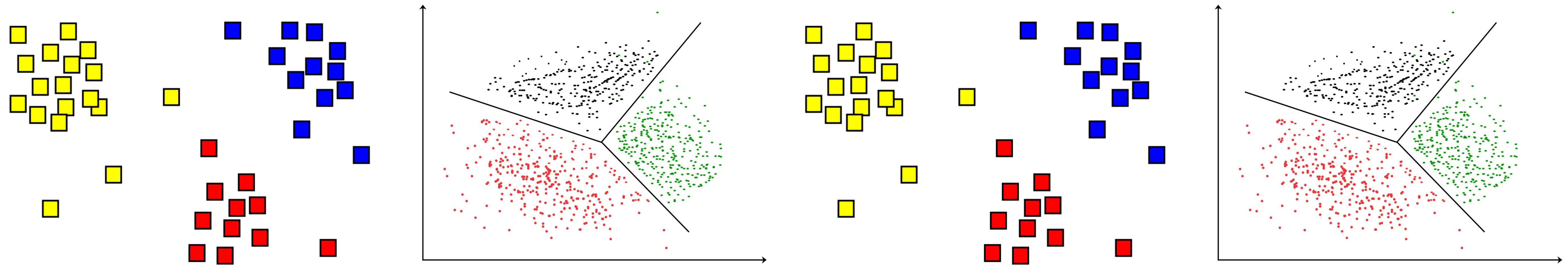
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 22: Clustering

Menu for Today (**March 31, 2020**)

Topics:

— Grouping
— Image Segmentation
— Agglomerative Clustering with a Graph
— Classification

Redings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 15.1, 15.2, 17.2
— **Next** Lecture: Deep Learning (N/A)

Reminders:

— **Assignment 5**: Scene Recognition with Bag of Words due **March 31th**
— **Assignment 6:** Deep Learning will be out **tonight** (latest tomorrow morning)
— Last **lecture** next Tuesday (**April 7th**), will make prep Quizzes available

Grouping in Human Vision

Humans routinely group features that belong together when looking at a scene. What are some cues that we use for grouping?

Grouping in Human Vision

Humans routinely group features that belong together when looking at a scene. What are some cues that we use for grouping?

— Similarity

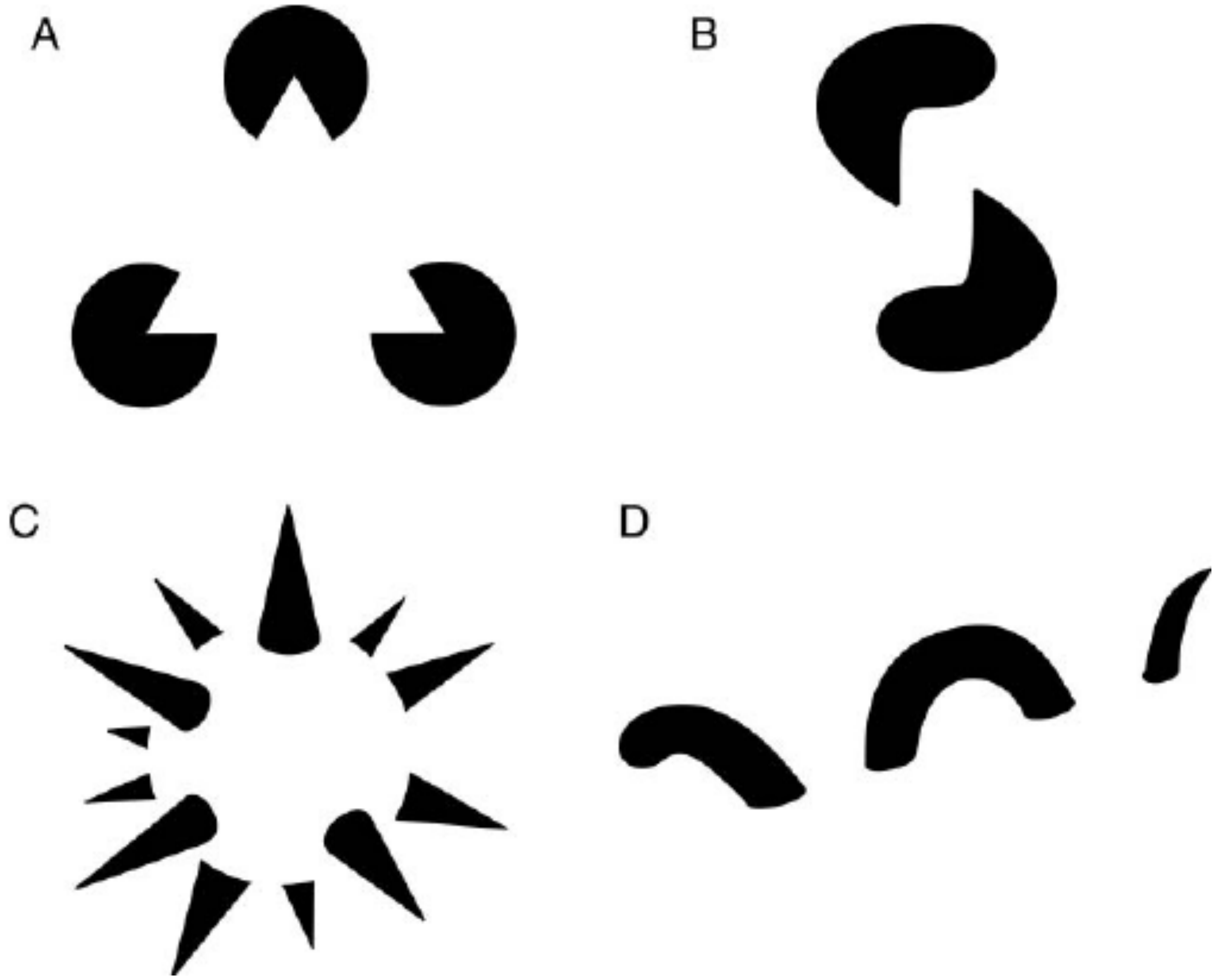
— Symmetry

— Common Fate

— Proximity

— ...

Grouping in Human Vision



A. Kanizsa triangle
B. Tse's volumetric worm
C. Idesawa's spiky sphere
D. Tse's "sea monster"

Figure credit: Steve Lehar

Grouping in Human Vision

Grouping in Human Vision

Slide credit: Kristen Grauman

Grouping in Human Vision

Benjamin Lee
@benfraserlee

Follow

Incredible way of making my two star review seem like I didn't hate the film

2:53 PM - 8 Sep 2015 from Montrose, CO

14,153 Retweets **13,994** Likes

Slide credit: Kristen Grauman

Clustering

It is often useful to be able to **group** together **image regions** with similar appearance (e.g. roughly coherent colour or texture)

— image compression

— approximate nearest neighbour search

— base unit for higher-level recognition tasks

— moving object detection in video sequences

— video summarization

Clustering

Clustering is a set of techniques to try to find components that belong together (i.e., components that form clusters).

— Unsupervised learning (access to data, but no labels)

Two basic clustering approaches are

— **agglomerative clustering**

— **divisive clustering**

Agglomerative Clustering

Each data point starts as a separate cluster. Clusters are recursively merged.

Algorithm:

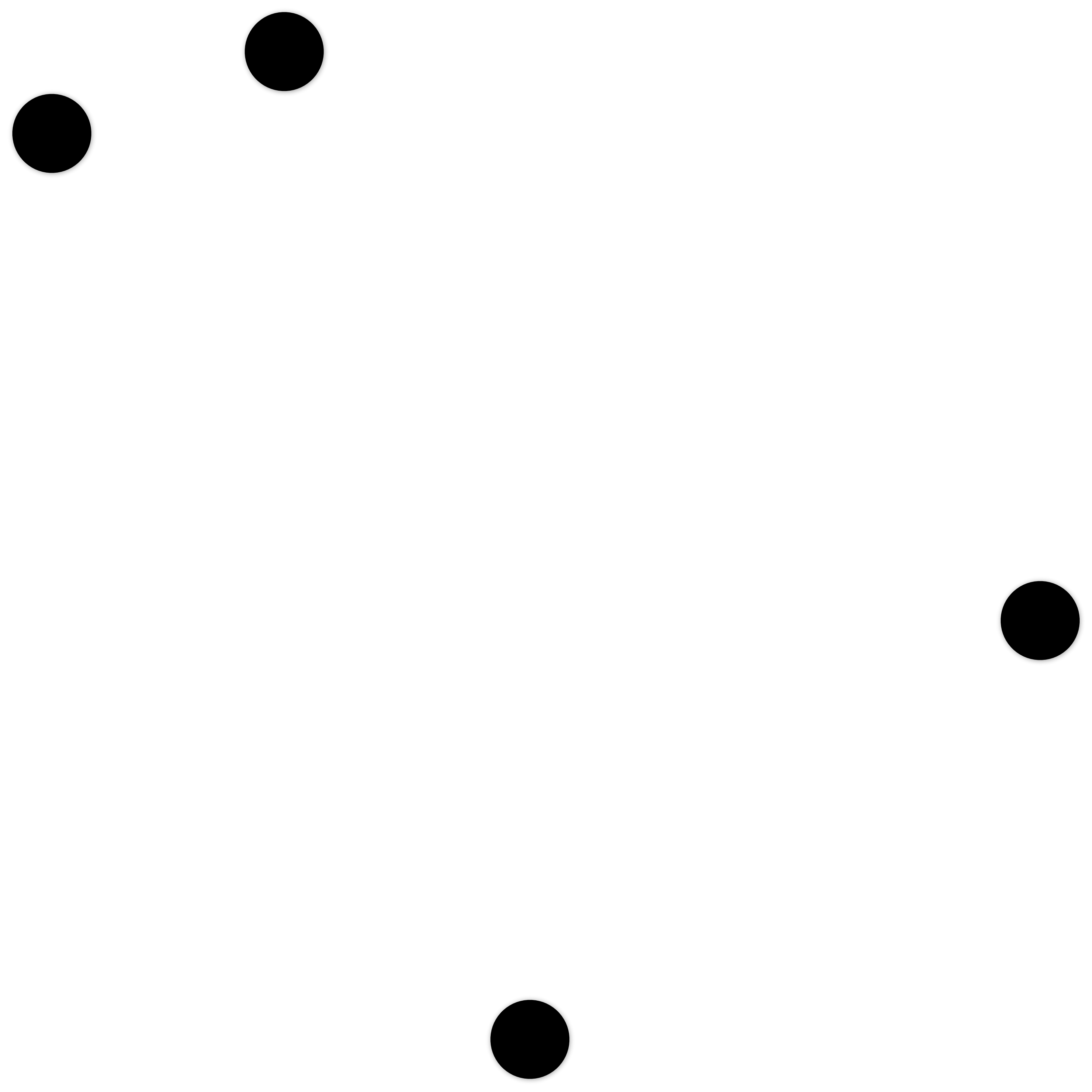
Make each point a separate cluster

Until the clustering is satisfactory

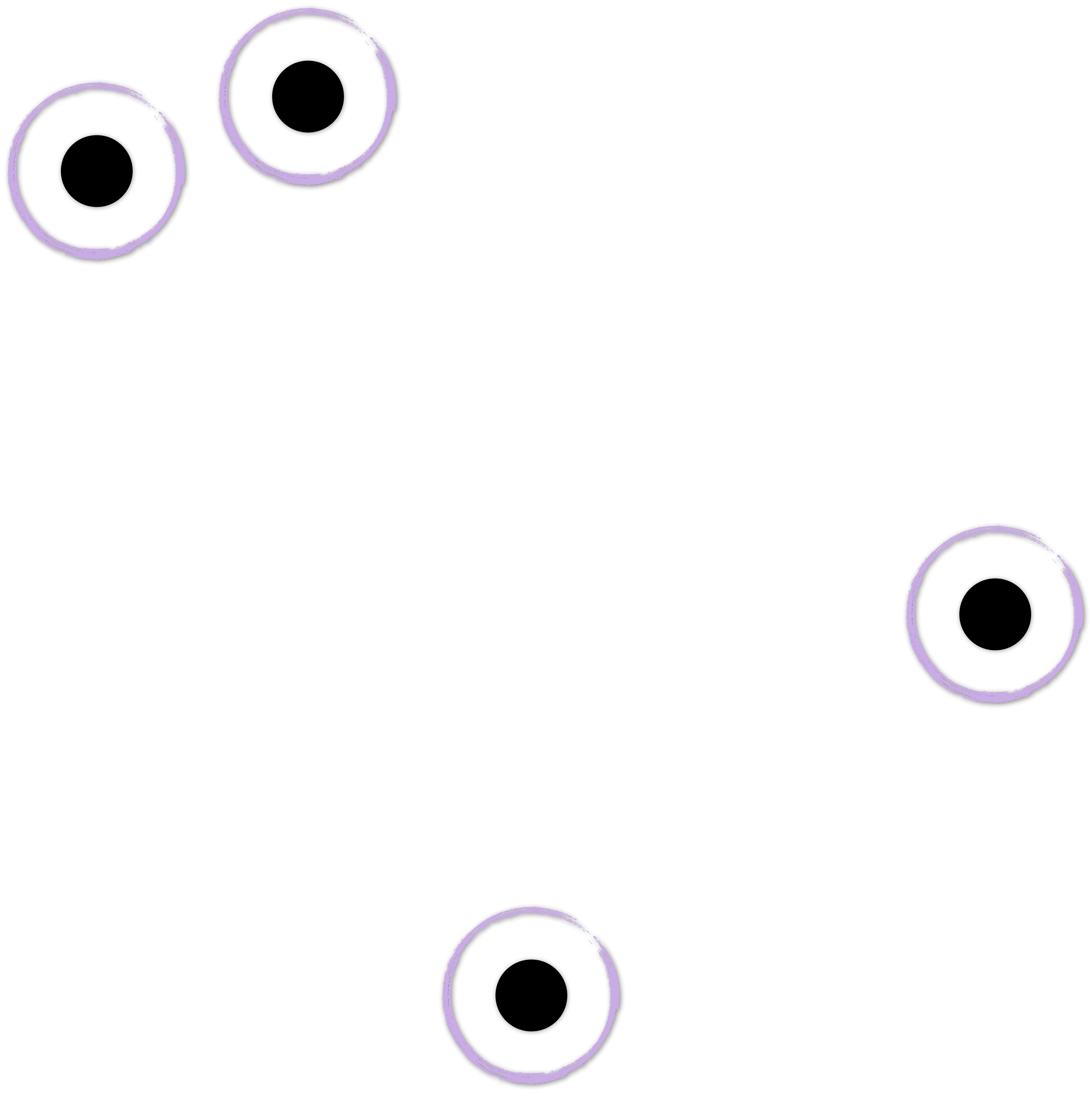
    Merge the two clusters with the smallest inter-cluster distance

