INTRODUCTION TO MACHINE LEARNING

Clustering with k-means

Clustering, what?

- **Cluster**: collection of objects
    - *Similar* within cluster
    - *Dissimilar* between clusters
- **Clustering**: grouping objects in clusters
    - No labels: *unsupervised* classification
    - Plenty possible clusterings

Clustering, why?

- Pattern Analysis
- Visualise Data
- pre-Processing Step
- Outlier Detection
- ...
- Targeted Marketing Programs
- Student Segmentations
- Data Mining
- ...

Clustering, how?

- **Measure of Similarity:** *d(..., ...)*
  - Numerical variables → Metrics: Euclidean, Manhattan, …
  - Categorical variables → Construct your own distance

- **Clustering Methods**
  - k-means
  - Hierarchical ← **Many variations**
  - …

Compactness and Separation

- **Within Cluster Sums of Squares (WSS):**

$$\text{WSS} = \sum_{i=1}^{N_C} \sum_{x \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_{\mathbf{C_i}})^2$$

$\bar{\mathbf{x}}_{\mathbf{C_i}}$ Cluster Centroid
$\mathbf{x}$ Object
C_i Cluster
N_C #Clusters

Measure of **compactness** → **Minimise WSS**

- **Between Cluster Sums of Squares (BSS):**

$$\text{BSS} = \sum_{i=1}^{N_C} |C_i| \cdot d(\bar{\mathbf{x}}_{\mathbf{C_i}}, \bar{\mathbf{x}})^2$$

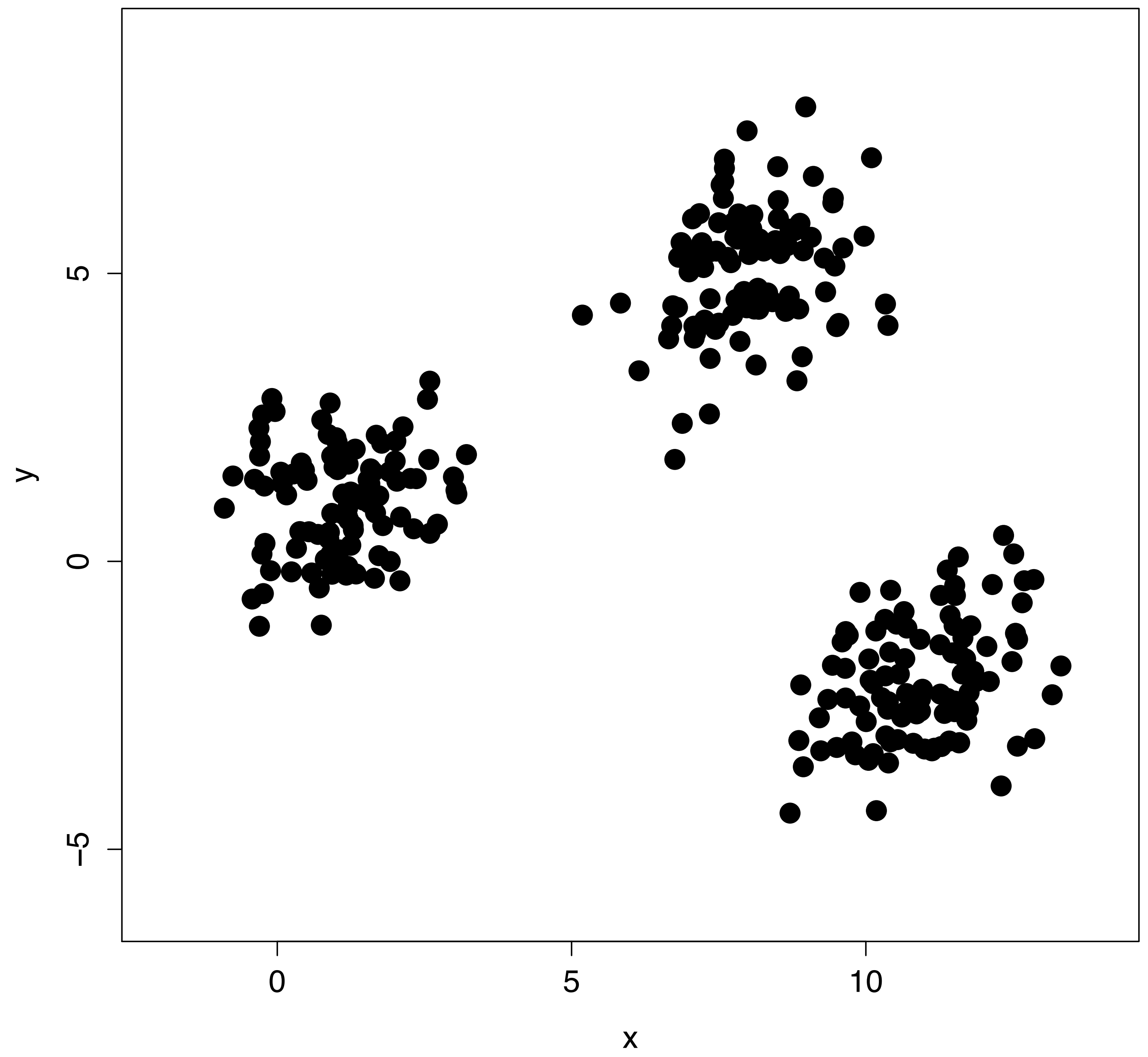
$\bar{\mathbf{x}}_{\mathbf{C_i}}$ Cluster Centroid
N_C #Clusters
$|C_i|$ #Objects in Cluster
$\bar{\mathbf{x}}$ Sample Mean

