Web Information Retrieval

Lecture 15
Clustering

Today's Topic: Clustering

- Document clustering
    - Motivations
    - Document representations
    - Success criteria
- Clustering algorithms
    - Partitional
    - Hierarchical

What is clustering?

- **Clustering**: the process of grouping a set of objects into classes of similar objects
  - Documents within a cluster should be similar
  - Documents from different clusters should be dissimilar
- The commonest form of **unsupervised learning**
  - Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples is given
- A common and important task that finds many applications in IR and other places

# A data set with clear cluster structure
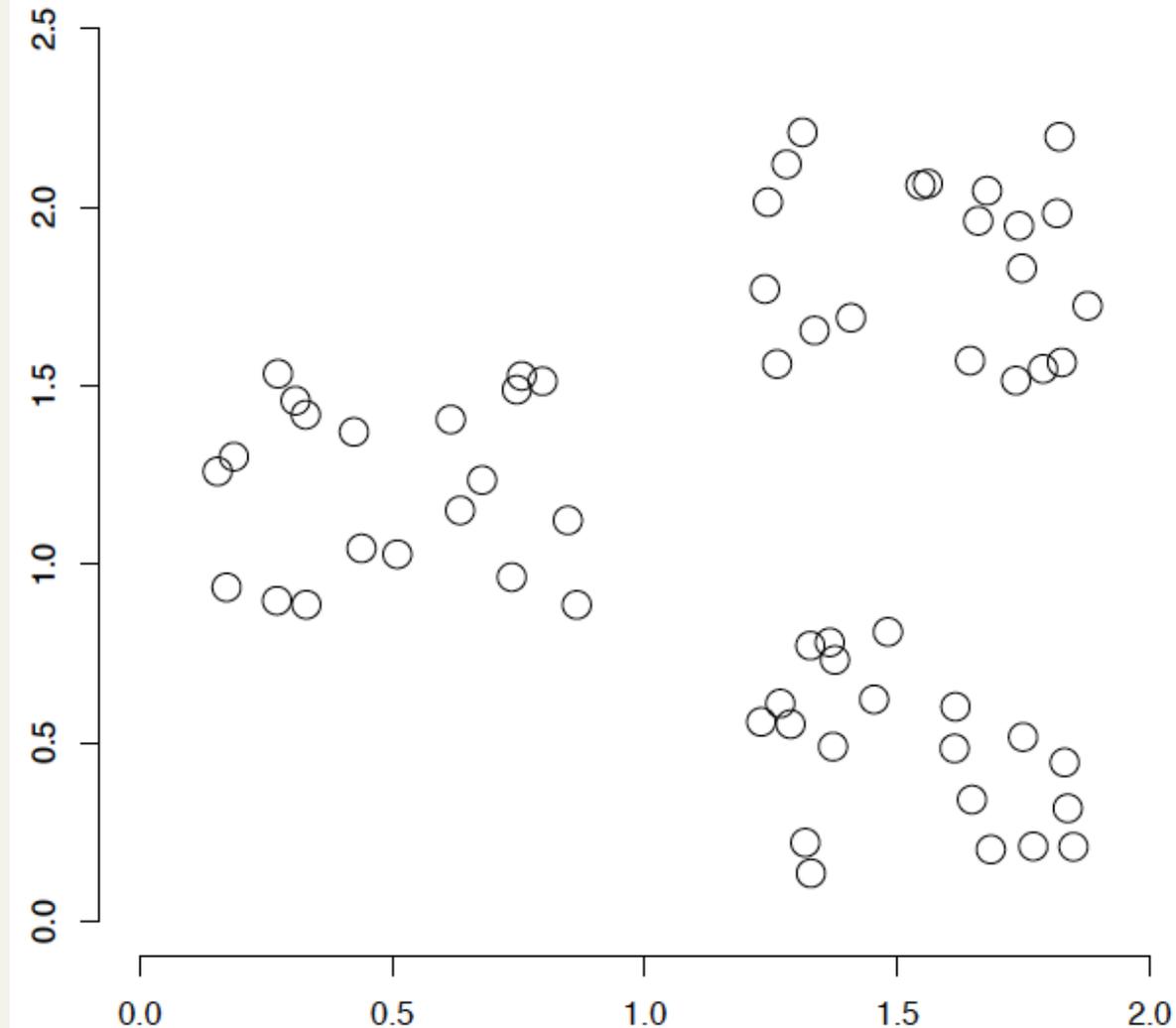
- How would you design an algorithm for finding the three clusters in this case?

Applications of clustering in IR

- Whole corpus analysis/navigation
  - Better user interface: search without typing
- For improving recall in search applications
  - Better search results
- For better navigation of search results
  - Effective "user recall" will be higher
- For speeding up vector space retrieval
  - Cluster-based retrieval gives faster search

Yahoo! Hierarchy *isn't* clustering but *is* the kind of output you want from clustering

`www.yahoo.com/Science`

… (30)

agriculture   biology   physics   CS   space

...

dairy

crops

forestry   agronomy

botany   cell

evolution

magnetism

relativity

AI   courses

HCI

craft

missions
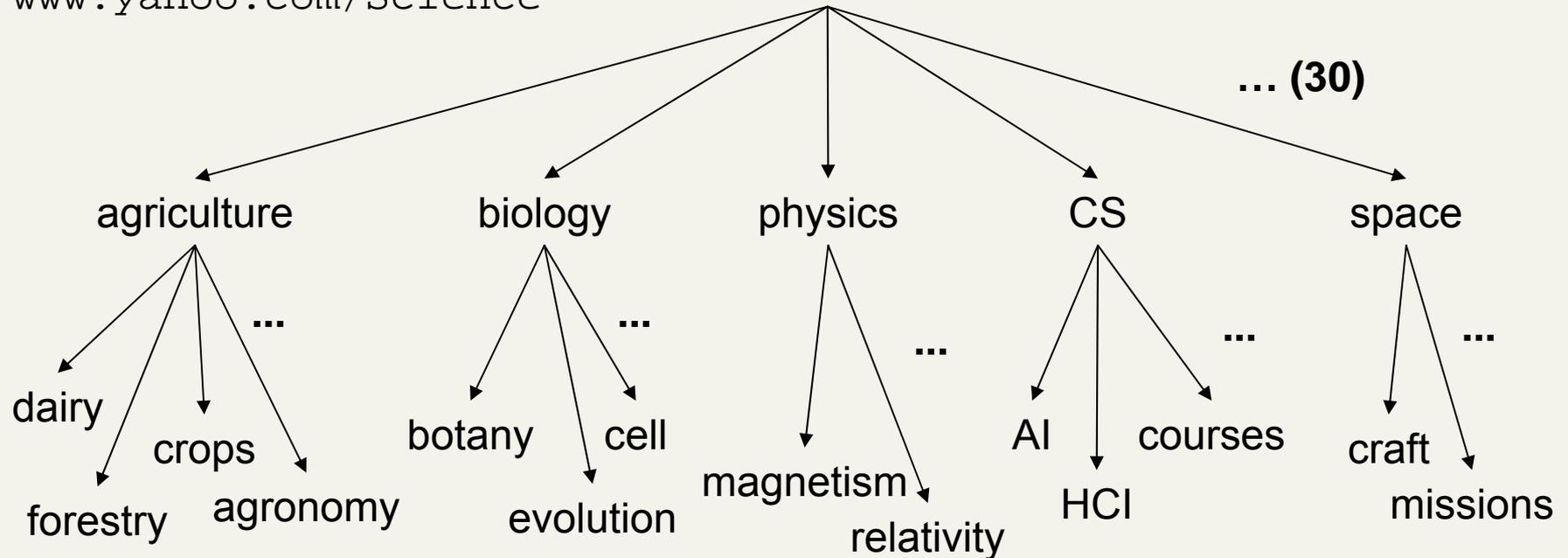
Google News: automatic clustering gives an effective news presentation metaphor

Scatter/Gather: Cutting, Karger, and Pedersen

New York Times News Service, August 1990

Scatter

| Education | Domestic | **Iraq** | Arts | Sports | **Oil** | **Germany** | Legal |

Gather

International Stories

Scatter

| Deployment | Politics | Germany | **Pakistan** | **Africa** | Markets | Oil | Hostages |

Gather

Smaller International Stories

Scatter

| Trinidad | W. Africa | S. Africa | Security | International | Lebanon | Pakistan | Japan |

For visualizing a document collection and its themes

- Wise et al, "Visualizing the non-visual" PNNL
- ThemeScapes, Cartia
  - [Mountain height = cluster size]

For better navigation of search results

- **For grouping search results thematically**
  - clusty.com / Vivisimo

Issues for clustering

- Representation for clustering
  - Document representation
    - Vector space? Normalization?
  - Need a notion of similarity/distance
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid "trivial" clusters - too large or small
      - In an application, if a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.