end

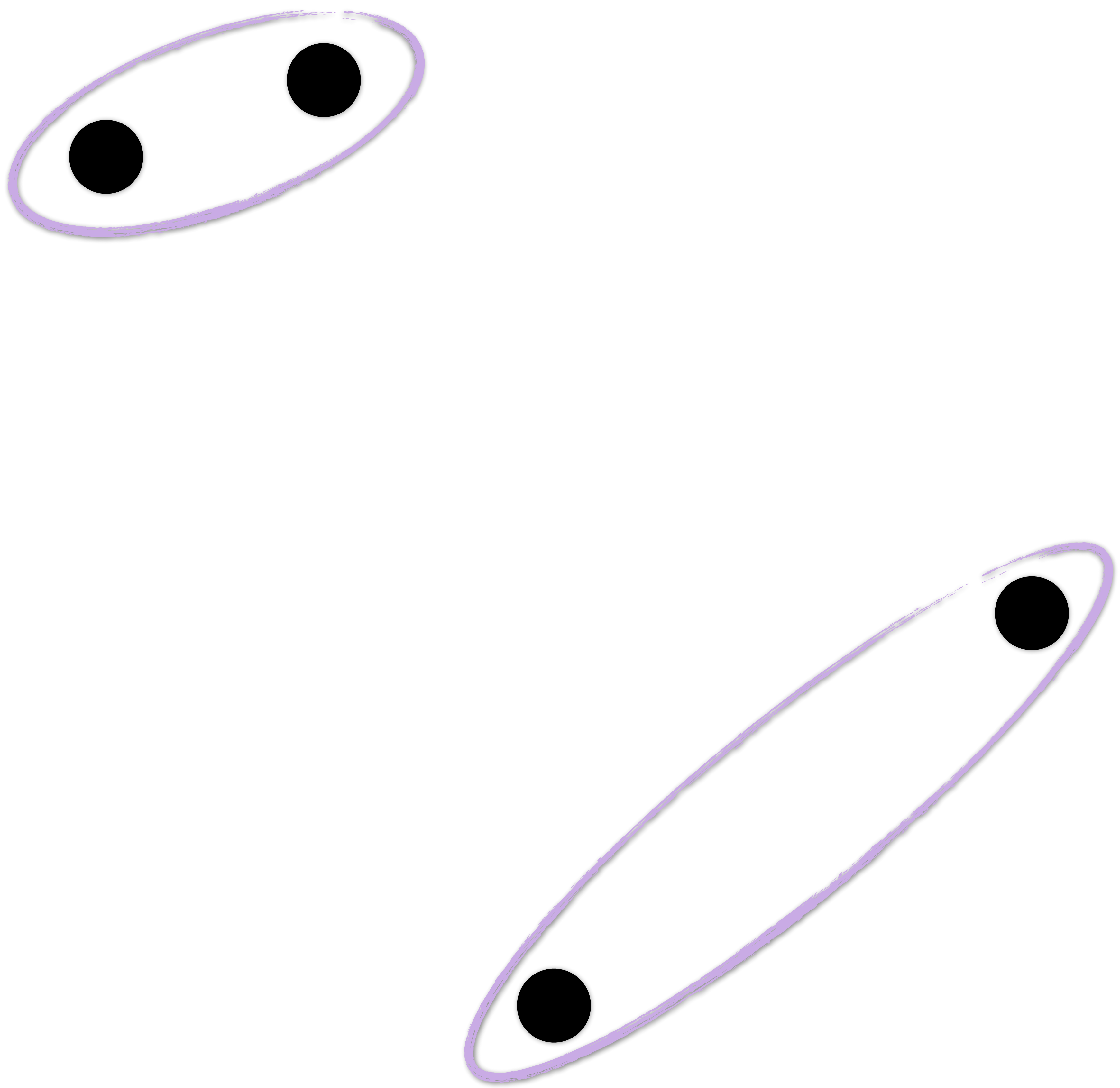
Agglomerative Clustering



Agglomerative Clustering



Agglomerative Clustering



Divisive Clustering

The entire data set starts as a single cluster. Clusters are recursively split.

Algorithm:

Construct a single cluster containing all points

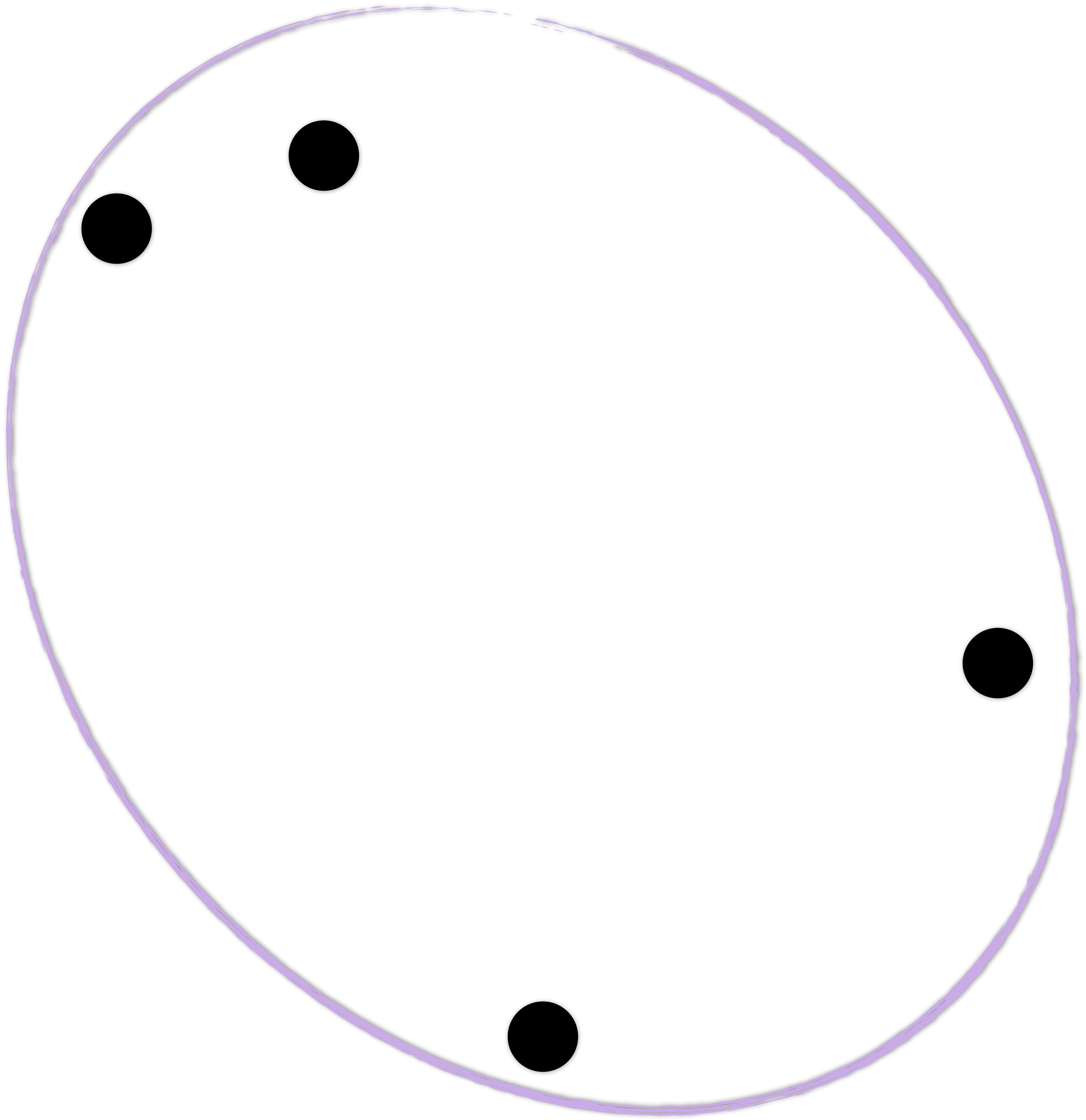
Until the clustering is satisfactory

        Split the cluster that yields the two components

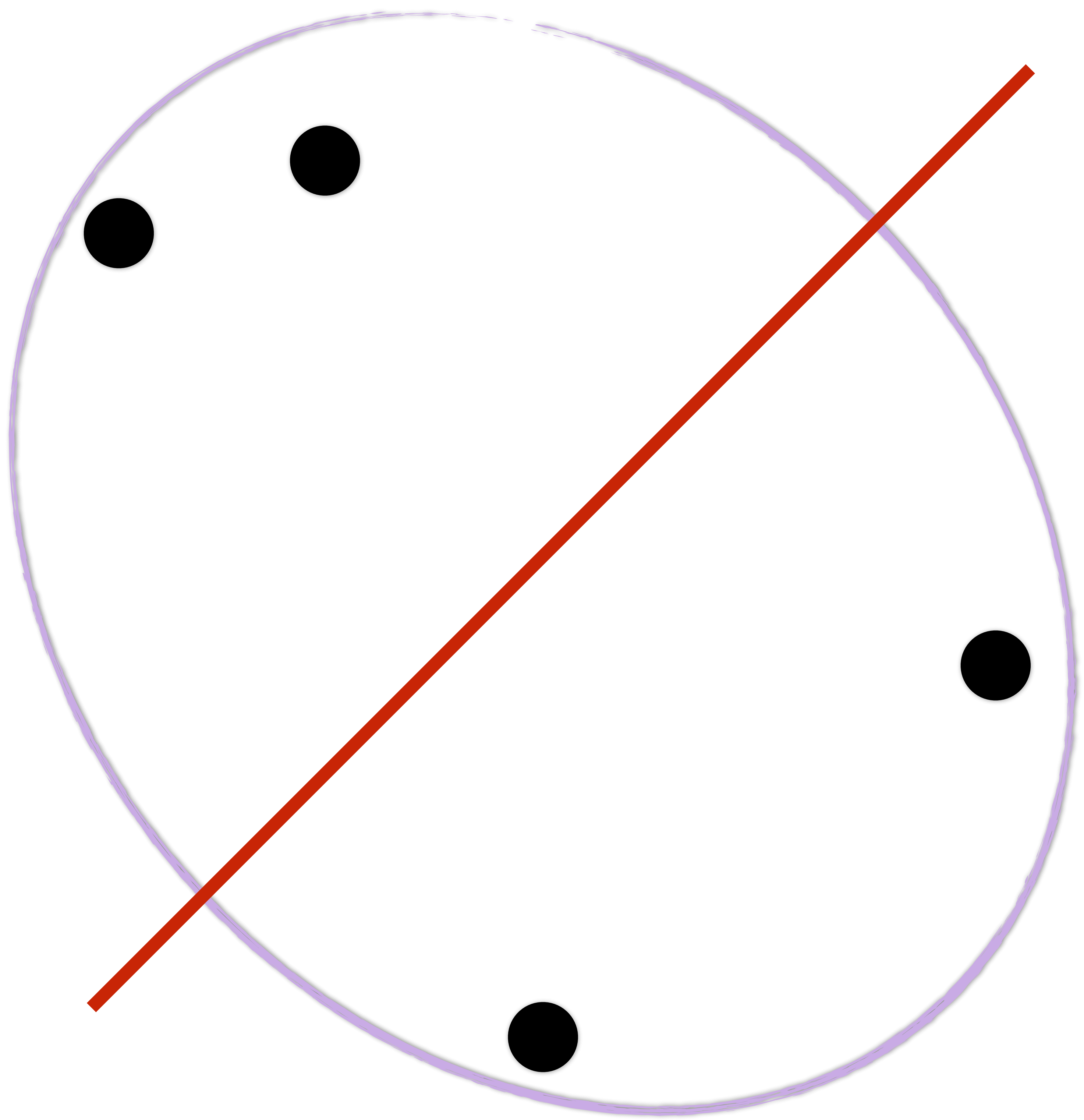
        with the largest inter-cluster distance