Measure of **separation** → **Maximise BSS**

k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

Let's take k = 3

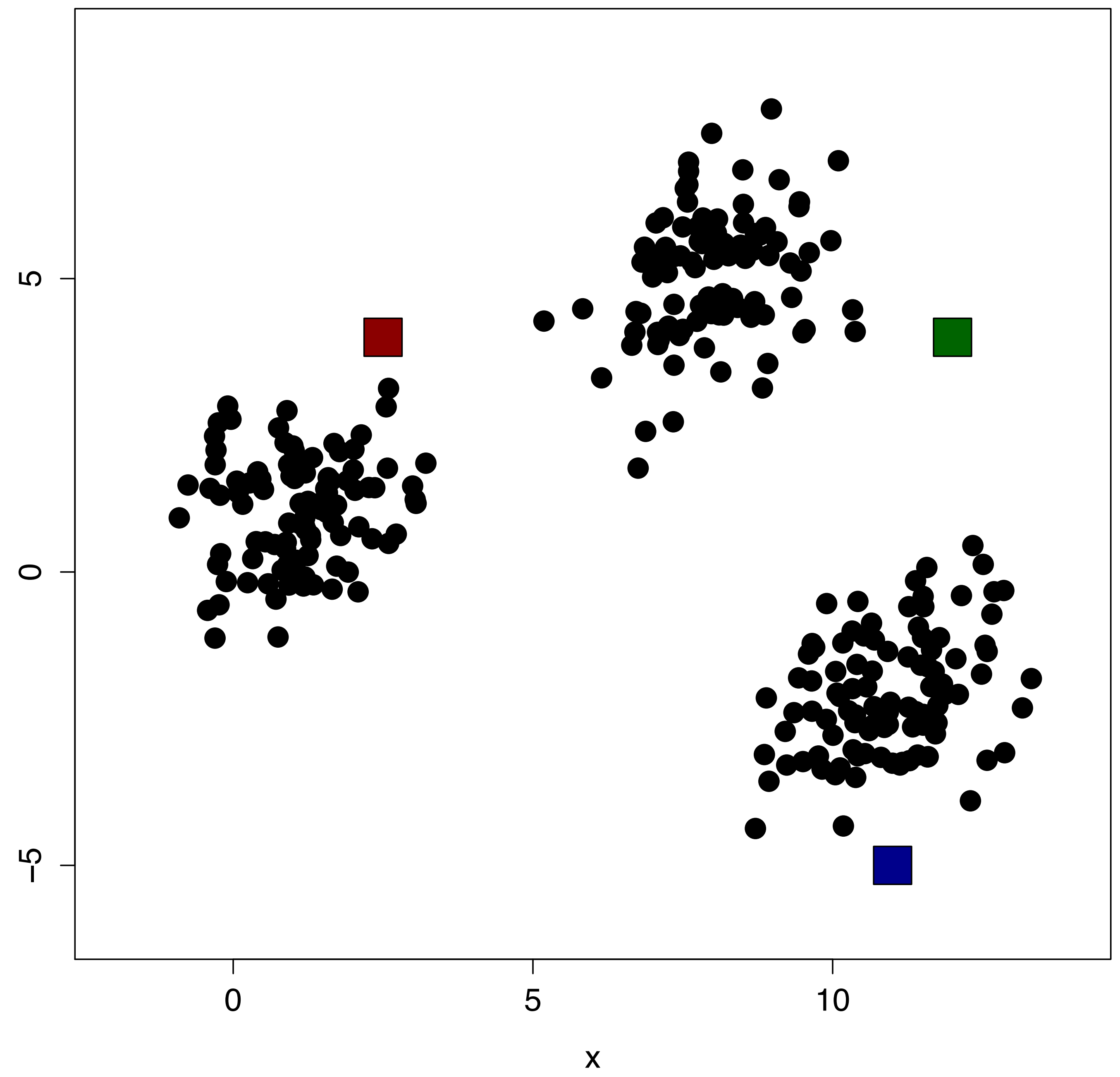


k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

1. Randomly assign k *centroids*

k = 3

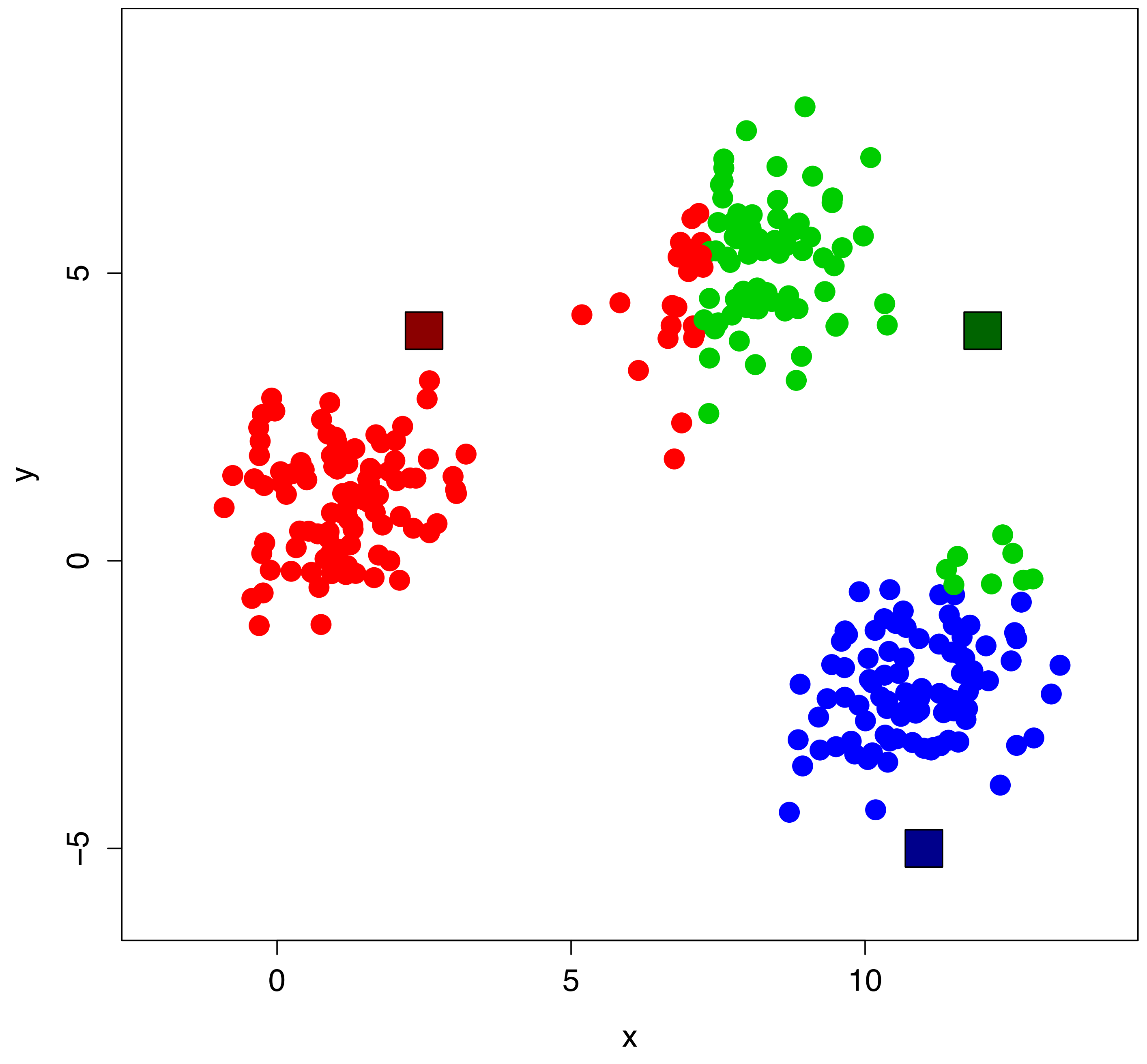


k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

1. Randomly assign k *centroids*
2. Assign data to *closest* centroid

k = 3

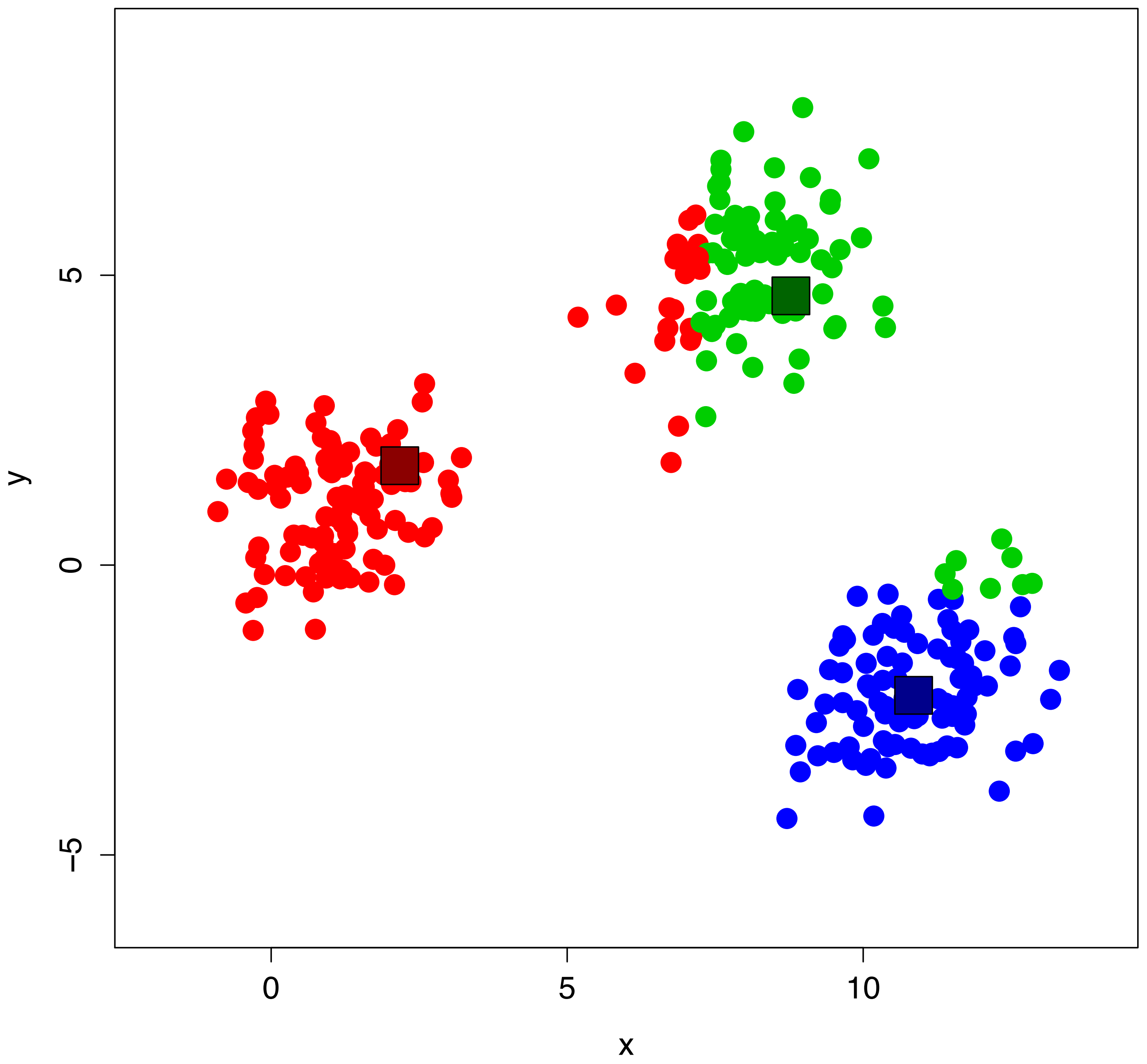


k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

1. Randomly assign k *centroids*
2. Assign data to *closest* centroid
3. Moves centroids to *average* location

k = 3

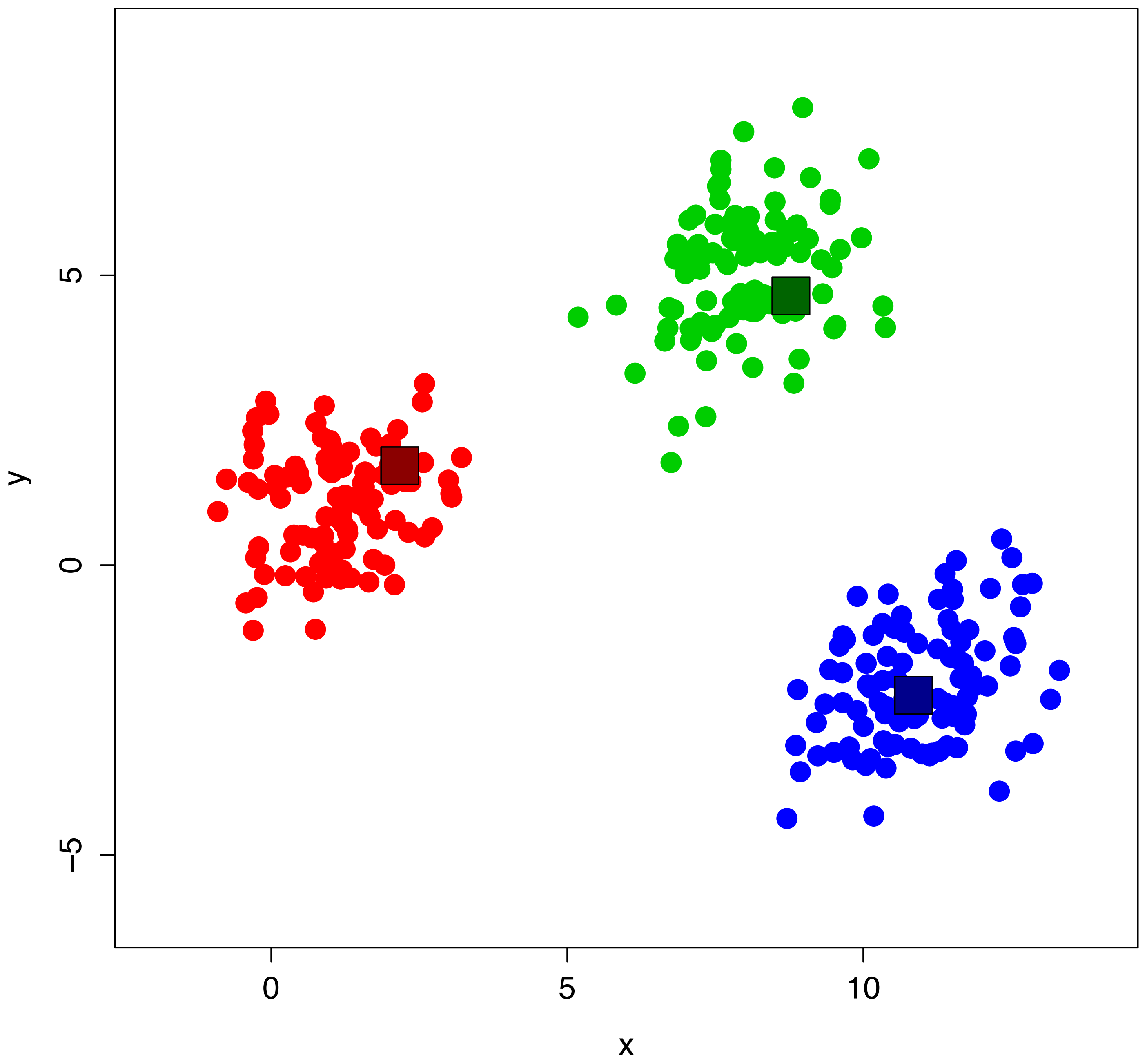


k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

1. Randomly assign k *centroids*
2. Assign data to *closest* centroid
3. Moves centroids to *average* location
4. Repeat step 2 and 3

k = 3

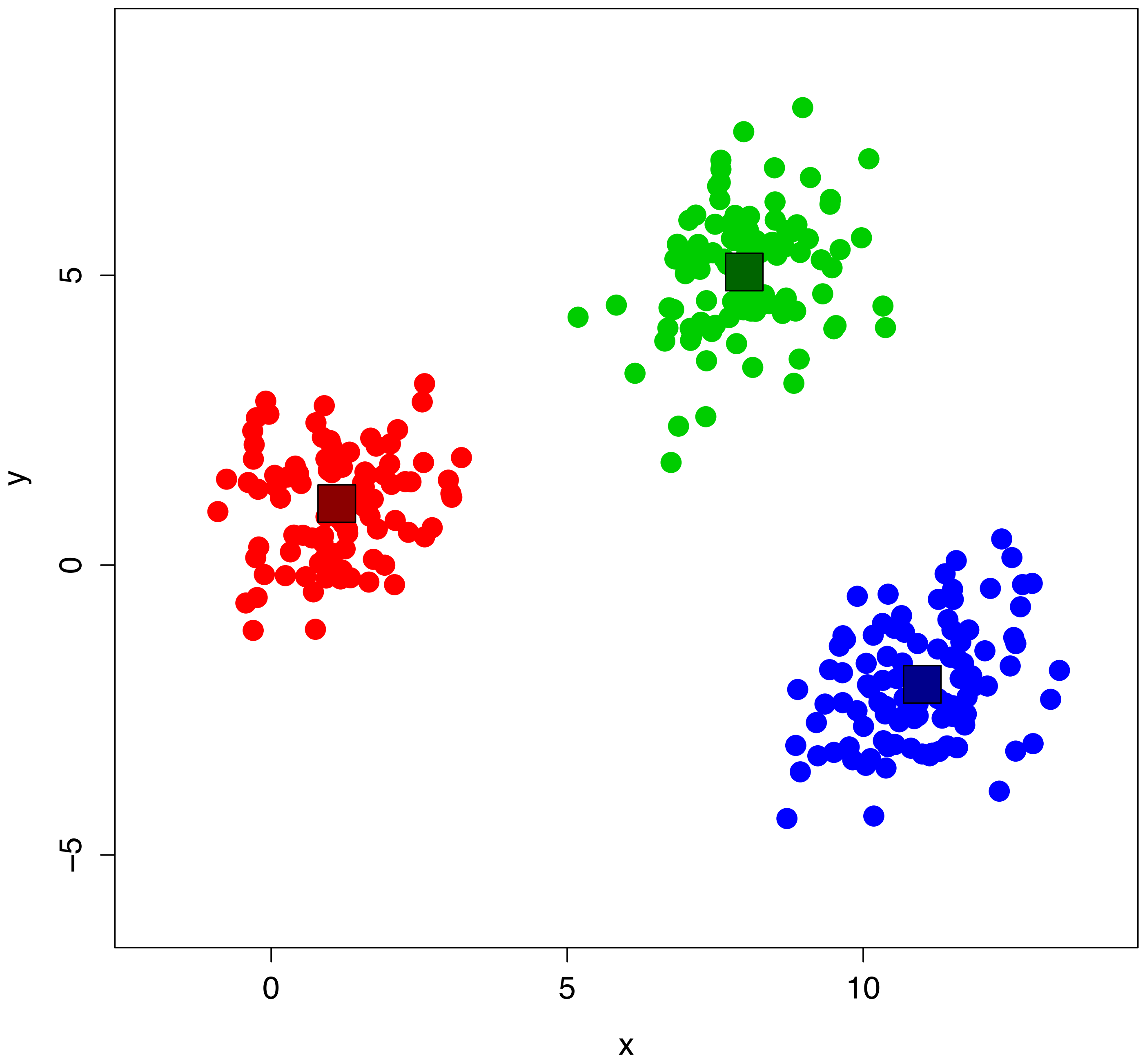


k-Means Algorithm

Goal: Partition data in k *disjoint* subsets

1. Randomly assign k *centroids*
2. Assign data to *closest* centroid
3. Moves centroids to *average* location
4. Repeat step 2 and 3

k = 3



→ The algorithm has converged!

