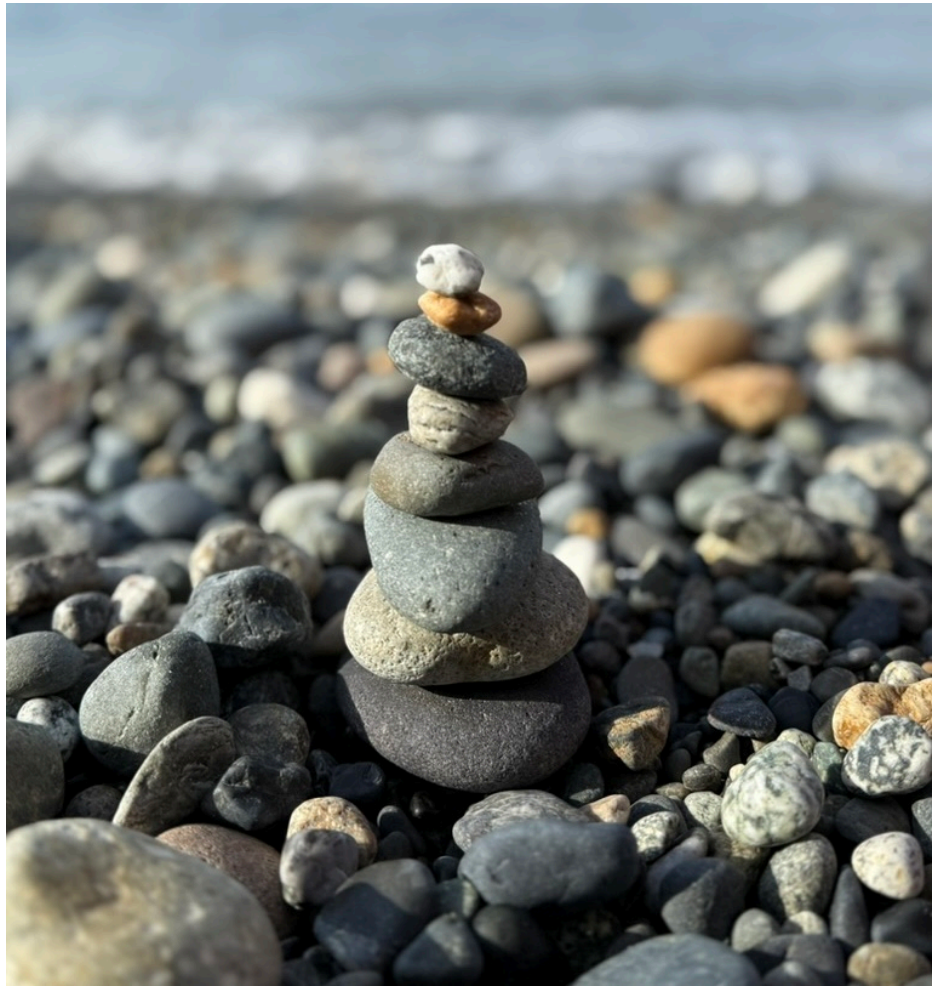
CPSC 330 Lecture 15: DBSCAN and hierarchical clustering

Varada Kolhatkar

UBC Computer Science

Focus on the breath!



Announcements

Today’s big questions

1. When does a **nearest-centre** grouping fail?
2. How can **density** define a cluster and identify noise?
3. How can we explore **nested groups** with a dendrogram?
4. What evidence makes a clustering **useful**?

iClicker 15.1: recalling K-Means

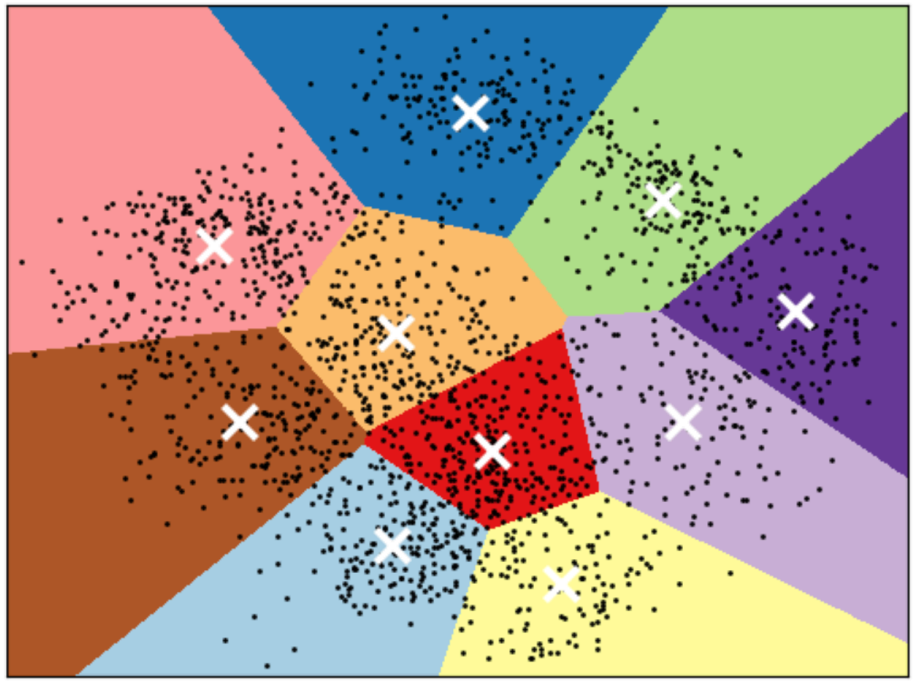
Select all statements that are true.

- a. With n examples and d features, K-Means learns k centres, each with d coordinates.
- b. The parameter k plays the same role in KNN and K-Means.
- c. Changing feature units can change a distance-based clustering.
- d. When choosing the number of clusters, we should prefer values that produce clusters of roughly equal size.