What makes docs "related"?

- Ideal: semantic similarity.
- Practical: statistical similarity
  - We will use cosine similarity.
  - Docs as vectors.
  - For many algorithms, easier to think in terms of a **distance** (rather than **similarity**) between docs.
  - We will use Euclidean distance.

Clustering Algorithms

- Flat algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - *K* means clustering
    - (Model based clustering)
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - (Top-down, divisive)

Hard vs. soft clustering

- Hard clustering: Each document belongs to exactly one cluster
  - More common and easier to do
- Soft clustering: A document can belong to more than one cluster.
  - Makes more sense for applications like creating browsable hierarchies
  - You may want to put a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes
  - You can only do that with a soft clustering approach.
- We won't do soft clustering today. See IIR 16.5, 18

Partitioning Algorithms

- Partitioning method: Construct a partition of n documents into a set of K clusters

- Given: a set of documents and the number K

- Find: a partition of K clusters that optimizes the chosen partitioning criterion

  - Globally optimal: exhaustively enumerate all partitions
  - Effective heuristic methods: K-means and K-medoids algorithms

K-Means

- Assumes documents are real-valued vectors.

- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.

  - (Or one can equivalently phrase it in terms of similarities)

K-Means Algorithm

Select K random docs $\{s_1, s_2, \ldots s_K\}$ as seeds.
Until clustering converges or other stopping criterion:

  For each doc d_i:

  Assign d_i to the cluster c_j such that $dist(d_i, s_j)$ is minimal.

  Update the seeds to the centroid of each cluster:

  For each cluster c_j

  $s_j = \mu(c_j)$

K Means Example (*K*=2)
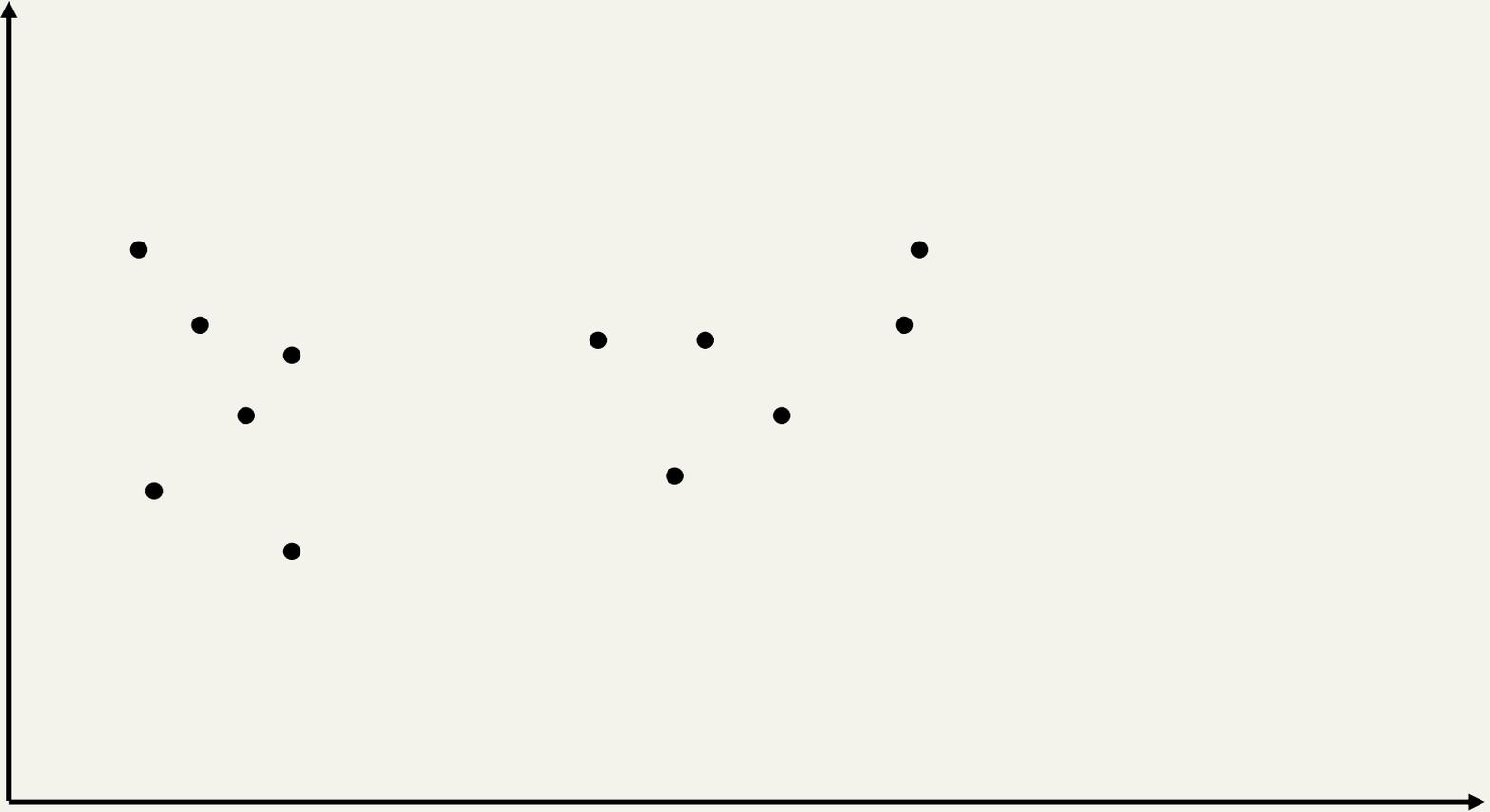
K Means Example (*K*=2)
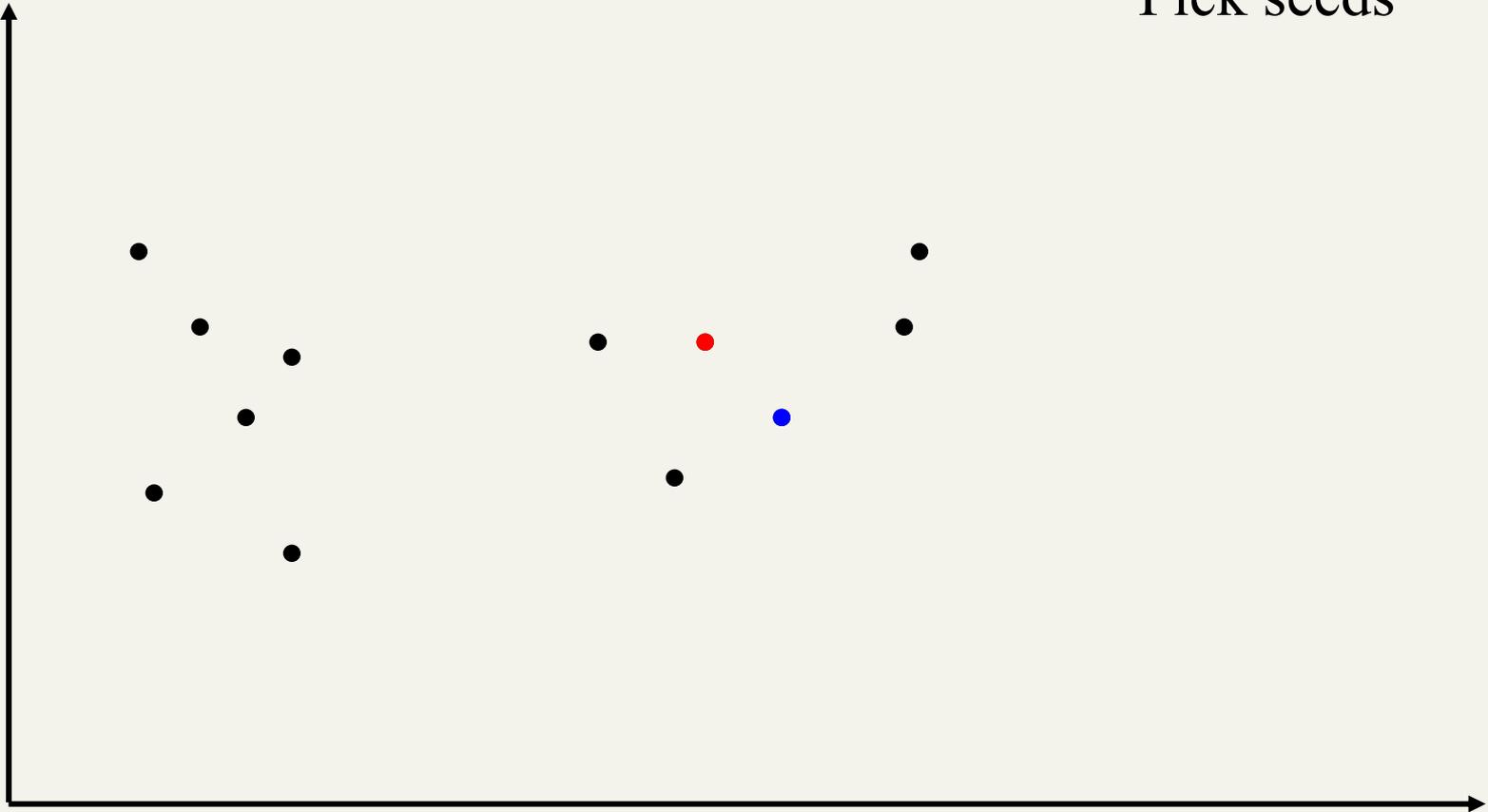
Pick seeds