end

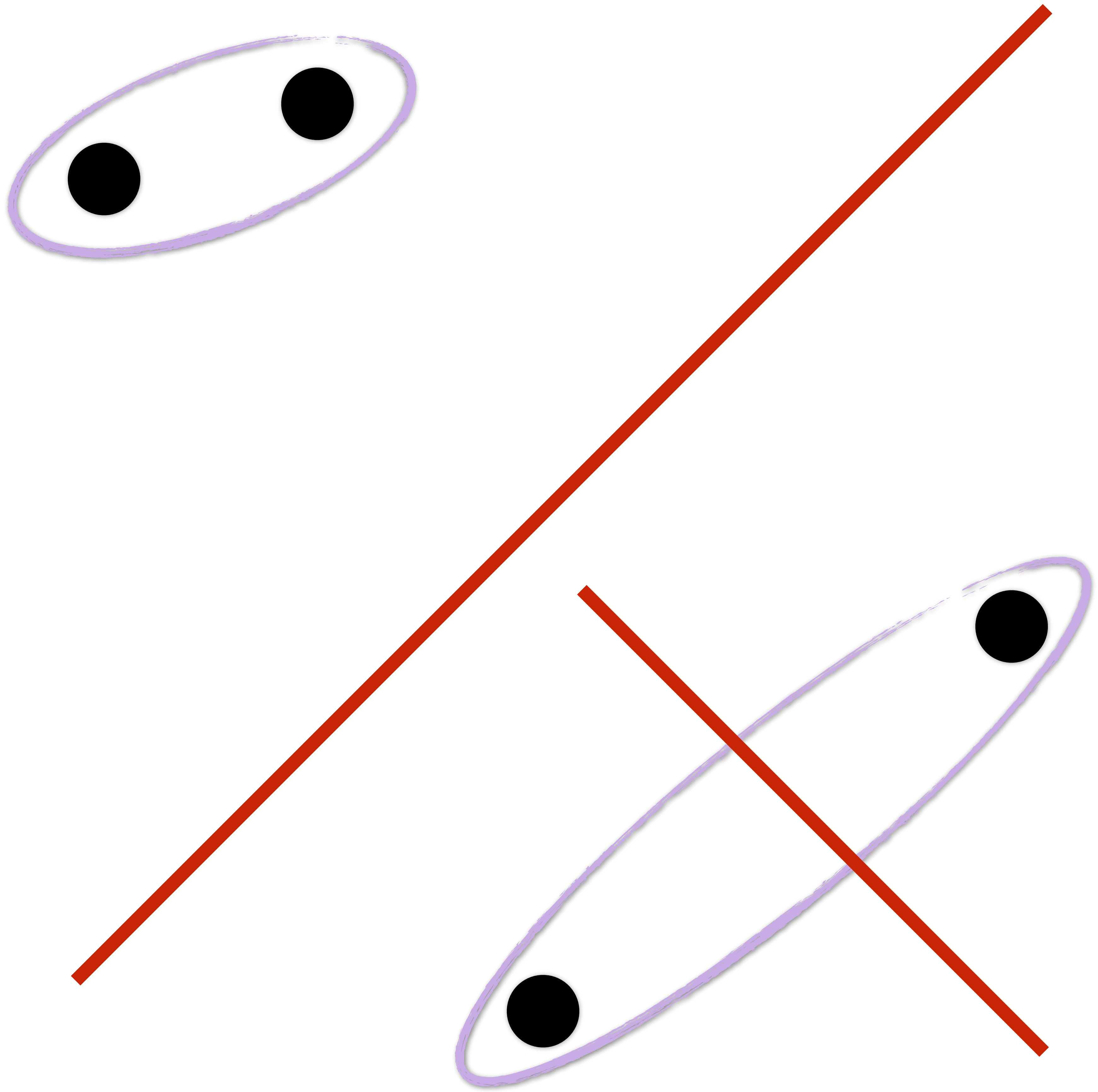
Divisive Clustering



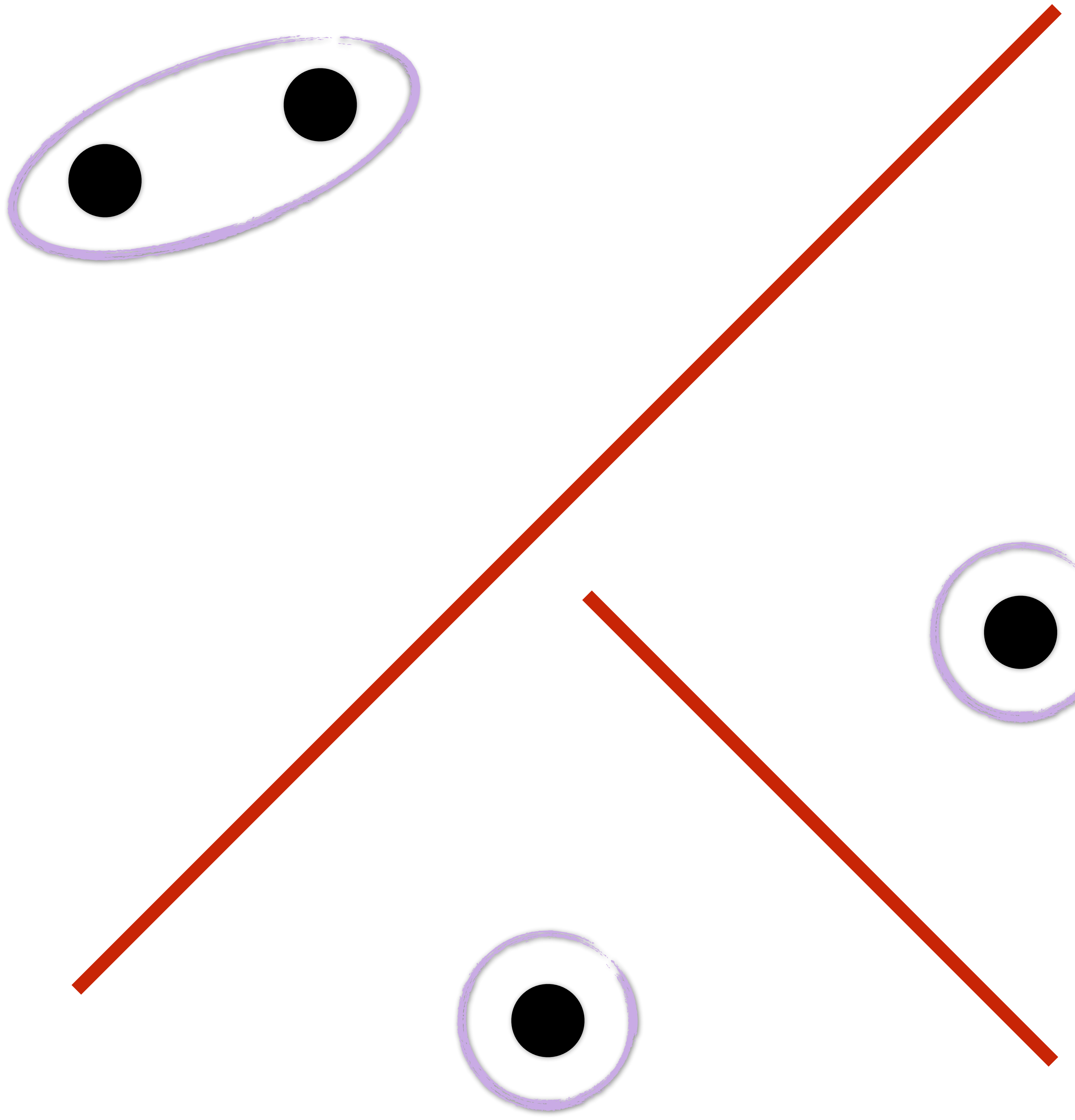
Divisive Clustering



Divisive Clustering



Divisive Clustering



Inter-Cluster Distance

How can we define the cluster distance between two clusters C_1 and C_2 in agglomerative and divisive clustering? Some common options:

the distance between the closest members of C_1 and C_2

$$\min d(a, b), a \in C_1, b \in C_2$$

– single-link clustering

the distance between the farthest members of C_1 and a member of C_2

$$\max d(a, b), a \in C_1, b \in C_2$$

– complete-link clustering

Inter-Cluster Distance

How can we define the cluster distance between two clusters C_1 and C_2 in agglomerative and divisive clustering? Some common options:

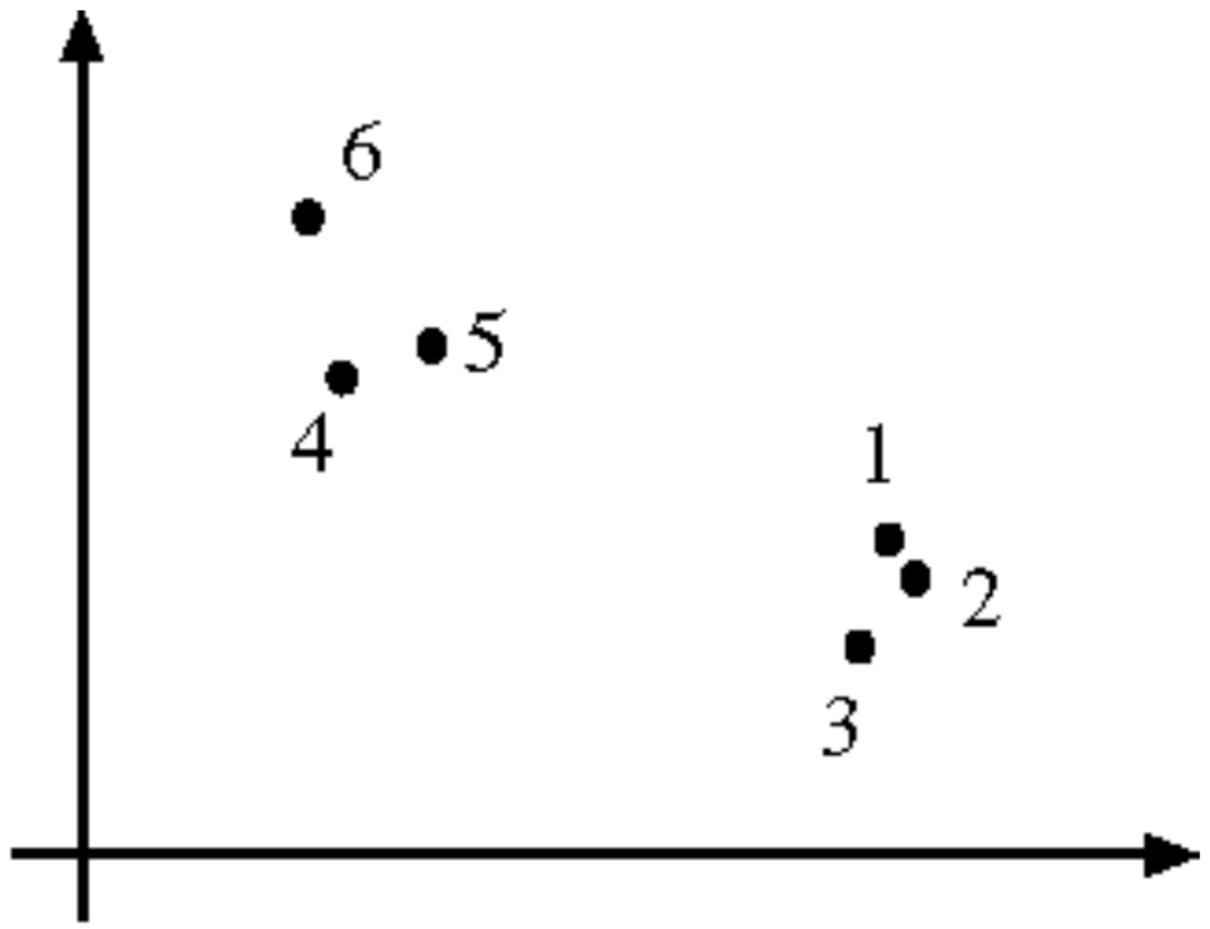
an average of distances between members of C_1 and C_2

