Clustering: Hierarchical clustering

Tom Edinburgh

te269

Example classes and lecture notes

- There are 2 more small group classes, this week and the following week

- You should know the group/time/room, please email me (te269) or Steven (stg20) if not

- This week's problem set: decision trees

  - Chapter 8 of Introduction to Statistical Learning with Python

  - Questions 4, 5, 9, 10 (extension Q12, ignore BART)

  - Pdf of the book and .csv files available at https://www.statlearning.com/

- Notes covering the first half of the course will be online on the Moodle soon

Clustering schedule

1. k-means

2. Hierarchical clustering

3. Gaussian mixture models, self-organising maps, spectral clustering

4. Graph-based clustering, density-based clustering, outlier detection

5. Clustering evaluation, hyperparameters, pros and cons, consensus clustering

Today: hierarchical clustering

- Overview

- Hierarchical agglomerative clustering

- Linkage

- Code example (k-means and hierarchical clustering)

- Vector quantisation

Questions: halfway through, at the end, or by email (te269)

Resources

- Introduction to Statistical Learning with Python, Chapter 12

- Slides adapted from:

  - Prof Stephen Eglen, Cambridge

  - Ethan Fetaya/James Lucas/Emad Andrews, Toronto

  - Ronan Cummins, Cambridge

Overview: hierarchical clustering

- Build a hierarchy of clusters

- Small clusters have parents

- Agglomerative (bottom-up) vs divisive (top-down)

- Some similarities and some differences to decision trees (supervised learning)

- We don't even really need to know the observations themselves, a matrix of distances is enough

- Hierarchical clustering results in a sequence of clusters

One cluster per observation ⟵⟶ One cluster with all observations

Hierarchical agglomerative clustering

- Start with each observation in its own separate cluster

- Merge the two clusters that are the most similar

- Repeat until there is only one cluster with all observations (greedy)