K Means Example (*K*=2)
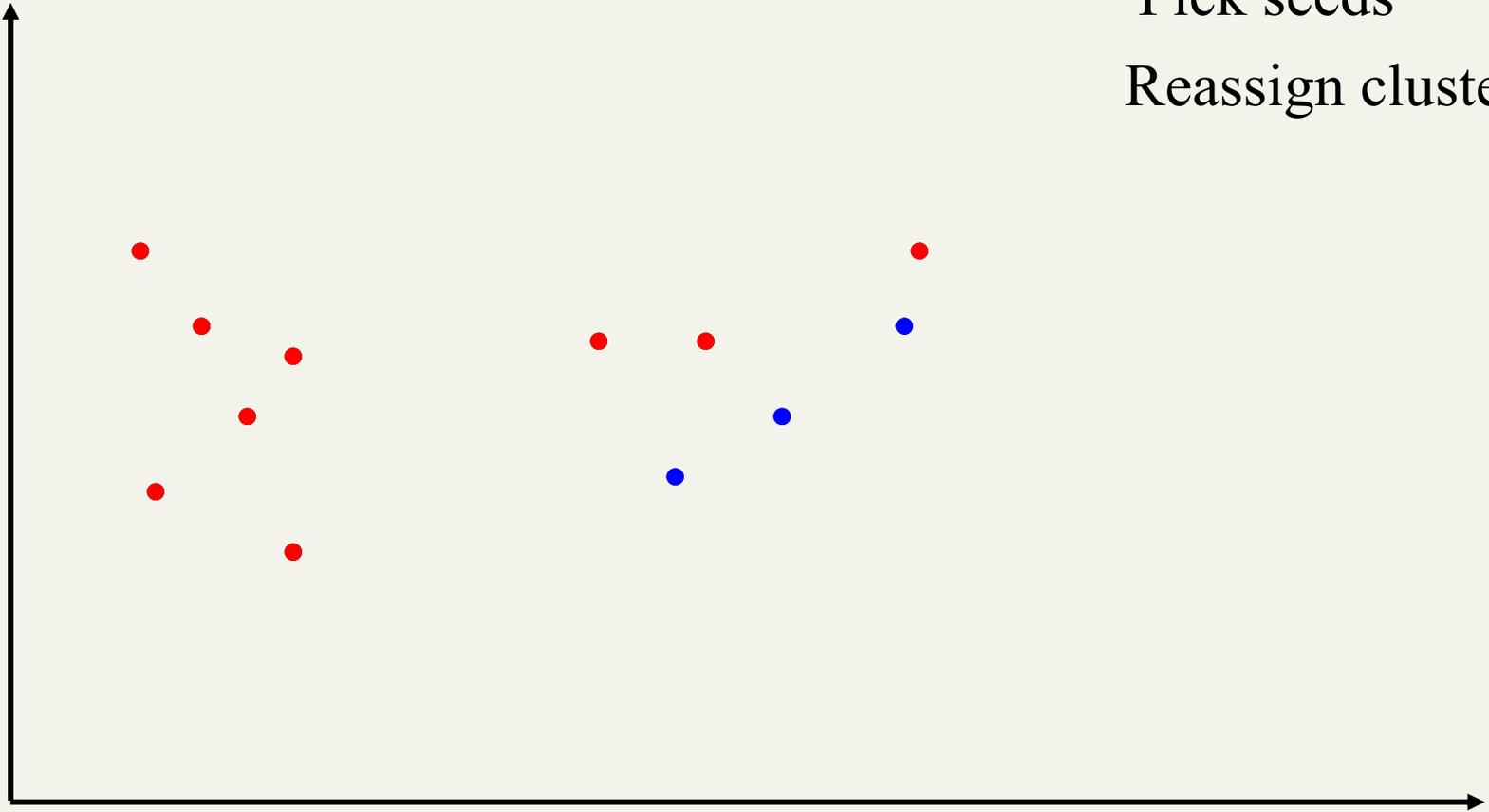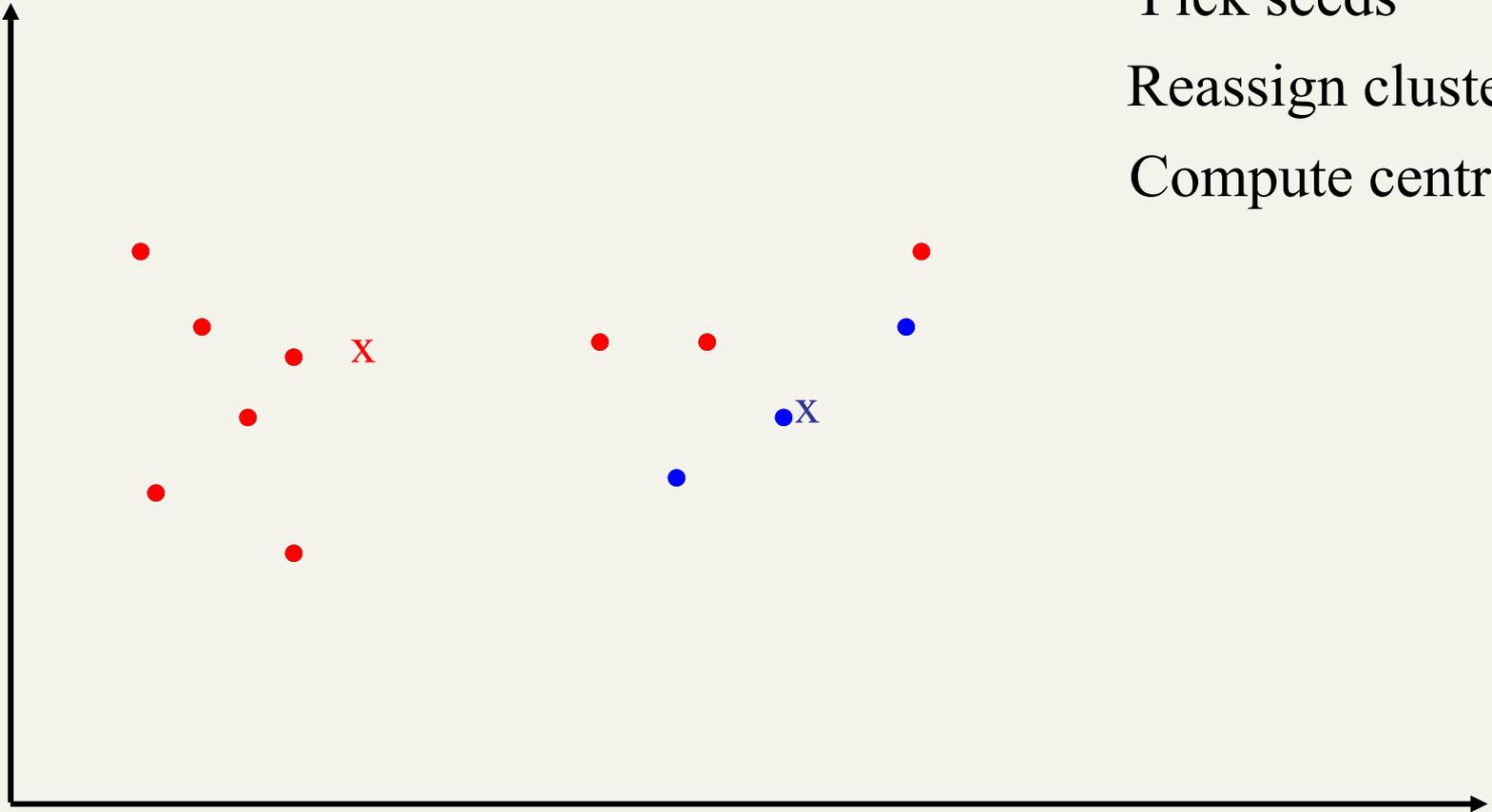
Pick seeds