K-Means limitations

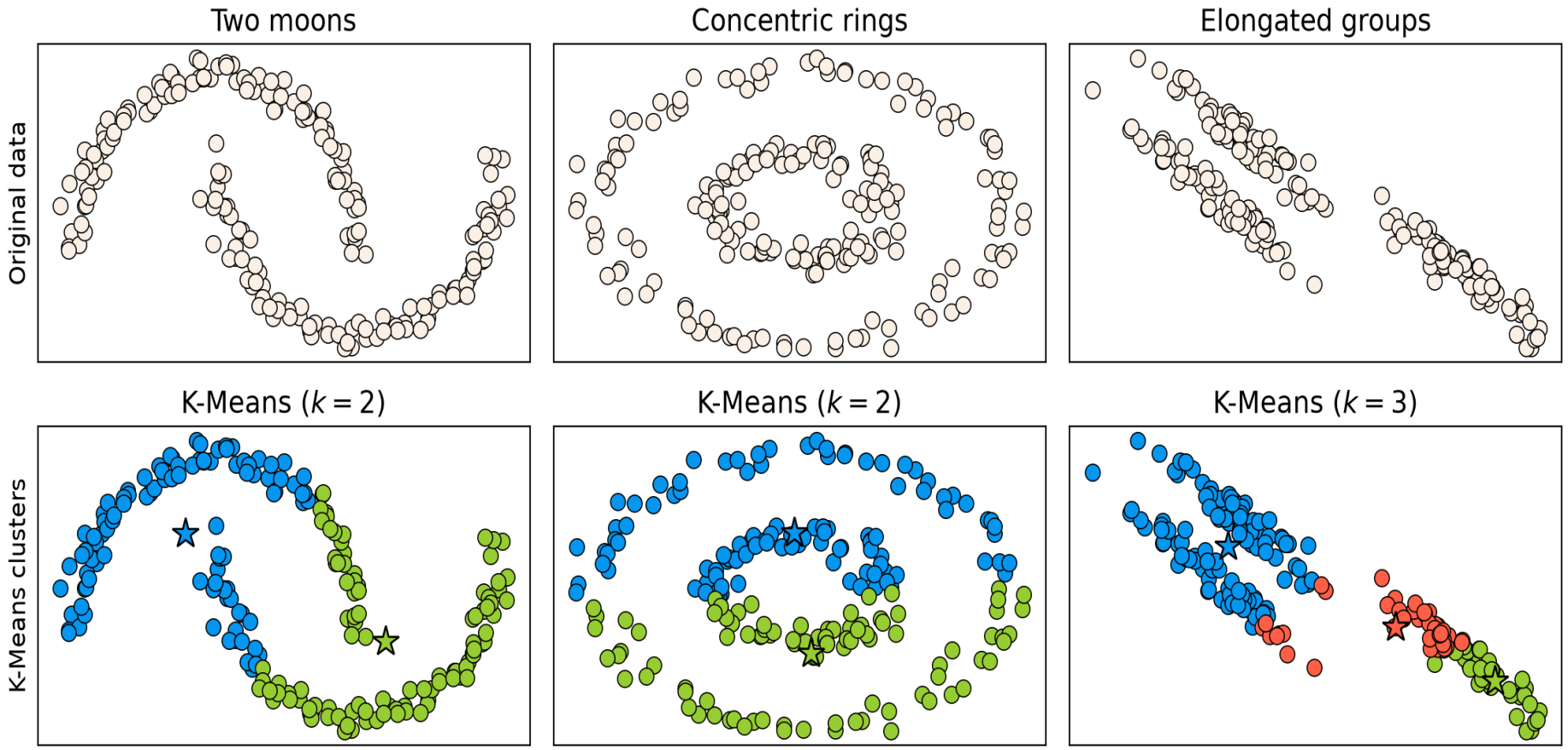
Shape of clusters

- Good for compact, roughly spherical clusters of comparable sizes.



When nearest-centre assignments fail

Even with the intended k, K-Means can split natural groups.

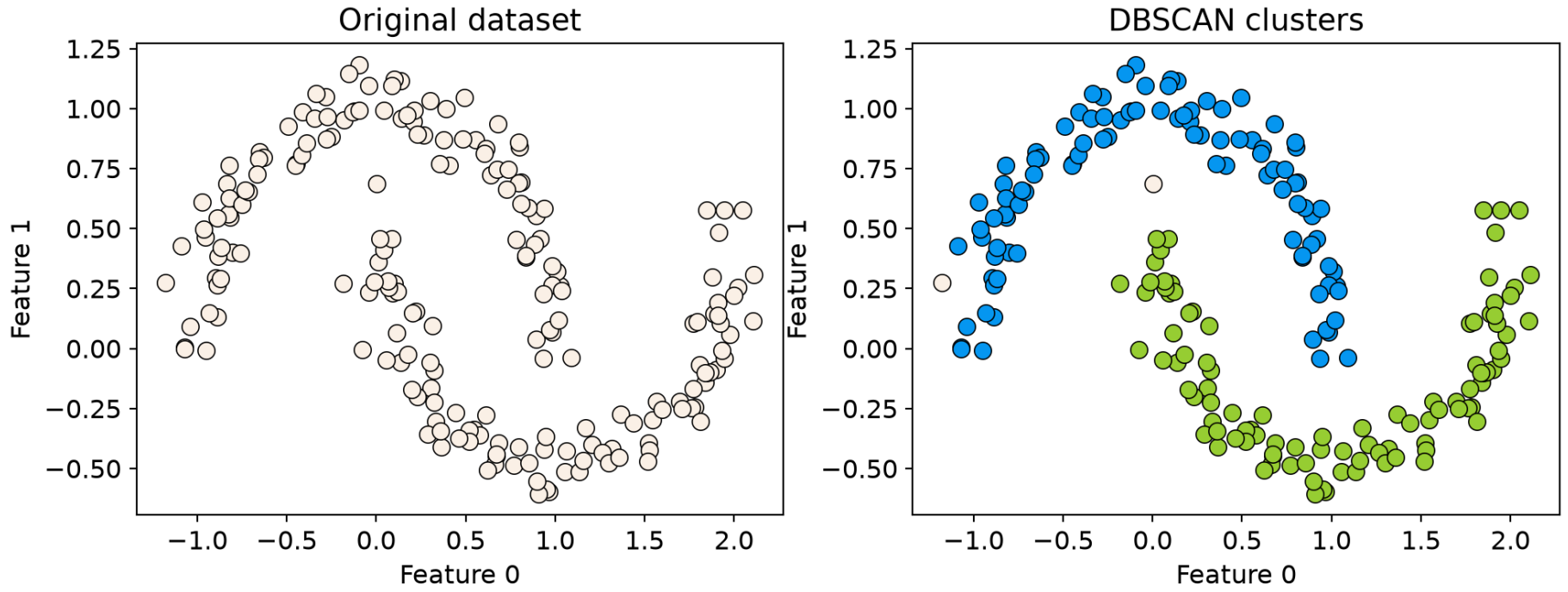


Can we do better?