$$\frac{1}{|C_1||C_2|}\sum_{a \in C_1}\sum_{b \in C_2} d(a, b)$$

– group average clustering

Dendrogram

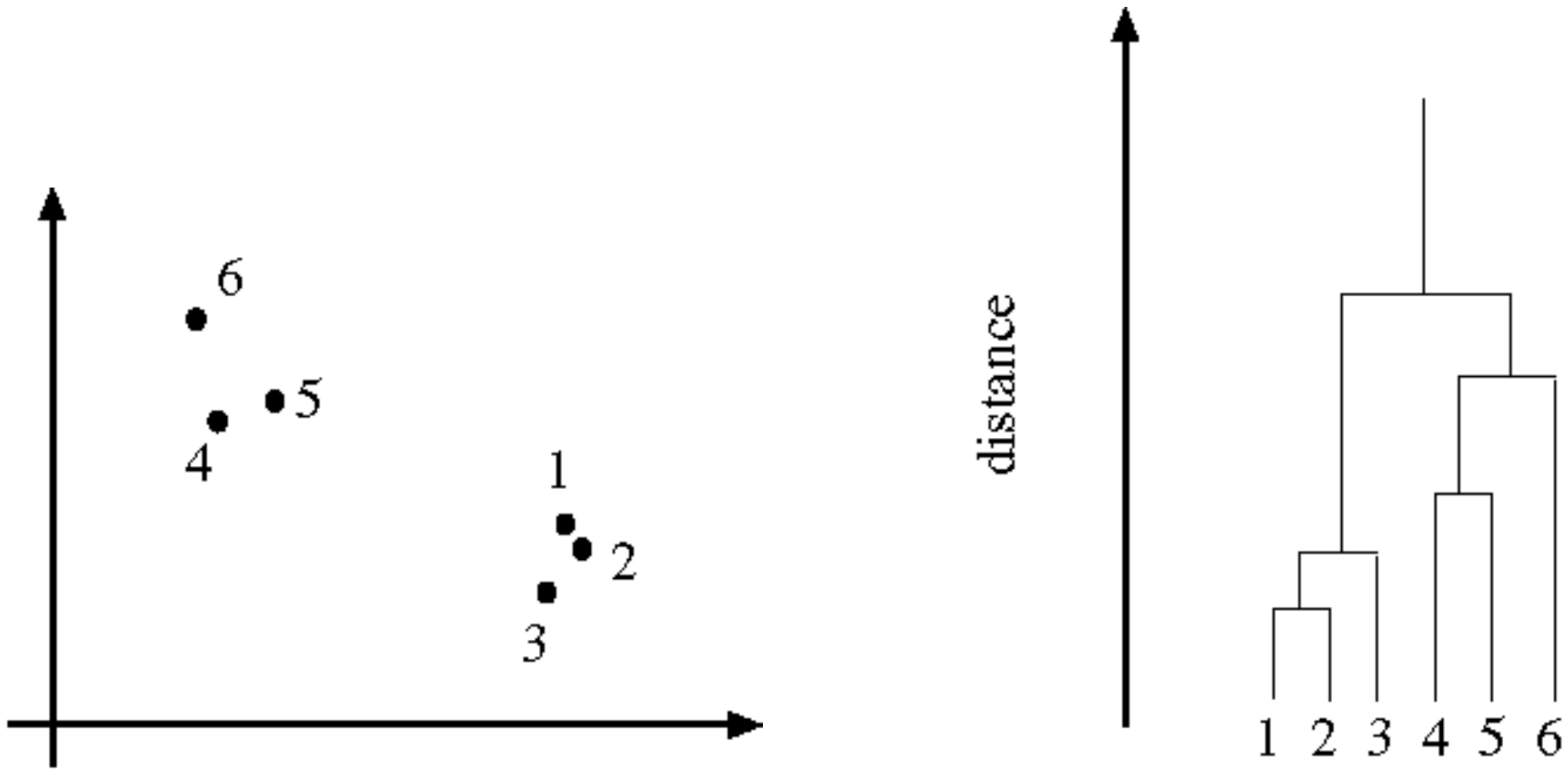
The algorithms described generate a hierarchy of clusters



Forsyth & Ponce (2nd ed.) Figure 9.15

Dendrogram

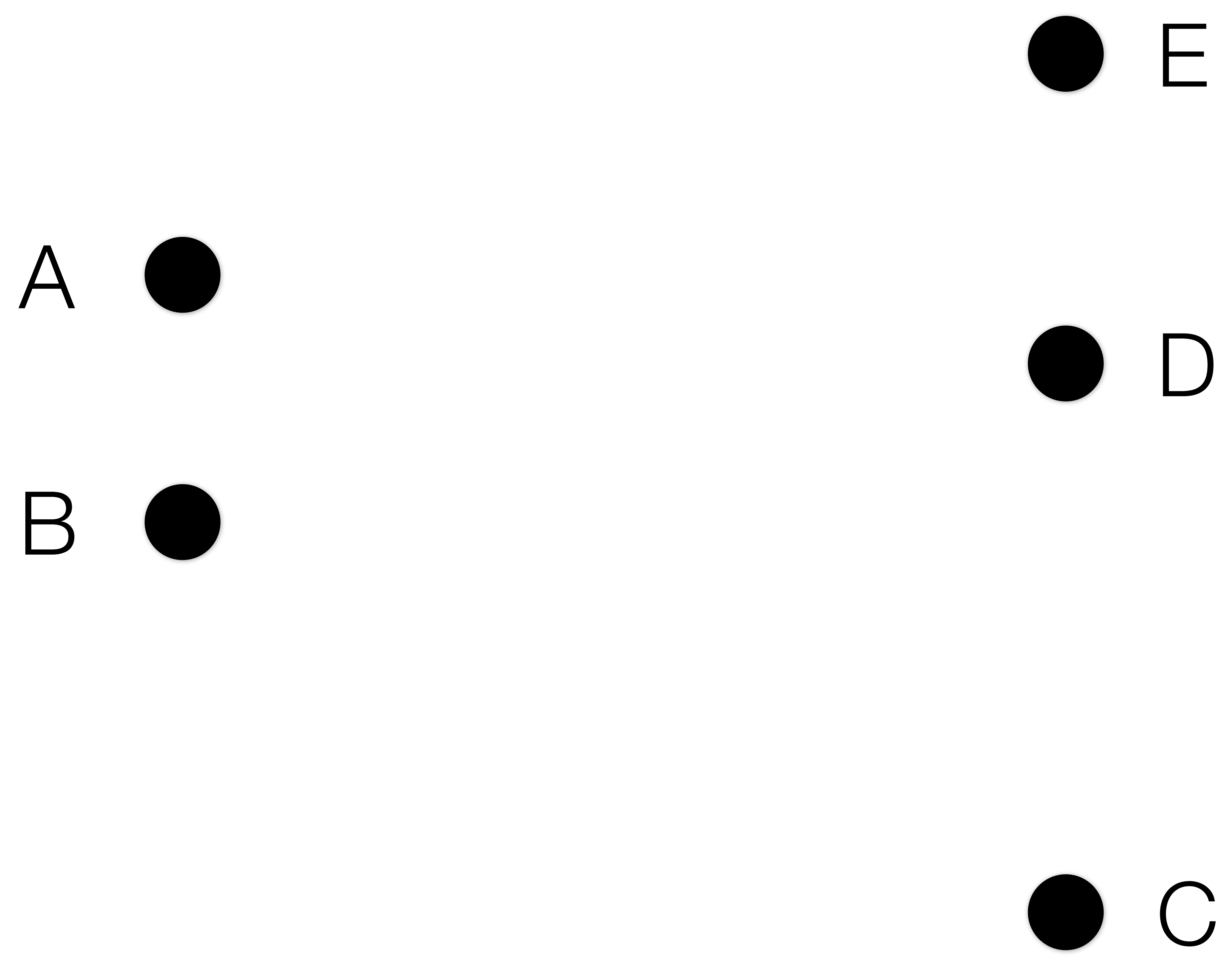
The algorithms described generate a hierarchy of clusters, which can be visualized with a **dendrogram**.



Forsyth & Ponce (2nd ed.) Figure 9.15

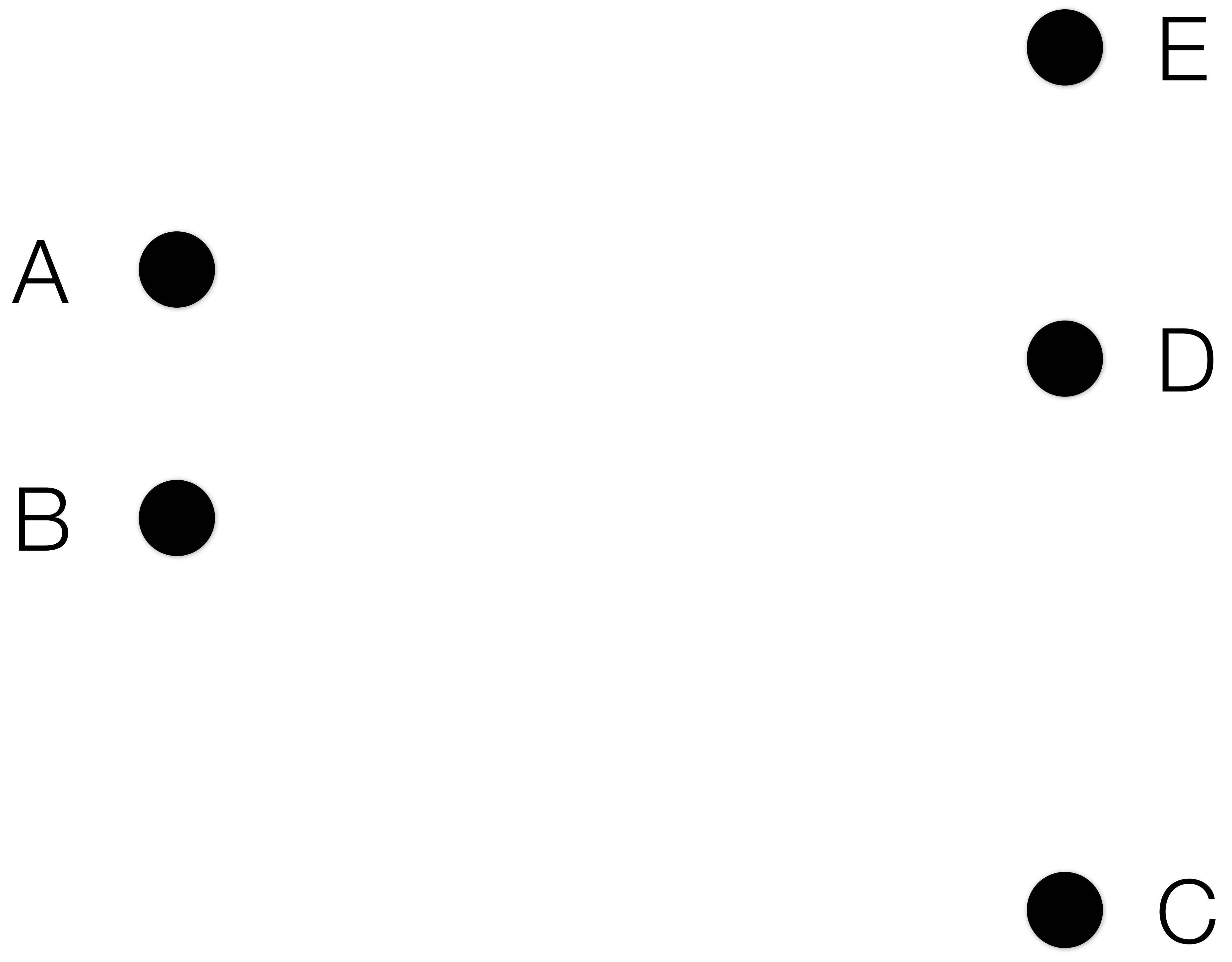
A Short **Exercise**

A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.



