Findings:

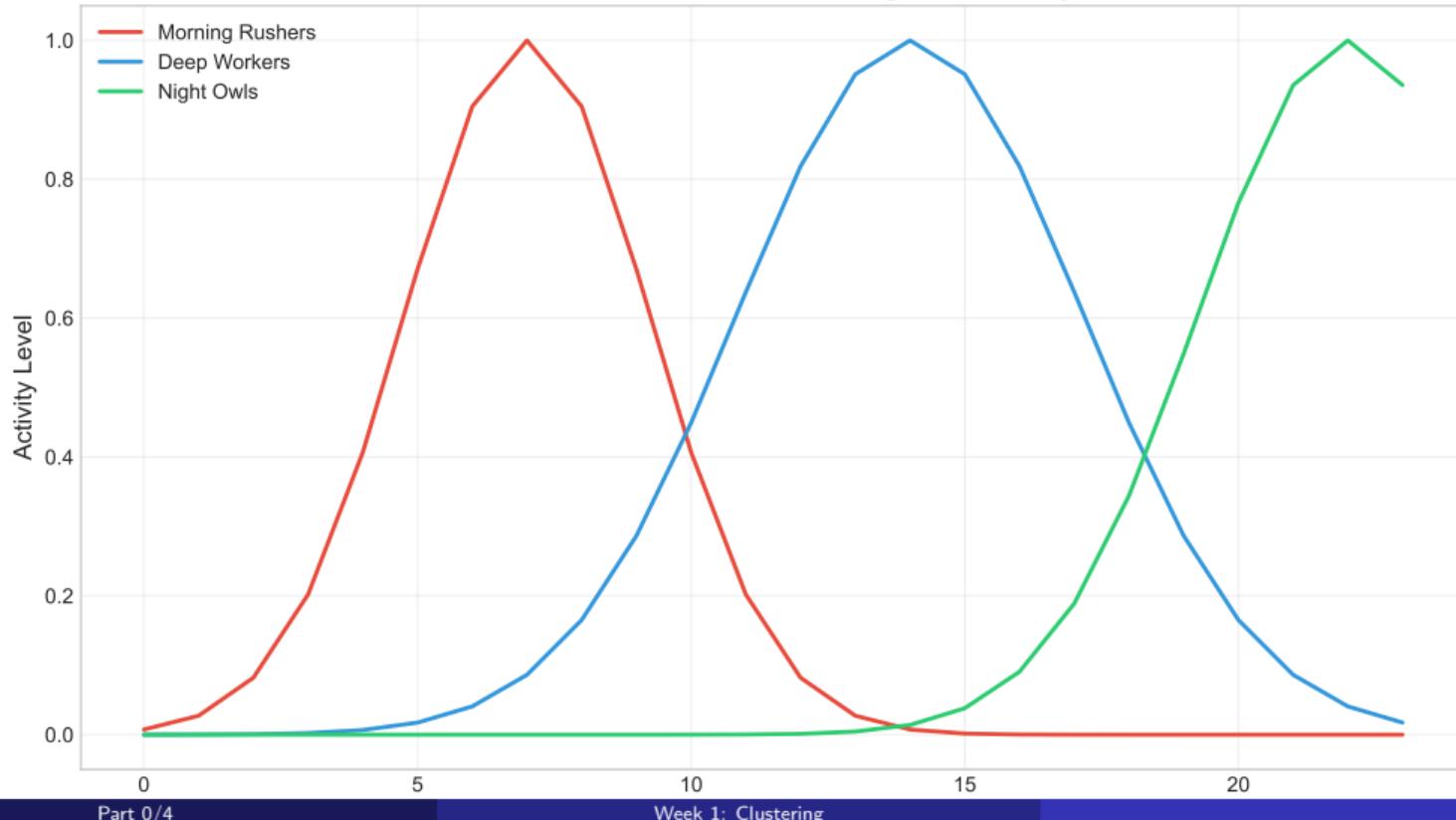
- Emerging tech: Early stage
- Disruptive: Scalability
- Incremental: Integration
- Platform-based: Network effects