- The history of how clusters merged forms a binary tree called a dendrogram

- We need a way to describe the similarity between clusters
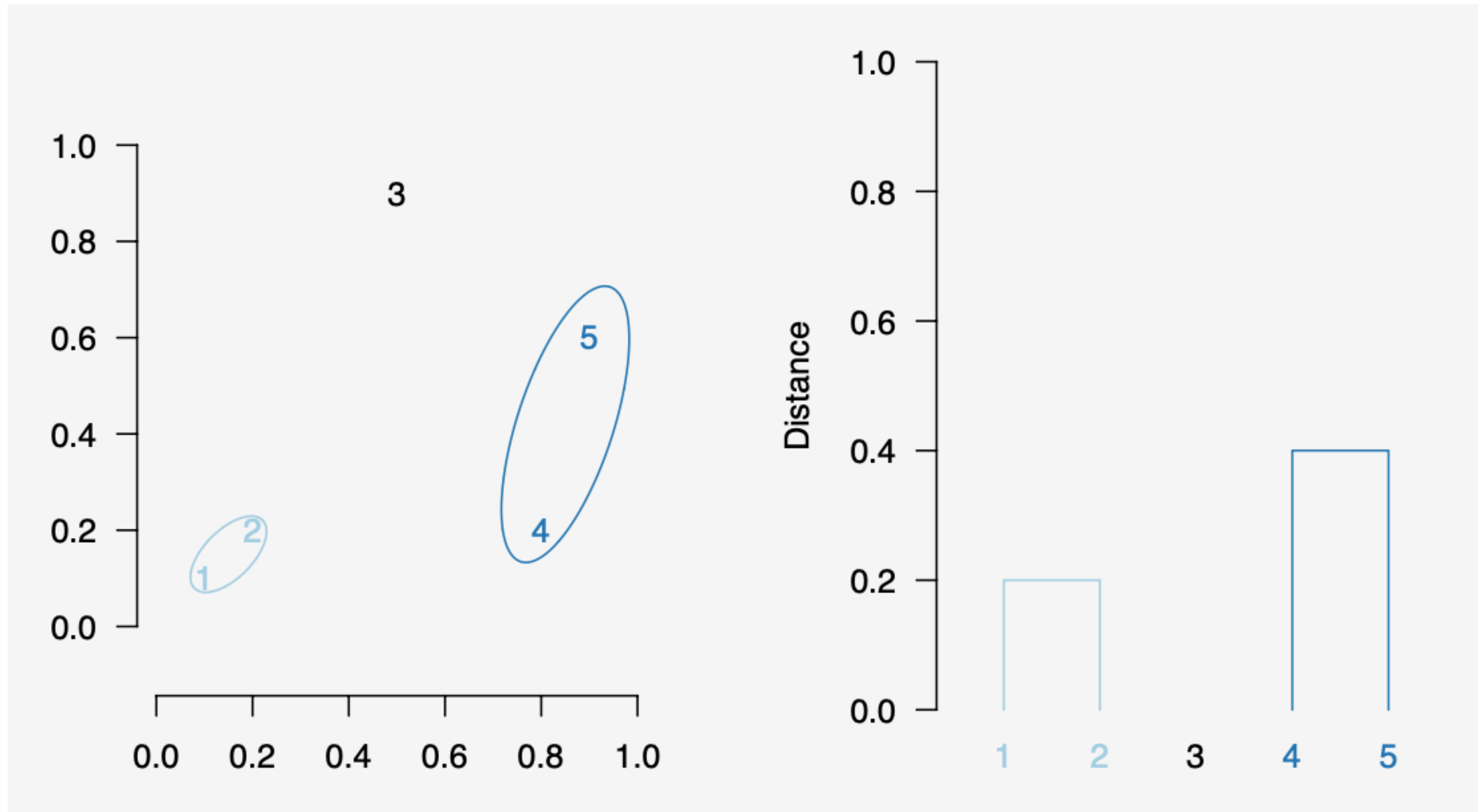
