Machine Learning for Innovation

Week 1: Finding Patterns with Clustering

BSc Data Science & AI Program

No prerequisites required

September 13, 2025

What We’ll Learn Today

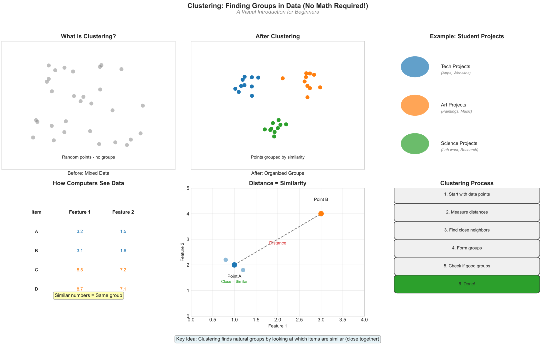
Finding Groups in Data

By the end of today:

- Understand what clustering does
- Know when to use it
- Apply 3 different methods
- Interpret the results
- Practice with real examples

No math required!

We’ll use pictures and examples instead.



Clustering: Finding Groups in Data (No Math Required!)

A Visual Introduction for Beginners

Item	Feature 1	Feature 2
A	3.2	1.5
B	3.1	1.6
C	8.5	7.2
D	8.7	7.1

Similar numbers = Same group

1. Start with data points
2. Measure distances
3. Find close neighbors
4. Form groups
5. Check if good groups
6. Done!

Key Idea: Clustering finds natural groups by looking at which items are similar (close together)

Part 1

Understanding the Problem

Why we need to find patterns

The Challenge: Too Much Information

Why computers help us see patterns

Imagine you have:

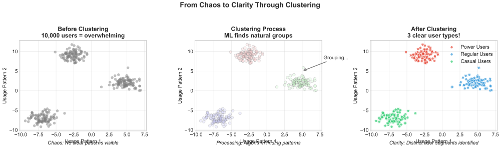
- 1000 customer reviews
- 500 product ideas
- 10,000 survey responses

The problem:

- Too much to read manually
- Hidden patterns we can't see
- Takes too long to analyze

The solution:

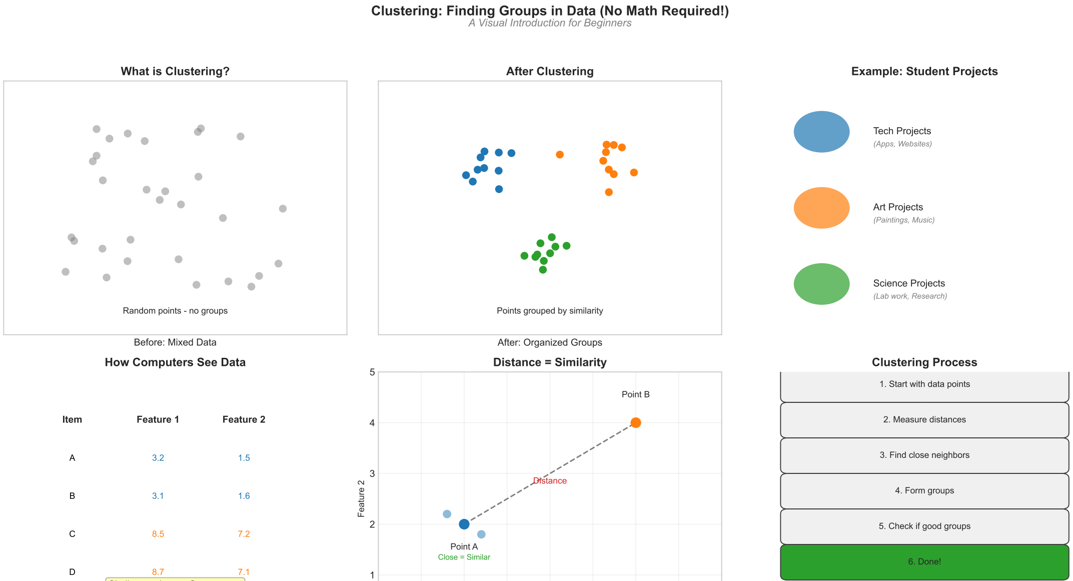
Let computers find the groups for us!



From chaos to organized groups

What is Clustering?