**Design implication:**  
One solution won't fit all!

# Innovation Patterns Revealed

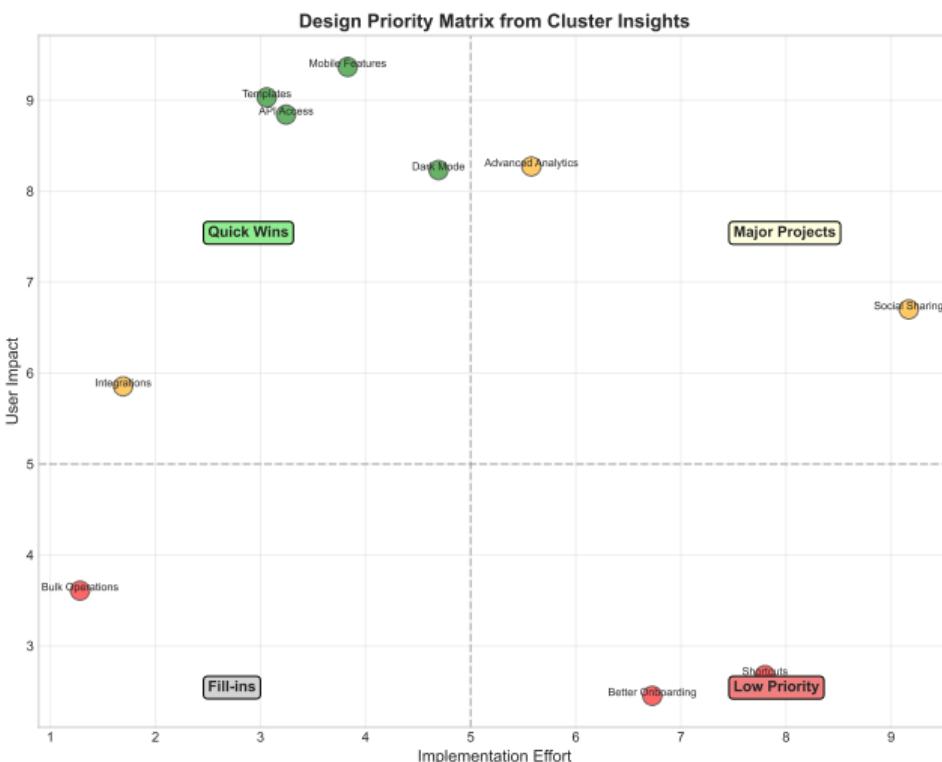
What Clusters Tell Us About Evolution

User Behavior Patterns Throughout the Day



# Design Priority Matrix

Where to Focus Your Efforts



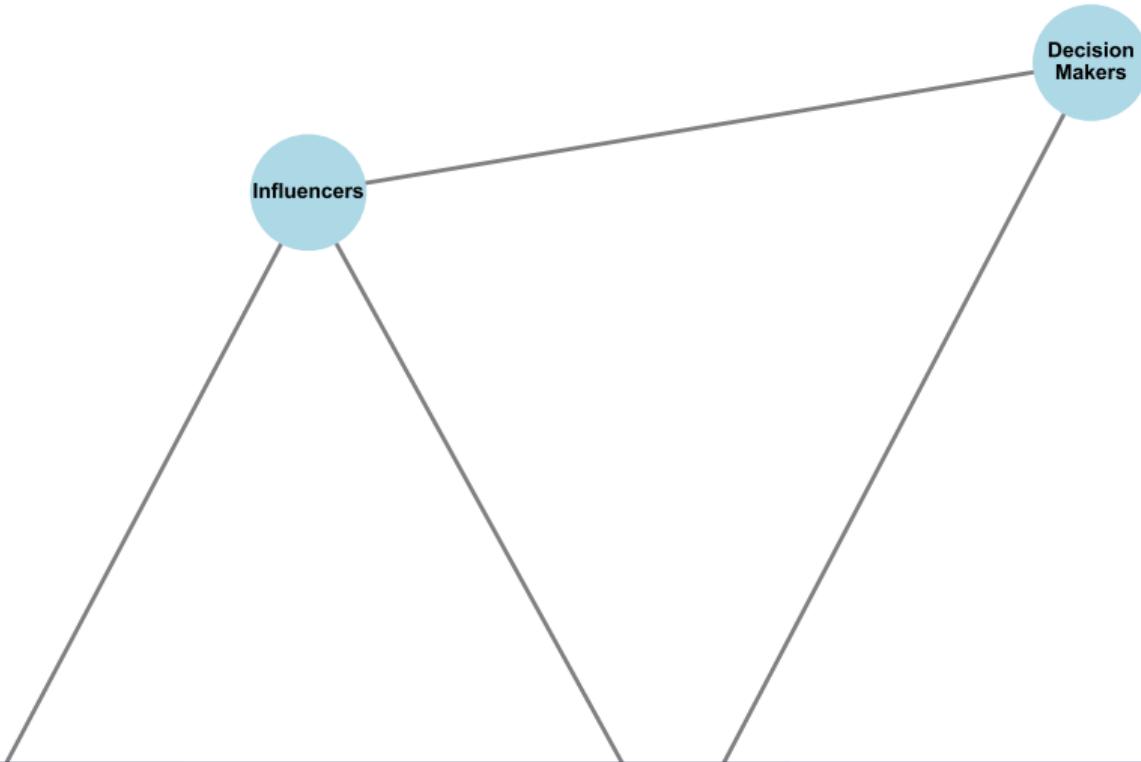
## Priority Quadrants:

- **High Impact + High Effort**  
Strategic initiatives
- **High Impact + Low Effort**  
Quick wins
- **Low Impact + Low Effort**  
Fill-ins
- **Low Impact + High Effort**  
Avoid

# Understanding Innovation Ecosystems

## Network Analysis of Innovation Connections

Stakeholder Network from Cluster Analysis



# Check Your Understanding - Part 3

## Application Knowledge Check

Progress: 3/3

### Match the Application

Match algorithm to use case:

- ① Customer segmentation → K-means
- ② Finding outliers → DBSCAN
- ③ Creating taxonomy → Hierarchical
- ④ Overlapping groups → GMM

### Design Thinking

How does clustering help in:

- **Empathize:** Find user groups
- **Define:** Identify patterns
- **Ideate:** Discover opportunities
- **Prototype:** Target solutions
- **Test:** Validate segments

**Excellent! Ready to practice with real data?**

*Next: Summary, real-world case studies, and hands-on practice exercise*

# Putting It All Together

From Theory to Practice

## You've learned:

- The clustering algorithms
- How to validate quality
- Design applications

## Now let's see it in action

How these techniques work in practice  
to find patterns in data

## PART 4

### Summary & Practice

What we'll do:

- See real-world success patterns
- Consolidate key learnings
- Practice with exercises
- Preview next week
- Explore resources

From learning to doing

# PART 4

Summary & Practice

Real-World Applications & Next Steps

# How Clustering is Used

## Common Applications and Results



## Academic Applications:

- Student performance analysis (*grouping by learning styles*)
- Research paper categorization (*organizing by topics*)
- Course recommendation systems (*matching students to courses*)
- Exam question classification (*grouping by difficulty*)

## Benefits You'll See:

- Better understanding of patterns
- Faster data analysis
- More accurate groupings
- Clearer insights from data

# Key Takeaways

## What We've Learned

### Technical Skills

- K-means clustering algorithm
- Choosing optimal K with elbow method
- Silhouette scores for validation
- DBSCAN for complex shapes
- Hierarchical clustering

### Design Applications

- Data-driven innovation archetypes
- Segment-specific journeys
- Opportunity identification
- Priority matrices
- Scaled innovation analysis

Clustering transforms data into actionable innovation insights

# Implementation Checklist

Ensuring Successful Clustering Projects

## Data Preparation

- Collect relevant features
- Handle missing values
- Standardize/normalize data
- Remove outliers if needed
- Feature engineering complete
- Data quality verified

## Quality Assurance

- Silhouette score  $> 0.5$
- Cluster sizes balanced
- Visual inspection done
- Stability tested
- Business sense verified
- Edge cases handled

## Algorithm Selection

- Choose distance metric
- Select clustering method
- Determine optimal K
- Validate with metrics