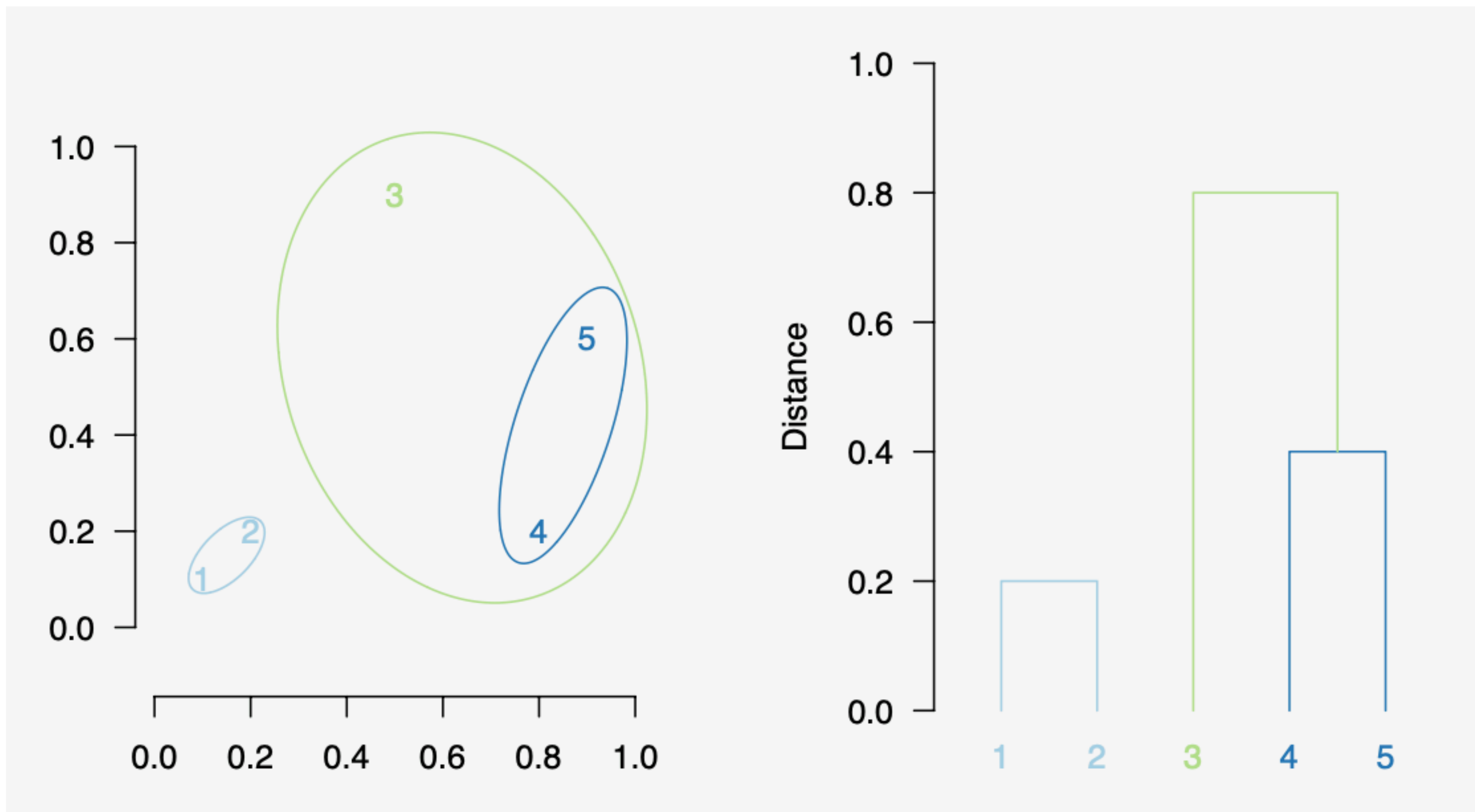 - Lots of similarity/dissimilarity measures
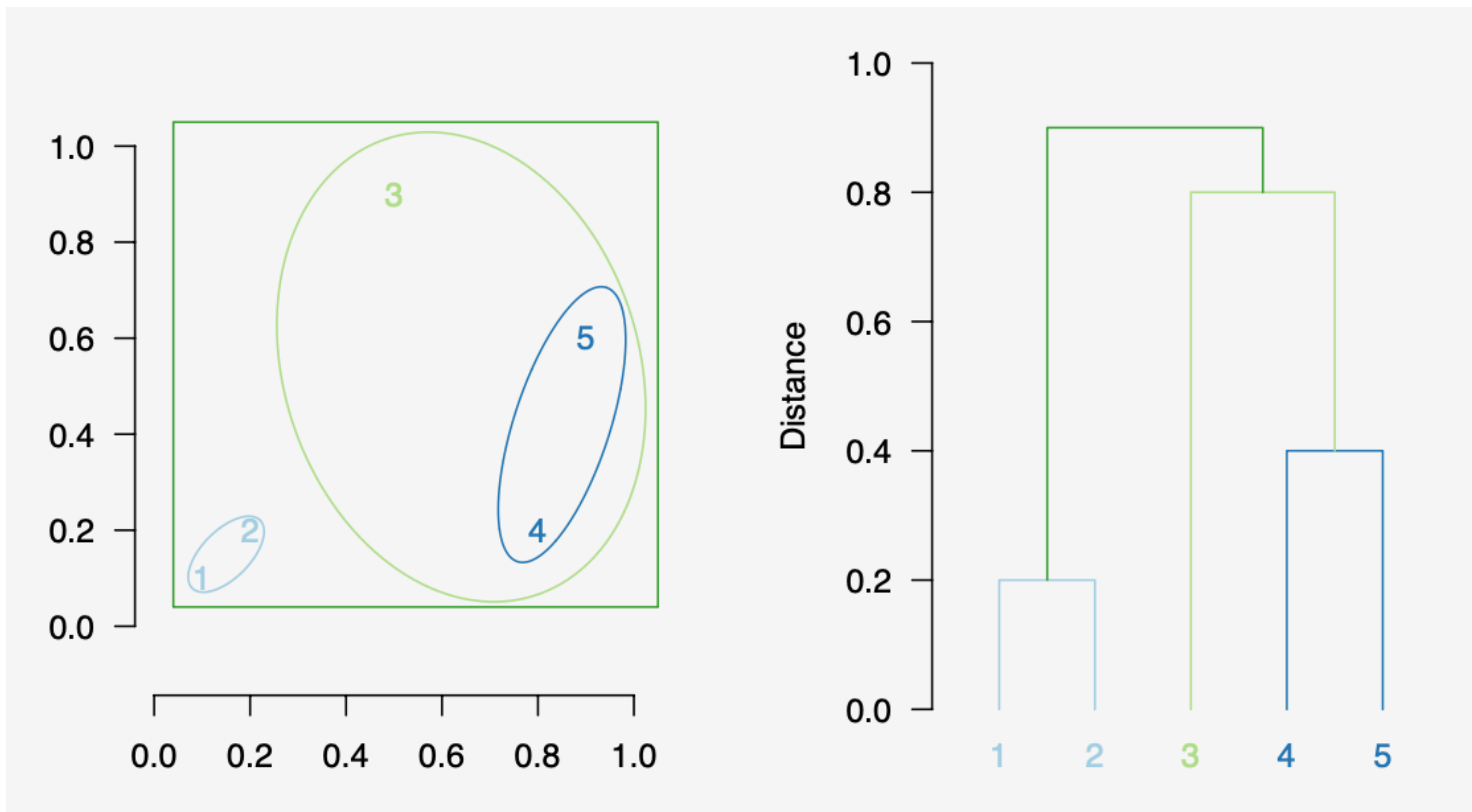
Example
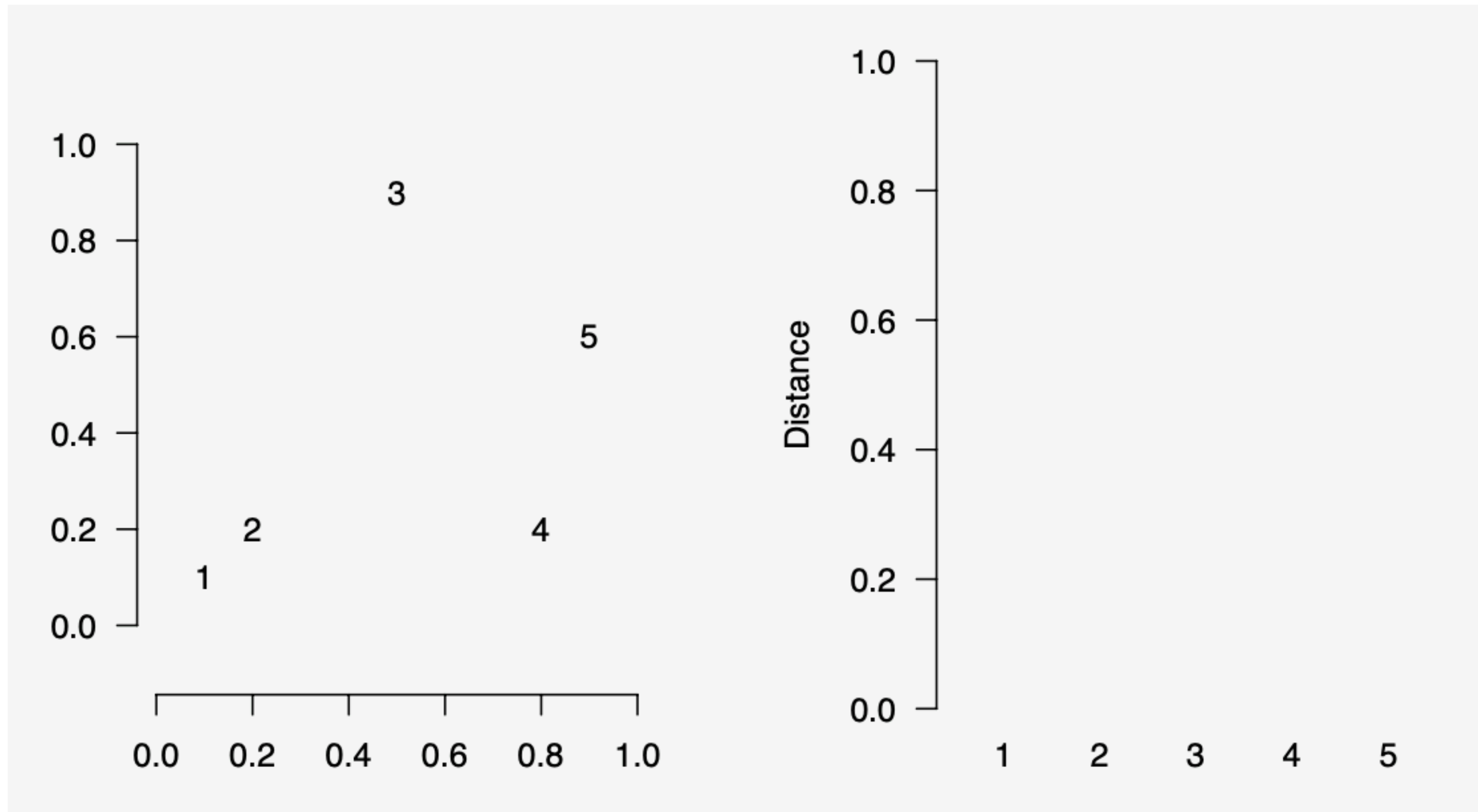
Example
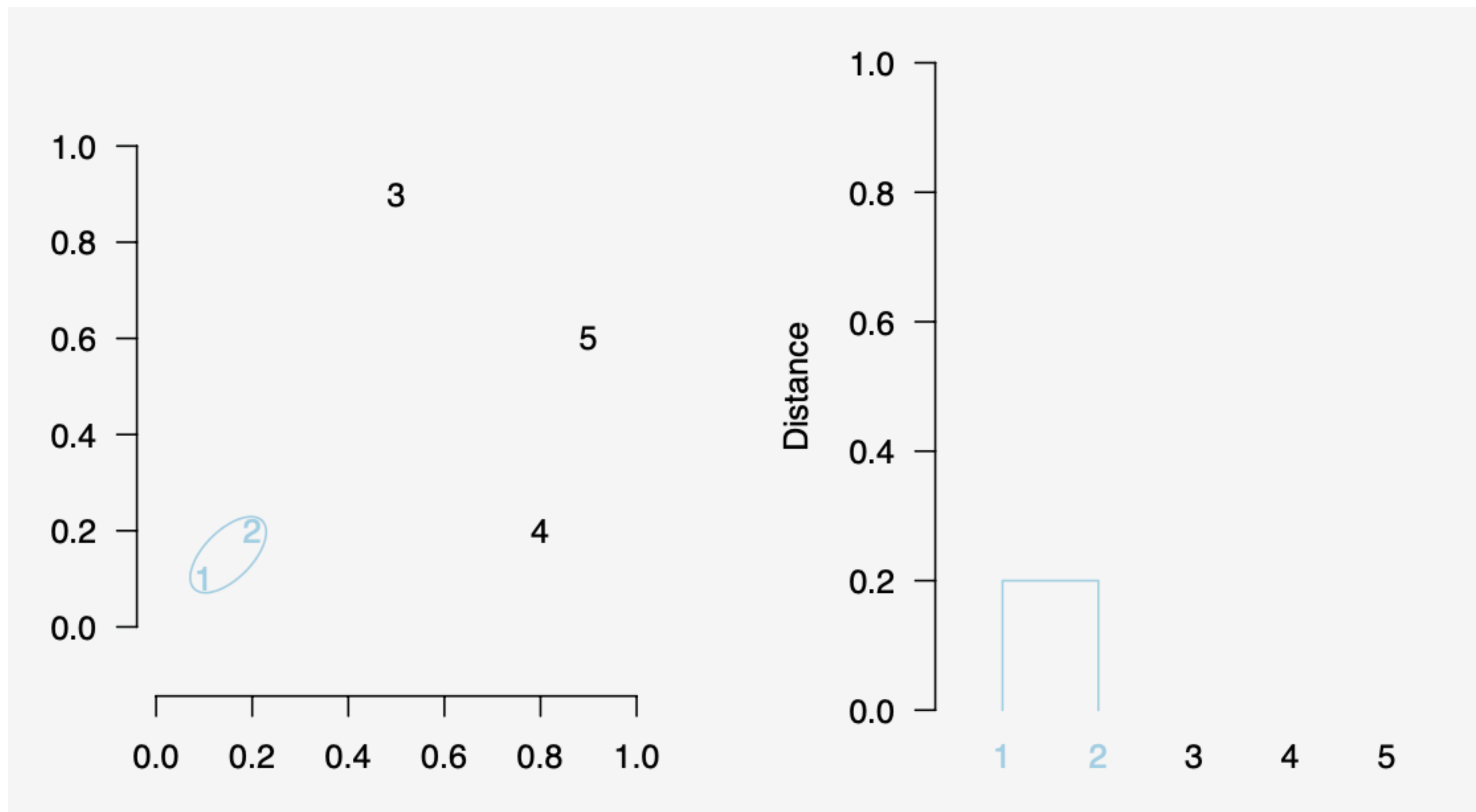
Example