DBSCAN

- **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
- Forms clusters from connected dense regions, leaving isolated points as noise.

```
X, y = make_moons(n_samples=200, noise=0.08, random_state=42)
dbscan = DBSCAN(eps=0.2, min_samples=5)
dbscan.fit(X)
plot_original_clustered(X, dbscan, dbscan.labels_)
```



Two choices define “locally dense”

- `eps`: radius of a point’s neighbourhood.
- `min_samples`: minimum number of examples in that neighbourhood, **including the point itself**, needed to make it a **core point**.

We choose these parameters; the **number of clusters emerges**.

Their values depend on the **features, scale, and distance metric**.

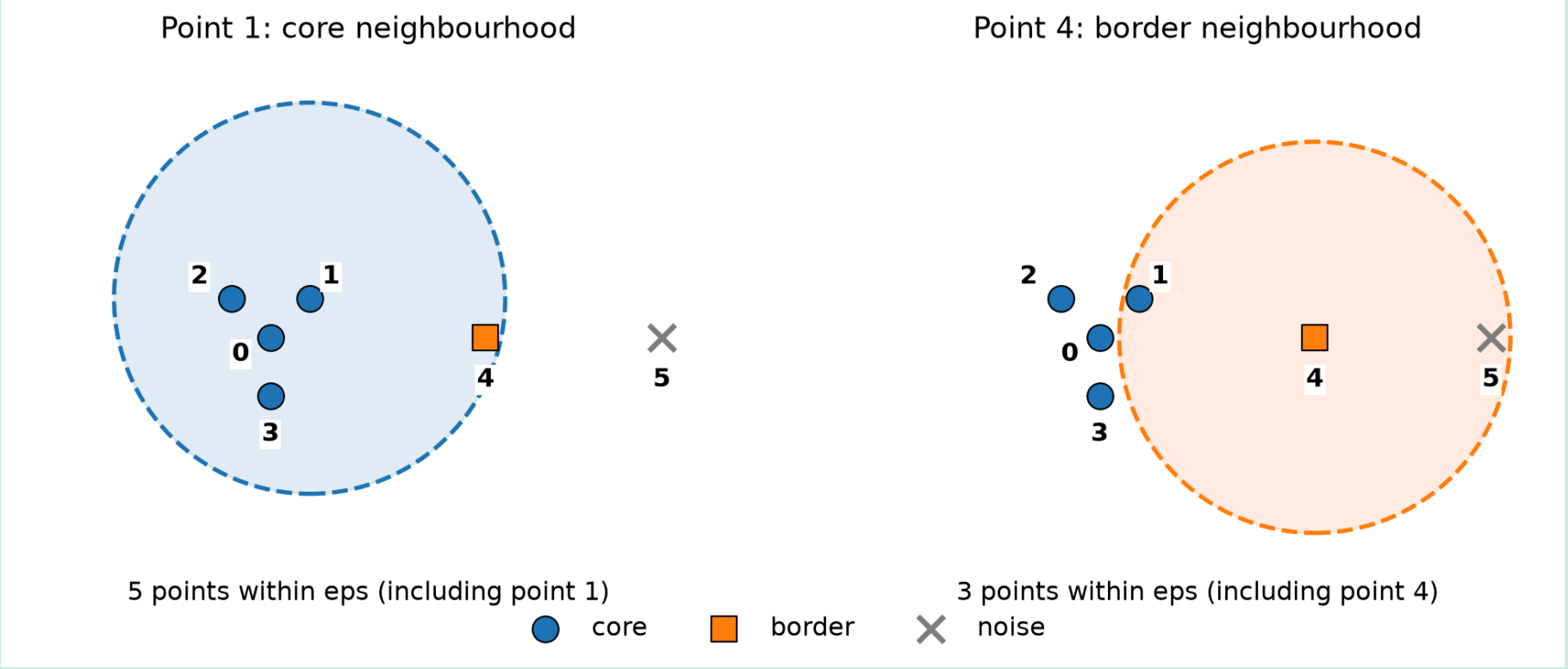
Core, border, and noise

- **Core:** at least `min_samples` examples within `eps`, counting itself.
- **Border:** not core, but within `eps` of a core point.
- **Noise:** neither core nor within `eps` of any core point.

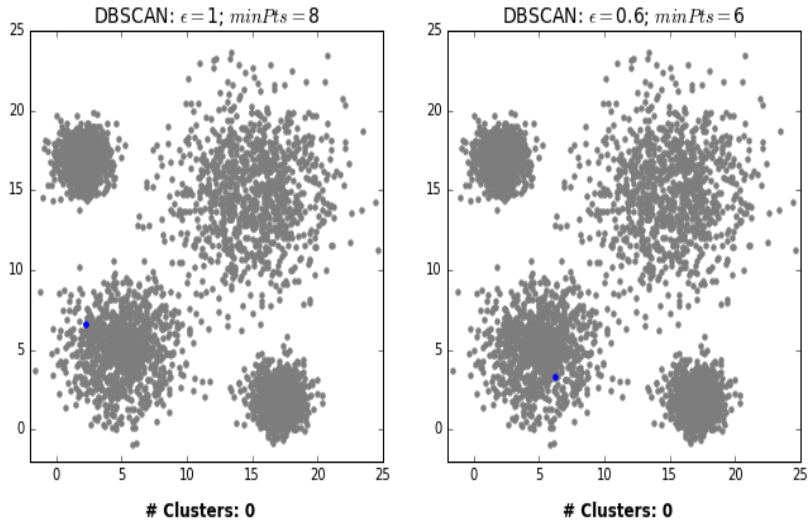
Only core points extend a cluster. Border points can join it.

Your turn: who can extend the group?

`eps=0.5`, `min_samples=4`. Why is point 5 noise even though it is near point 4?



How DBSCAN grows clusters



1. Visit an unprocessed point.
2. If core, start a cluster and add its neighbours.
3. Expand through any core neighbours.
4. Repeat until no further expansion is possible.
5. Continue with unprocessed points.

Points never reached from core points remain **noise**.

DBSCAN in scikit-learn

```
model = DBSCAN(eps=0.22, min_samples=6)
labels = model.fit_predict(X_moons)
# After fitting, these labels are also in model.labels_
```

- One label per input row; **`-1` means noise**.
- Other integers are arbitrary group identifiers.
- Standard DBSCAN has **no `predict` method** for new points.

Your turn: human DBSCAN

Keep the same students and positions for every run.