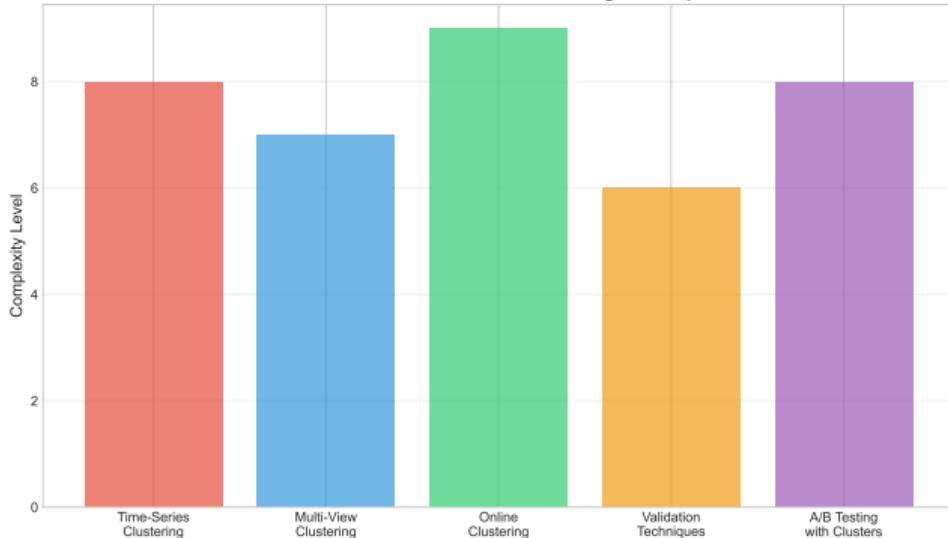
## Common Pitfalls

- Forgetting to scale features
- Wrong distance metric
- Forcing unnatural K
- Ignoring outliers

# Next Week: Advanced Clustering

Going Deeper into Innovation Patterns

Week 2 Preview: Advanced Clustering Techniques



## Week 2 Topics:

- Density-based clustering
- Gaussian mixture models
- Clustering validation
- Feature engineering
- Real-time clustering

## Design Focus:

- Dynamic innovation tracking
- Evolving innovation landscapes
- Predictive opportunity analysis
- Micro-innovation detection

# Resources & Further Reading

Deepen Your Understanding

## Technical Resources

### Papers:

- MacQueen, J. (1967). K-means
- Ester et al. (1996). DBSCAN
- Rousseeuw (1987). Silhouettes

### Tools:

- scikit-learn clustering
- Orange data mining
- KNIME analytics

## Design Resources

### Books:

- "Design Thinking" - Tim Brown
- "Sprint" - Jake Knapp
- "Lean UX" - Jeff Gothelf

### Applications:

- Miro (journey mapping)
- Figma (archetype creation)
- Optimal Workshop

Questions? Let's discuss!

# Glossary of Technical Terms

## Key Concepts Quick Reference

### Clustering Algorithms:

- **K-Means:** Partitions data into K predefined clusters
- **DBSCAN:** Density-based spatial clustering
- **Hierarchical:** Builds cluster tree (dendrogram)
- **GMM:** Gaussian Mixture Models, soft clustering

### Key Parameters:

- **K:** Number of clusters
- **eps:** Neighborhood radius (DBSCAN)
- **min\_samples:** Minimum points for density
- **n\_init:** Number of random initializations

### Evaluation Metrics:

- **Silhouette:** Cluster cohesion vs separation [-1,1]
- **Inertia:** Sum of squared distances to centroids
- **Davies-Bouldin:** Ratio of within to between distances
- **Calinski-Harabasz:** Ratio of dispersions

### Innovation Terms:

- **Empathy Mapping:** Understanding user perspectives
- **Pain Points:** User problems/frustrations
- **User Archetypes:** Representative user groups
- **Innovation Ecosystem:** Connected stakeholders

# Implementation Checklist

Your Step-by-Step Guide to Success

## Data Preparation:

- Collect innovation feedback data
- Clean and remove duplicates
- Handle missing values
- Normalize/standardize features
- Create feature vectors

## Algorithm Selection:

- Analyze data distribution
- Choose appropriate algorithm
- Set initial parameters
- Prepare validation strategy