Example

Example

Linkage

- The linkage is a similarity function between clusters (not observations)

- Lots of options, e.g.

  - Single linkage: distance between closest observations in each cluster

  - Complete linkage: distance between furthest observations in each cluster

  - Average linkage: average distance between each pair of observations

  - Centroid linkage: distance between cluster centroids

  - Ward linkage: minimum increase in sum of squares distance from centroids

Single linkage, complete linkage and average linkage

- Notation: cluster set A contains $n_A = |A|$ and has centroid $c_A = 1/n_A \sum_{x \in A} x$. Then:

  - Single linkage: $d(A, B) = \min_{x \in A, y \in B} d(x, y)$

  - Complete linkage: $d(A, B) = \max_{x \in A, y \in B} d(x, y)$

  - Average linkage: $d(A, B) = 1/(n_A n_B) \sum_{x \in A} \sum_{y \in B} d(x, y)$

Centroid linkage and Ward linkage

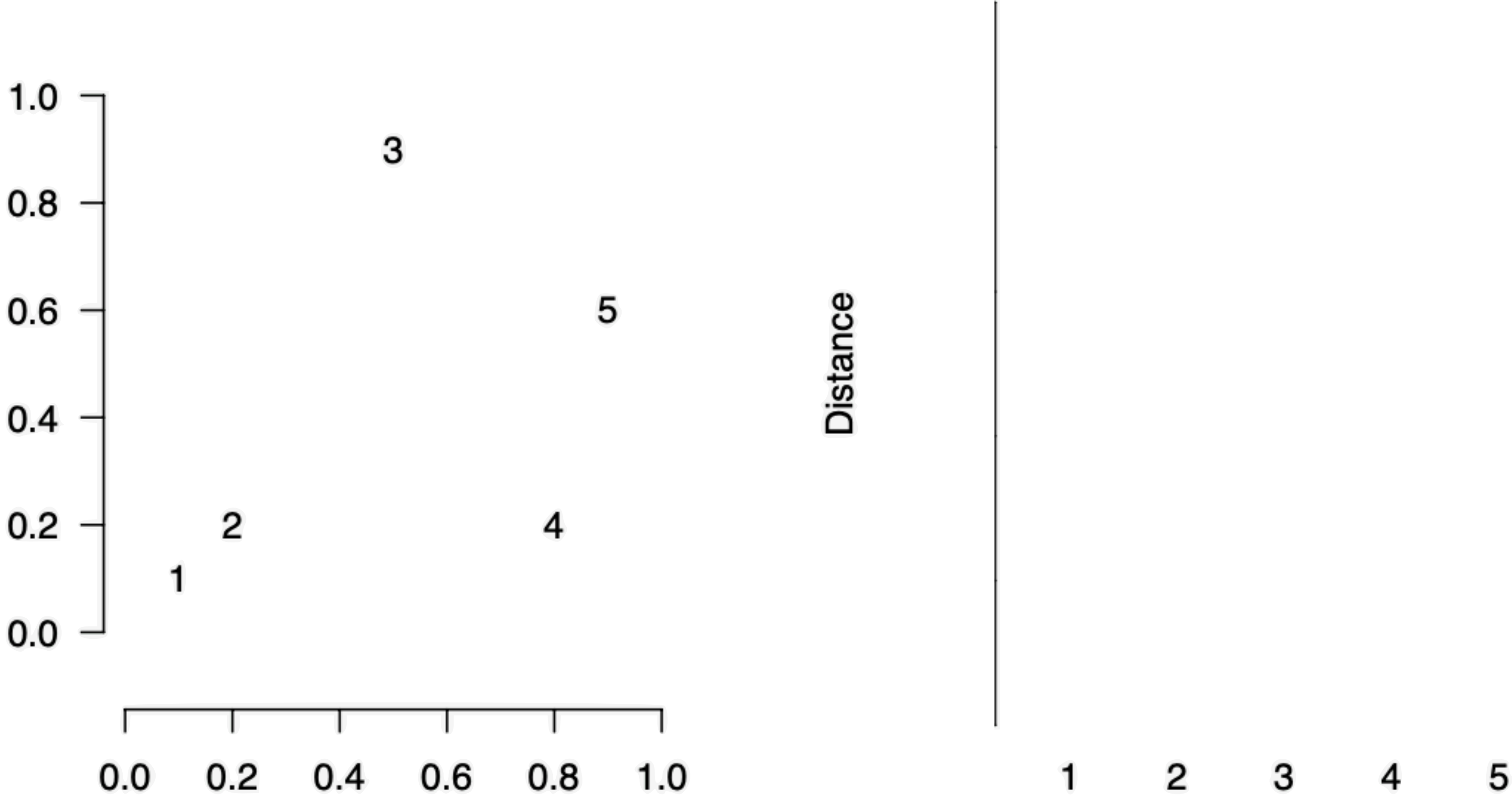- Notation: cluster set A contains $n_A = |A|$ and has centroid

$$c_A = 1/n_A \sum_{x \in A} x. \text{ Then:}$$

- Centroid linkage: $d(A, B) = \|c_A - c_B\|^2$
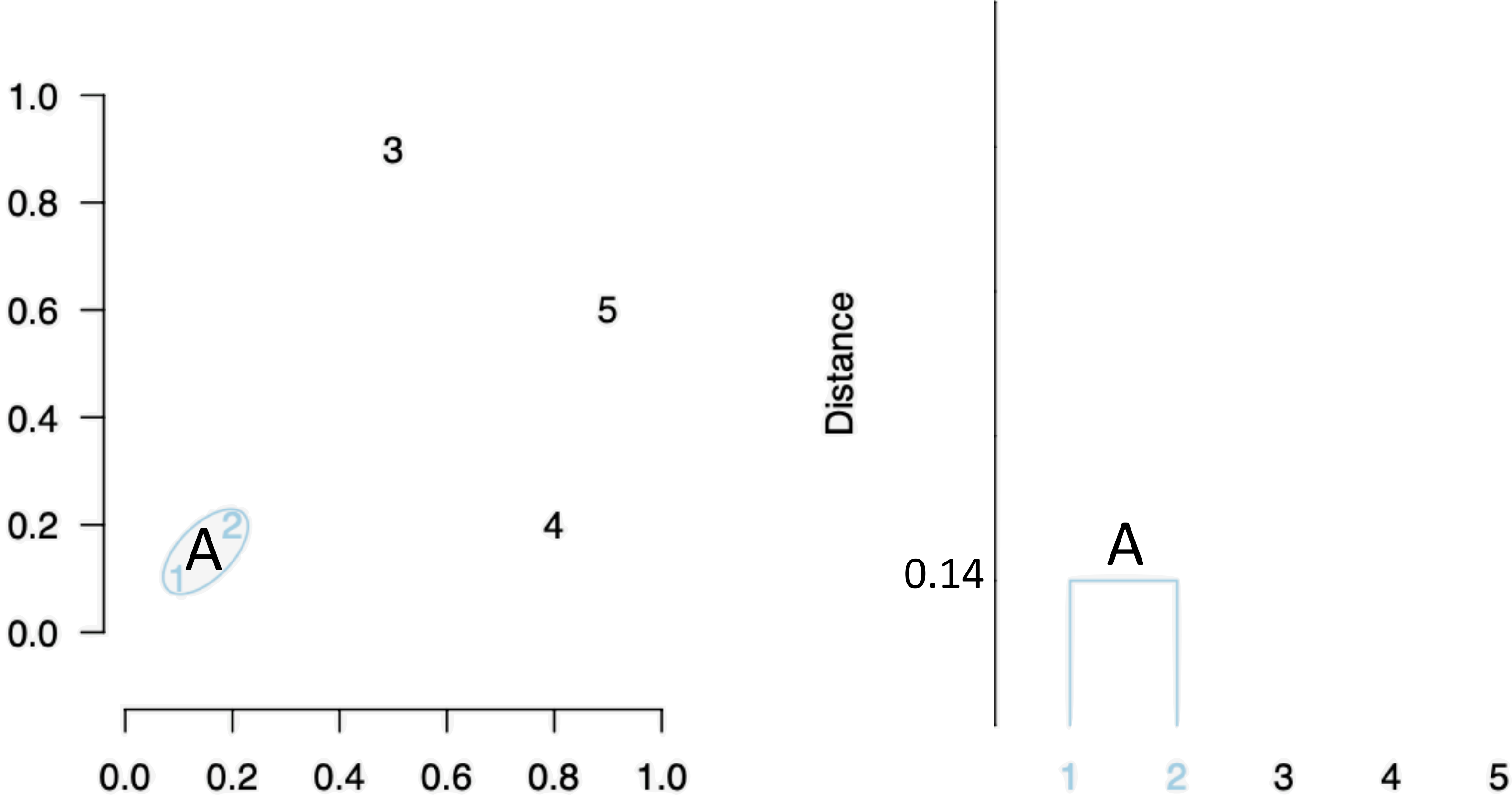
- Ward linkage:

$$d(A, B) = \sum_{x \in A \cup B} \|x - c_{A \cup B}\|^2 - \sum_{x \in A} \|x - c_A\|^2 - \sum_{x \in B} \|x - c_B\|^2$$