Choosing k

- **Goal:** Find **k** that minimizes WSS
- **Problem**: WSS keeps decreasing as **k** increases!
- **Solution**: WSS starts decreasing slowly

$$\left.\begin{array}{r}\text{WSS starts decreasing slowly}\\ \text{WSS / TSS} < 0.2\end{array}\right\}\ \text{Fix } \mathbf{k}$$

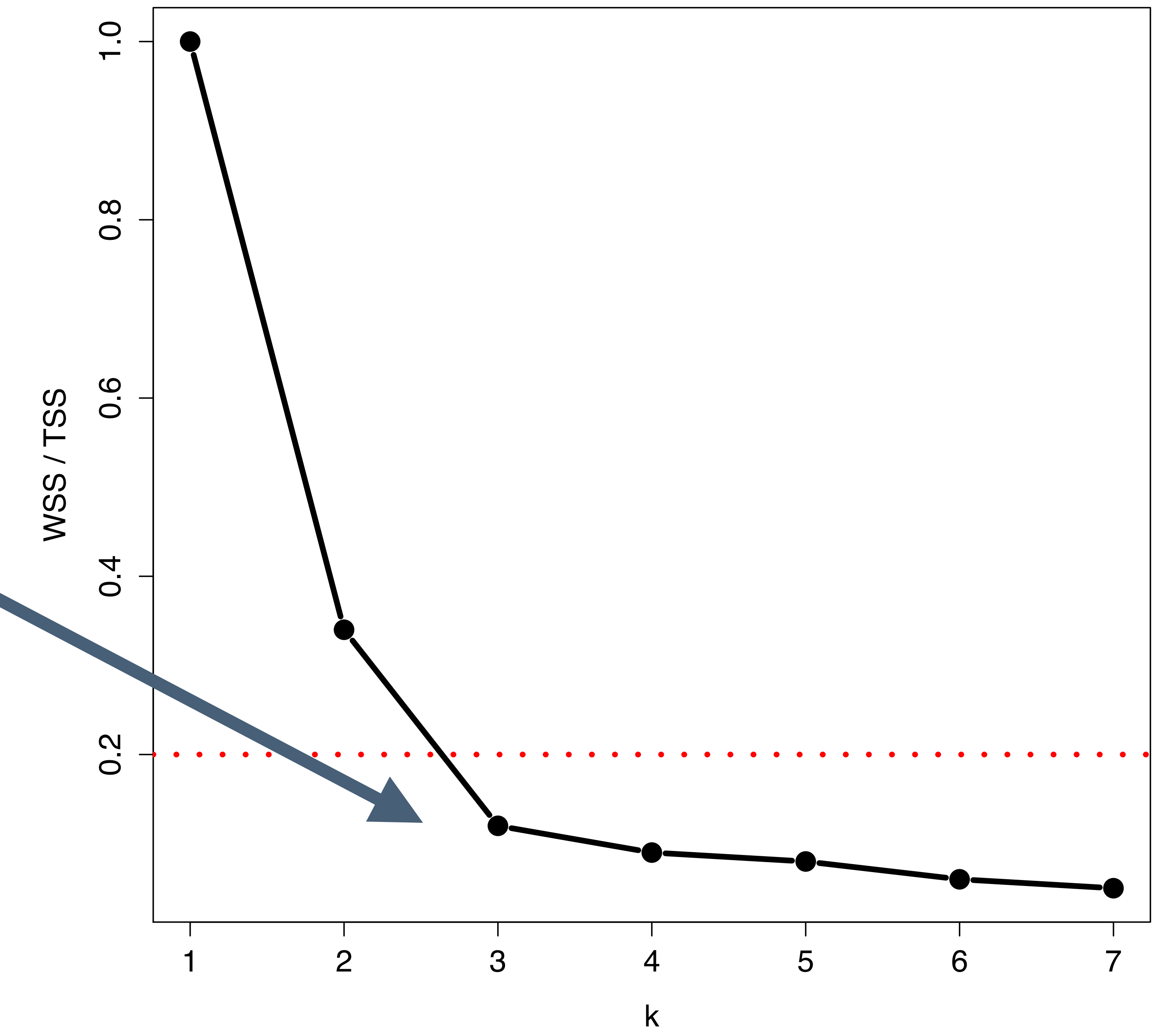
$$\text{TSS} = \text{WSS} + \text{BSS}$$

Choosing k: Scree Plot

Scree Plot: Visualizing the ratio WSS / TSS as function of k

Look for the *elbow* in the plot

Choose **k** = 3



k-Means in R

```
> my_km <- kmeans(data, centers, nstart)
```

- **centers:** Starting centroid or #clusters
- **nstart:** #times R restarts with different centroids

Distance: Euclidean metric

```
> my_km$tot.withinss     ← WSS

> my_km$betweenss        ← BSS
```

Let's practice!

Performance and Scaling

Cluster Evaluation

Not trivial! There is no truth

- No **true** labels
- No **true** response

Evaluation methods? Depends on the **goal**

Goal: Compact and Separated ⟵ **Measurable!**

Cluster Measures

WSS and BSS: Good indication

Underlying idea:

- Variance within clusters
- Separation between clusters

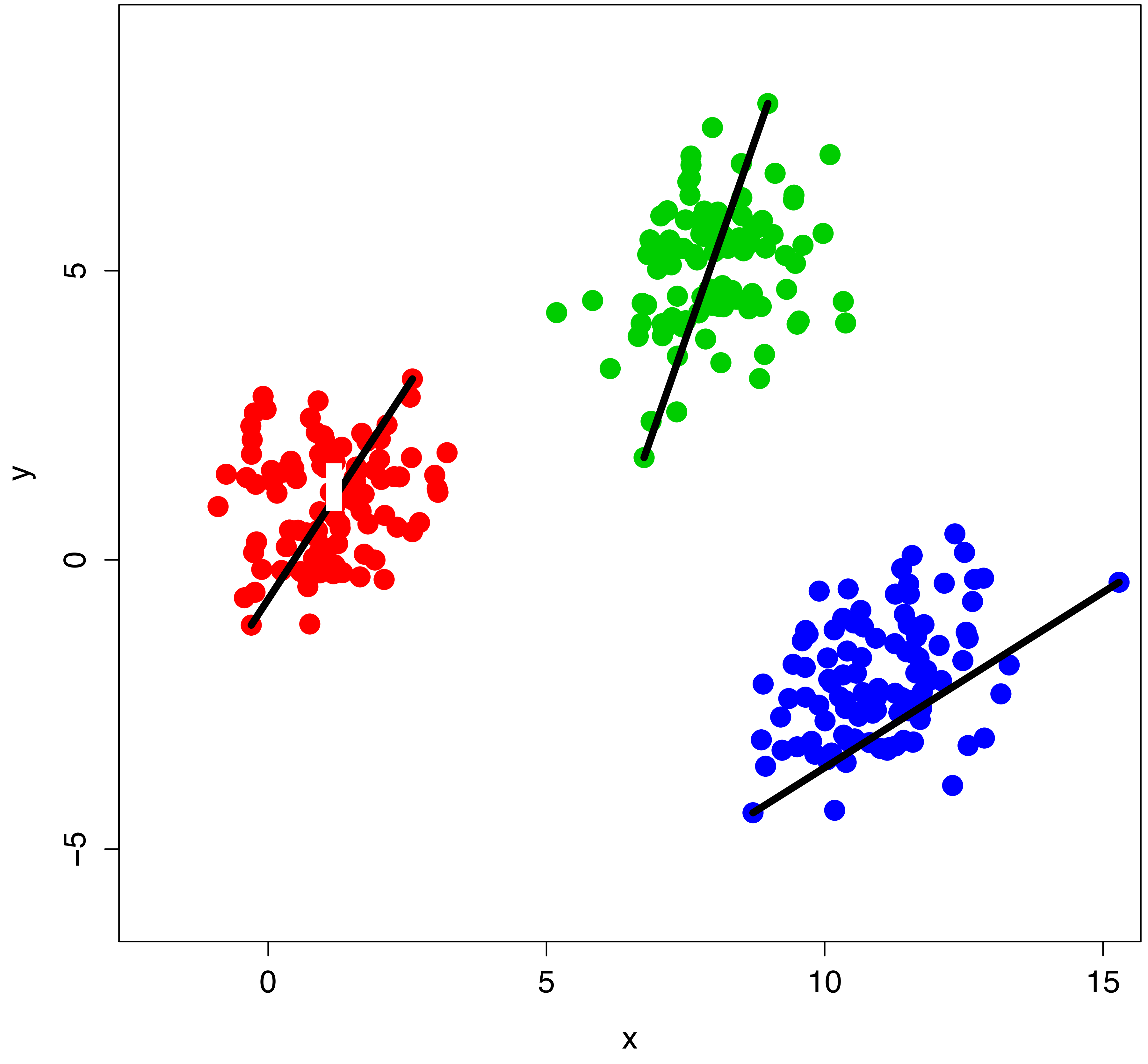
Compare

Alternative:

- Diameter
- Intercluster Distance

Diameter

$$\text{Dia}_i = \max_{x,y \in C_i} d(x, y)$$



x, y : Objects

C_i : Cluster

d : Distance (objects)