## Implementation:

- Run clustering algorithm
- Calculate evaluation metrics
- Visualize results (PCA/t-SNE)
- Validate with domain experts
- Iterate and refine

## Innovation Application:

- Map clusters to user personas
- Identify innovation opportunities
- Create targeted solutions
- Design prototype features
- Test with user groups

**Ready? Start with data preparation and work your way down!**

# Appendix: K-Means Mathematics (Optional)

The Mathematical Foundation - For Those Interested

## What K-means tries to minimize:

$$J = \sum_{i=1}^n \sum_{j=1}^k w_{ij} \|x_i - \mu_j\|^2$$

*In simple terms: Make points close to their group centers*

Where:

- $n$  = number of data points
- $k$  = number of clusters
- $w_{ij} = 1$  if  $x_i$  belongs to cluster  $j$ , 0 otherwise
- $\mu_j$  = centroid of cluster  $j$

## Update Rules:

- ① Assignment:  $c^{(i)} = \arg \min_j \|x^{(i)} - \mu_j\|^2$
- ② Update:  $\mu_j = \frac{1}{|S_j|} \sum_{i \in S_j} x^{(i)}$

# Appendix: Distance Metrics (Optional)

Different Ways to Measure "How Far Apart" Things Are

**Euclidean Distance:**

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

**Manhattan Distance:**

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

**Minkowski Distance:**

$$d(x, y) = \left( \sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

**Cosine Similarity:**

$$\cos(\theta) = \frac{x \cdot y}{\|x\| \cdot \|y\|}$$

**Jaccard Distance:**

$$J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

**Mahalanobis Distance:**

$$d(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)}$$

# Appendix: Silhouette Score Explained

How We Know If Groups Are Good

## Silhouette Score for point $i$ :

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

Where:

- $a(i)$  = average distance to points in same cluster
- $b(i)$  = average distance to points in nearest neighbor cluster

## Interpretation:

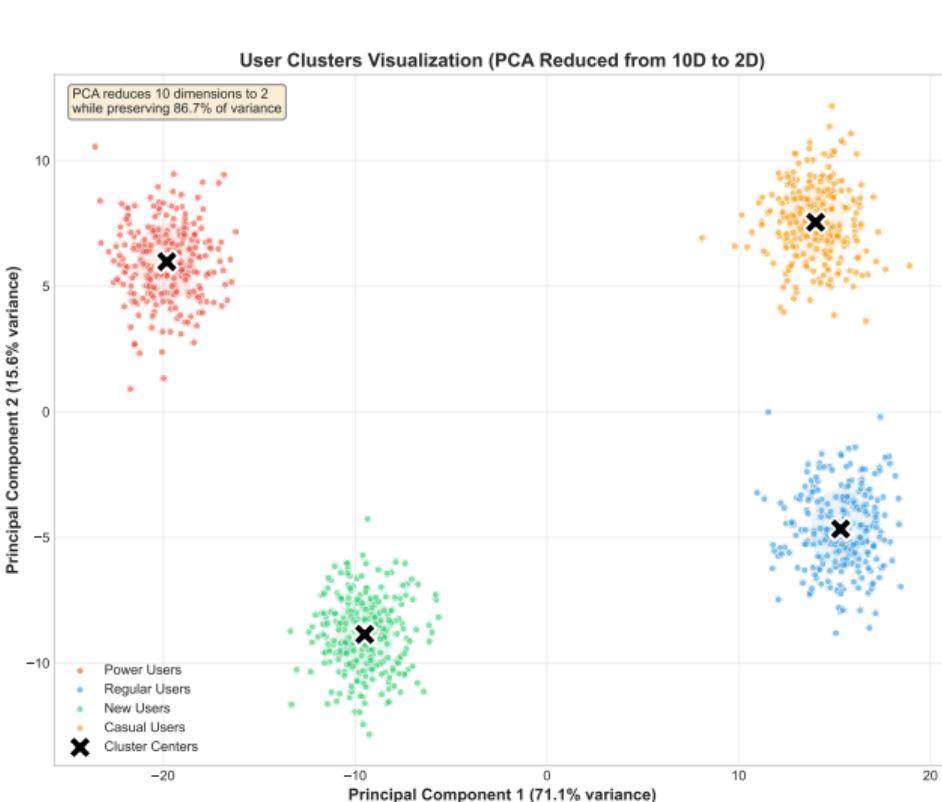
- $s(i) \approx 1$ : Well clustered
- $s(i) \approx 0$ : On border between clusters
- $s(i) \approx -1$ : Misclassified