Example (centroid linkage)

d	2	3	4	5
1	0.14	0.89	0.71	0.94
2		0.76	0.60	0.81
3			0.76	0.50
4				0.41

Example (centroid linkage)

d	2	3	4	5
1	0.14	0.89	0.71	0.94
2		0.76	0.60	0.81
3			0.76	0.50
4				0.41

d	3	4	5
A	0.83	0.65	0.87
3		0.76	0.50
4			0.41

Example (centroid linkage)
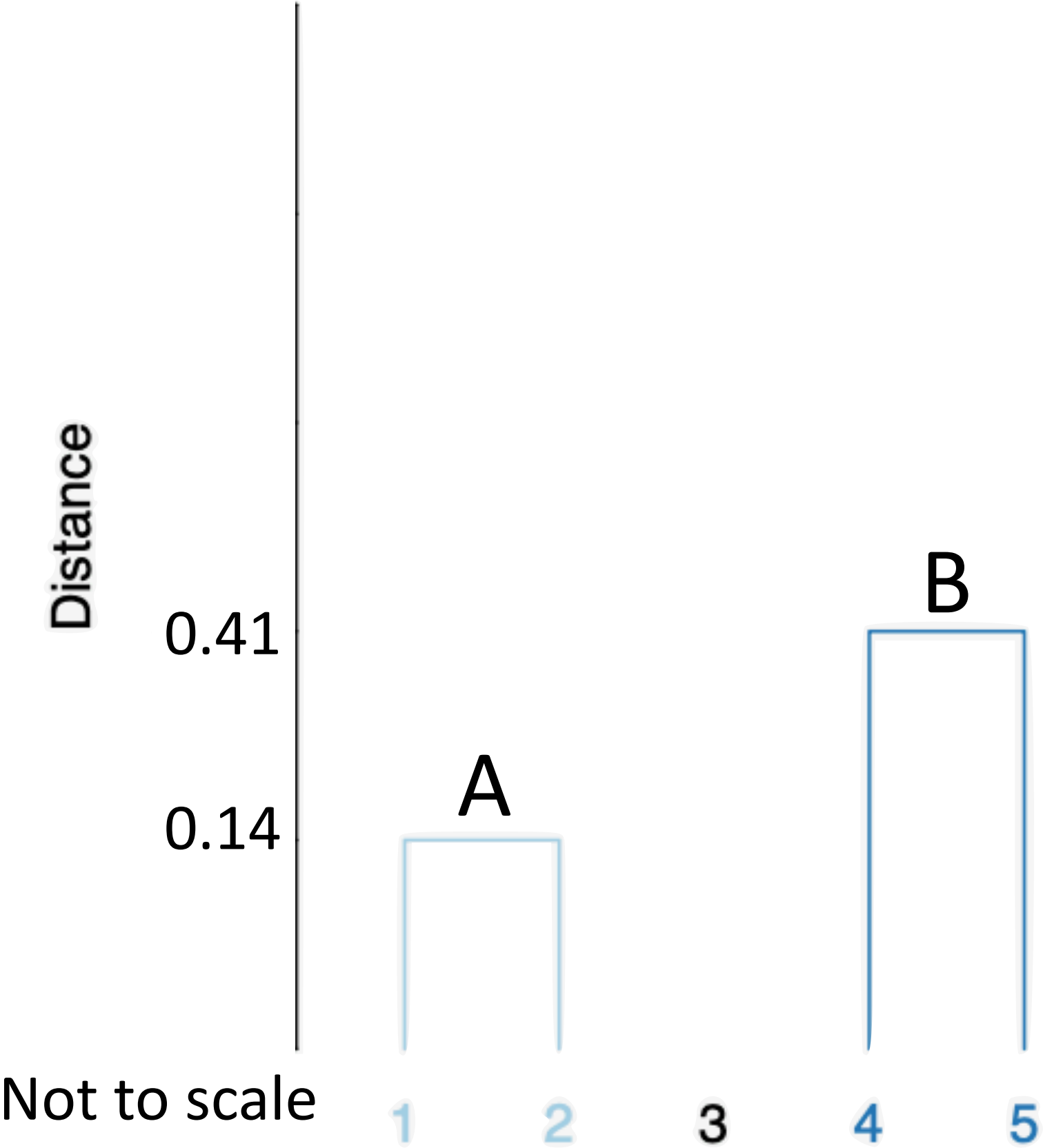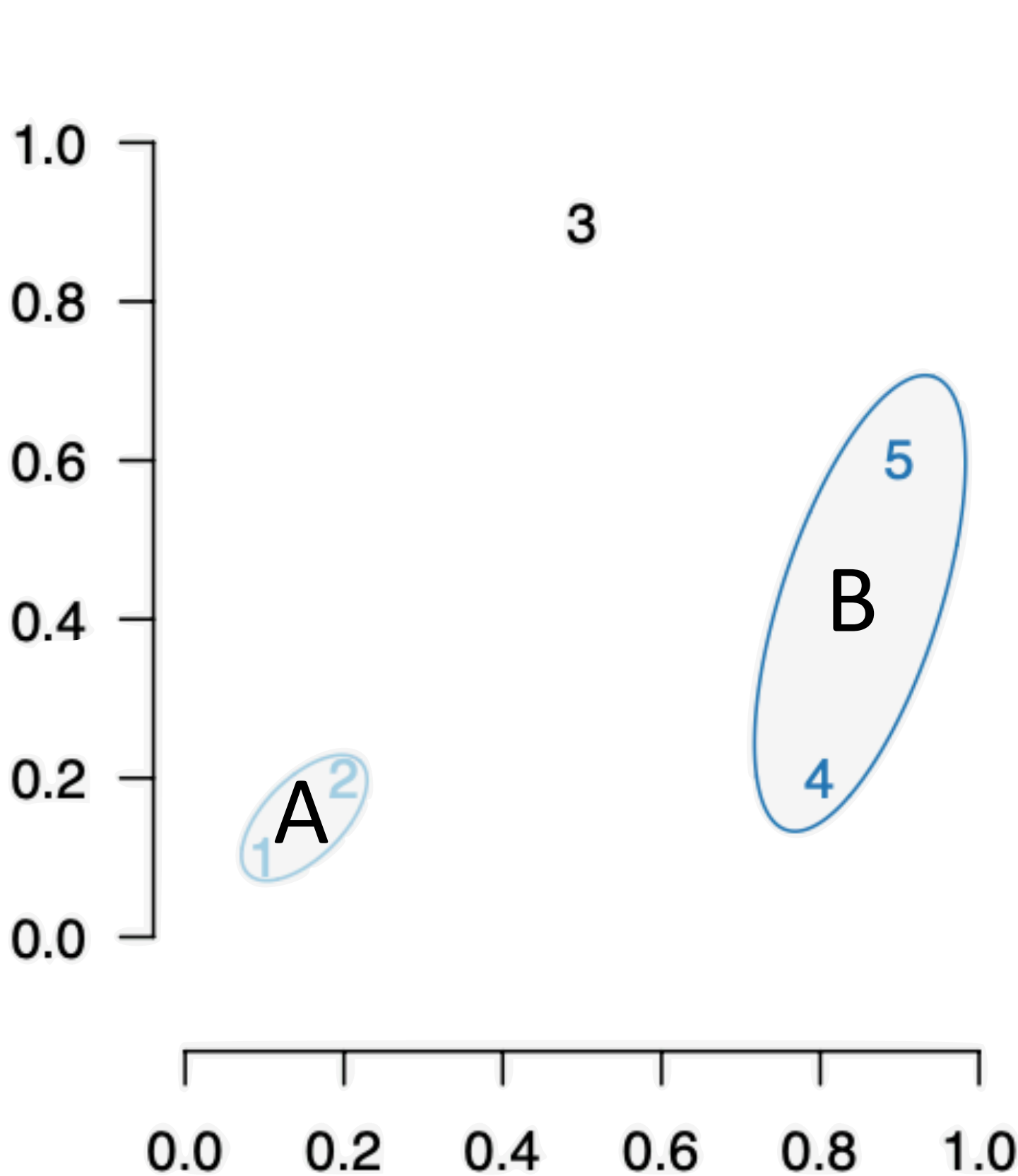
d	2	3	4	5
1	0.14	0.89	0.71	0.94
2		0.76	0.60	0.81
3			0.76	0.50
4				0.41

d	3	4	5
A	0.83	0.65	0.87
3		0.76	0.50
4			0.41

d	3	B
A	0.83	0.74
3		0.61

Example (centroid linkage)