Reassign clusters

K Means Example (*K*=2)

Pick seeds

Reassign clusters

Compute centroids

K Means Example (K=2)

Pick seeds

Reassign clusters

Compute centroids

Reassign clusters
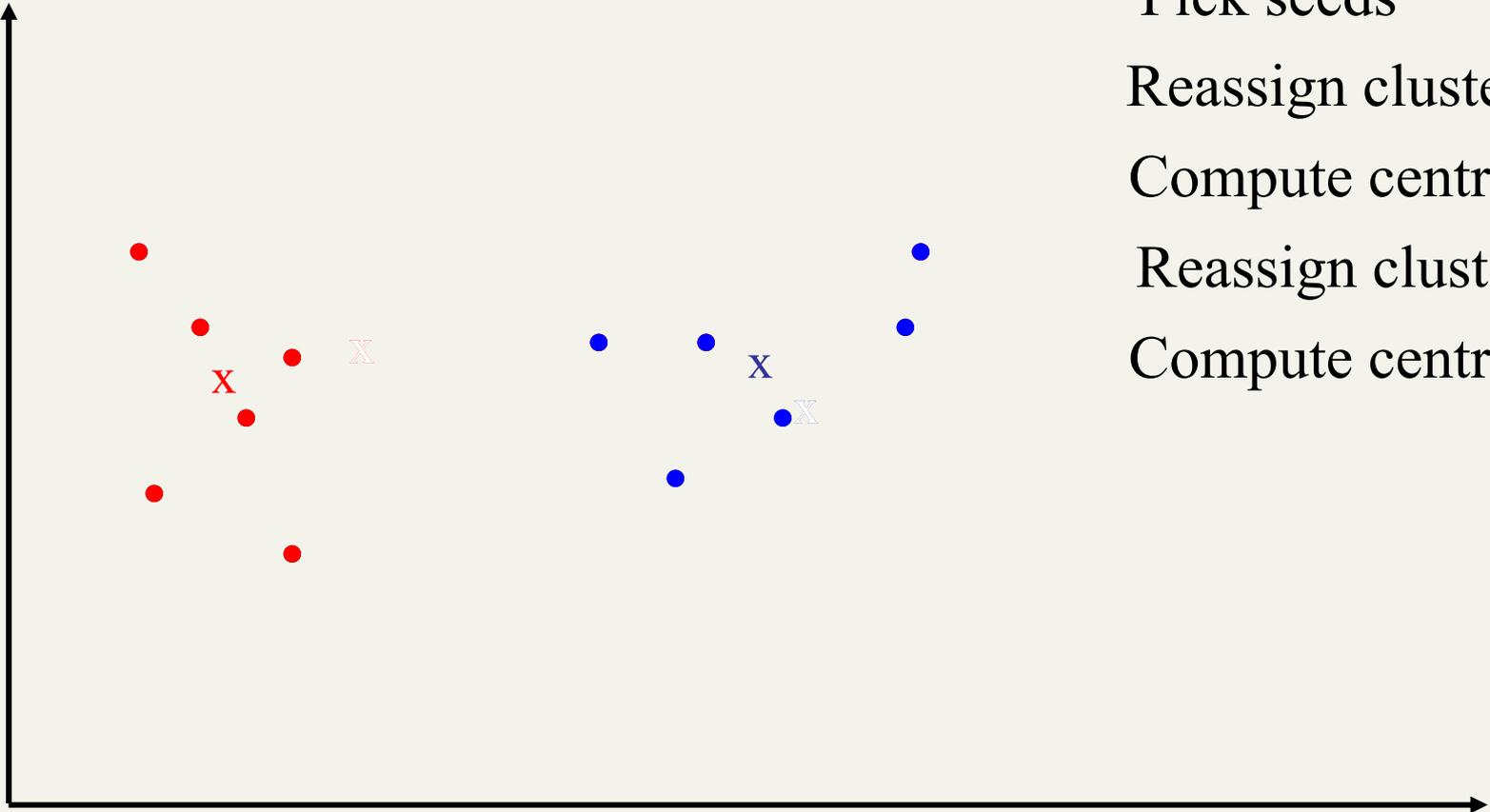
K Means Example (*K*=2)

Pick seeds

Reassign clusters

Compute centroids

Reassign clusters

Compute centroids
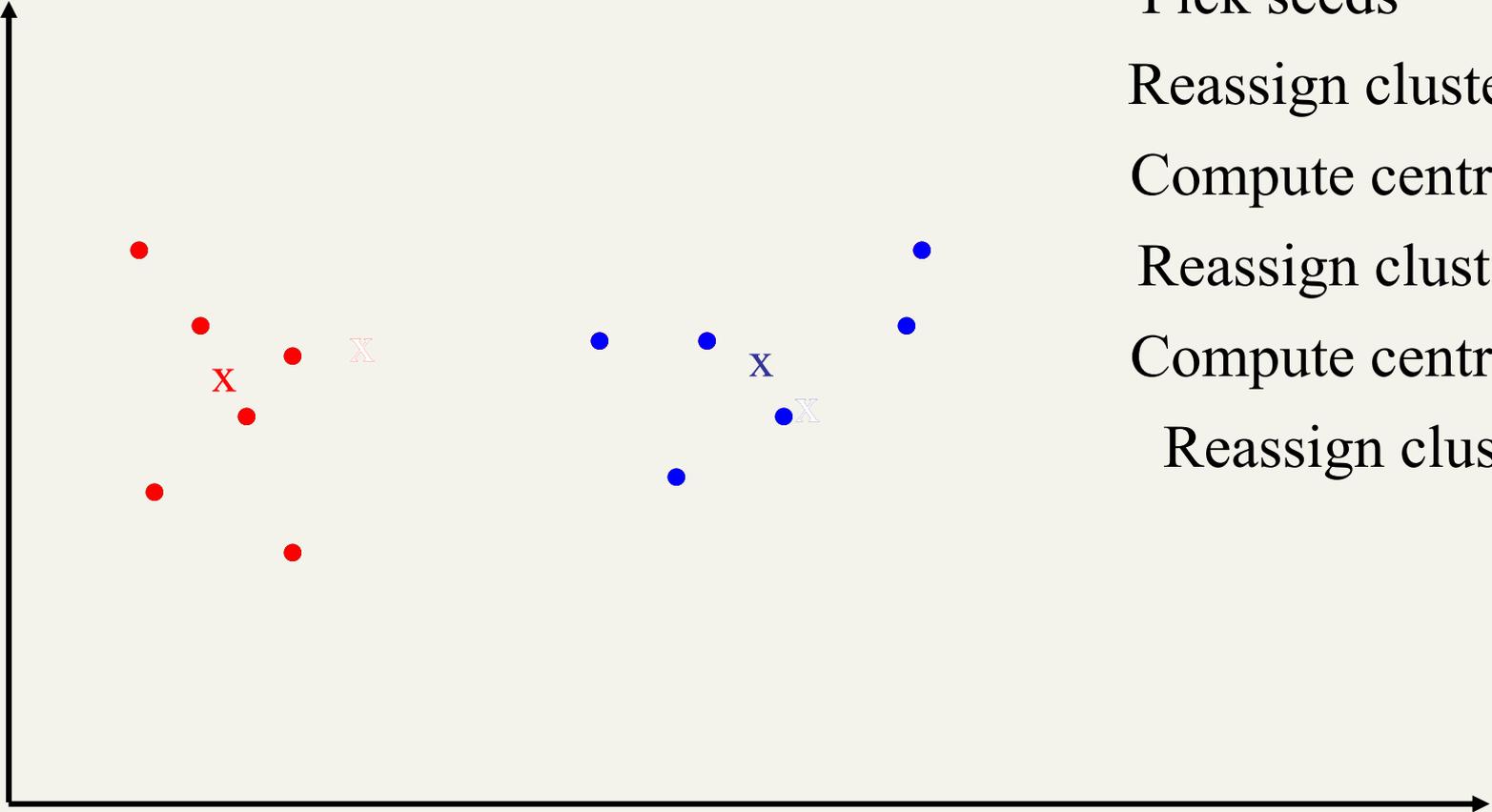
K Means Example (*K*=2)

Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters

K Means Example (*K*=2)

Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters
Converged!

Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Doc partition unchanged.
  - Centroid positions don't change.

Issues for clustering

- Why should the *K*-means algorithm ever reach a *fixed point*?

  - A state in which clusters don't change.

- *K*-means is a special case of a general procedure known as the *Expectation Maximization (EM) algorithm*.

  - EM is known to converge.