## Overall Score:

$$S = \frac{1}{n} \sum_{i=1}^n s(i)$$

# Appendix: Visualizing High-Dimensional Data

Making Complex Data Viewable in 2D



## PCA Process:

- ① Standardize data
- ② Compute covariance matrix
- ③ Find eigenvectors/values
- ④ Select top 2 components
- ⑤ Transform data

## Variance Explained:

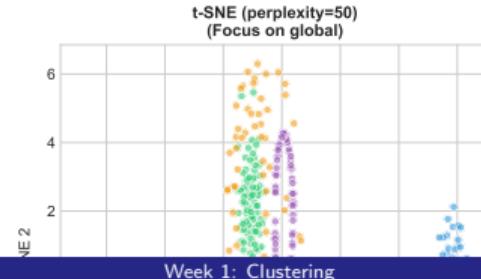
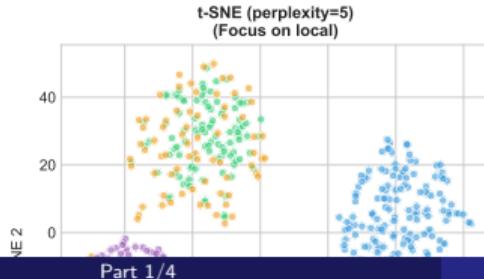
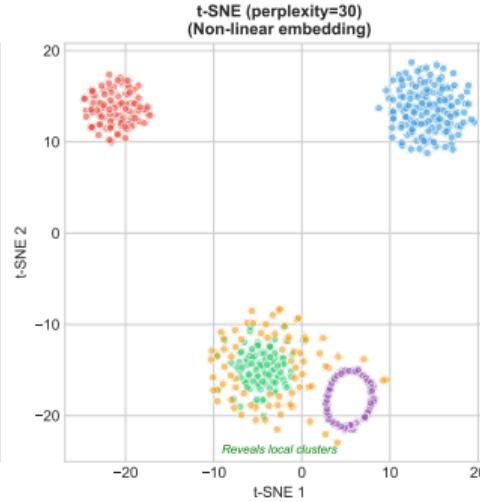
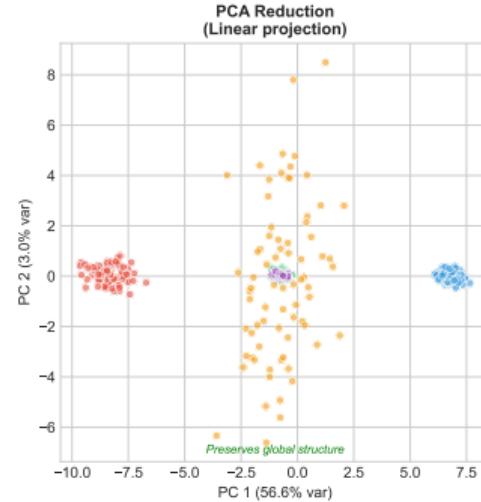
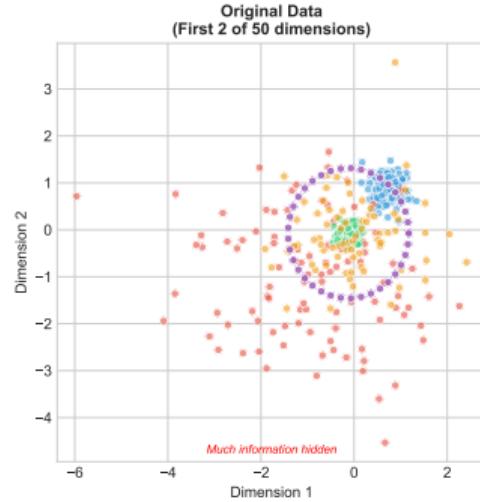
- PC1: 45.2%
- PC2: 28.7%
- Total: 73.9%