d	2	3	4	5
1	0.14	0.89	0.71	0.94
2		0.76	0.60	0.81
3			0.76	0.50
4				0.41

d	3	4	5
A	0.83	0.65	0.87
3		0.76	0.50
4			0.41

d	3	B
A	0.83	0.74
3		0.61

d	C
A	0.72

Not to scale

Example (centroid linkage)

d	2	3	4	5
1	0.14	0.89	0.71	0.94
2		0.76	0.60	0.81
3			0.76	0.50
4				0.41

d	3	4	5
A	0.83	0.65	0.87
3		0.76	0.50
4			0.41

d	3	B
A	0.83	0.74
3		0.61

d	C
A	0.72

Distance

Not to scale
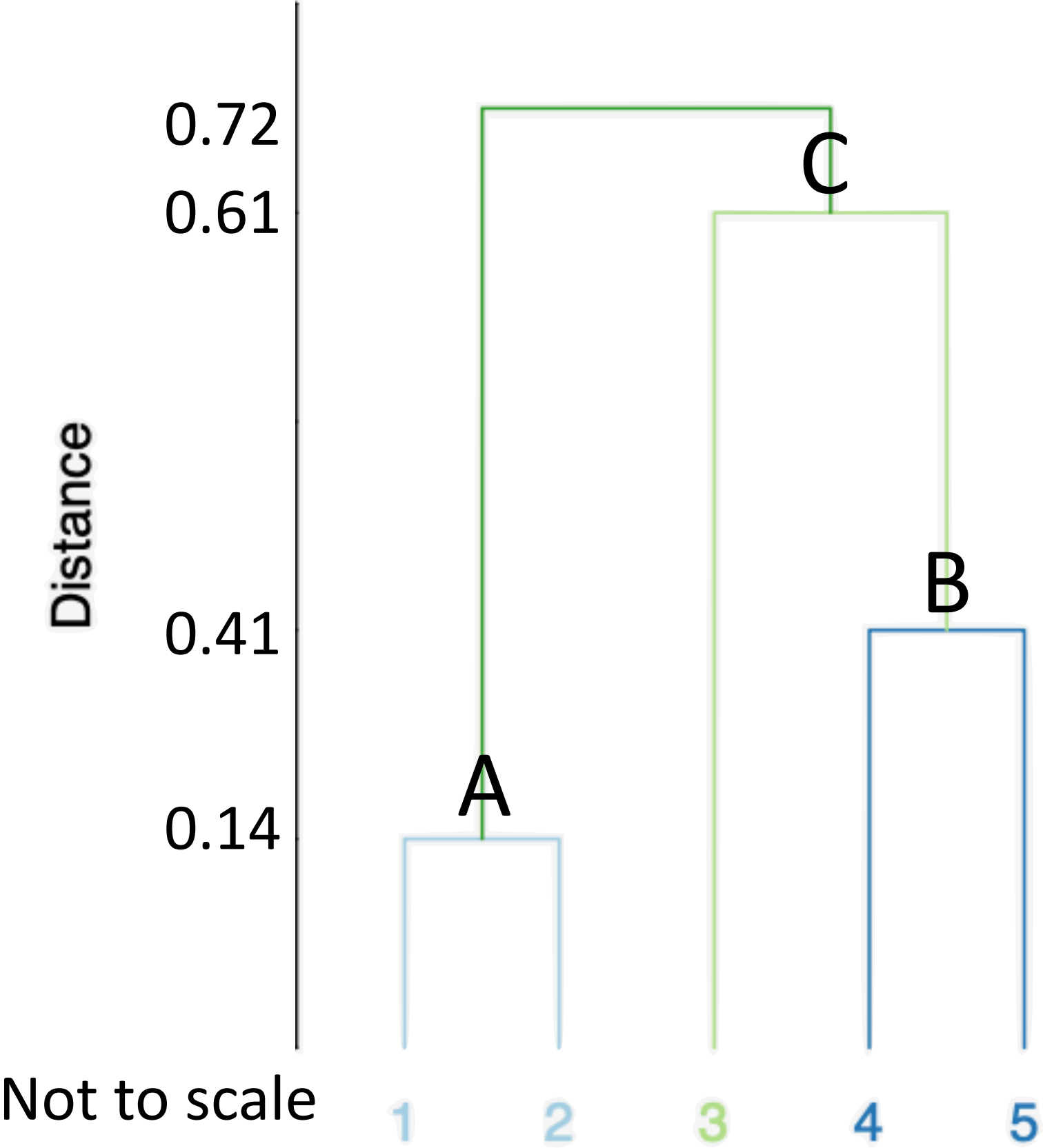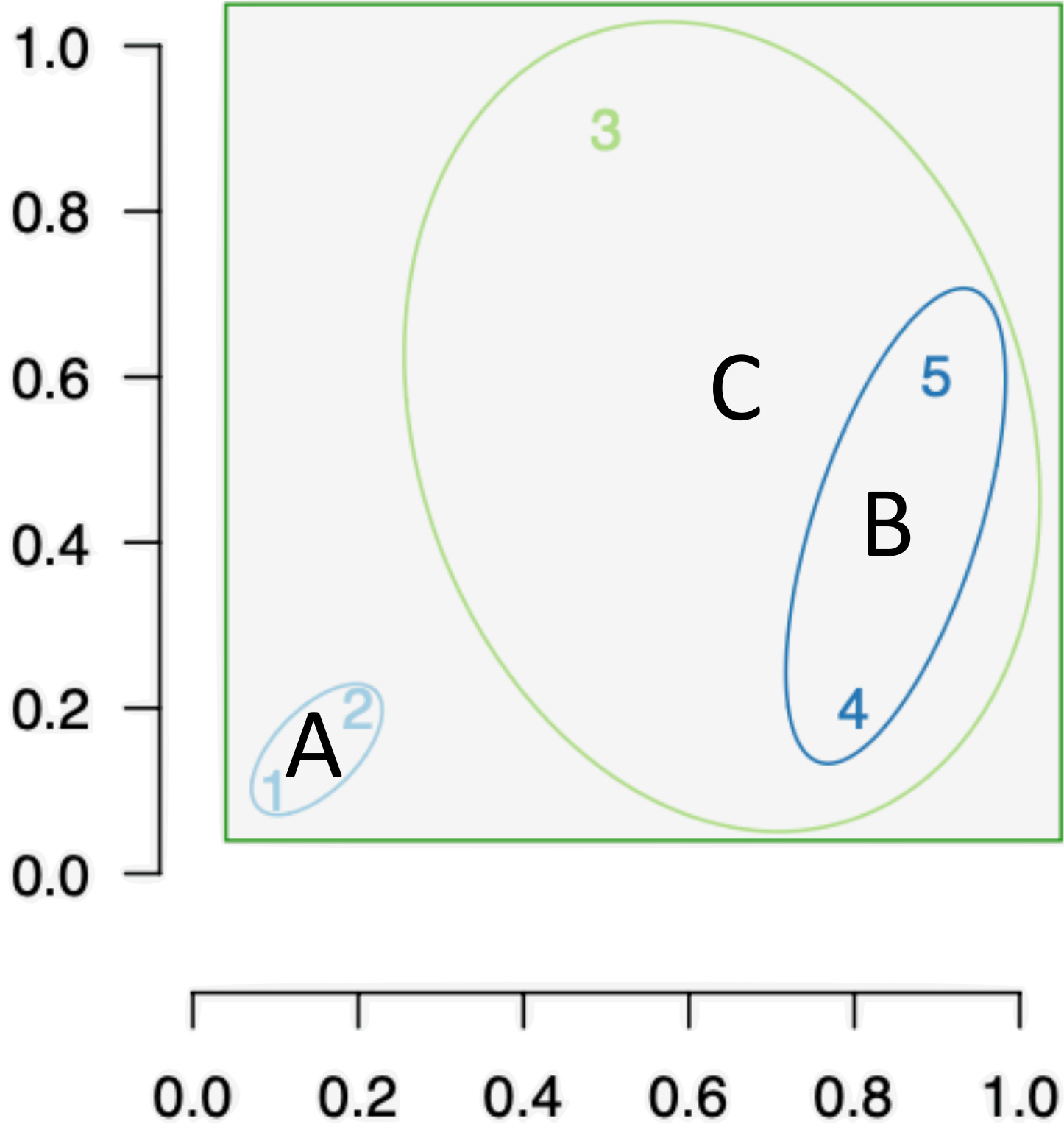
Linkage: some less common options