Measure of Compactness

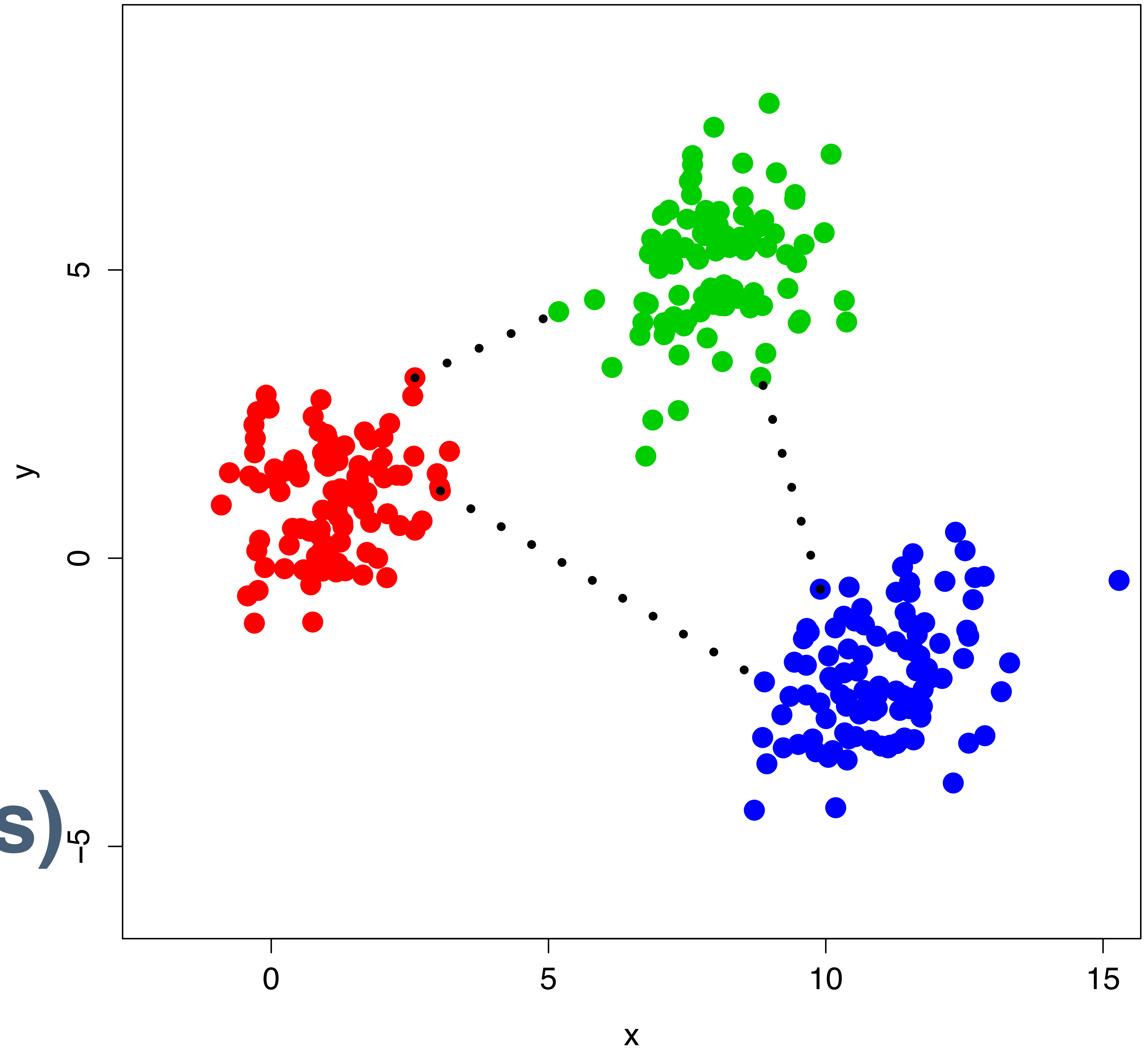
Intercluster Distance

$$\delta(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

x , y : Objects

C_i , C_j : Clusters

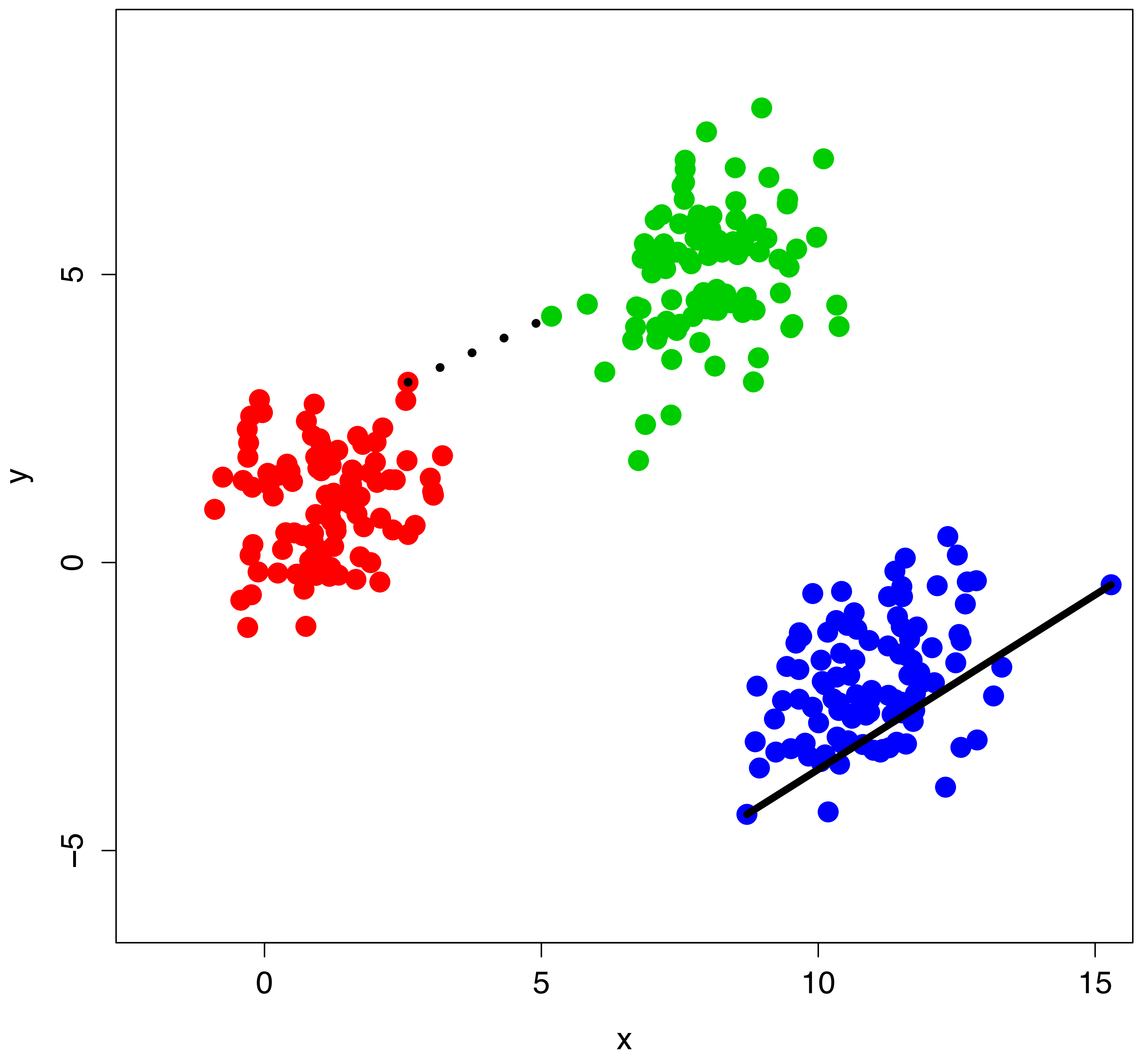
d : Distance (objects)



Measure of Separation

Dunn's Index

$$\frac{\min\limits_{1 \leq i < j \leq k} \delta(C_i, C_j)}{\max\limits_{1 \leq m \leq k} \mathrm{Dia}_m}$$



Dunn's Index

$$\frac{\min\limits_{1 \leq i < j \leq k} \delta(C_i, C_j)}{\max\limits_{1 \leq m \leq k} \mathrm{Dia}_m}$$

Higher Dunn → Better separated / more compact

Notes:

- High computational cost
- Worst case indicator

Alternative measures

- **Internal Validation:** based on intrinsic knowledge
  - BIC Index
  - Silhouette's Index
- **External Validation:** based on previous knowledge
  - Hulbert's Correlation
  - Jaccard's Coefficient

Evaluating in R

Libraries: cluster and clValid

Dunn's Index:

```
> dunn(clusters = my_km, Data = ...)
```

- **clusters**: cluster partitioning vector
- **Data**: original dataset

Scale Issues

Metrics are often scale dependent!

Which pair is most **similar**? (**Age, Income, IQ**)

- X1 = (28, 72000, 120)
- X2 = (56, 73000, 80)
- X3 = (29, 74500, 118)

→

- **Intuition**: (X1, X3)
- **Euclidean**: (X1, X2)

Solution: **Rescale** income / 1000$

Standardizing

Problem: Multiple variables on different scales

Solution: Standardize your data

1. Subtract the mean
2. Divide by the standard deviation

```
> scale(data)
```

Note: Standardizing → Different interpretation

Let's practice!

INTRODUCTION TO MACHINE LEARNING

Hierarchical Clustering

Hierarchical Clustering

Hierarchy:

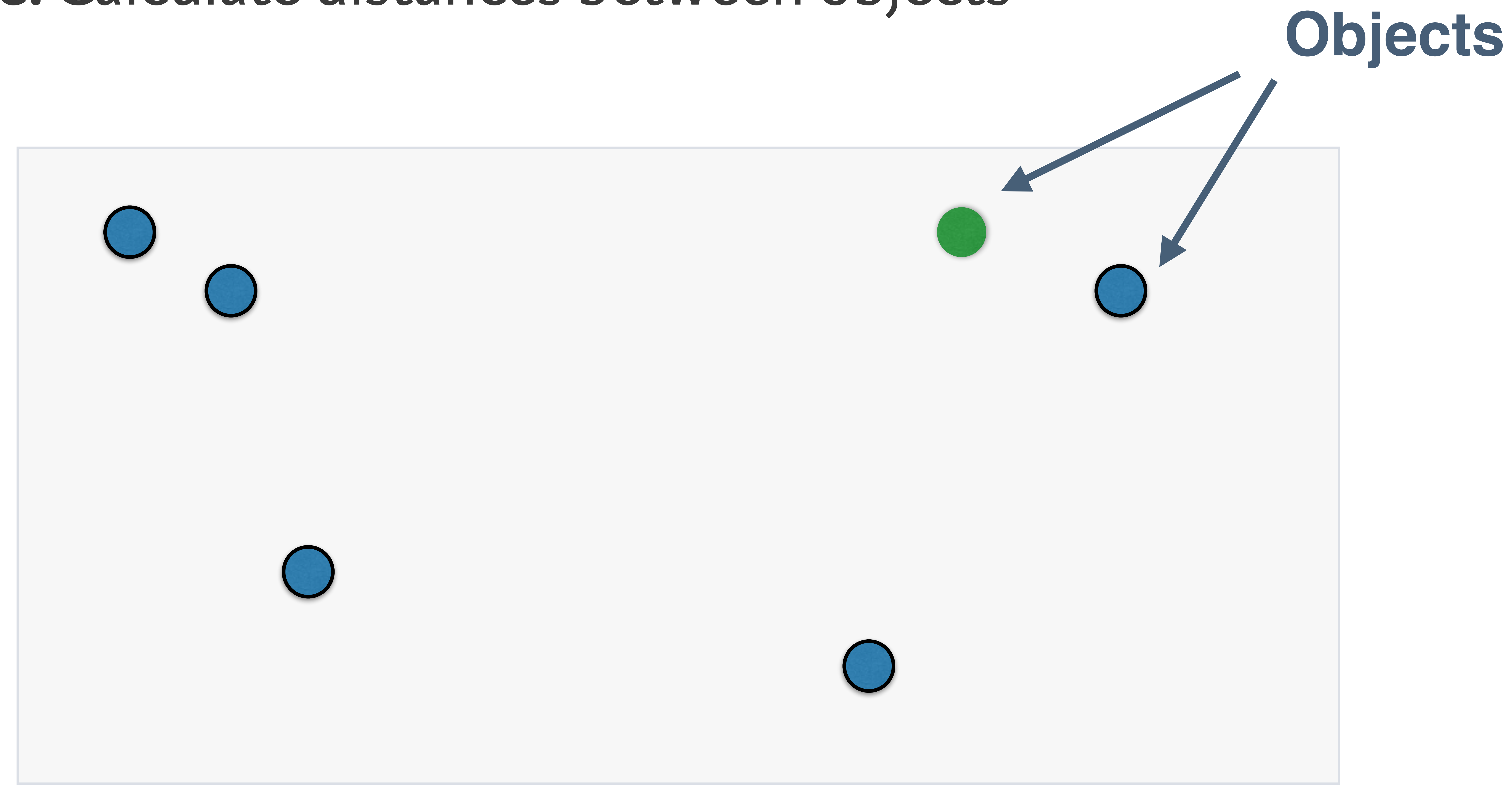
- Which objects cluster first?
- Which cluster pairs merge? When?

Bottom-up:

- Starts from the objects
- Builds a hierarchy of clusters

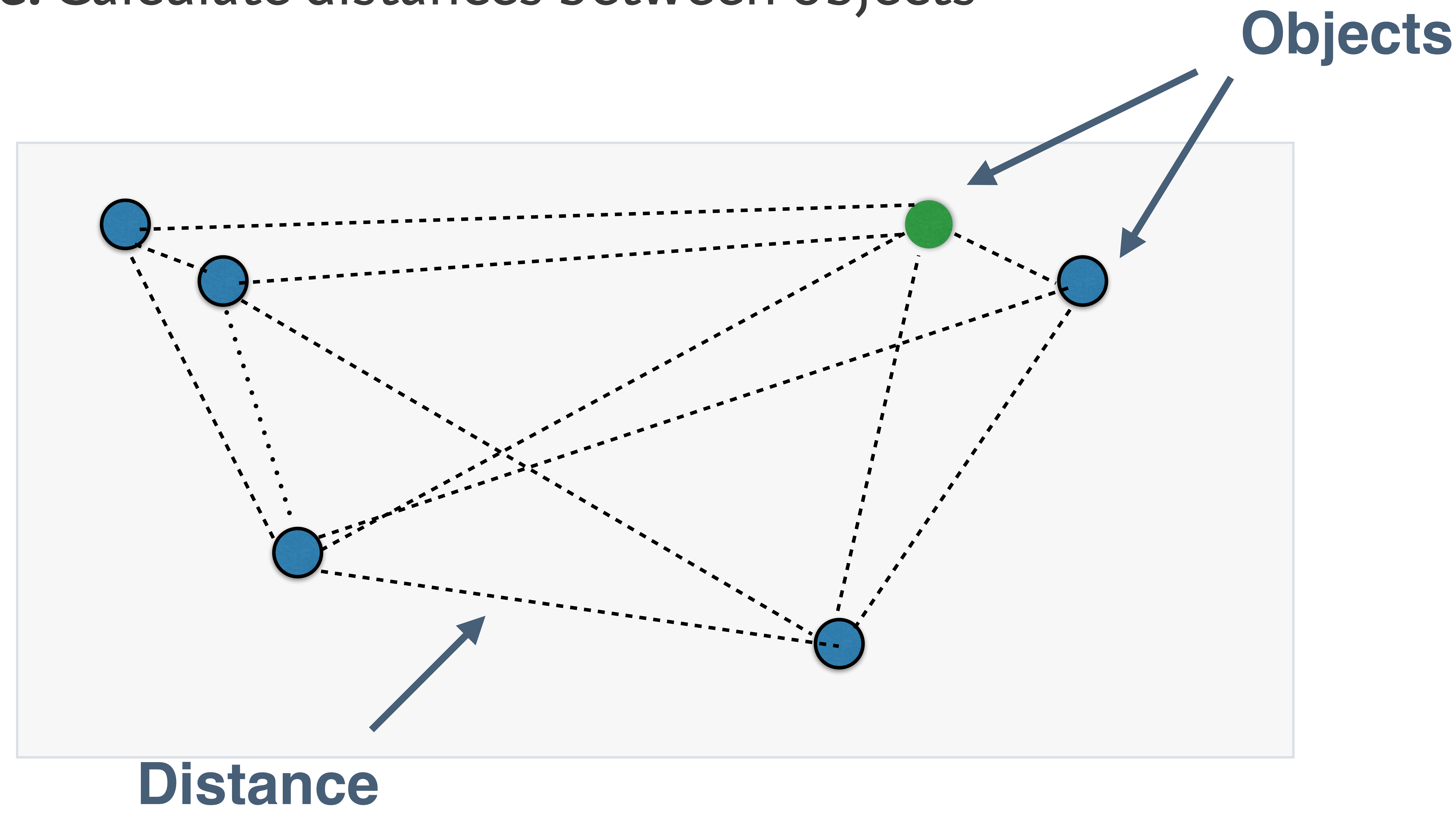
Bottom-Up: Algorithm

Pre: Calculate distances between objects



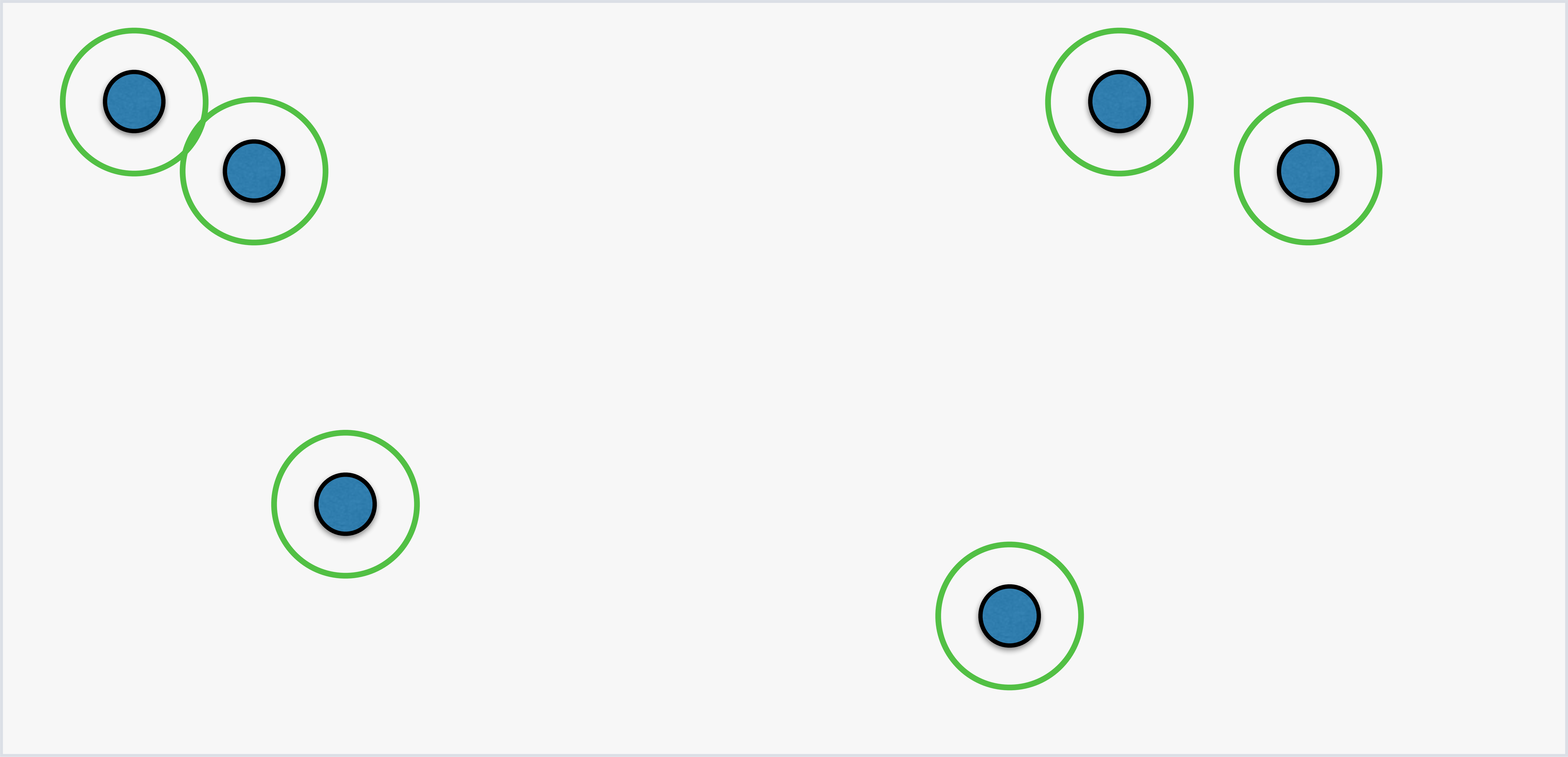
Bottom-Up: Algorithm

Pre: Calculate distances between objects



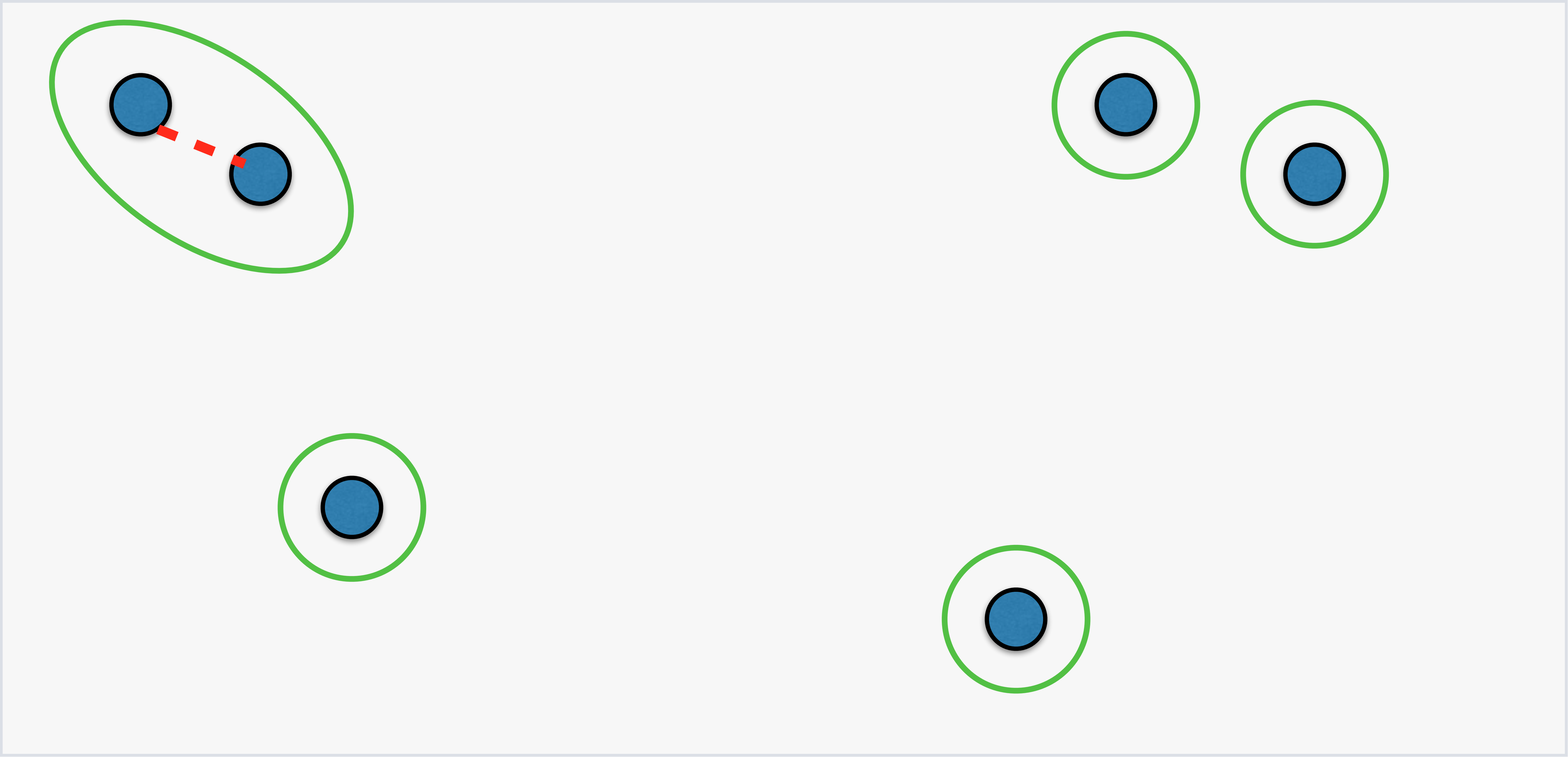
Bottom-Up: Algorithm

1. Put every object in its own cluster



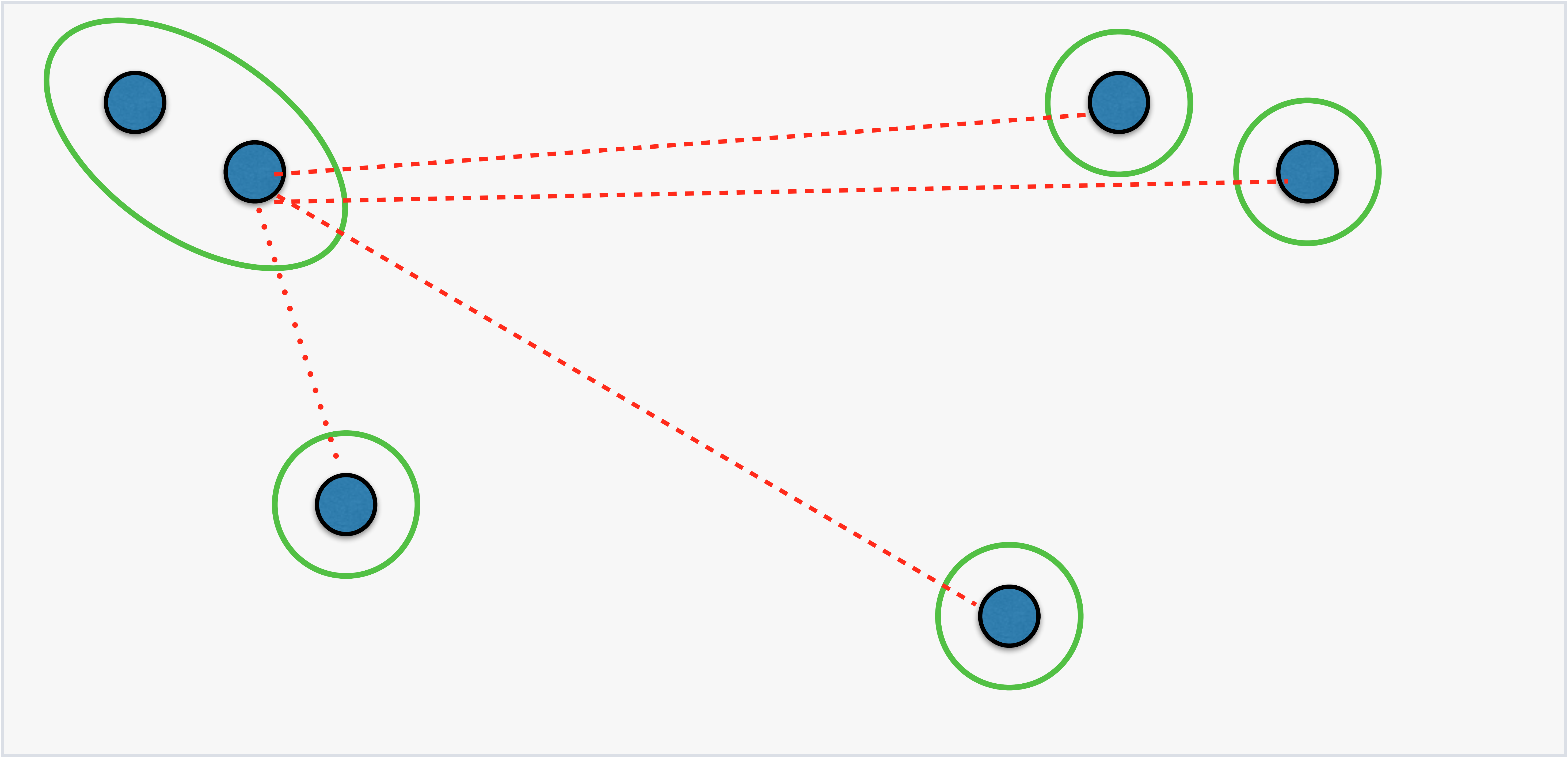