# Dimensionality Reduction: PCA vs t-SNE

Revealing Hidden Patterns in High-Dimensional Innovation Space

## Dimensionality Reduction: PCA vs t-SNE for Innovation Data

Revealing Hidden Patterns in High-Dimensional Innovation Space



### Method Comparison

	PCA	t-SNE
Speed	Fast	Slow
Scalability	Excellent	Limited
Interpretation	Clear axes	No axes meaning

# Appendix: How DBSCAN Works

Finding Groups Based on How Close Points Are

## Key Parameters:

- $\epsilon$  (eps): Maximum distance between points
- MinPts: Minimum points to form dense region

## Point Classification:

- **Core point:** Has  $\geq$  MinPts within  $\epsilon$
- **Border point:** Within  $\epsilon$  of core point
- **Noise point:** Neither core nor border

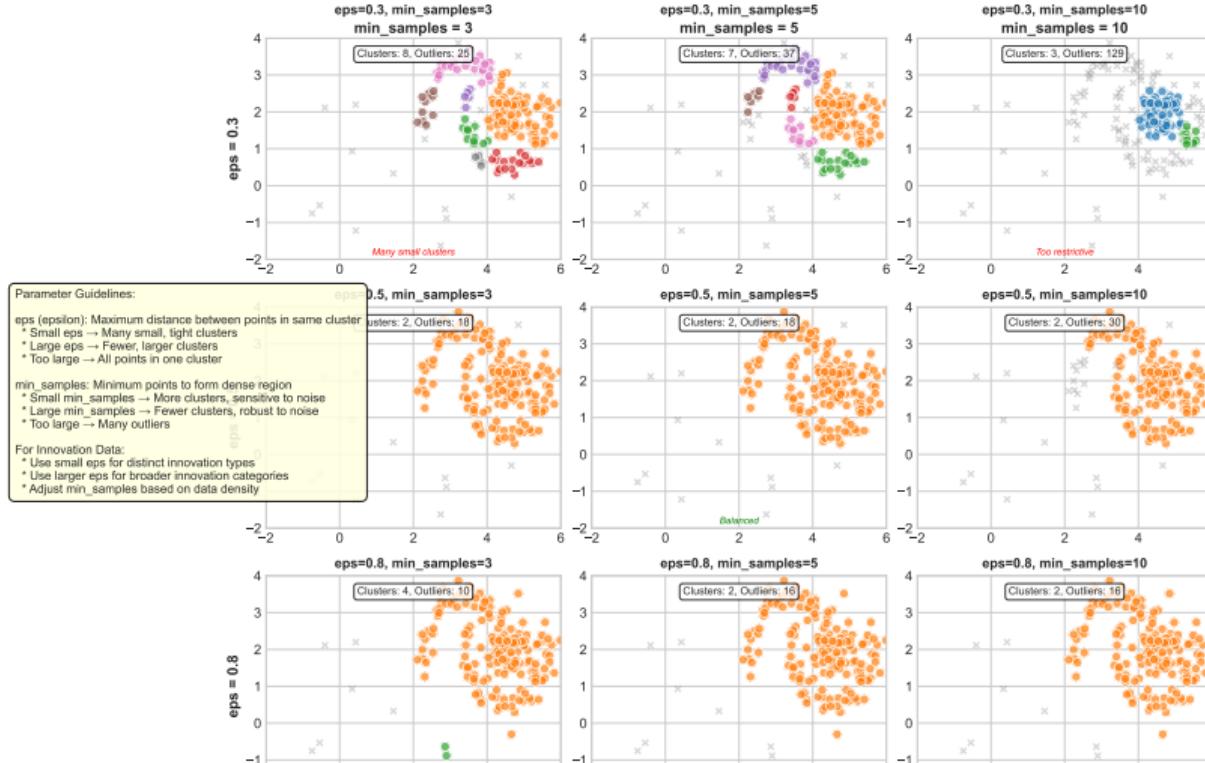
## Algorithm Steps:

- ① Find all core points
- ② Form clusters from core points within  $\epsilon$
- ③ Assign border points to clusters
- ④ Mark remaining as noise