- Weighted average linkage: $d(A, B \cup C) = (d(A, B) + d(B, C))/2$

- Minimum increase in variance: $d(A, B) = \text{var}(A \cup B) - \text{var}(A) - \text{var}(B)$,
$$\text{var}(A) = 1/n_A \sum_{x \in A} \|x - c_A\|^2$$

- Mini-max linkage: $d(A, B) = \min_{x \in A \cup B}(\max_{y \in A \cup B} d(x, y))$

- Hausdorff linkage:
$$d(A, B) = \max(\max_{x \in A} \min_{y \in B} d(x, y), \max_{y \in B} \min_{x \in A} d(x, y))$$

Hierarchical Clustering With Prototypes via Minimax Linkage

Jacob Bien, Ph.D. Student and Robert Tibshirani, Professor

Questions?

Code example

- Jupyter notebook about hierarchical clustering

- On GitHub (https://github.com/tedinburgh/ads2023), Moodle, and course GitLab

Dendrogram

- How should we plot the dendrogram? For the same sequence of clusters, there are lots of orderings along the x-axis (2^{n-1} orderings for n observations)

- Nearby leaves are not necessarily similar

- Can affect visualisation/interpretation!

- There are some heuristics for this

  - GW method

  - Optimal leaf ordering (OLO)
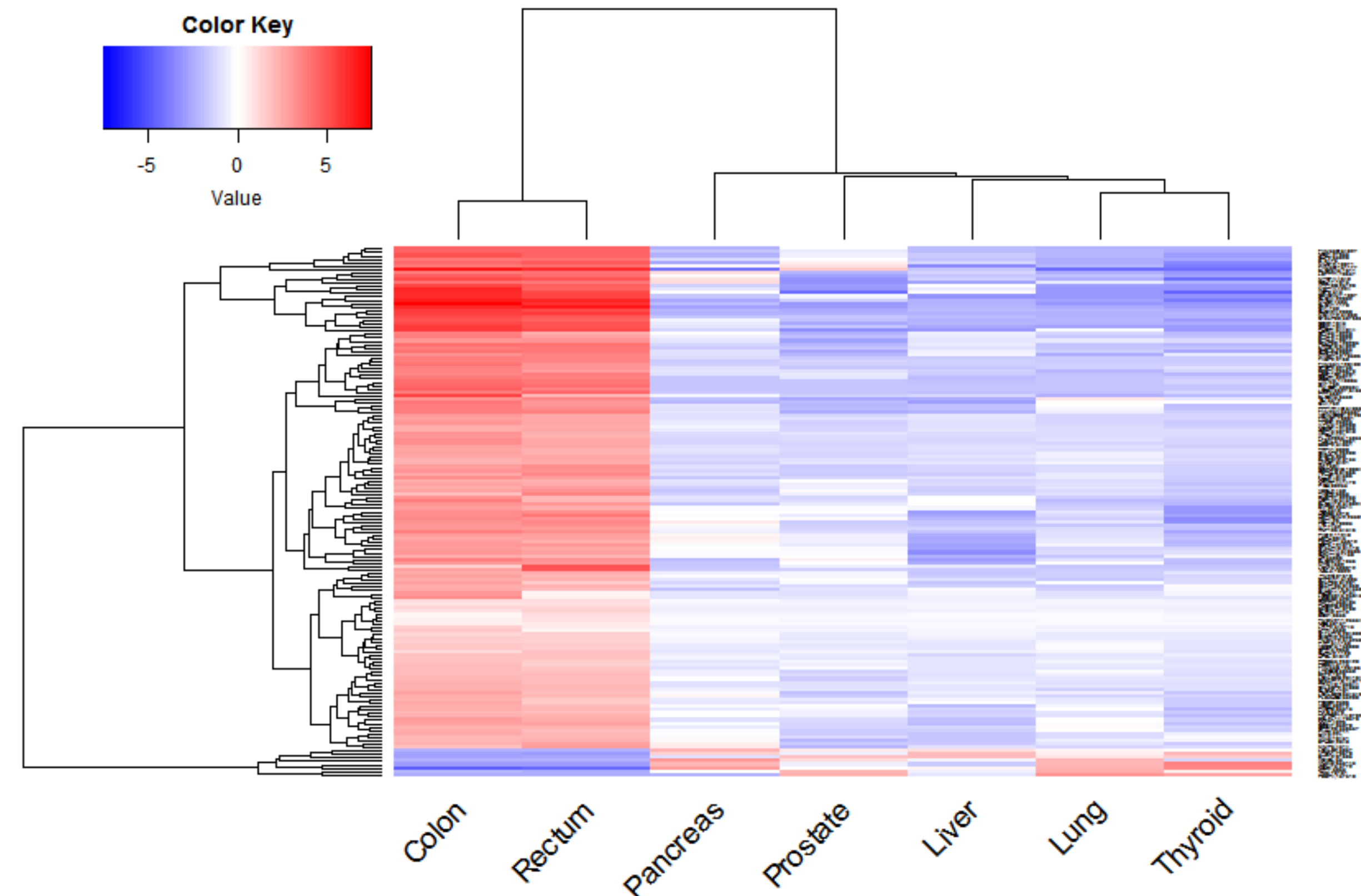
Hierarchical divisive clustering

- Basic algorithm is DIANA (divisive analysis clustering)

- Start with all data in the same cluster

- At each iteration, choose one existing cluster (with the largest diameter) and build a new 'splinter group' nested cluster inside it

- Observations within the larger cluster can migrate to the new nested cluster

- The existing cluster gets 'hollowed out' as loose observations within it move into nested clusters (as more nested clusters are created)

- The nested 'splinter group' clusters are children of the existing cluster (compared to creating parent clusters for agglomerative clustering)

Computational complexity

- Calculate the distance between each pair of observations ($n \times n$), so $O(n^2)$, which is expensive if there are a lot of observations

- Single linkage: for n iterations, find the minimum distance (sort operation, $O(n \log n)$), overall $O(n^2) + O(n \times n \log n) = O(n^2 \log n)$

- Exhaustive search e.g. for divisive clustering $O(2^n)$

- Optimised algorithms e.g. SLINK/CLINK are $O(n^2)$ (doesn't calculate distance matrix)

Single linkage and minimum spanning trees

- The naïve algorithmic approach for single linkage clustering is essentially the same as Kruskal's algorithm for minimum spanning trees

- This finds the minimum distance connecting all vertices of an undirected edge-weighted graph