Bottom-Up: Algorithm

2. Find the closest pair of clusters → Merge them



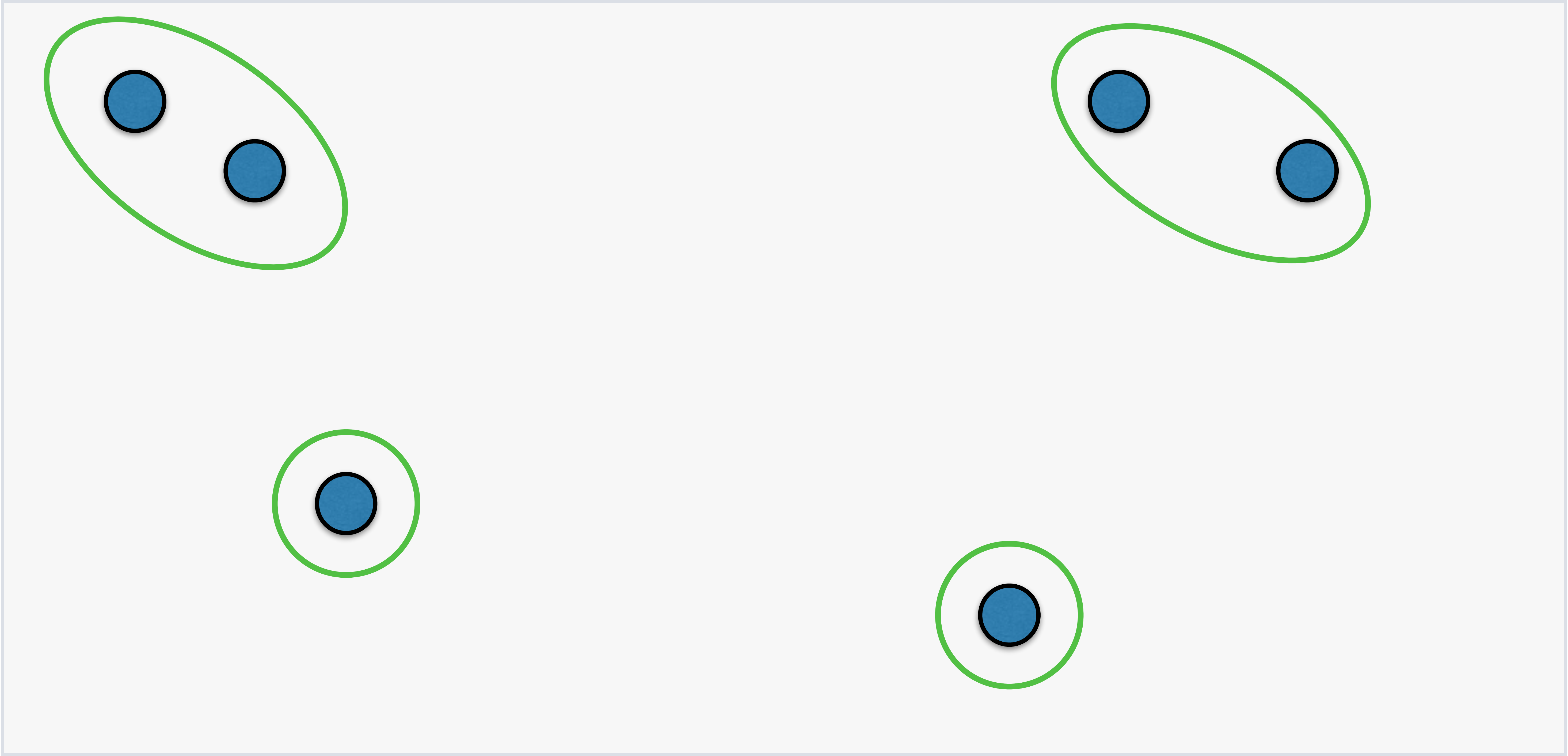
Bottom-Up: Algorithm

3. Compute distances between new cluster and old ones



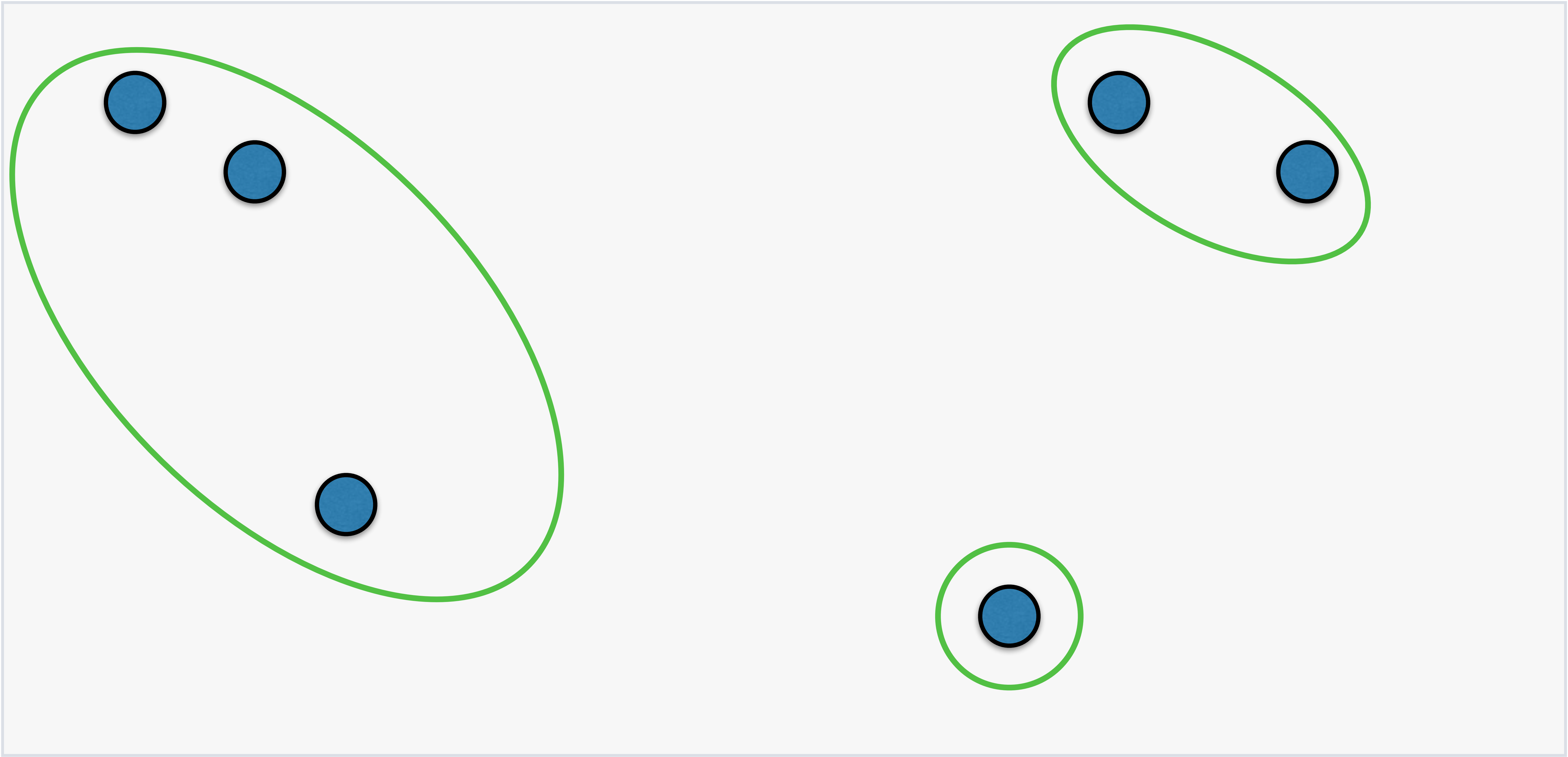
Bottom-Up: Algorithm

4. Repeat steps two and three



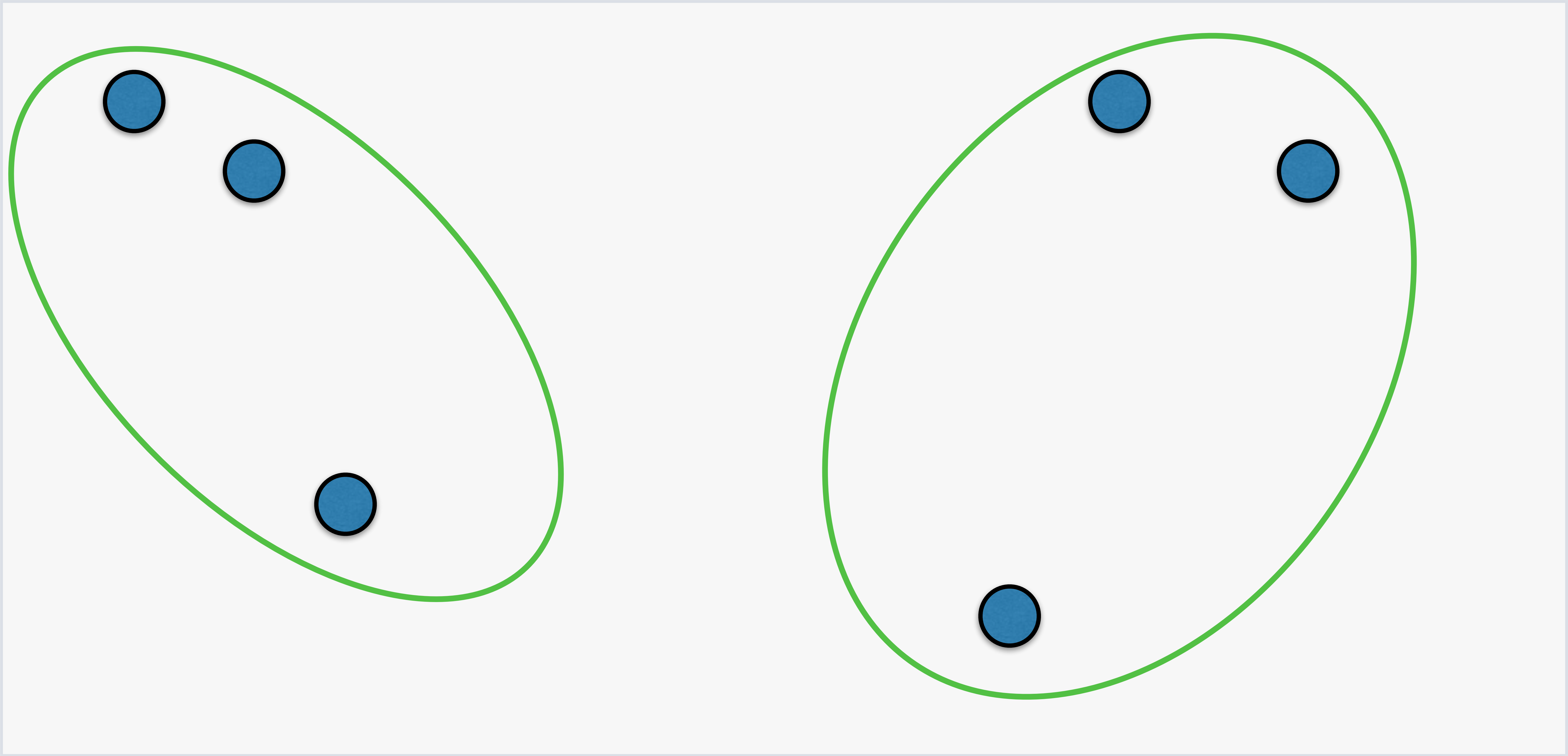
Bottom-Up: Algorithm

4. Repeat steps two and three