- Each student is a point; distance is physical seat distance.
- Start with `eps` about 1 m and `min_samples=4`.
- Count neighbours **including yourself**; identify all core points first.
- Give connected core points one label, then label their border neighbours.
- Everyone else is noise. **Border points do not spread labels.**

Repeat: change one parameter

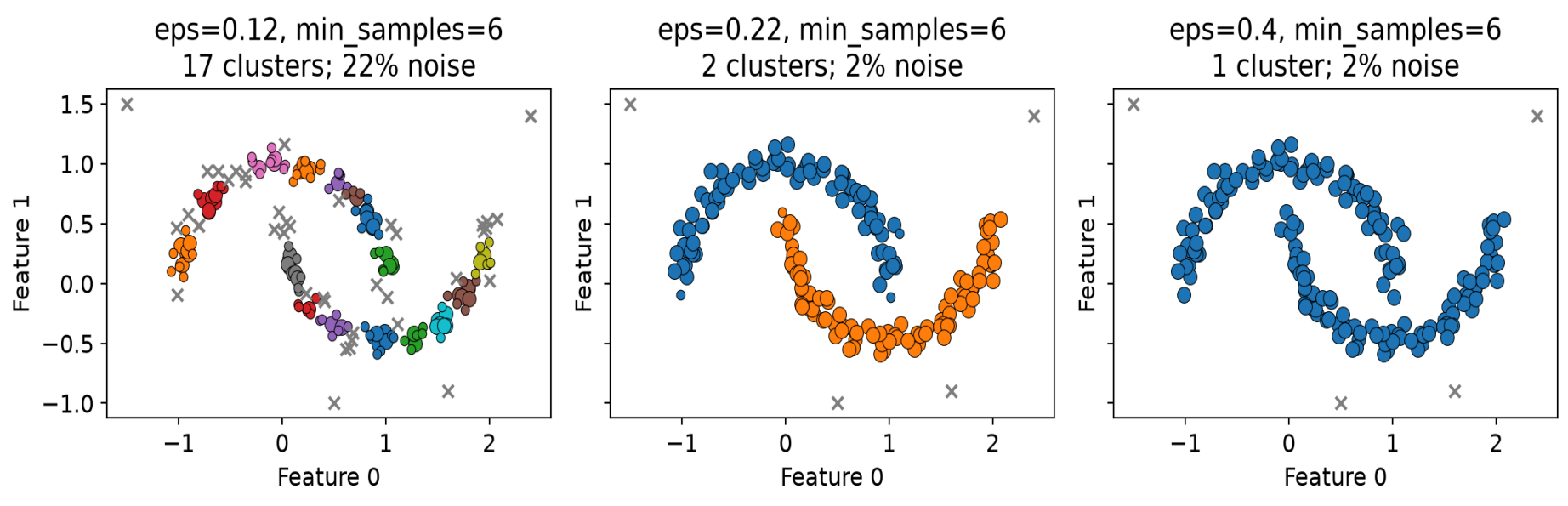
Run	`eps`	`min_samples`	What changed?
Baseline	~1 m	4	Identify core, border, noise
Wider neighbourhood	~3 m	4	Which groups connect?
Stricter density	~1 m	15	Who remains core?

Reset labels between runs. Keep positions fixed.

Predict: increase eps

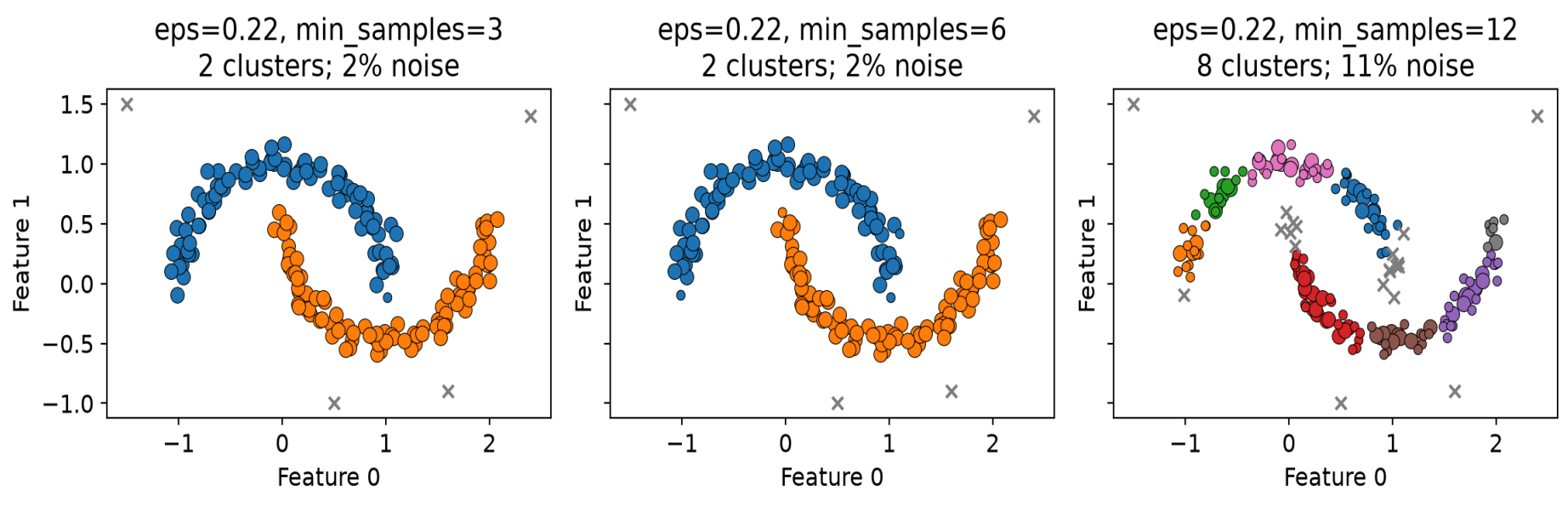
Hold `min_samples=6` fixed. Which setting fragments or merges the moons?

Large dots: core. **Small dots:** border. **Grey crosses:** noise.



Predict: increase `min_samples`

Hold `eps=0.22` fixed. Does every parameter change alter the partition?