A Short **Exercise**

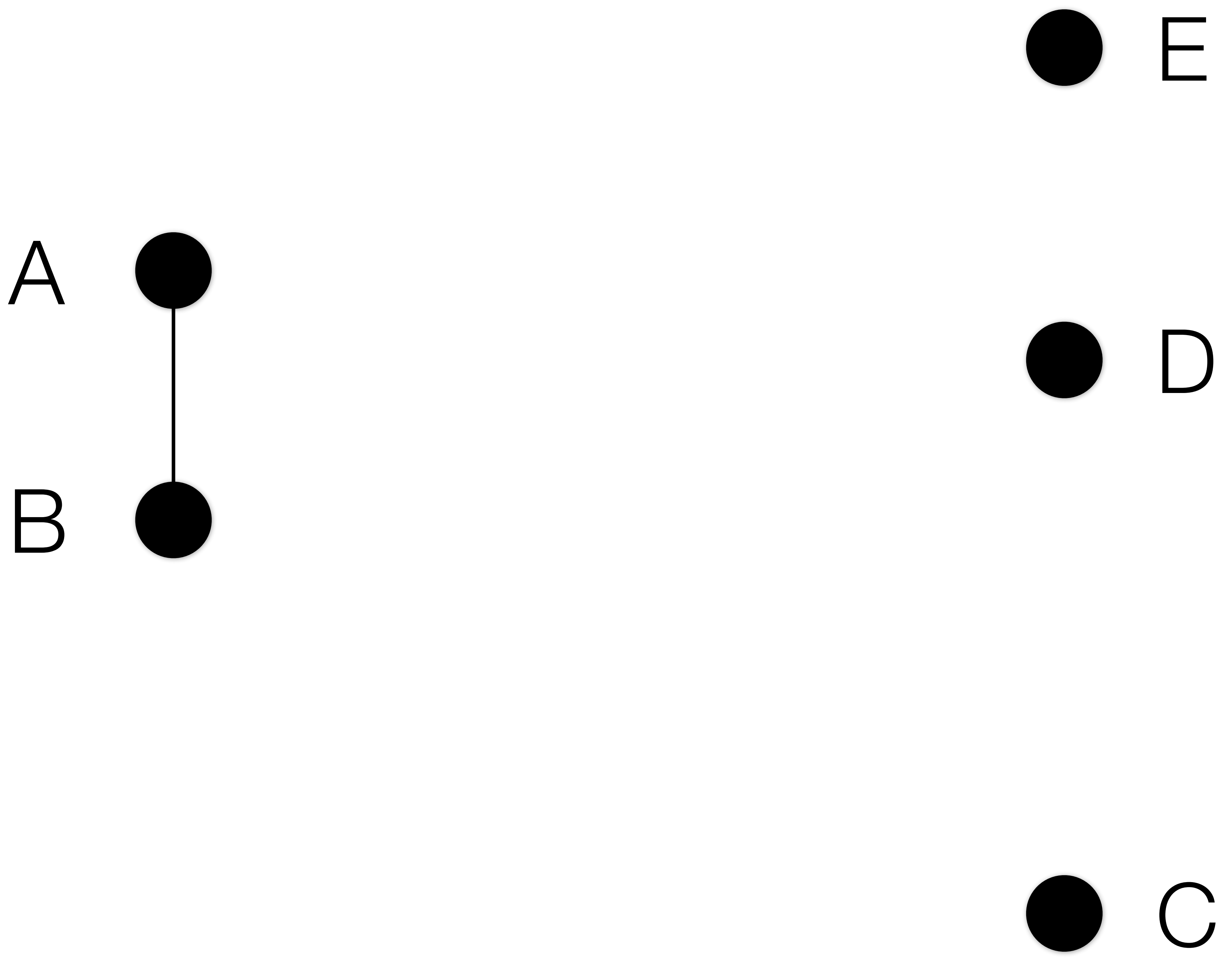
A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.

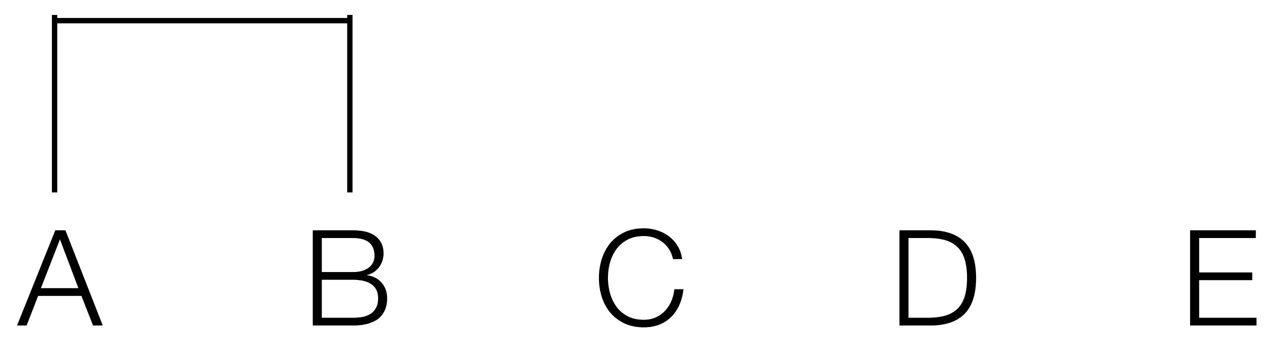


A B C D E

A Short **Exercise**

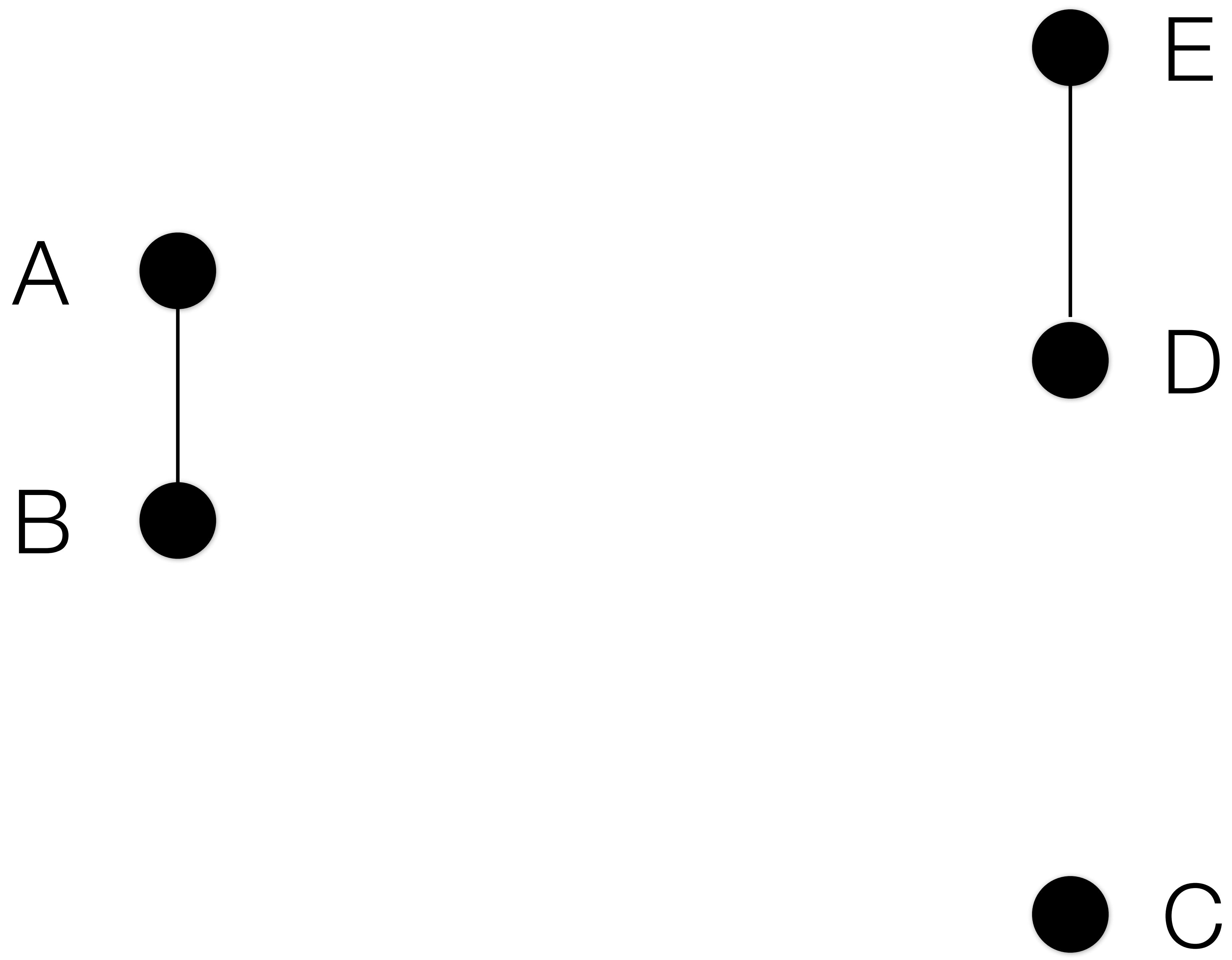
A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.

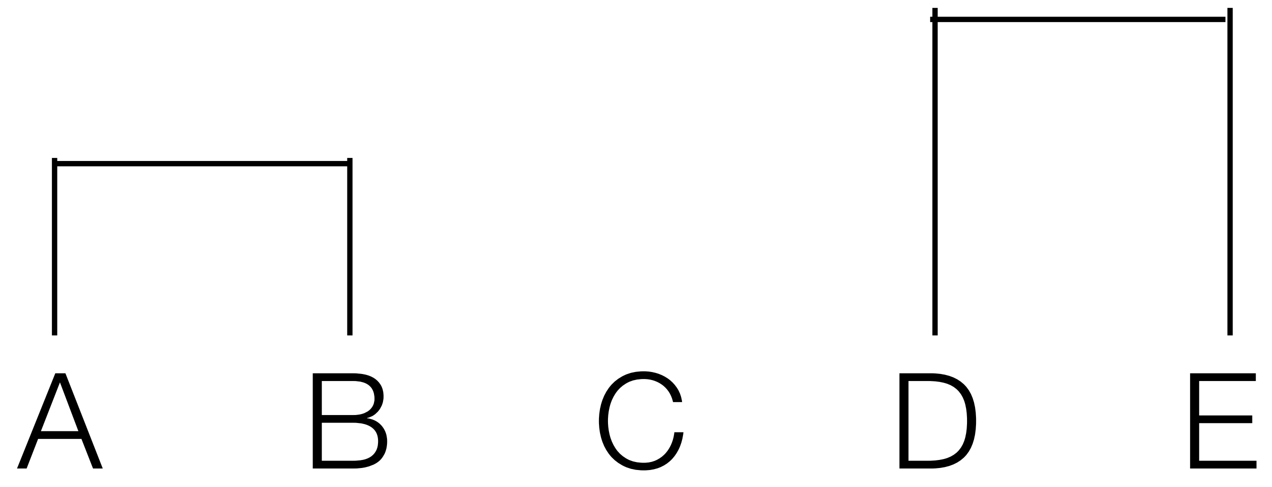




A Short **Exercise**

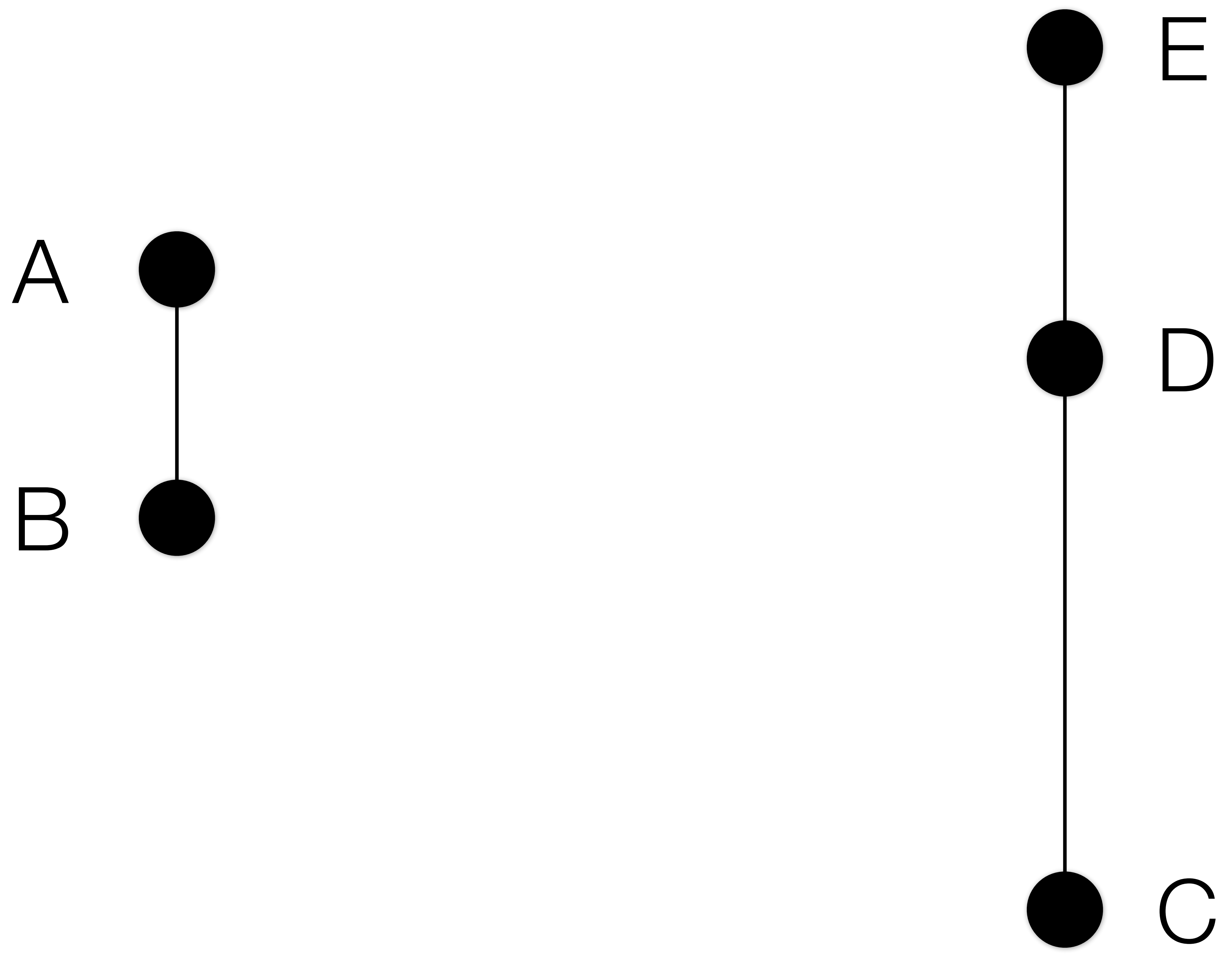
A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.