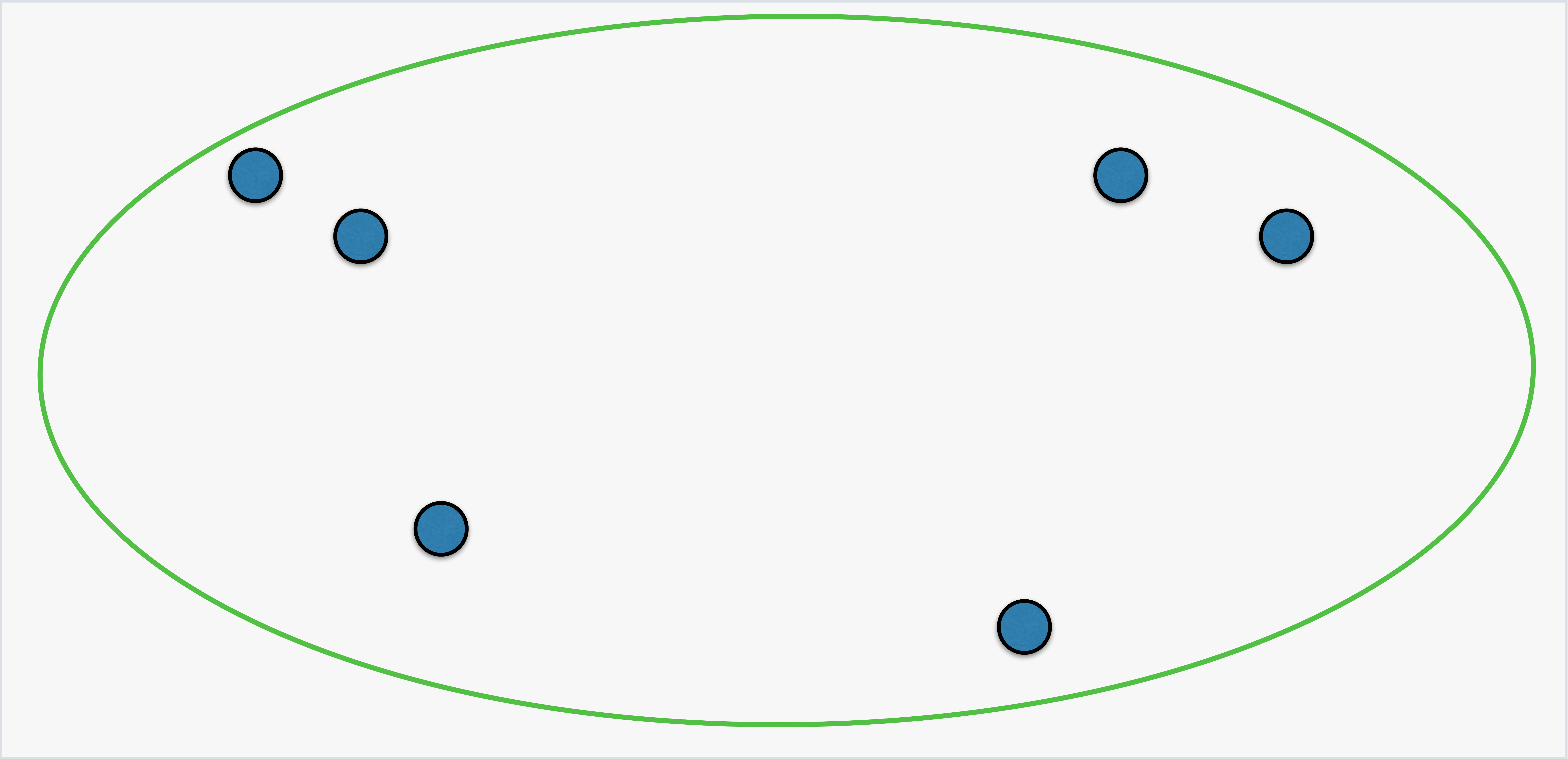
Bottom-Up: Algorithm

4. Repeat steps two and three



Bottom-Up: Algorithm

4. Repeat steps two and three → One cluster



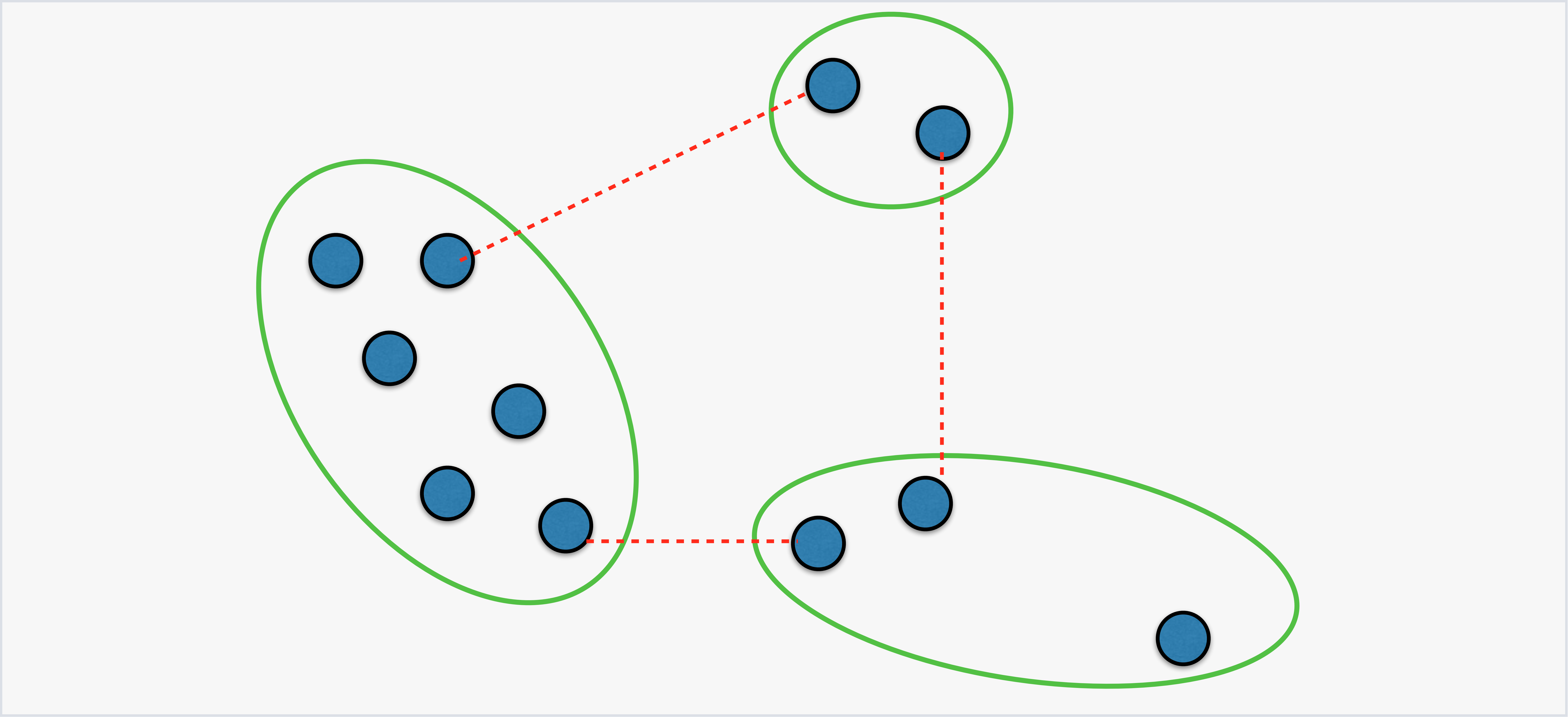
Linkage-Methods

- **Simple-Linkage:** *minimal* distance between clusters
- **Complete-Linkage:** *maximal* distance between clusters
- **Average-Linkage:** *average* distance between clusters