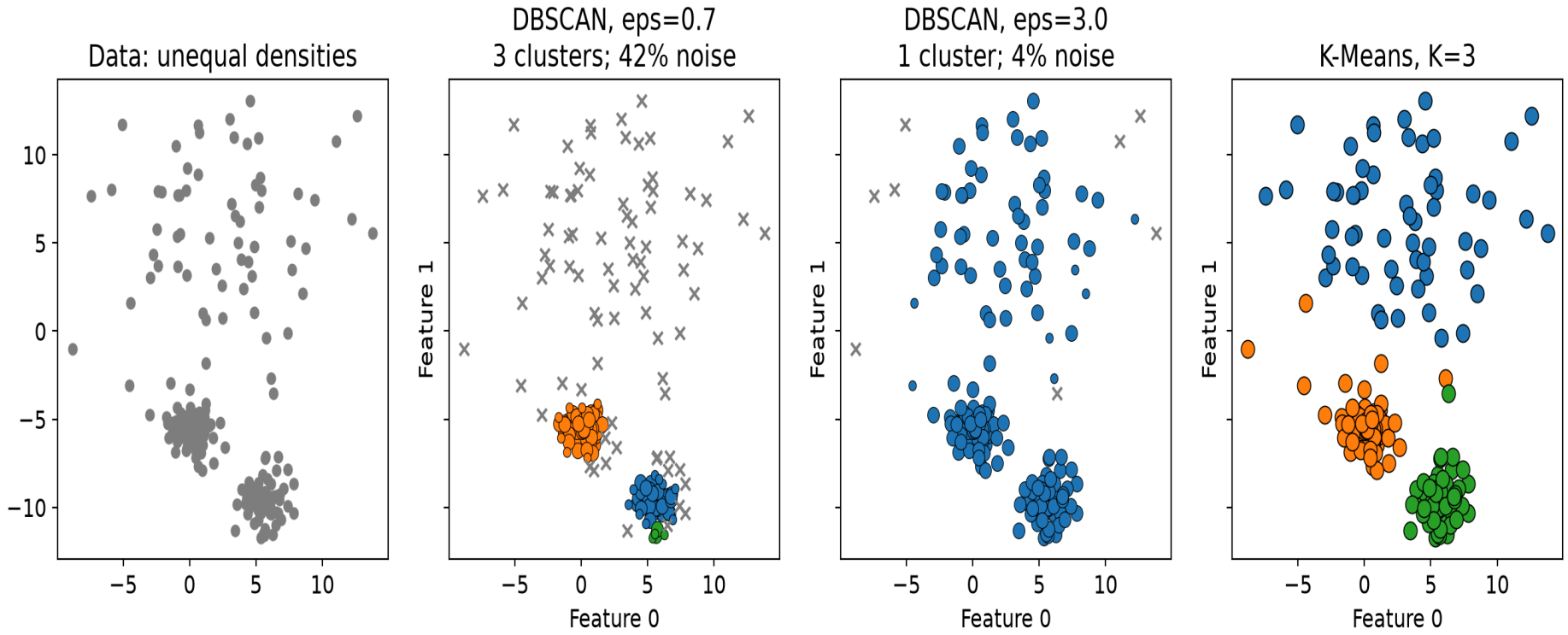
How should we tune DBSCAN?

1. Choose a meaningful **representation, scaling, and metric**.
2. Try a small grid of `eps` and `min_samples` values.
3. Inspect **cluster sizes, noise fraction, and representative examples**.
4. Check whether useful interpretations persist at nearby settings.

Can we use our usual grid search or random search?

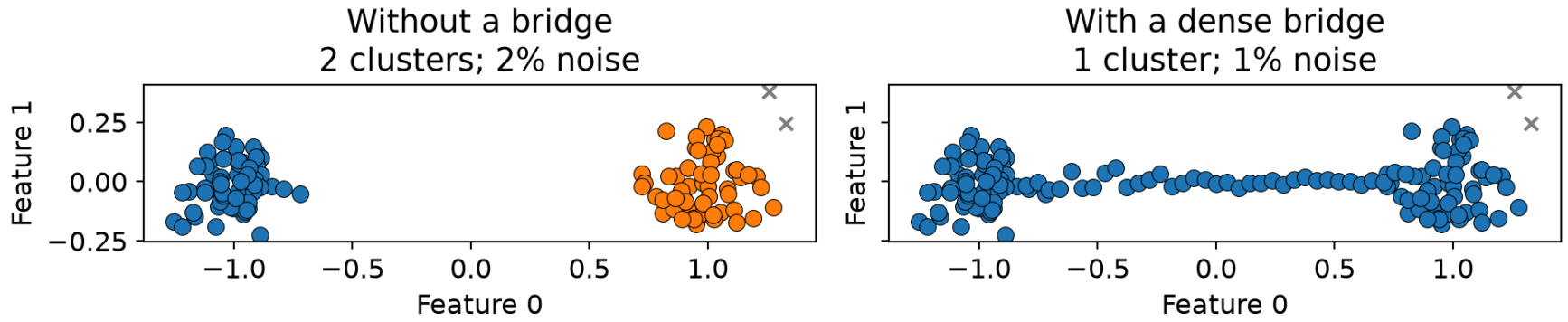
One density threshold may not suit every group

A small radius can discard a sparse group; a large radius can merge distinct groups.



A dense bridge can join groups

DBSCAN follows **core-point connections**, even when we would prefer two groups.



What if we want nested groups?

Agglomerative hierarchical clustering

Start with **one cluster per example**.

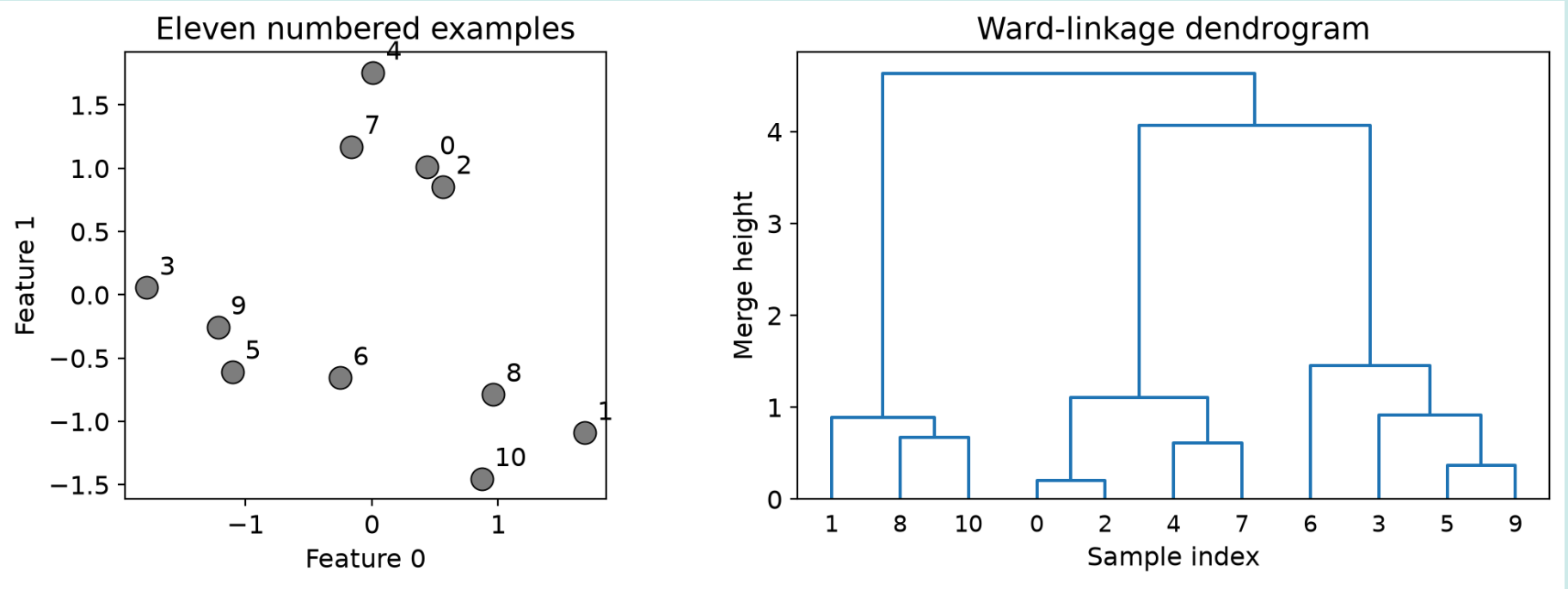
1. Merge the two closest clusters.
2. Recalculate which clusters are closest.
3. Repeat until all examples form one cluster.