  - Number of iterations could be large.

    - But in practice usually isn't

Time Complexity

- Computing distance between two docs is O(m) where m is the dimensionality of the vectors.

- Reassigning clusters: O(Kn) distance computations, or O(Knm).

- Computing centroids: Each doc gets added once to some centroid: O(nm).

- Assume these two steps are each done once for I iterations: O$(IKnm)$.

Seed Choice

- Results can vary based on random seed selection.

- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
  - Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
  - Try out multiple starting points
  - Initialize with the results of another method.

Example showing sensitivity to seeds
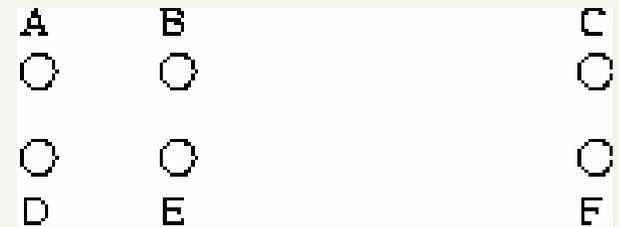
In the above, if you start with B and E as centroids you converge to {A,B,C} and {D,E,F}
If you start with D and F you converge to {A,B,D,E} {C,F}

How Many Clusters?

- Number of clusters K is given
  - Partition n docs into predetermined number of clusters
- Finding the "right" number of clusters is part of the problem
  - Given docs, partition into an "appropriate" number of subsets.
  - E.g., for query results - ideal value of K not known up front – though UI may impose limits.
- Can usually take an algorithm for one flavor and convert to the other.

K not specified in advance

- Say, the results of a query.

- Solve an optimization problem: penalize having lots of clusters
  - application dependent, e.g., compressed summary of search results list.

- Tradeoff between having more clusters (better focus within each cluster) and having too many clusters

Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of documents.
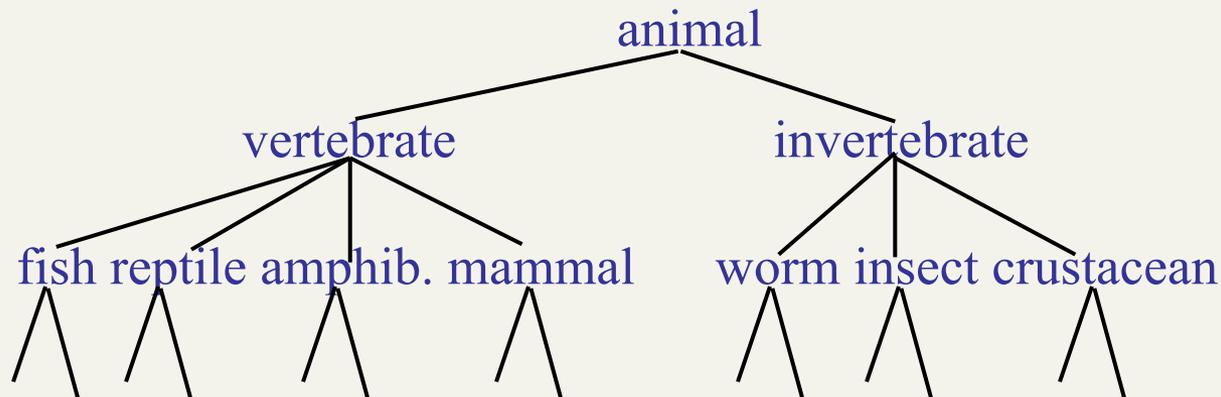
- One approach: recursive application of a partitional clustering algorithm.

Dendogram: Hierarchical Clustering

- Clustering obtained by cutting the dendrogram at a desired level: each **connected** component forms a cluster.
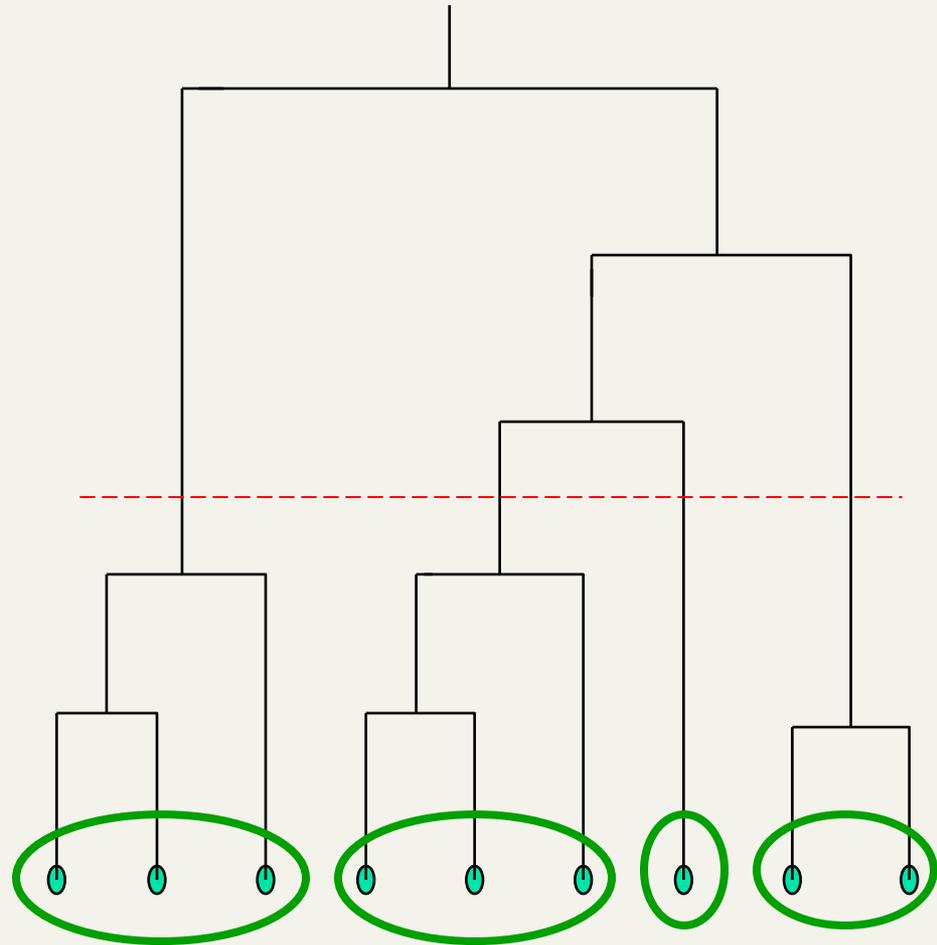
Hierarchical Agglomerative Clustering (HAC)

- Starts with each doc in a separate cluster
  - then repeatedly joins the **closest pair** of clusters, until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

Closest pair of clusters

- Many variants to defining closest pair of clusters
- **Single-link**
  - Similarity of the *most* cosine-similar (single-link)
- **Complete-link**
  - Similarity of the "furthest" points, the *least* cosine-similar
- **Centroid**
  - Clusters whose centroids (centers of gravity) are the most cosine-similar
- **Average-link**
  - Average cosine between pairs of elements

Single Link Agglomerative Clustering

- Use maximum similarity of pairs:

$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

- Can result in "straggly" (long and thin) clusters due to chaining effect.

- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \max(sim(c_i, c_k), sim(c_j, c_k))$$