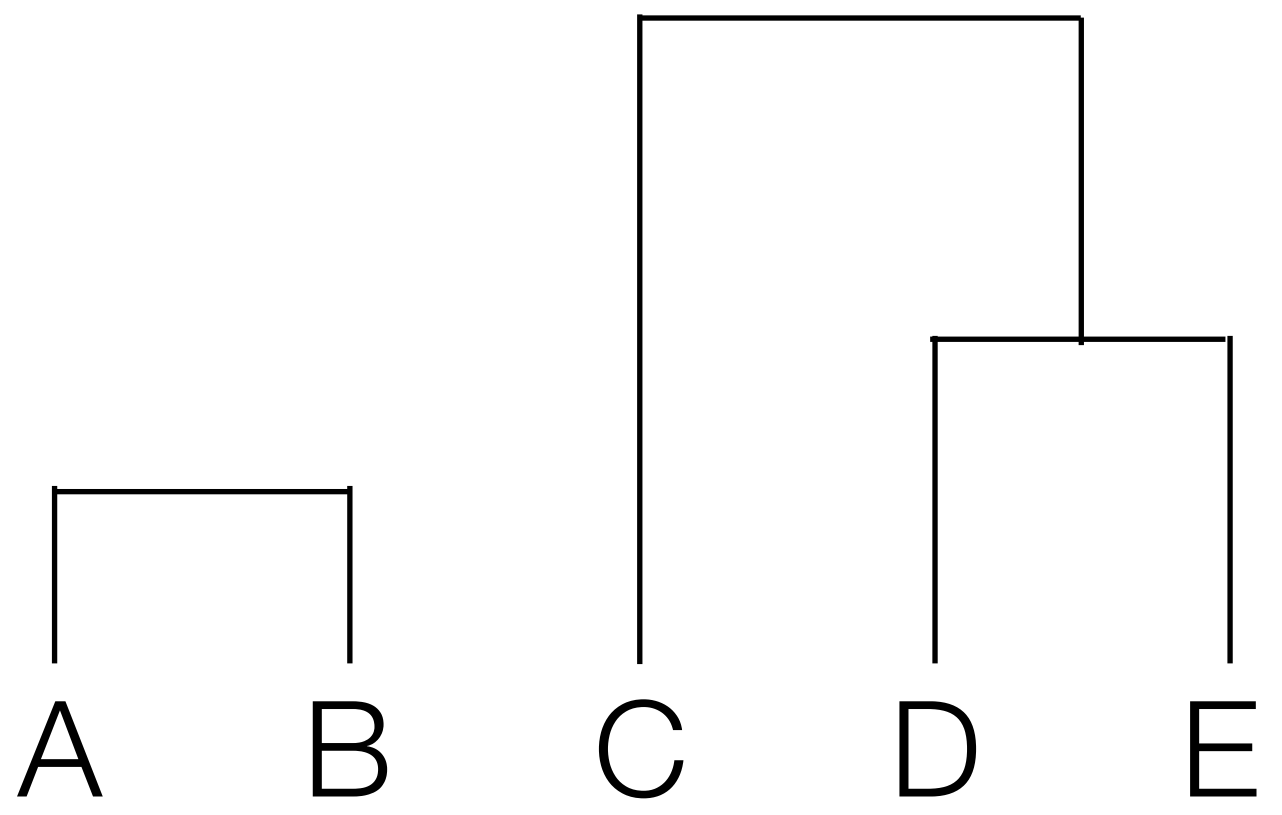




A Short **Exercise**

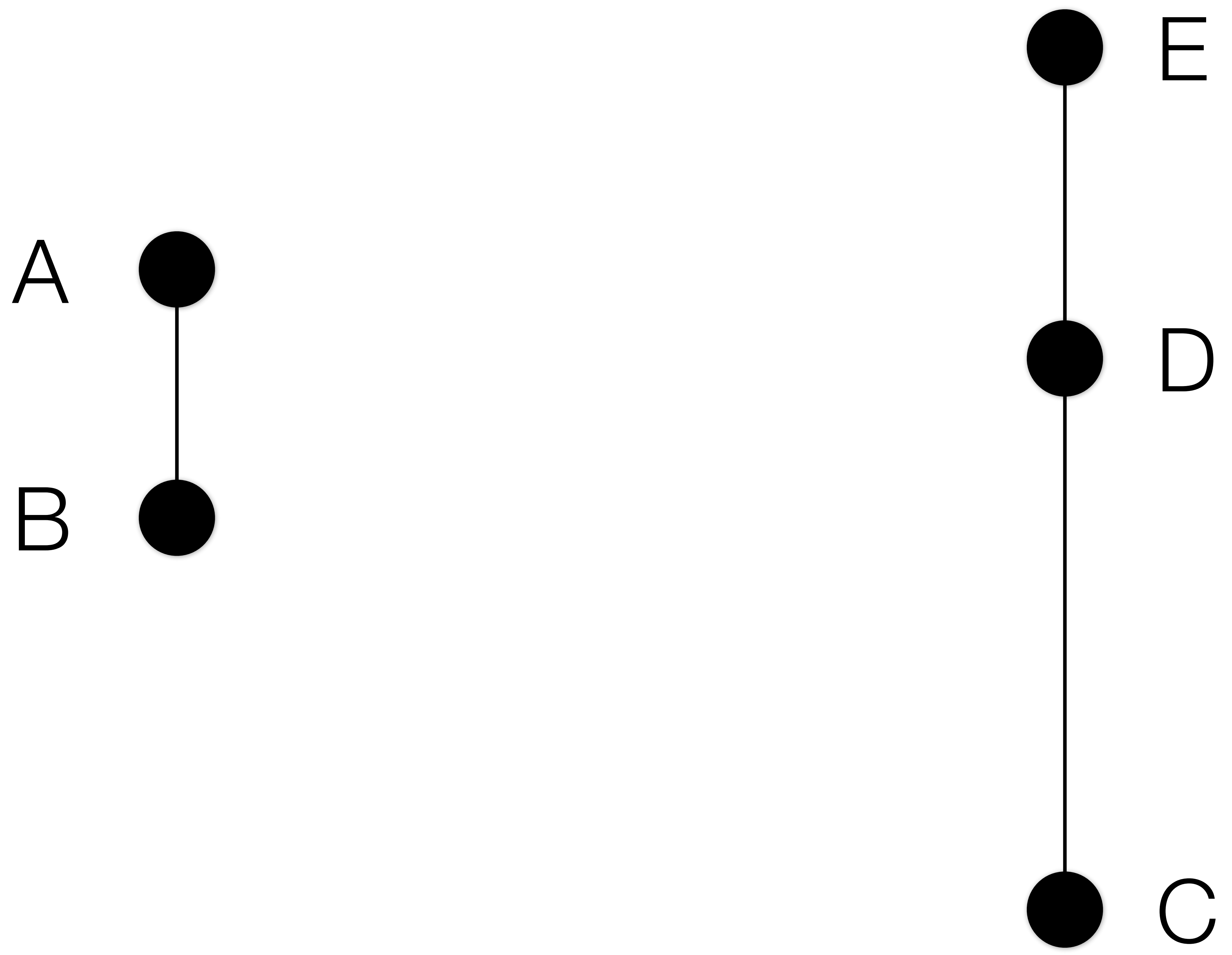
A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.

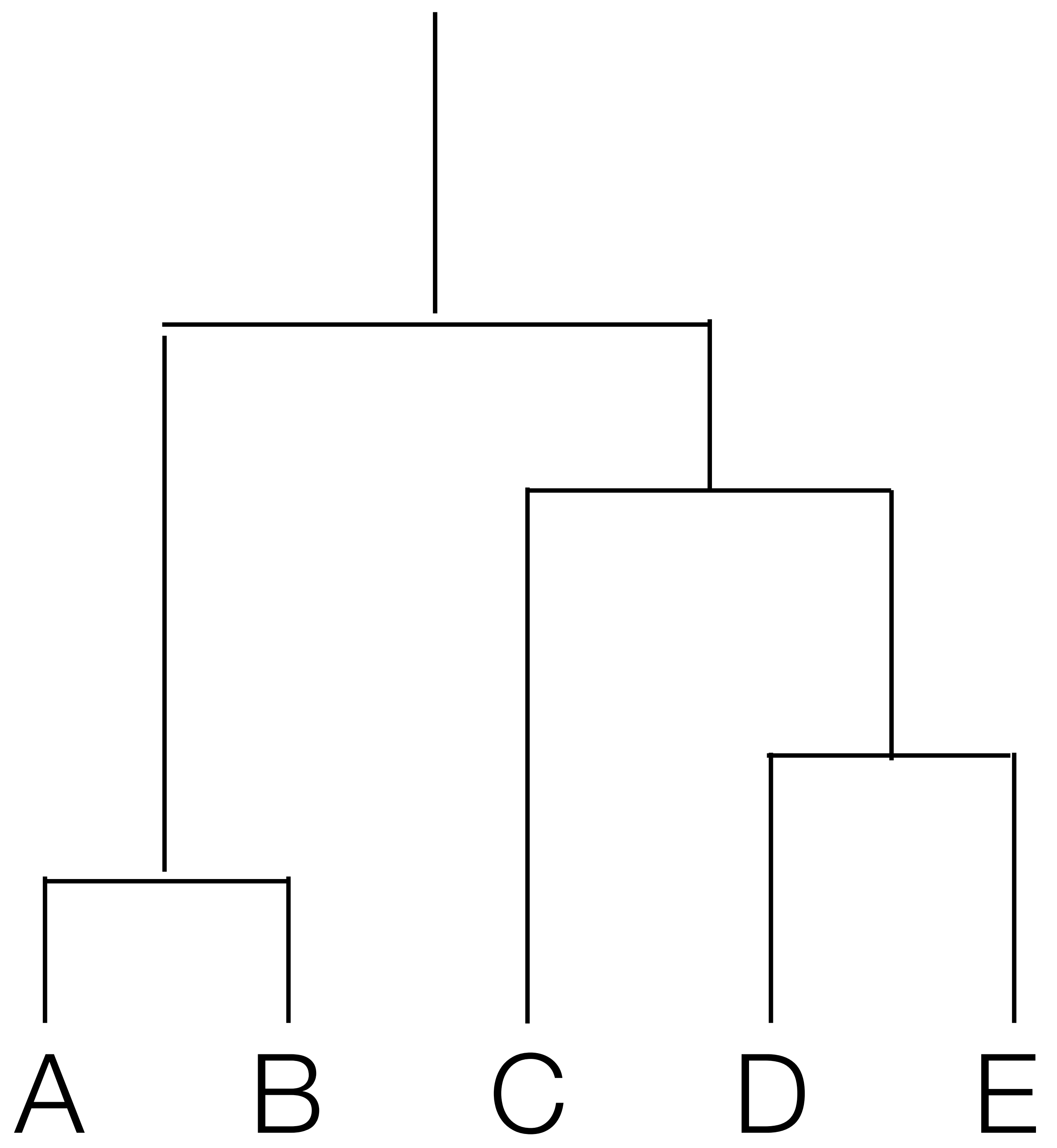




A Short **Exercise**

A simple dataset is shown below. Draw the dendrogram obtained by agglomerative clustering with single-link (closest member) inter-cluster distance.