After $n - 1$ merges, we have a **hierarchy of nested groups**.

Merges are not undone. We can choose a level of detail afterward.

Your turn: read the merge sequence

Find the first pair that merges. Where are those points in the scatter plot?



Reading a dendrogram

- **Leaves:** individual examples, identified by original row number.
- **Connections:** merges of two points or groups.
- **Height:** merge dissimilarity under the chosen linkage rule.

Horizontal spacing **does not measure similarity**. Branches can rotate.

Large vertical gaps suggest possible cuts, rather than proving a correct number of groups.

What does “closest clusters” mean?

A **distance metric** compares points. A **linkage criterion** compares clusters.

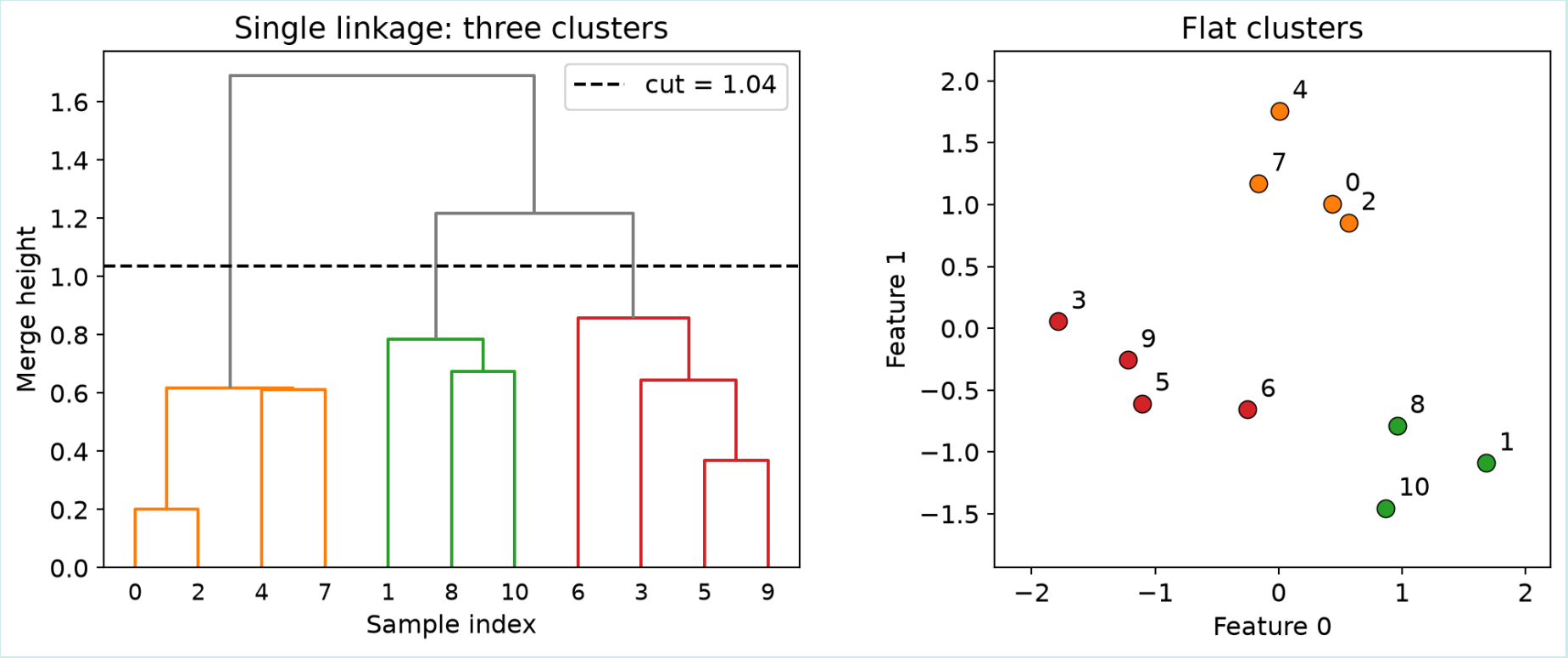
Linkage	**Merge the pair with the smallest…**
Single	Minimum cross-cluster point distance
Complete	Maximum cross-cluster point distance
Average	Average cross-cluster point distance
Ward	Increase in total within-cluster squared deviation

Ward requires Euclidean distance and favours compact groups. It does not enforce equal sizes.