→ Different Clusterings

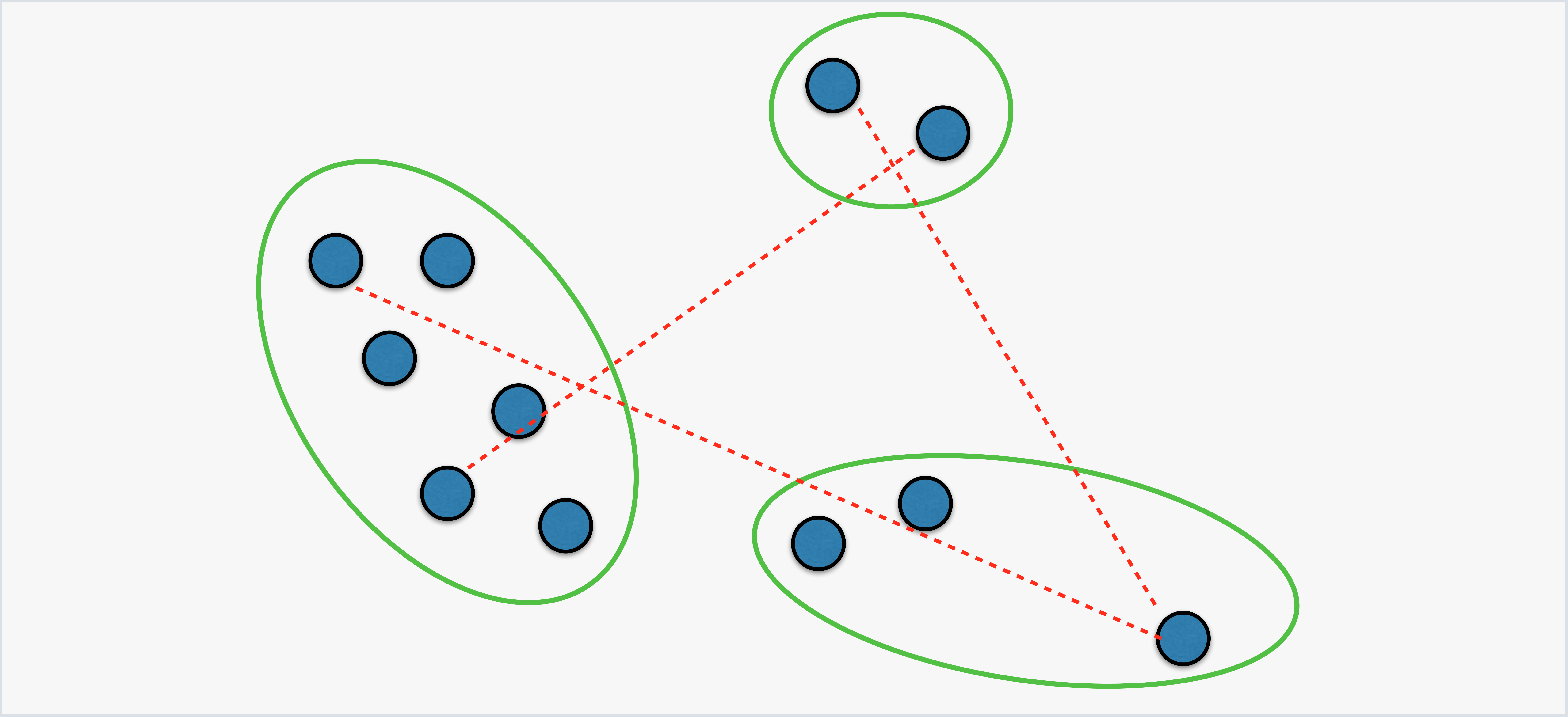
Simple-Linkage

Minimal distance between objects in each clusters

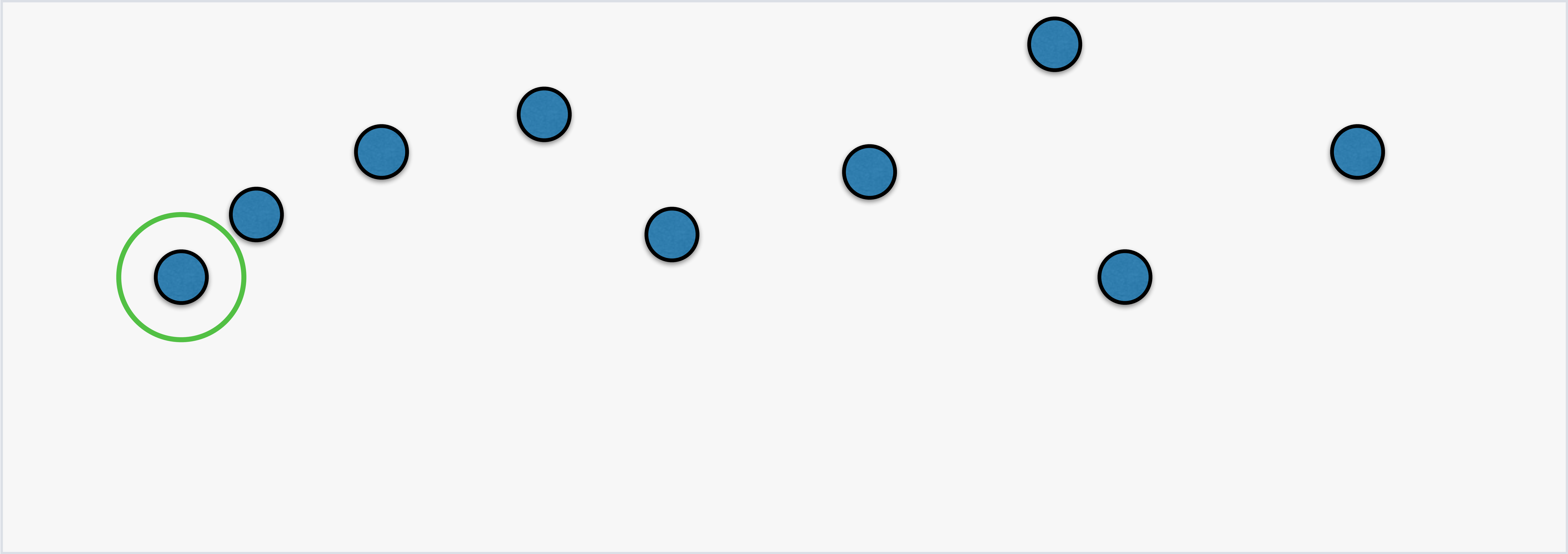


Complete-Linkage

Maximal distance between objects in each cluster

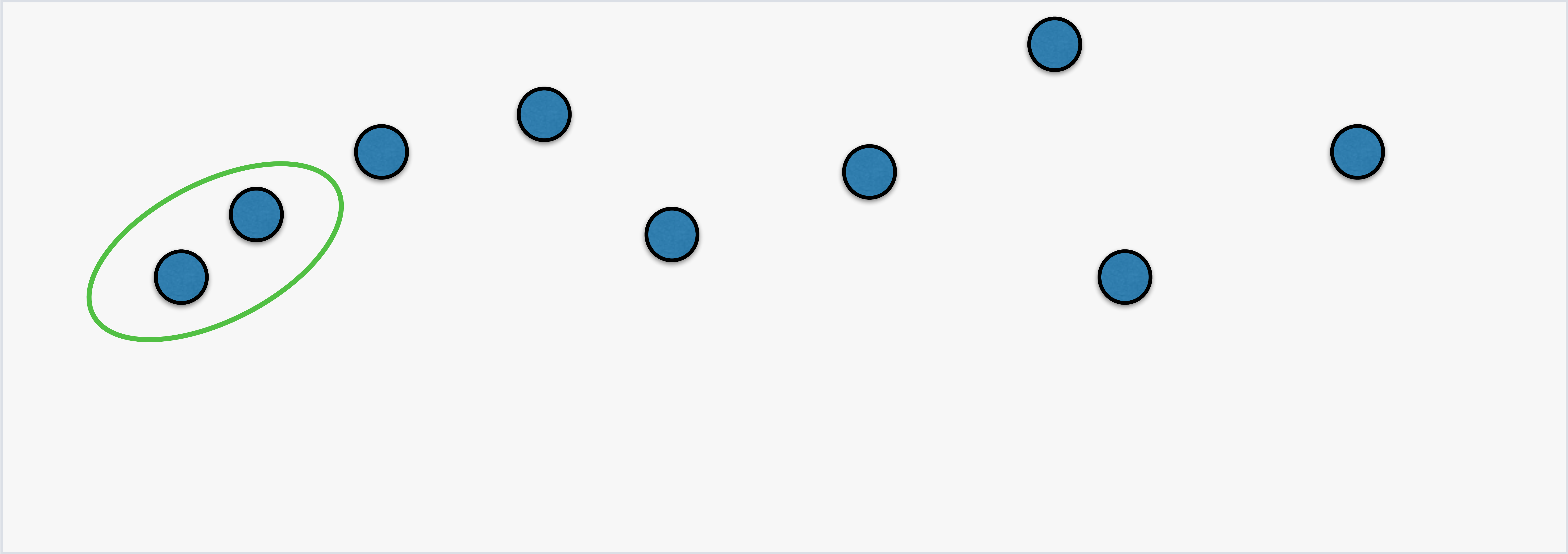


Single-Linkage: Chaining



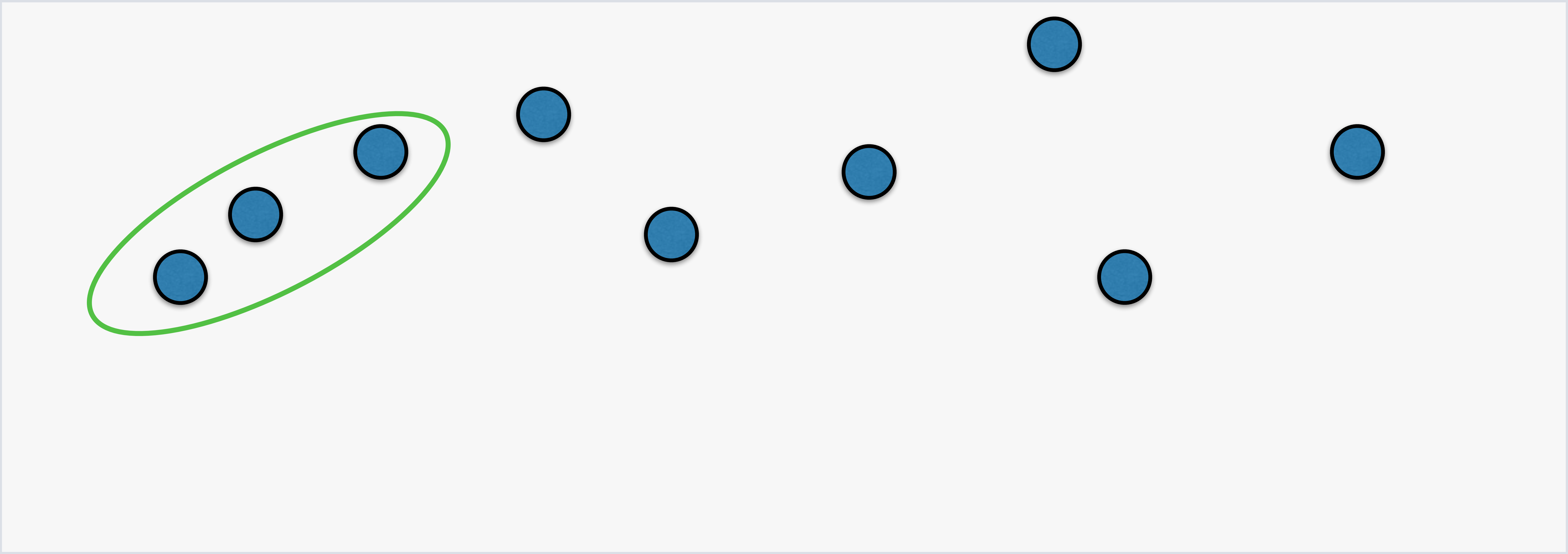
- Often **undesired**

Single-Linkage: Chaining



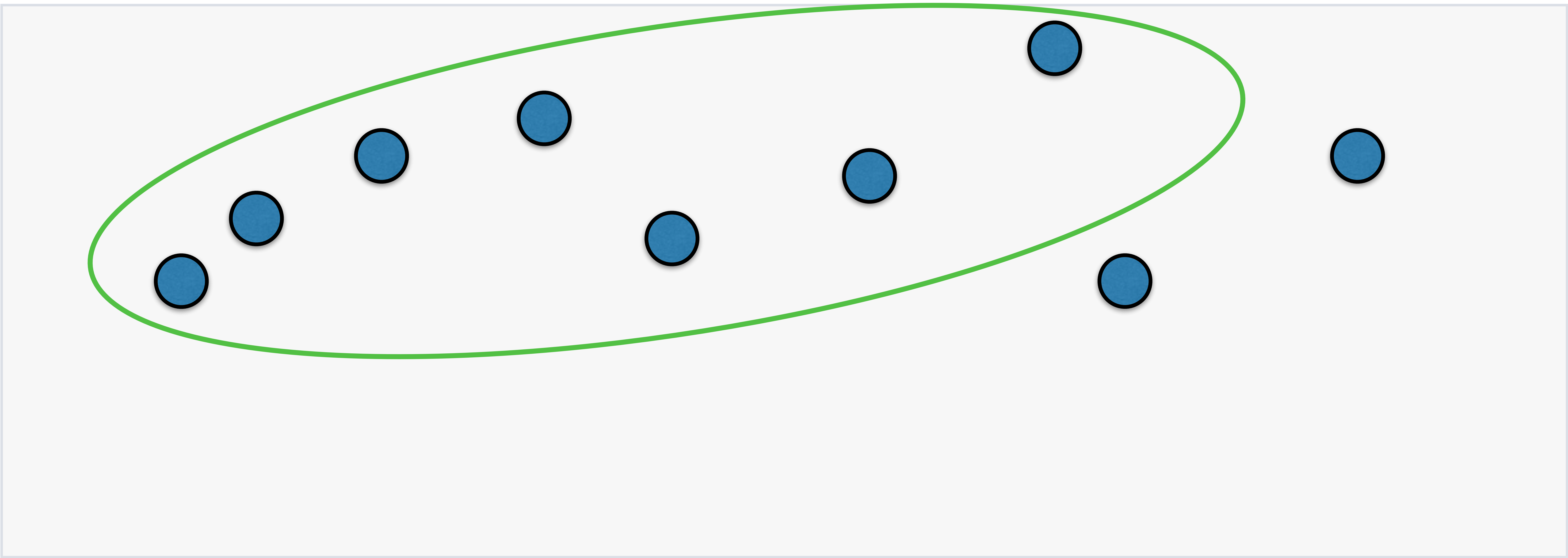
- Often **undesired**

Single-Linkage: Chaining



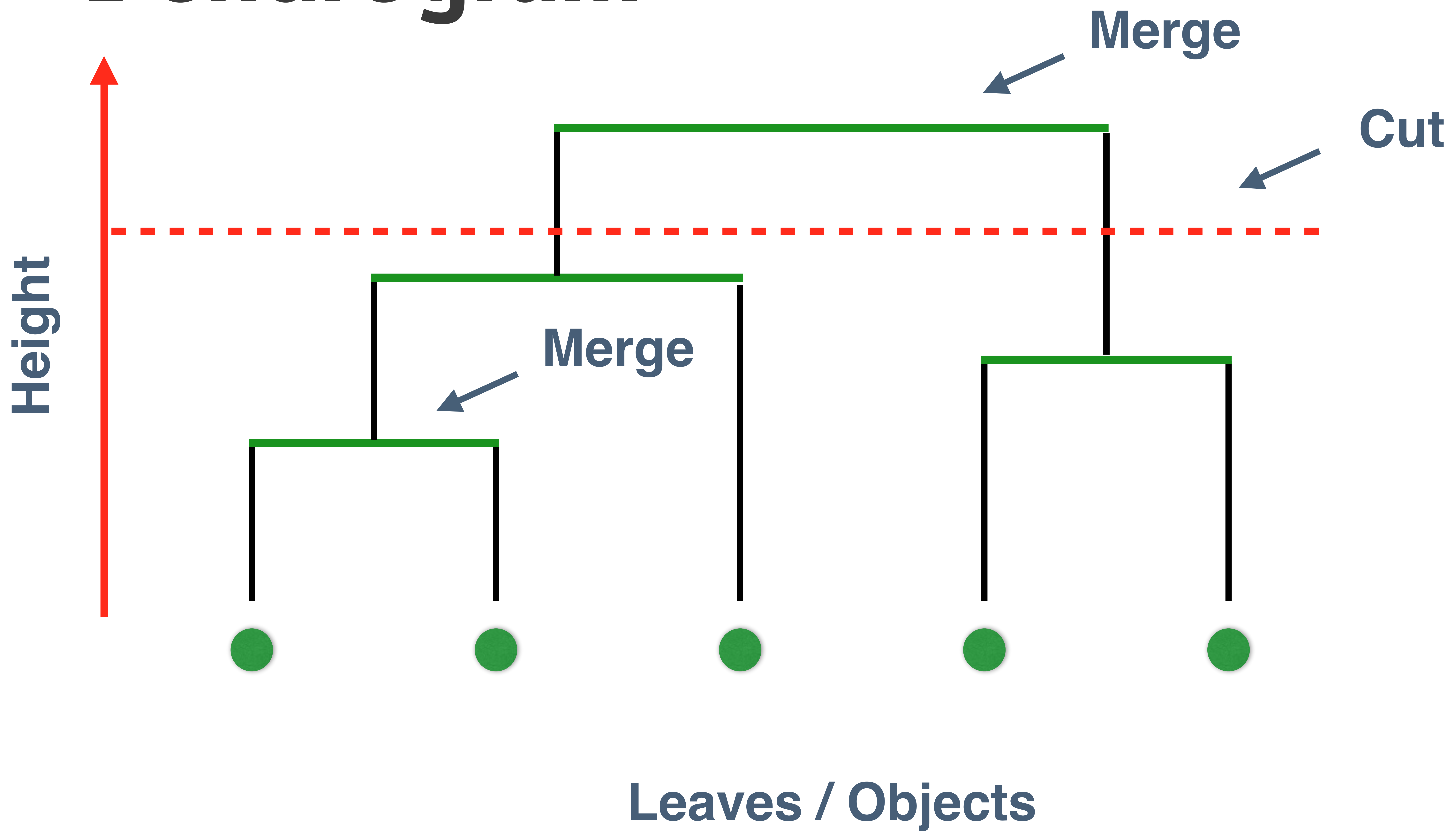
- Often **undesired**

Single-Linkage: Chaining



- Often **undesired**
- Can be great outlier detector

Dendrogram



Hierarchical Clustering in R

Library: stats

```
> dist(x, method)
```

- x: dataset
- method: distance

```
> hclust(d, method)
```

- d: distance matrix
- method: linkage

Hierarchical: Pro and Cons

- Pros
  - In-depth analysis
  - Linkage-methods → Different pattern
- Cons
  - High computational cost
  - Can never undo merges

k-Means: Pro and Cons

- Pros
    - Can undo merges
    - Fast computations
- Cons
    - Fixed #Clusters
    - Dependent on starting centroids

Let's practice!