Single Link Example
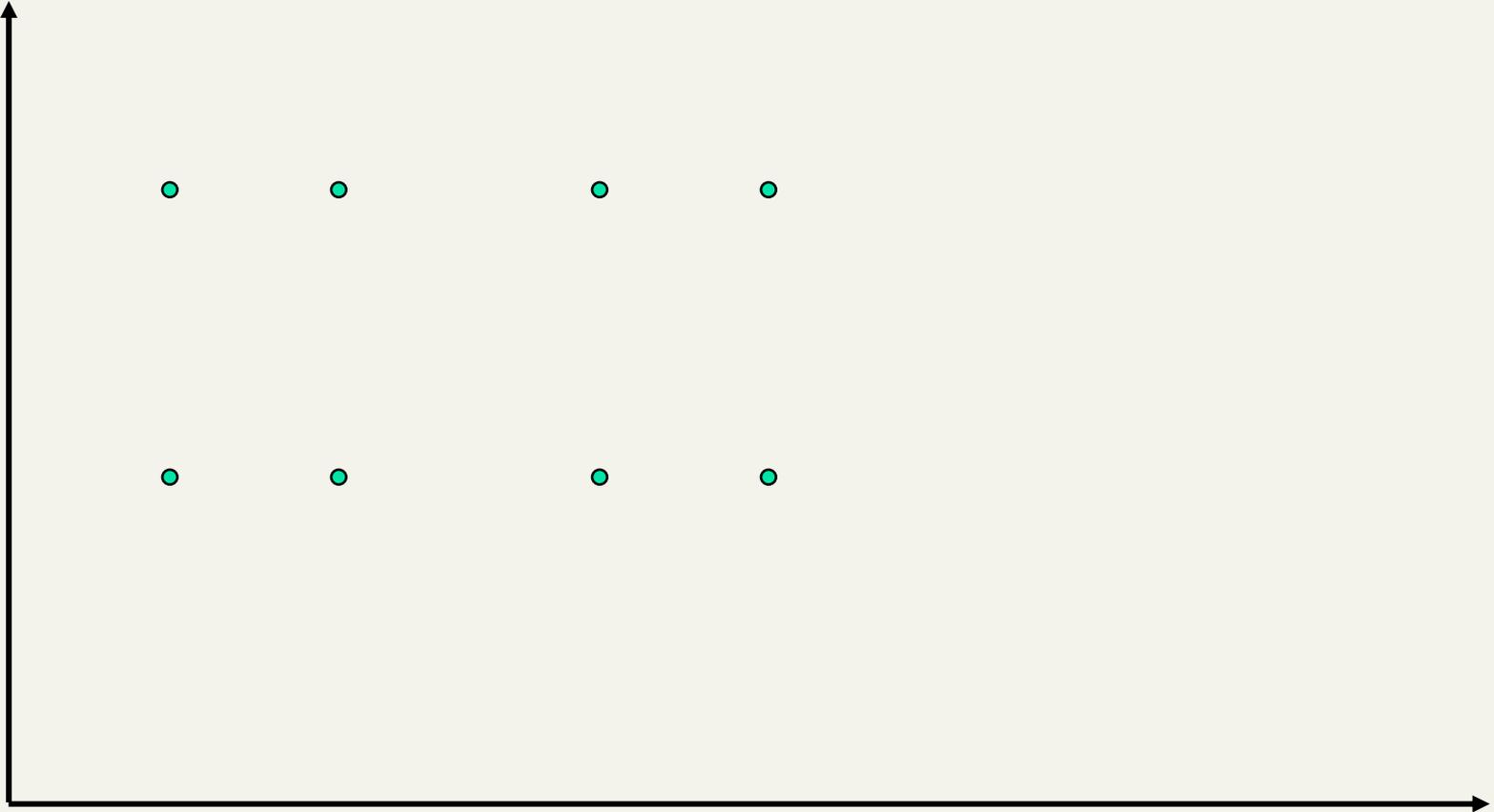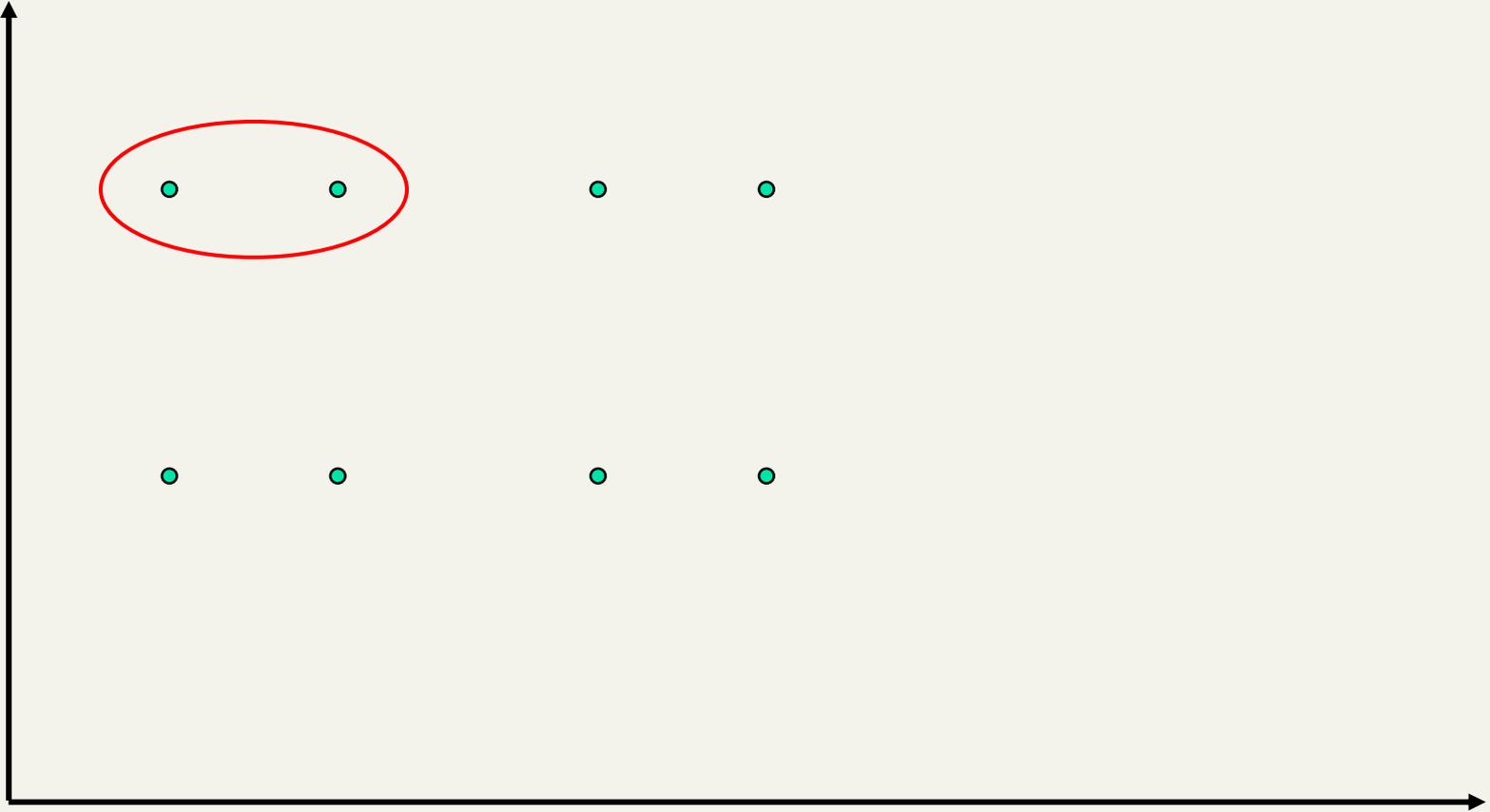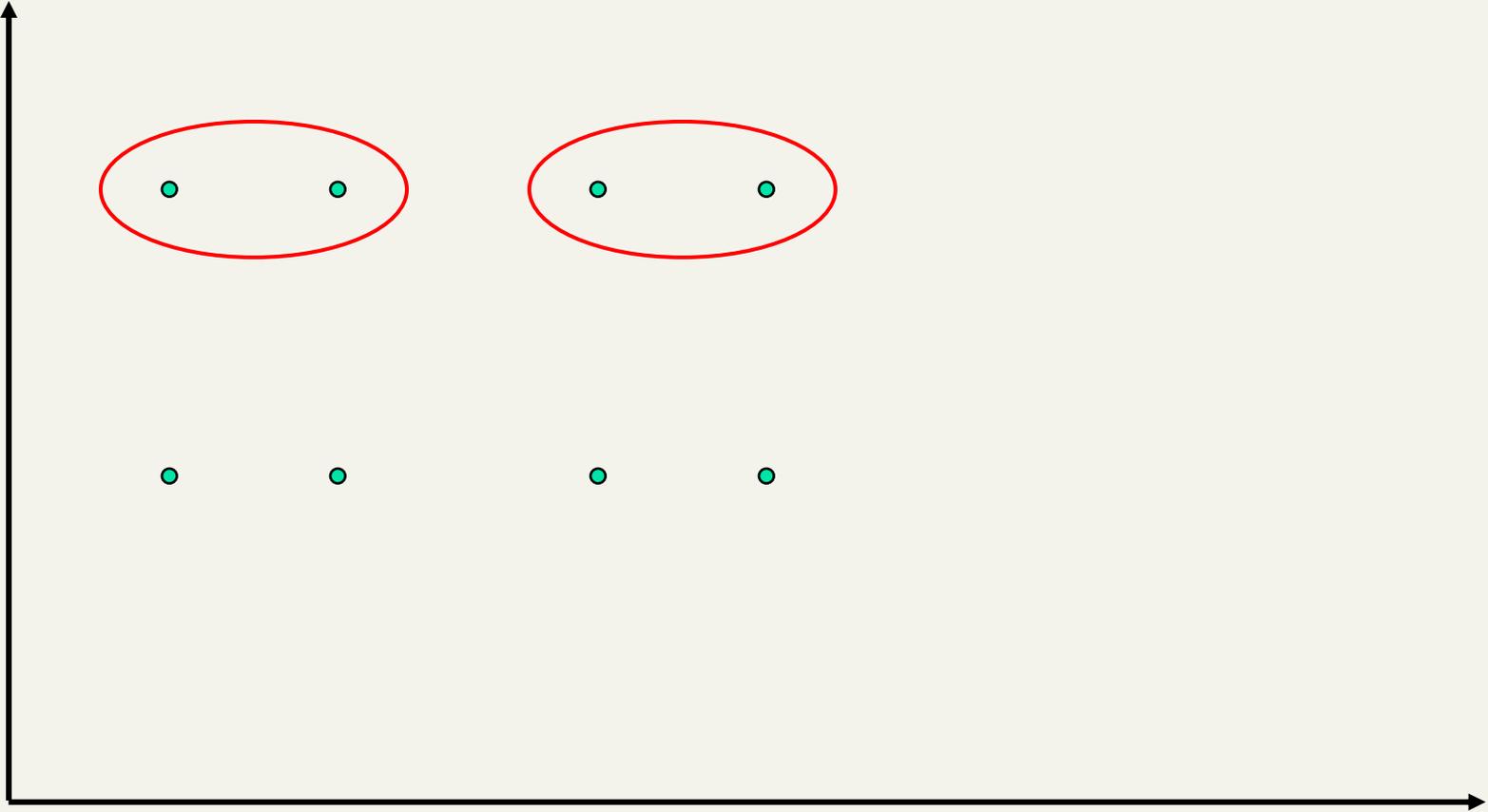
Single Link Example