Example: single linkage, three clusters

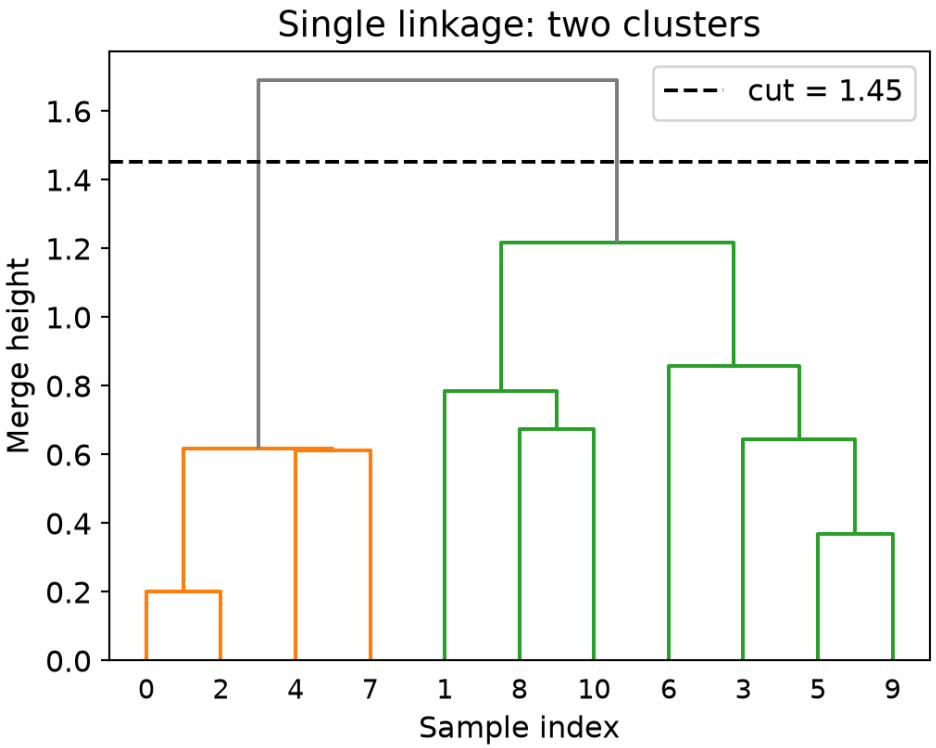
To go from **three clusters to two**, which clusters would you merge?

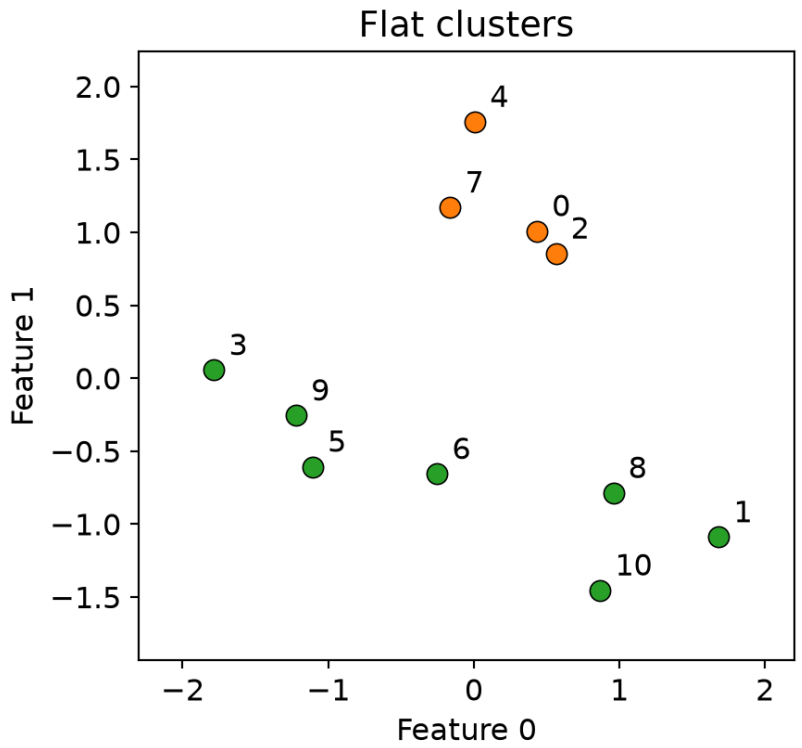
Single linkage compares the **closest pair of points across each pair of clusters**.



Example: single linkage, two clusters

Points **6 and 8** are the closest cross-cluster pair: merge their two clusters.

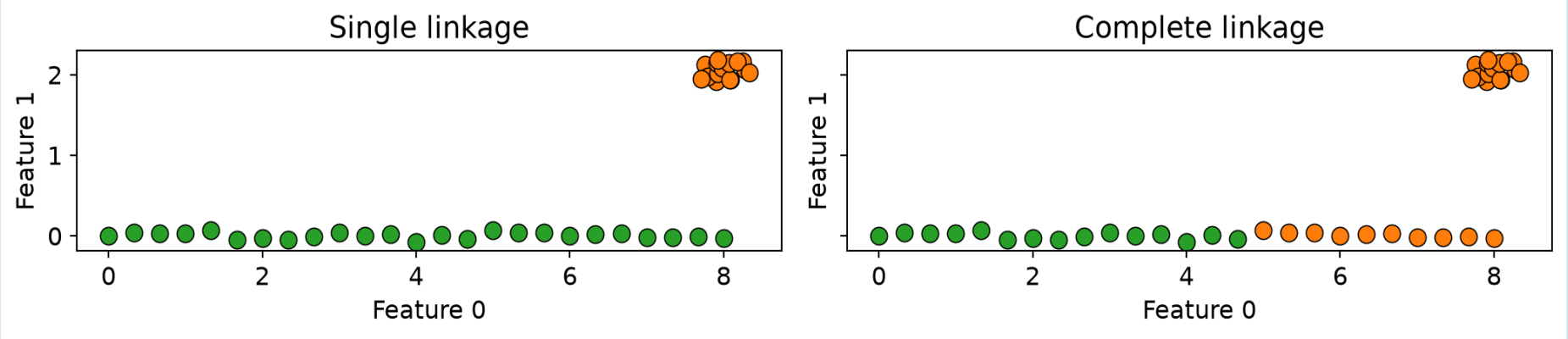




The **same hierarchy** gives two clusters when we raise the cut past the next merge.

Your turn: does linkage change the result?

Same data, Euclidean distance, and **two clusters**. Which rule keeps the long chain together?



- **Single:** one nearby pair can join groups → loose, elongated clusters.
- **Complete:** the farthest pair determines closeness → tighter clusters.