K-Means Clustering

Assume we know how many clusters there are in the data - denote by K

Each cluster is represented by a cluster center, or mean

Our objective is to minimize the representation error (or quantization error) in letting each data point be represented by some cluster center

Minimize

$$\sum_{i \in clusters} \left\{ \sum_{j \in i^{th}\ cluster} ||x_j - \mu_i||^2 \right\}$$

K-Means Clustering

K-means clustering alternates between two steps:

1. Assume the cluster centers are known (fixed). Assign each point to the closest cluster center.

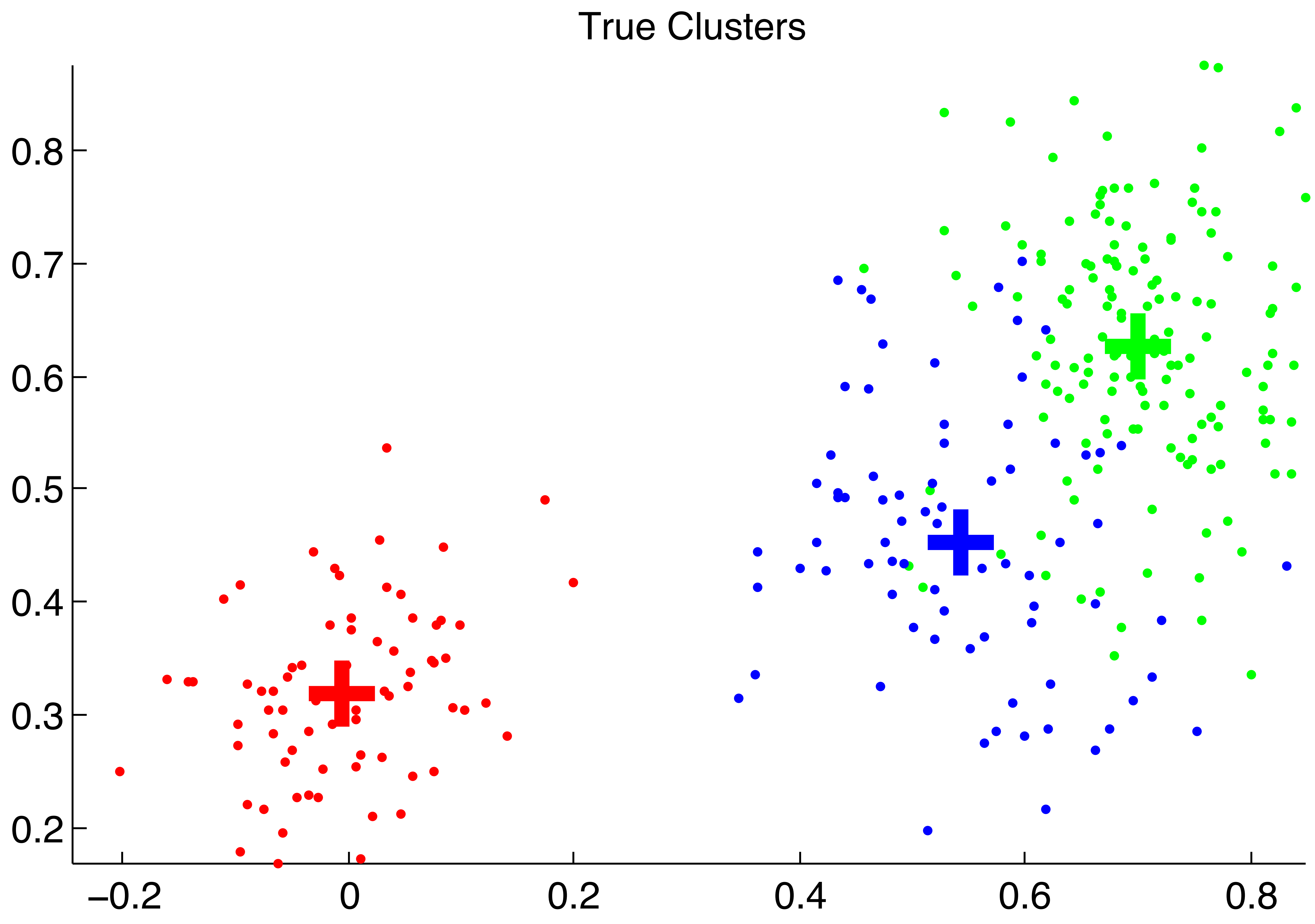
2. Assume the assignment of points to clusters is known (fixed). Compute the best center for each cluster, as the mean of the points assigned to the cluster.

The algorithm is initialized by choosing K random cluster centers

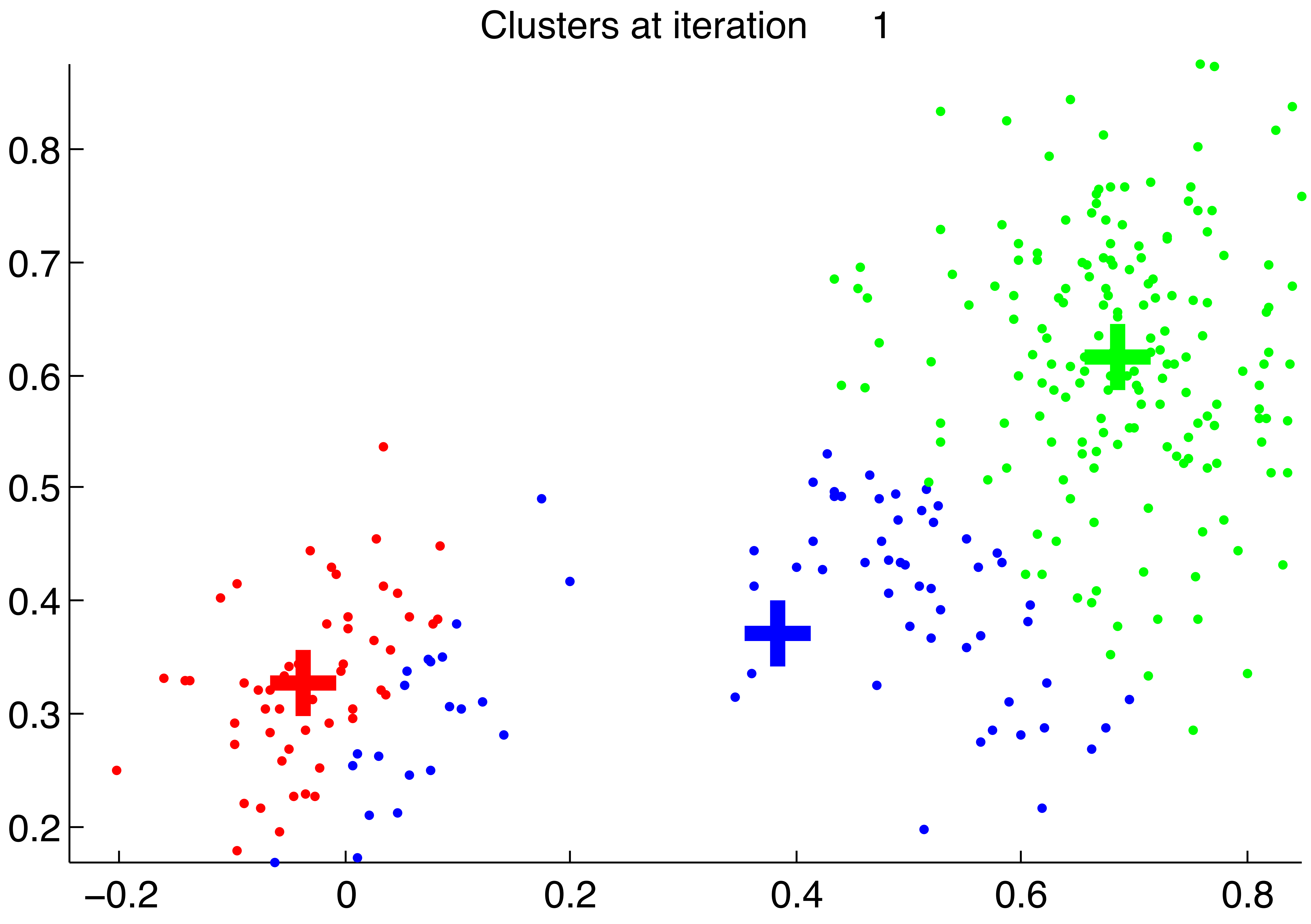
K-means converges to a local minimum of the objective function