# Single Link Example

Single Link Example
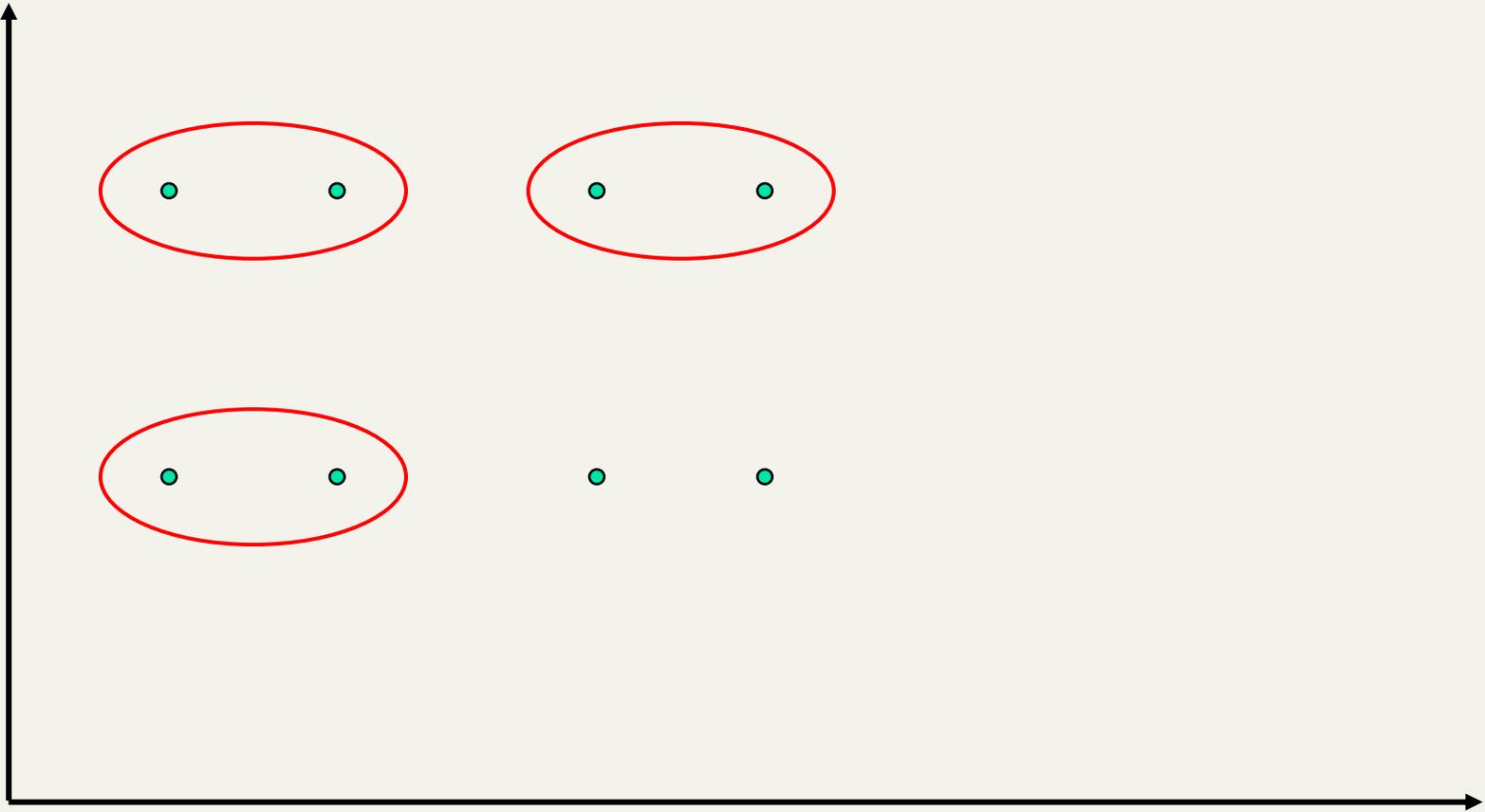
Single Link Example
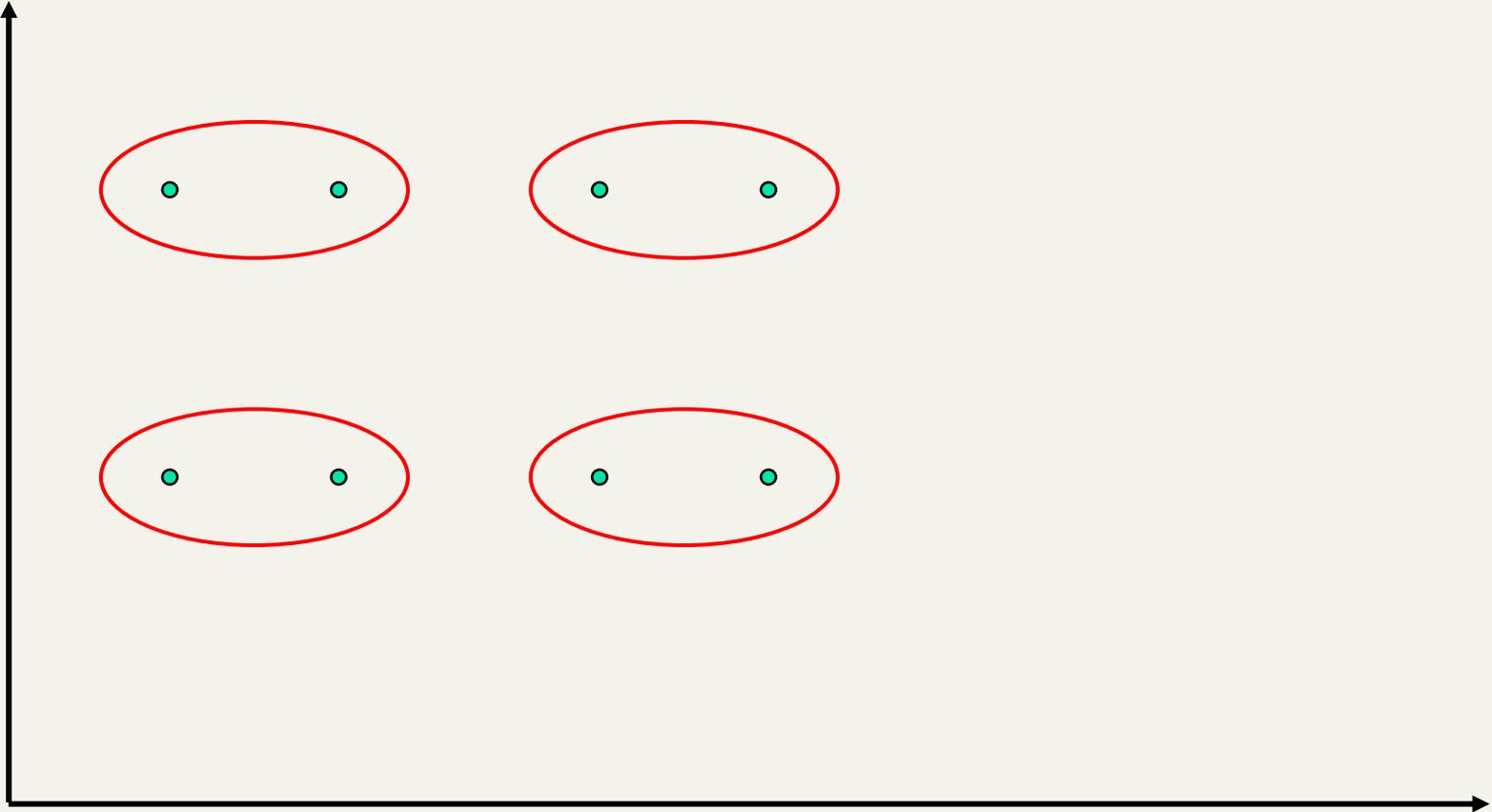