# DBSCAN Parameter Tuning

Impact of eps and min\_samples on Clustering Results

DBSCAN Parameter Tuning: Impact on Innovation Clustering



# Appendix: Python Implementation

## Ready-to-Use Code Snippets

### K-Means Clustering

```
1 from sklearn.cluster import KMeans
2 import numpy as np
3
4 # Generate sample data
5 X = np.random.randn(1000, 2)
6
7 # Configure and fit K-means
8 kmeans = KMeans(
9     n_clusters=3,
10    random_state=42,
11    n_init=10
12 )
13 labels = kmeans.fit_predict(X)
14
15 # Get cluster centers
16 centers = kmeans.cluster_centers_
17
18 # Calculate inertia
19 inertia = kmeans.inertia_
```

### DBSCAN Clustering

```
1 from sklearn.cluster import DBSCAN
2
3 # Configure DBSCAN
4 dbscan = DBSCAN(
5     eps=0.3,
6     min_samples=5,
7     metric='euclidean'
8 )
9
10 # Fit and predict
11 labels = dbscan.fit_predict(X)
12
13 # Analyze results
14 outliers = labels == -1
15 n_clusters = len(set(labels)) - 1
16
17 print(f"Clusters: {n_clusters}")
18 print(f"Outliers: {sum(outliers)}")
19 print(f"Covariance: {100*(1-sum(outliers)/len(labels)):.1f} %")
```

# Appendix: Evaluation Metrics Code

## Measuring Clustering Quality

### Silhouette Analysis

```
1 from sklearn.metrics import silhouette_score
2 from sklearn.metrics import silhouette_samples
3
4 # Calculate overall score
5 score = silhouette_score(X, labels)
6 print(f"Silhouette Score: {score:.3f}")
7
8 # Per-sample scores
9 sample_scores = silhouette_samples(X, labels)
10
11 # Per-cluster average
12 for i in range(n_clusters):
13     cluster_scores = sample_scores[labels == i]
14     avg = cluster_scores.mean()
15     print(f"Cluster {i}: {avg:.3f}")
```

### Finding Optimal K

```
1 # Elbow method
2 inertias = []
3 silhouettes = []
4 K_range = range(2, 10)
5
6 for k in K_range:
7     km = KMeans(n_clusters=k,
8                  random_state=42)
9     labels = km.fit_predict(X)
10
11     inertias.append(km.inertia_)
12     silhouettes.append(
13         silhouette_score(X, labels)
14     )
15
16 # Find elbow point
17 # Plot inertias vs K
18 # Choose K at elbow
```

# Appendix: Implementation Guidelines

## Practical Considerations

### Data Preparation

- Standardize features
- Handle missing values
- Remove outliers (if needed)
- Feature selection/engineering
- Consider scaling methods

### Validation Methods

- Silhouette score
- Davies-Bouldin index
- Calinski-Harabasz score
- Visual inspection
- Domain expert review

### Algorithm Selection

- K-means: Spherical, similar size
- DBSCAN: Arbitrary shapes
- Hierarchical: Nested structure
- GMM: Overlapping clusters

### Common Pitfalls

- Not scaling features
- Wrong distance metric
- Ignoring outliers
- Over-clustering
- Forcing clusters

# Glossary of Technical Terms

## Key Concepts Reference

### Algorithms:

- **K-means:** Partitions data into K spherical clusters
- **DBSCAN:** Density-based clustering for arbitrary shapes
- **GMM:** Gaussian Mixture Models for soft clustering
- **Hierarchical:** Tree-based clustering approach

### Metrics:

- **Silhouette:** Measures cluster separation (-1 to 1)
- **Inertia:** Sum of squared distances to centroids
- **Davies-Bouldin:** Ratio of within to between cluster distance

### Concepts:

- **Centroid:** Center point of a cluster
- **Elbow Method:** Technique to find optimal K
- **Outlier:** Data point not belonging to any cluster
- **Convergence:** When algorithm stops improving

### Preprocessing:

- **Standardization:** Zero mean, unit variance
- **Normalization:** Scale to [0,1] range
- **PCA:** Principal Component Analysis
- **t-SNE:** t-distributed Stochastic Neighbor Embedding