— Results are initialization dependent

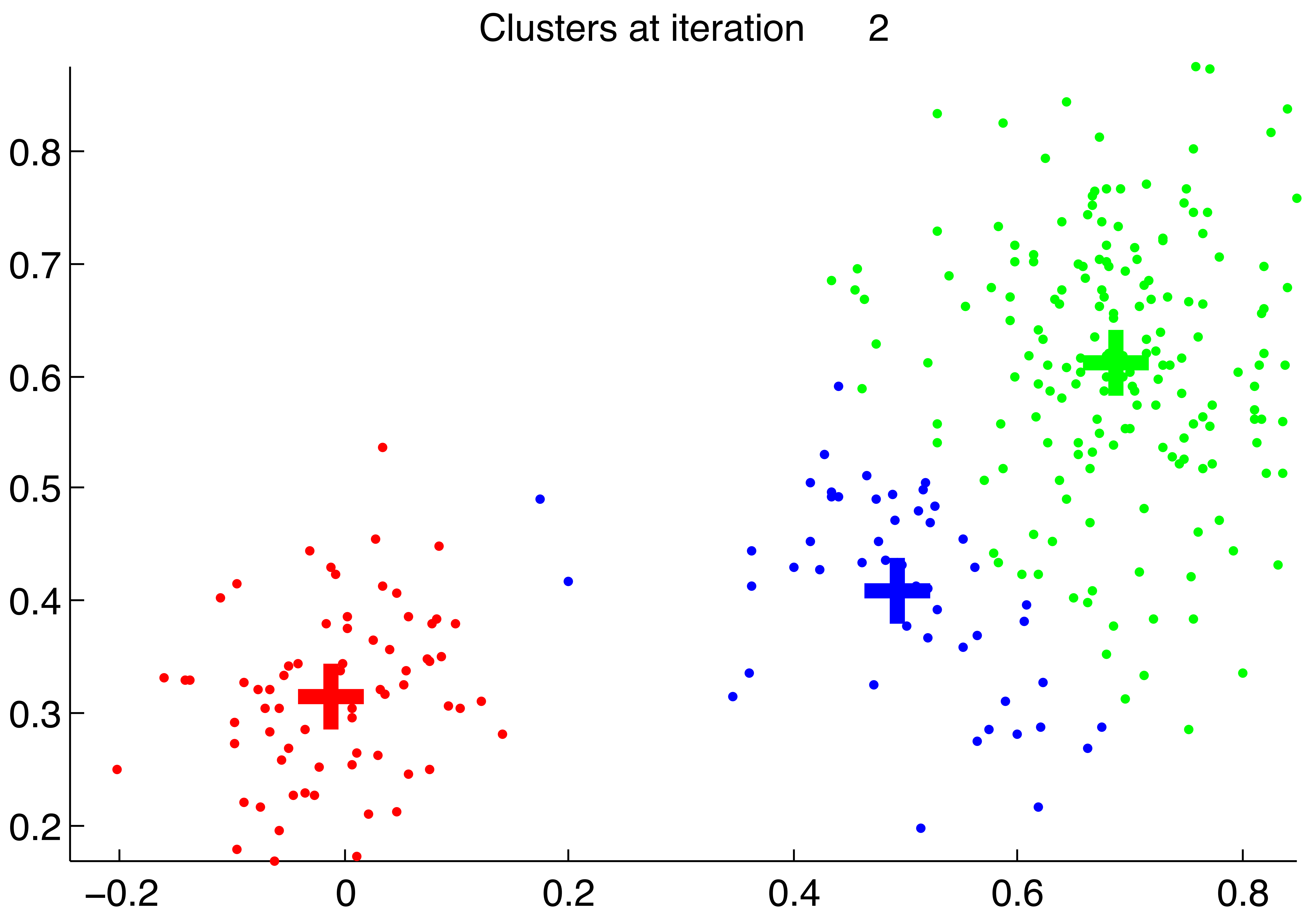
Example 1: K-Means Clustering



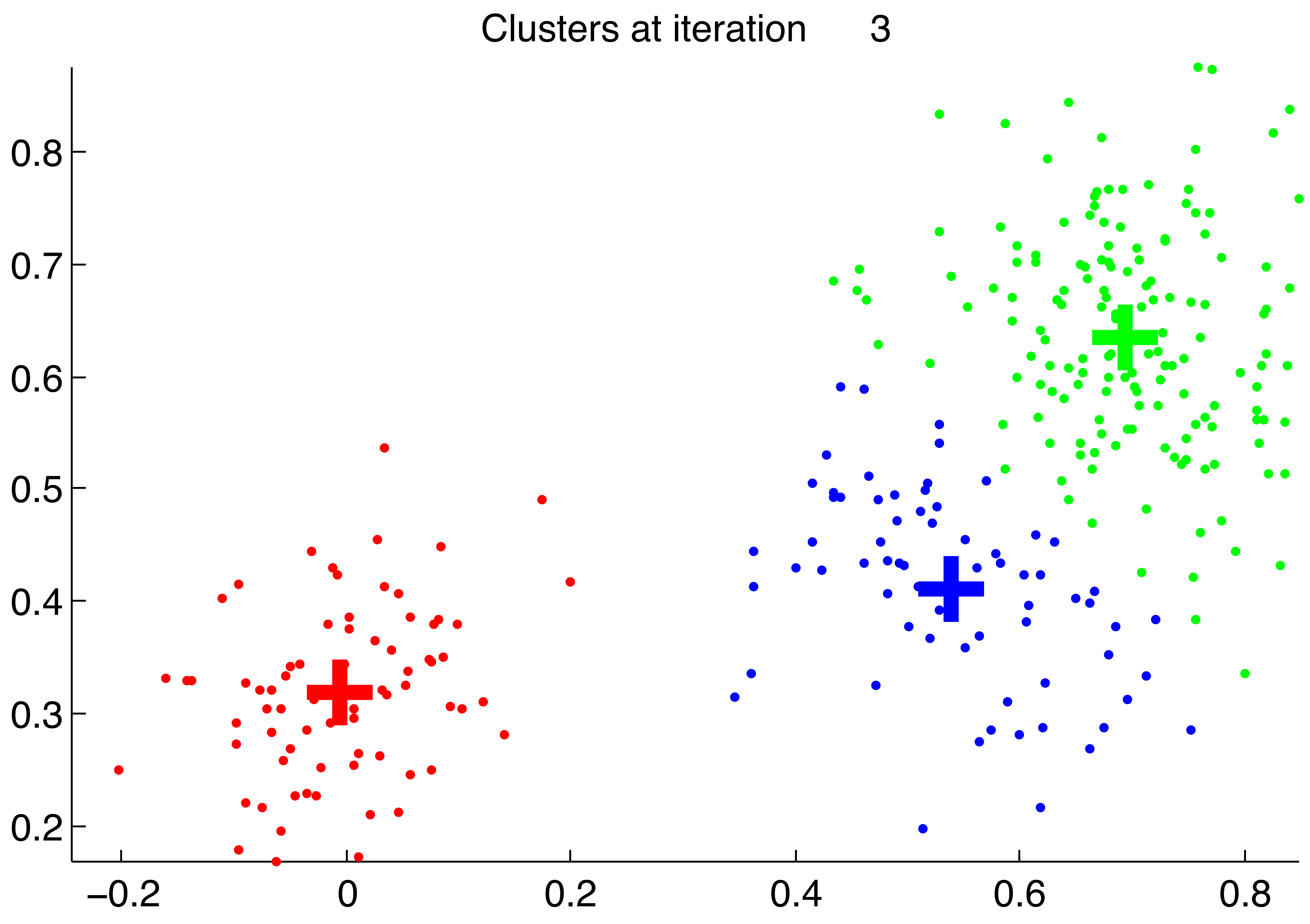
Example 1: K-Means Clustering



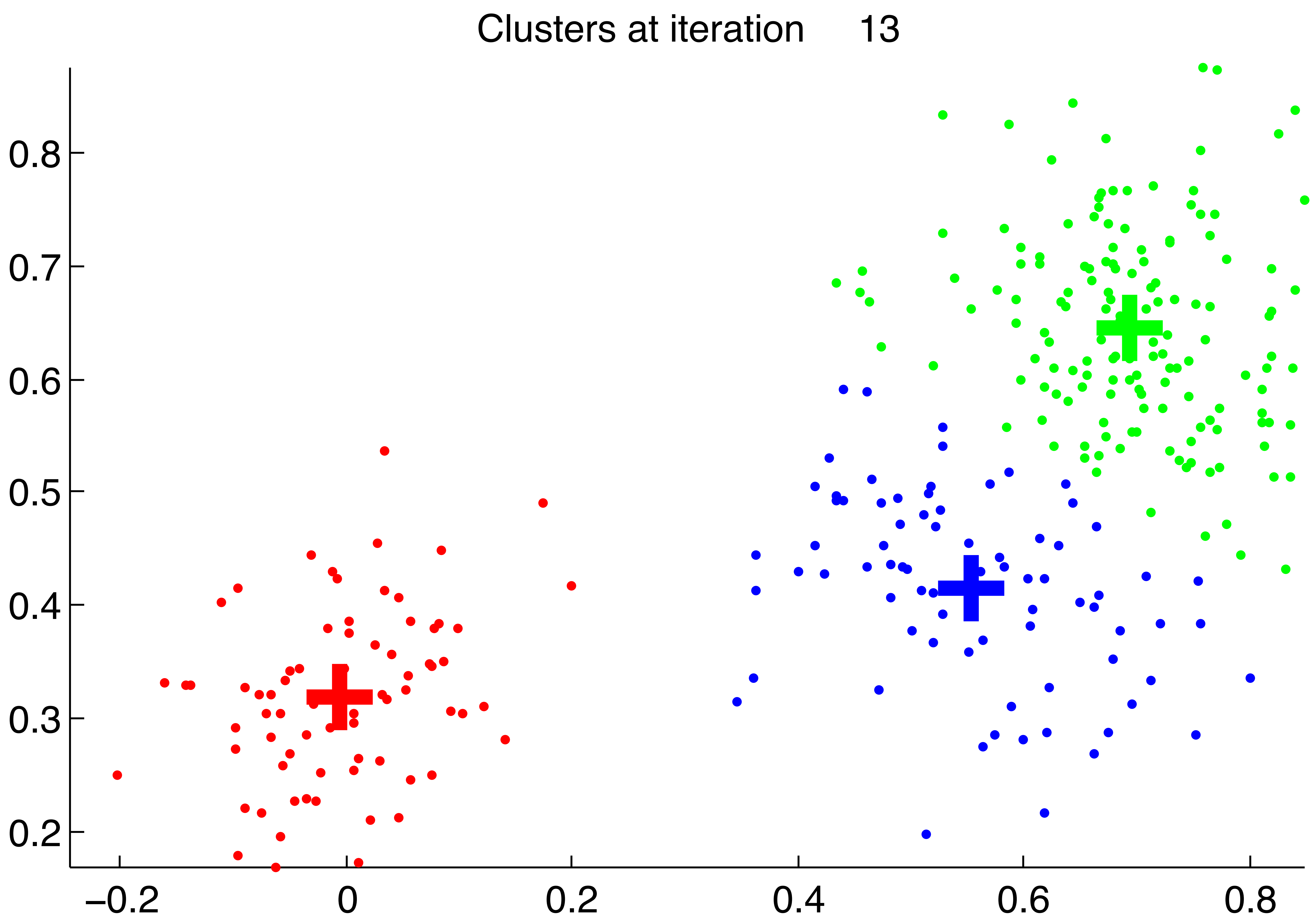
Example 1: K-Means Clustering



Example 1: K-Means Clustering



Example 1: K-Means Clustering



Example 2: Mixed Vegetables

Original Image

Segmentation Using Colour

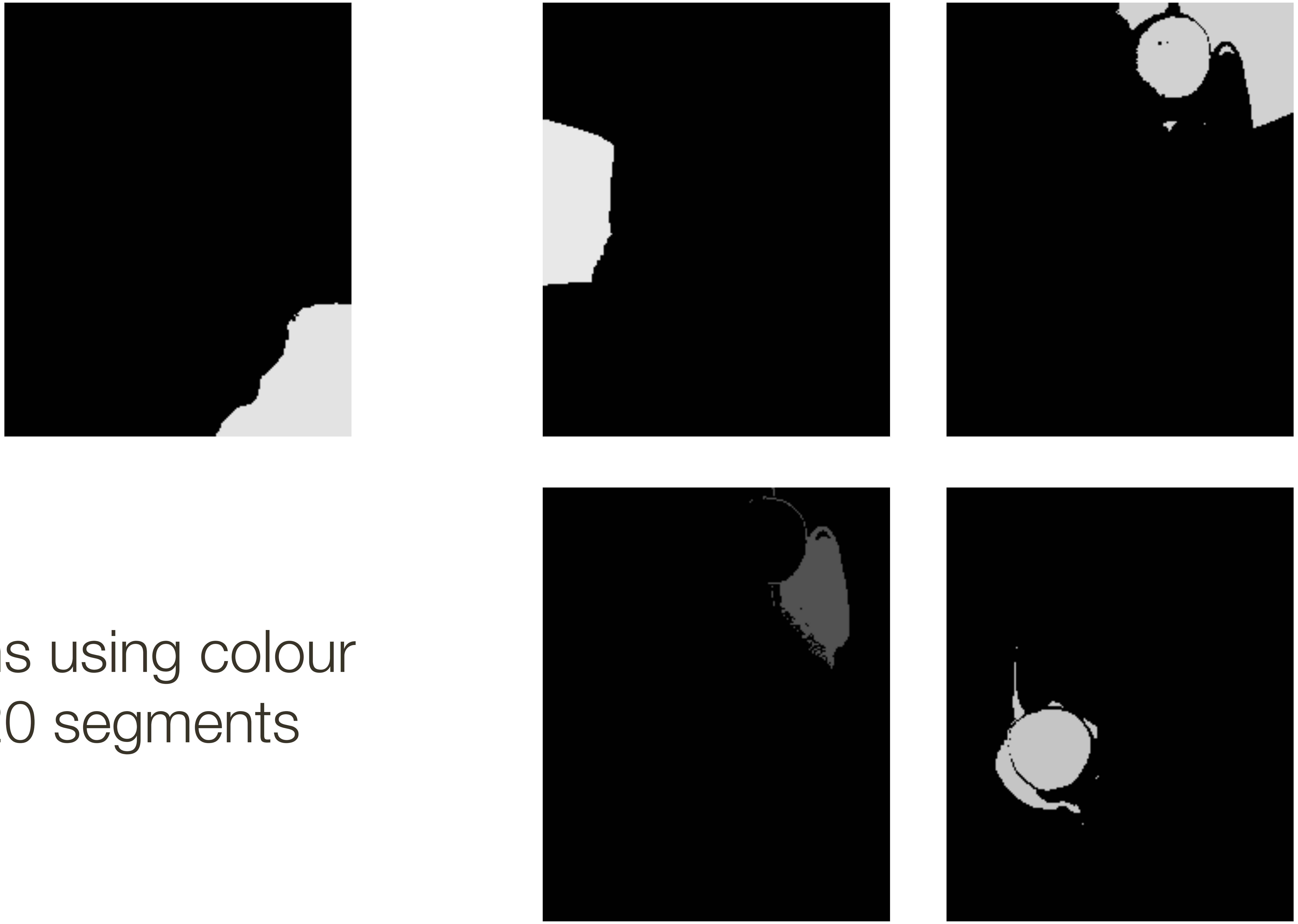
K-means using colour alone, 11 segments

Example 2: Mixed Vegetables

K-means using colour alone, 11 segments

Forsyth & Ponce (2nd ed.) Figure 9.18

Example 2: Mixed Vegetables



K-means using colour alone, 20 segments

Forsyth & Ponce (2nd ed.) Figure 9.19

An **Exercise**

Sketch an example of a 2D dataset for which agglomerative clustering performs well (finds the two true clusters) but K-means clustering fails.

An **Exercise**

Sketch an example of a 2D dataset for which agglomerative clustering performs well (finds the two true clusters) but K-means clustering fails.

Discussion of K-Means

Advantages:

— Algorithm always converges

— Easy to implement

Disadvantages:

— The number of classes, K, needs to be given as input

— Algorithm doesn’t always converge to the (globally) optimal solution

— Limited to compact/spherical clusters

Segmentation by Clustering

We just saw a simple example of segmentation based on colour and position, but segmentation typically makes use of a richer set of features.

— texture

— corners, lines, …

— geometry (size, orientation, ...)

Agglomerative Clustering with a Graph

Suppose we represent an image as a weighted graph.

Any pixels that are **neighbours** are connected by an edge.

Each edge has a weight that measures the similarity between the pixels
— can be based on colour, texture, etc.
— low weights → similar, high weights → different

We will segment the image by performing an agglomerative clustering guided by this graph.

Agglomerative Clustering with a Graph

Recall that we need to define the inter-cluster distance for agglomerative clustering. Let

$$d(C_1, C_2) = \min_{v_1 \in C_1, v_2 \in C_2, (v_1, v_2) \in \epsilon} w(v_1, v_2)$$

We also need to determine when to stop merging.

Agglomerative Clustering with a Graph

Denote the 'internal difference' of a cluster as the largest weight in the minimum spanning tree of the cluster, M(C):

$$int(C) = \max_{e \in M(C)} w(e)$$

Agglomerative Clustering with a Graph

Denote the 'internal difference' of a cluster as the largest weight in the minimum spanning tree of the cluster, M(C):

$$int(C) = \max_{e \in M(C)} w(e)$$

This is not going to work for small clusters: $int(C) + \tau(C)$

where $\tau(C) = \frac{k}{|C|}$

Agglomerative Clustering with a Graph

Algorithm: (Felzenszwalb and Huttenlocher, 2004)

Make each point a separate cluster.
Sort edges in order of non-decreasing weight so that $w(e_1) \geq w(e_2) \geq \cdots \geq w(e_r)$
For $i = 1$ to r
    If both ends of e_i lie in the same cluster
        Do nothing
    Else
        One end is in cluster C_l and the other is in cluster C_m
        If $d(C_l, C_m) \leq MInt(C_l, C_m)$
            Merge C_l and C_m Report the remaining set of clusters.

Report the remaining set of clusters.

Agglomerative Clustering with a Graph

Image credit: KITTI Vision Benchmark

Summary

To use standard clustering techniques we must define an **inter-cluster** distance measure

A **dendrogram** visualizes a hierarchical clustering process

K-means is a clustering technique that iterates between

1. Assume the cluster centers are known. Assign each point to the closest cluster center.

2. Assume the assignment of points to clusters is known. Compute the best cluster center for each cluster (as the mean).

K-means clustering is initialization dependent and converges to a local minimum