Distance and linkage are separate choices

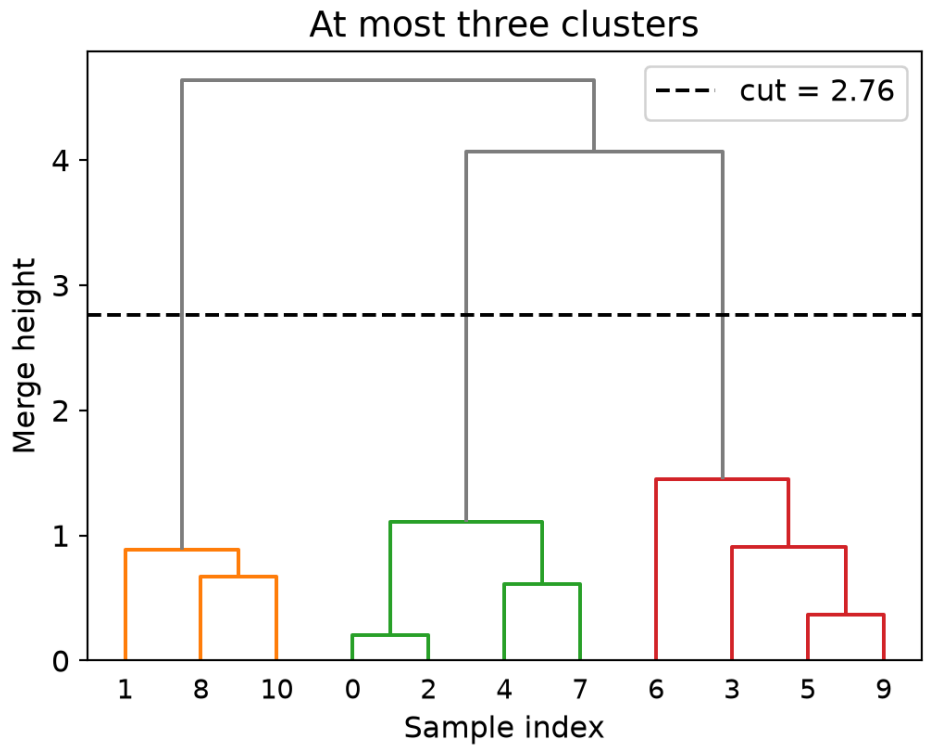
```
1 Z = linkage(X, method="complete", metric="cosine")
```

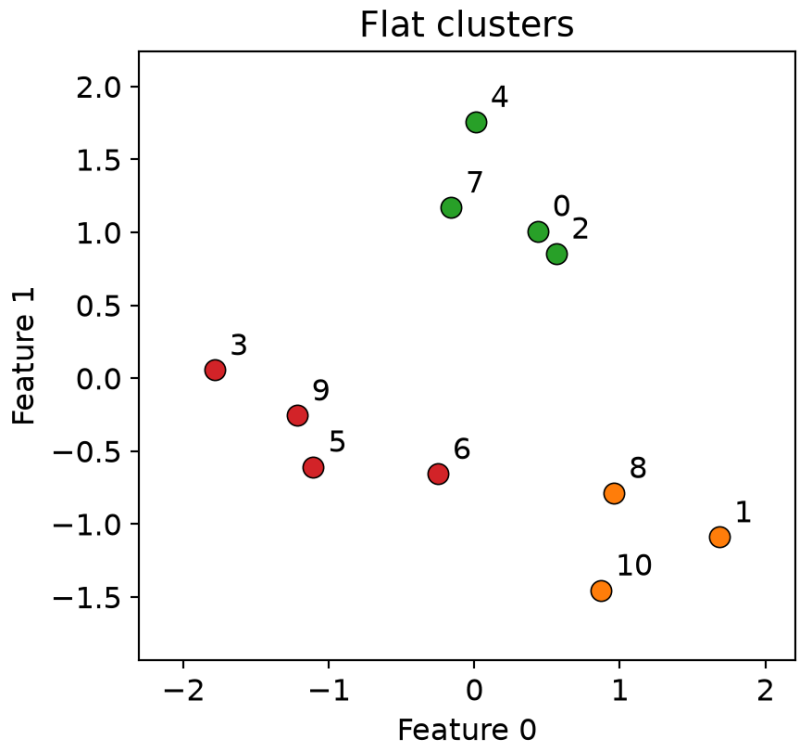
- **Cosine:** dissimilarity between individual feature vectors.
- **Complete:** compare clusters using their farthest pair.

Choose the metric for the representation. **Ward cannot use cosine distance.**

From the tree to flat cluster labels

Cut horizontally: each connected group below the cut becomes a cluster.





Two ways to cut the same hierarchy

```
# At most three clusters
labels = fcluster(Z, t=3, criterion="maxclust")

# A merge-height threshold
labels = fcluster(Z, t=0.7, criterion="distance")
```

Both reuse **the same fitted hierarchy `Z`**.

Changing dendrogram colours alone does not extract labels.

Optional: a smaller display

```
1 dendrogram(Z, truncate_mode="lastp", p=8)
```

- Truncation collapses branches **for display**.
- All observations still contribute to the fitted hierarchy.
- It does **not** reduce fitting cost; standard methods often need roughly quadratic memory.

iClicker 15.2: interpreting clusters

Select all statements that are true.

- a. Hierarchical clustering avoids choosing random initial centres.
- b. Truncating a dendrogram reduces the memory needed to fit the hierarchy.
- c. DBSCAN’s cluster count emerges from its density settings.
- d. We can combine clustering results by taking a majority vote of their numeric labels.
- e. A high silhouette score proves that clusters capture the meaning we care about.

Which grouping is useful?

Your turn: compare the methods

With a neighbour: fill in the table and give one reason for each choice.

Consideration	K-Means	DBSCAN	Hierarchical
Meaning of a group	?	?	?
Main choices	?	?	?
Noise handling	?	?	?
Assign new points with `predict`?	?	?	?

Evaluating clusters: evidence beyond one score

- Inspect representative members **and exceptions** in each group.
- Check nearby parameter settings and reasonable preprocessing changes.
- Use silhouette as supporting evidence; it can undervalue curved groups.
- For DBSCAN, report the **noise fraction** alongside any score.

Stable, separated groups can still reflect the wrong similarity.

Class demo: inspect and explain groups

Compare density settings and hierarchy cuts. For each result:

1. Describe what members share and inspect exceptions.
2. Explain what changed when the settings changed.
3. Decide whether the grouping serves the task.

Today's takeaways

- **DBSCAN:** core points extend density-connected groups; border points join; noise remains unassigned.
- **Density settings:** tune together and inspect noise, stability, and examples. Varying densities and bridges can cause trouble.
- **Hierarchical clustering:** repeated merging gives nested groups; linkage defines closeness and a cut gives flat labels.
- **Representation defines similarity:** useful groups require interpretation, beyond a numerical score.

Exit question

Two DBSCAN settings both produce three clusters.

One marks half the dataset as noise; the other marks almost none.

What would you inspect before choosing, and why is cluster count alone insufficient?

Happy Halloween