A simple explanation



Item	Feature 1	Feature 2
A	3.2	1.5
B	3.1	1.6
C	8.5	7.2
D	8.7	7.1

1. Start with data points
2. Measure distances
3. Find close neighbors
4. Form groups
5. Check if good groups
6. Done!

Where We Use Clustering

Everyday examples

Business:

- Group similar customers
- Organize products in store
- Find spending patterns

Science:

- Group similar genes
- Classify star types
- Identify weather patterns

Daily Life:

- Music playlists (similar songs)
- Friend suggestions (similar interests)
- News categories (similar topics)

Innovation:

- Group similar ideas
- Find user needs
- Identify opportunities

All these use clustering to find patterns!

Part 2

How Clustering Works

Three different approaches

Three Ways to Find Groups

Each has strengths and weaknesses

Clustering Algorithms: Simple Comparison

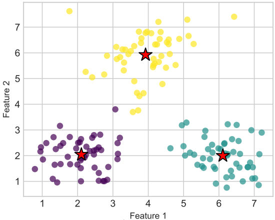
Three Different Ways to Find Groups

K-Means: How it Works

1. Pick 3 center points
2. Assign each point to nearest center
3. Move centers to middle of groups
4. Repeat until stable

Like organizing by neighborhoods

K-Means Result



Stars = Centers

When to Use K-Means

Good for:

- Round groups
- Similar sizes
- Fast results

Not good for:

- Weird shapes
- Different sizes

Speed & Difficulty

Speed: FAST

Difficulty: EASY

You need to know:

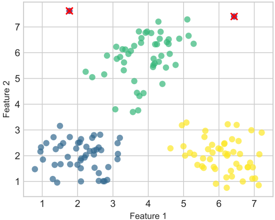
- How many groups (K)

DBSCAN: How it Works

1. Look at each point
2. Count neighbors nearby
3. If enough neighbors → core point
4. Connect core points → groups

Like finding crowds at a party

DBSCAN Result



X = Outliers

When to Use DBSCAN

Good for:

- Any shape groups
- Finding outliers
- Unknown group count

Not good for:

- Different densities
- Need exact K groups

Speed & Difficulty

Speed: MEDIUM

Difficulty: MEDIUM

You need to know:

- Distance (eps)
- Min neighbors

Hierarchical: How it Works

Hierarchical Result

When to Use Hierarchical

Speed & Difficulty

K-Means: The Most Common Method

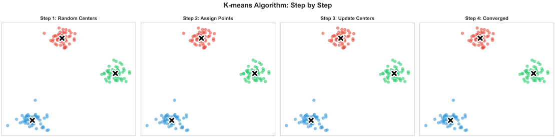
Like organizing by neighborhoods

How it works:

1. Decide how many groups (K)
2. Place K center points randomly
3. Assign each point to nearest center
4. Move centers to middle of their group
5. Repeat until stable

Good for:

- Round, similar-sized groups
- When you know how many groups
- Need fast results



Watch the centers move to find groups

DBSCAN: Finding Any Shape

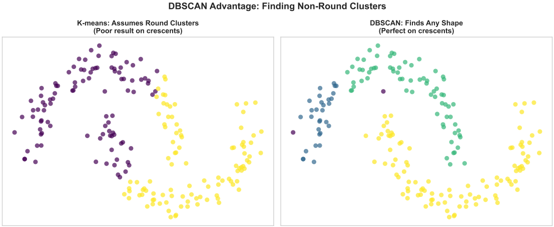
Like finding crowds at a party

How it works:

1. Look at each point
2. Count neighbors within distance
3. If enough neighbors → core point
4. Connect core points → groups
5. Points with few neighbors → outliers

Good for:

- Weird-shaped groups
- Finding outliers
- Don't know how many groups



Can find any shape, even crescents!

How Good Are Our Groups?

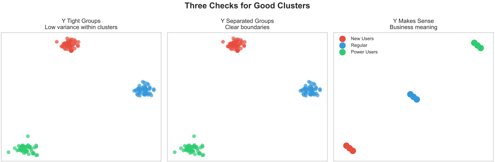
Checking if clustering worked

What makes groups good?

- Points in same group are similar
- Different groups are different
- Groups make sense for your problem

Simple checks:

1. **Visual**: Do groups look separated?
2. **Size**: Are groups reasonable sizes?
3. **Meaning**: Can you explain each group?
4. **Stability**: Same result if run again?



Silhouette Score:
-1 = Bad (overlapping)
0 = OK (touching)
+1 = Good (separated)

Which Method Should I Use?