- Faster algorithms for single-linkage HAC correspond to more efficient algorithms for MSTs
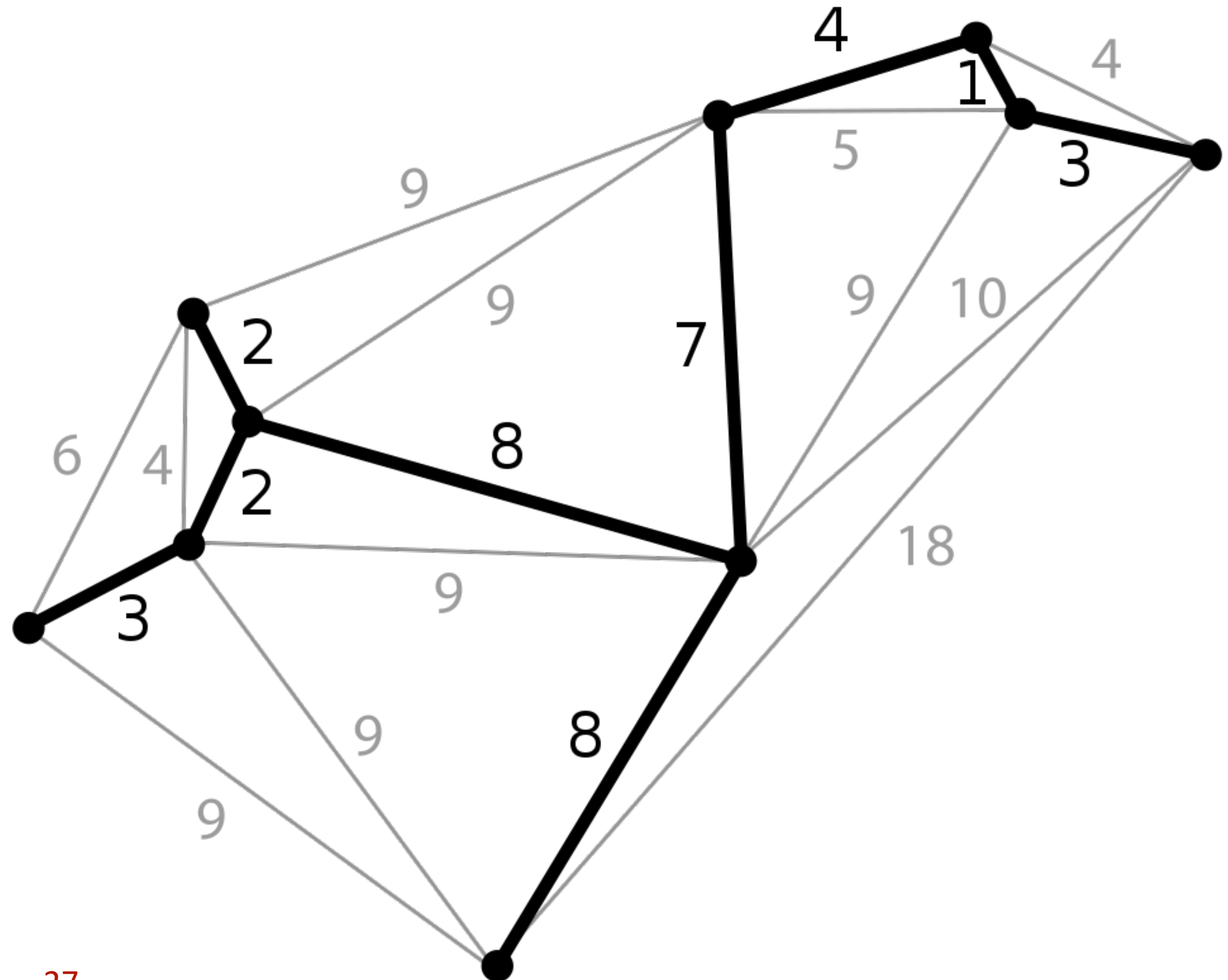
Vector quantisation

- Split a high-dimensional input (e.g. an image) into smaller chunks

- Perform clustering on the smaller chunks

- Replace each chunk with a representation/exemplar for that cluster (usually the centroid or medoid)

- Effectively, choose a smaller set of points to represent a larger set of points

- Useful for lossy data compression (e.g. MPEG, a precursor to JPEG)

28
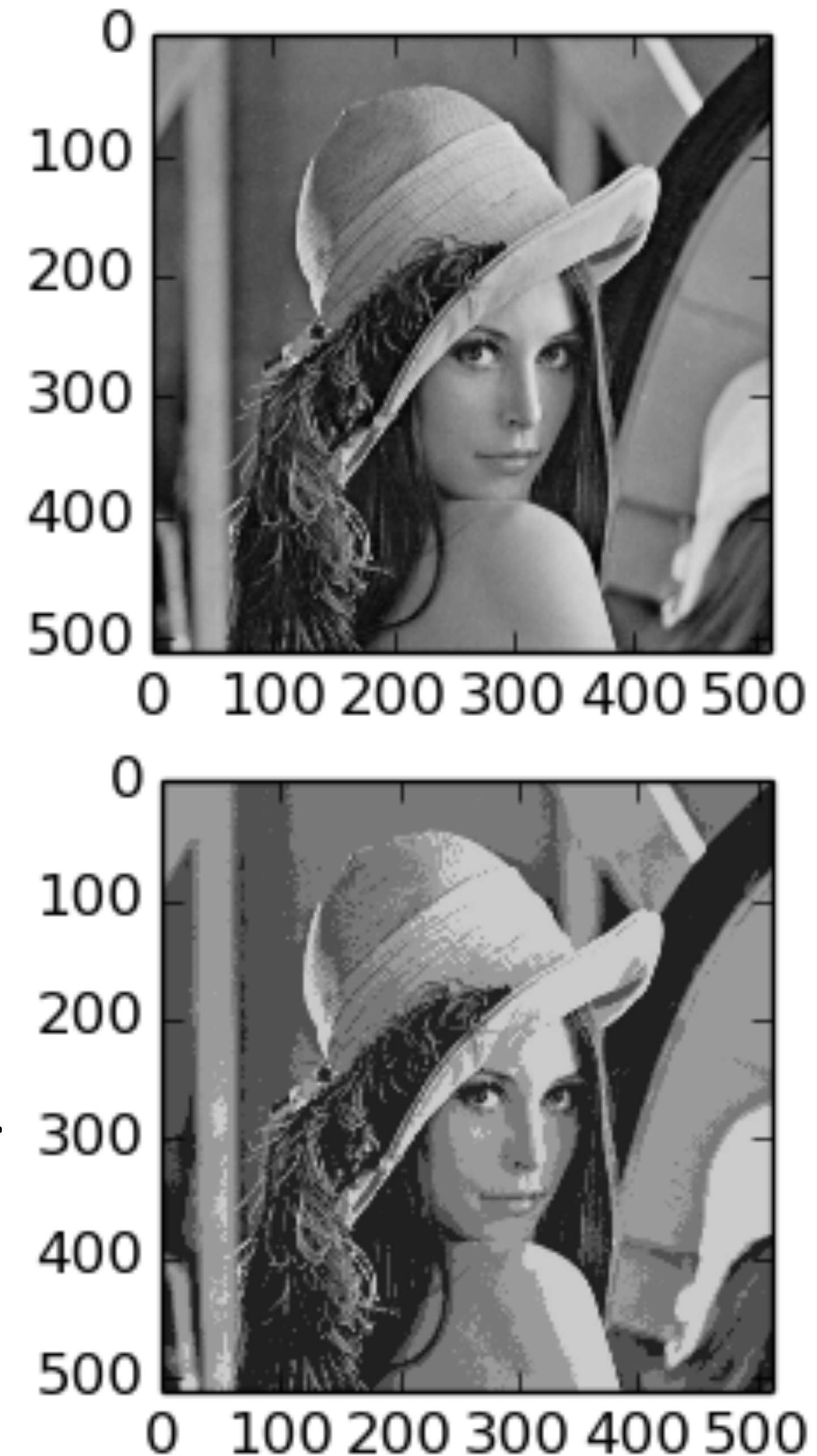
Questions?

- Feel free to email me at <u>te269@cam.ac.uk</u>

Next time

- Clustering

    - Gaussian mixture models and Expectation-Maximisation

    - Self-organising maps

    - Spectral clustering