A simple decision guide

Quick Decision Tree

Use K-Means if:

- Need speed
- Know number of groups
- Groups are round
- Similar sizes expected

Use DBSCAN if:

- Groups have weird shapes
- Want to find outliers
- Don't know group count
- Groups have different densities

Use Hierarchical if:

- Small dataset (¡500 points)
- Want to see all groupings
- Need tree structure
- Can wait for results

Not sure? Try K-Means first - it's simplest!

Part 3

Let's Practice

A simple example you can follow

Practice: Student Study Patterns

Follow along with this example

Practice Example: Grouping Students by Study Habits

A Simple Clustering Exercise

The Problem

A teacher wants to understand

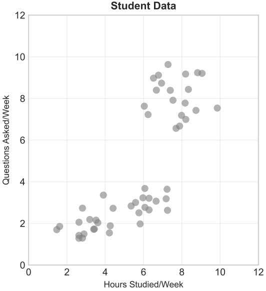
different study patterns in class.

Data collected:

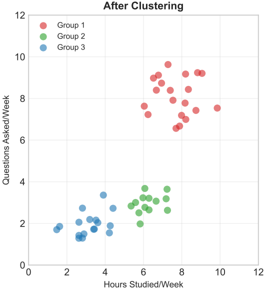
- Hours studied per week
- Number of questions asked

50 students total

Goal: Find study groups



Each dot = 1 student



What We Found

Group 1:

Need extra help

• Low hours, few questions

Group 2:

Independent learners

• Good hours, few questions

Group 3:

Highly engaged

• Many hours, many questions

Your Turn: Step 1

Load the data:

```
import pandas as pd
data = pd.read_csv("students.csv")
```

Look at it:

```
print(data.head())
```

You should see:

```
  hours  questions
```

Your Turn: Step 2

Prepare the data:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(data)
```

Why scale?

Makes features comparable

Apply clustering:

Your Turn: Step 3

See the results:

```
import matplotlib.pyplot as plt
plt.scatter(data["hours"],
            data["questions"],
            c=labels)
plt.xlabel("Hours")
plt.ylabel("Questions")
plt.show()
```

Check Your Work

□ Did you find 3 groups?

□ Are groups visually separated?

□ Do groups make sense?

What to look for:

- Clear differences between groups
- Similar students in same group
- Groups tell a story

Your Turn: Exercise

Try it yourself

Dataset: Store Products

You have data about 200 products:

- Price (0-100)
- Customer rating (1-5 stars)
- Sales per month

Your tasks:

1. Load the data
2. Apply K-Means with K=3
3. Plot the results
4. Describe each group
5. Try K=4, which is better?

Starter code:

```
import pandas as pd
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Load data
data = pd.read_csv(’products.csv’)

# Apply clustering
kmeans = KMeans(n_clusters=3)
labels = kmeans.fit_predict(data)

# Plot results
plt.scatter(data[’price’],
data[’rating’],
c=labels)
plt.show()
```

Common Mistakes to Avoid

Learn from these errors

Mistake 1: Forgetting to scale

- Problem: Price (0-100) vs Rating (1-5)
- Price dominates because bigger numbers
- Solution: Always scale your data first!

Mistake 2: Wrong K

- Too few groups: Miss important patterns
- Too many groups: Overly complex
- Solution: Try different K values

Mistake 3: Ignoring outliers

- One weird point can ruin groups
- K-Means pulls centers toward outliers
- Solution: Check for outliers first

Mistake 4: Not checking results

- Algorithm always gives an answer
- Doesn't mean it's meaningful!
- Solution: Always visualize and interpret

Remember: The computer finds patterns, YOU decide if they make sense!

Week 1 Summary

What you learned today

Concepts

- Clustering finds groups
- No labels needed
- Distance = similarity
- Multiple methods exist

Methods

- K-Means (fast, simple)
- DBSCAN (any shape)
- Hierarchical (tree view)
- Each has trade-offs

Skills

- Choose algorithm
- Apply clustering
- Check quality
- Interpret results

You can now find patterns in data!

Next week: More advanced clustering techniques

Resources for Practice

Where to learn more

Practice datasets:

- Iris flowers (150 samples, 4 features)
- Wine quality (178 samples, 13 features)
- Mall customers (200 samples, 5 features)

Python libraries:

- scikit-learn (all algorithms)
- pandas (data handling)
- matplotlib (visualization)

Online resources:

- scikit-learn.org/stable/modules/clustering
- Google Colab (free Python online)
- Kaggle Learn (free courses)

Help available:

- Office hours: Wed 3-5pm
- Course forum
- Study groups

Questions? Just ask - no question is too simple!