Single Link Example
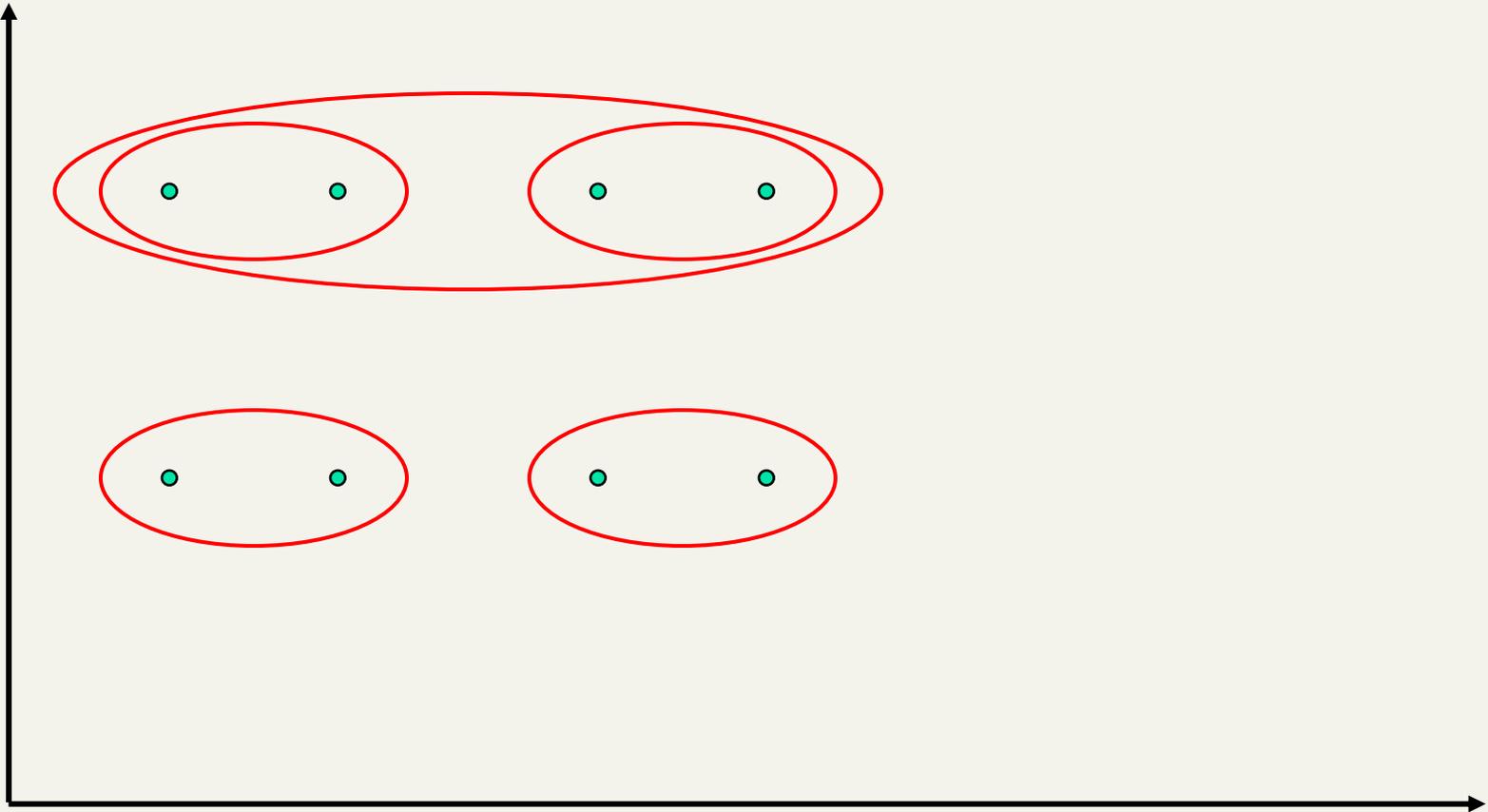
Single Link Example
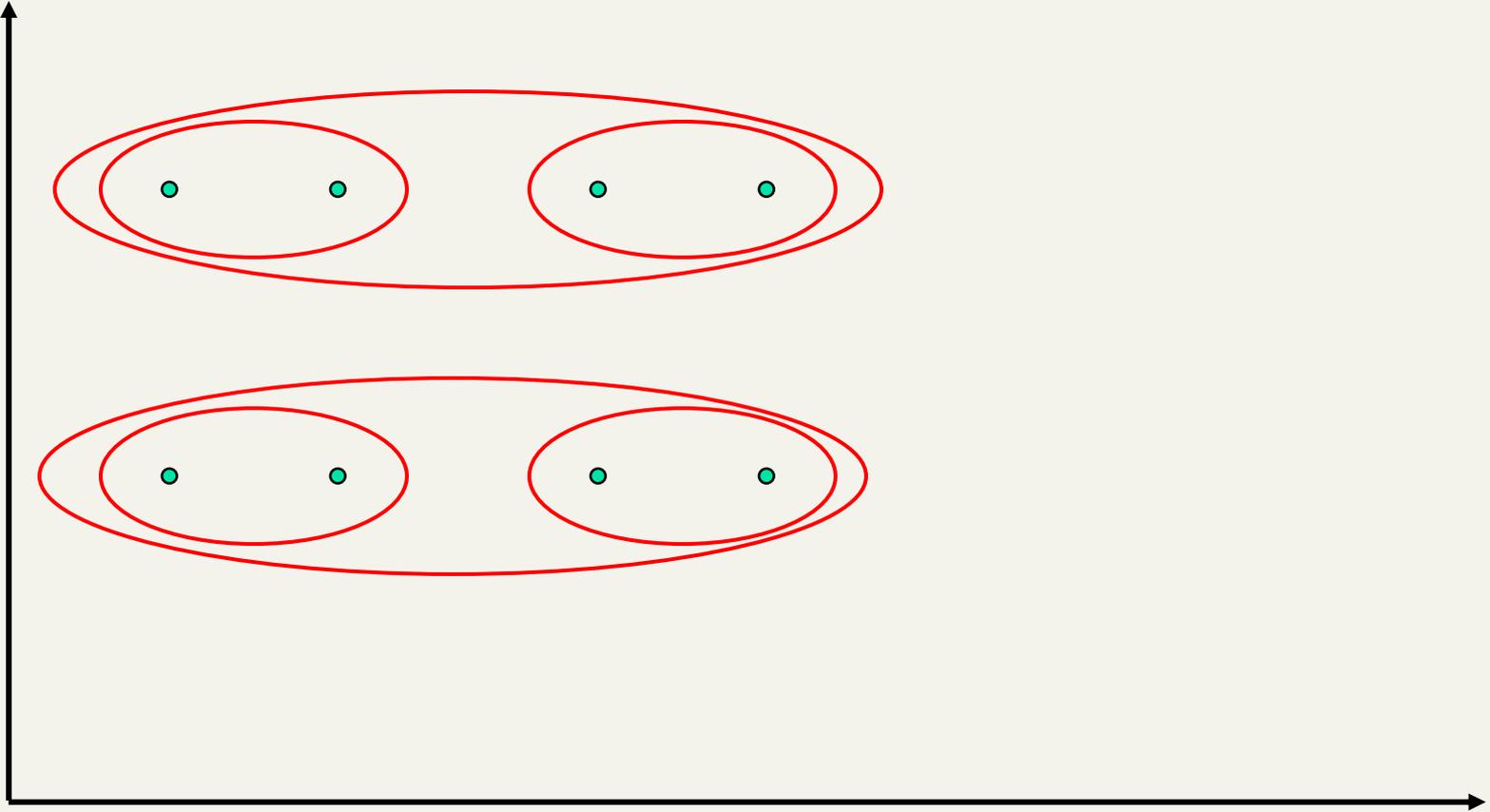
Single Link Example
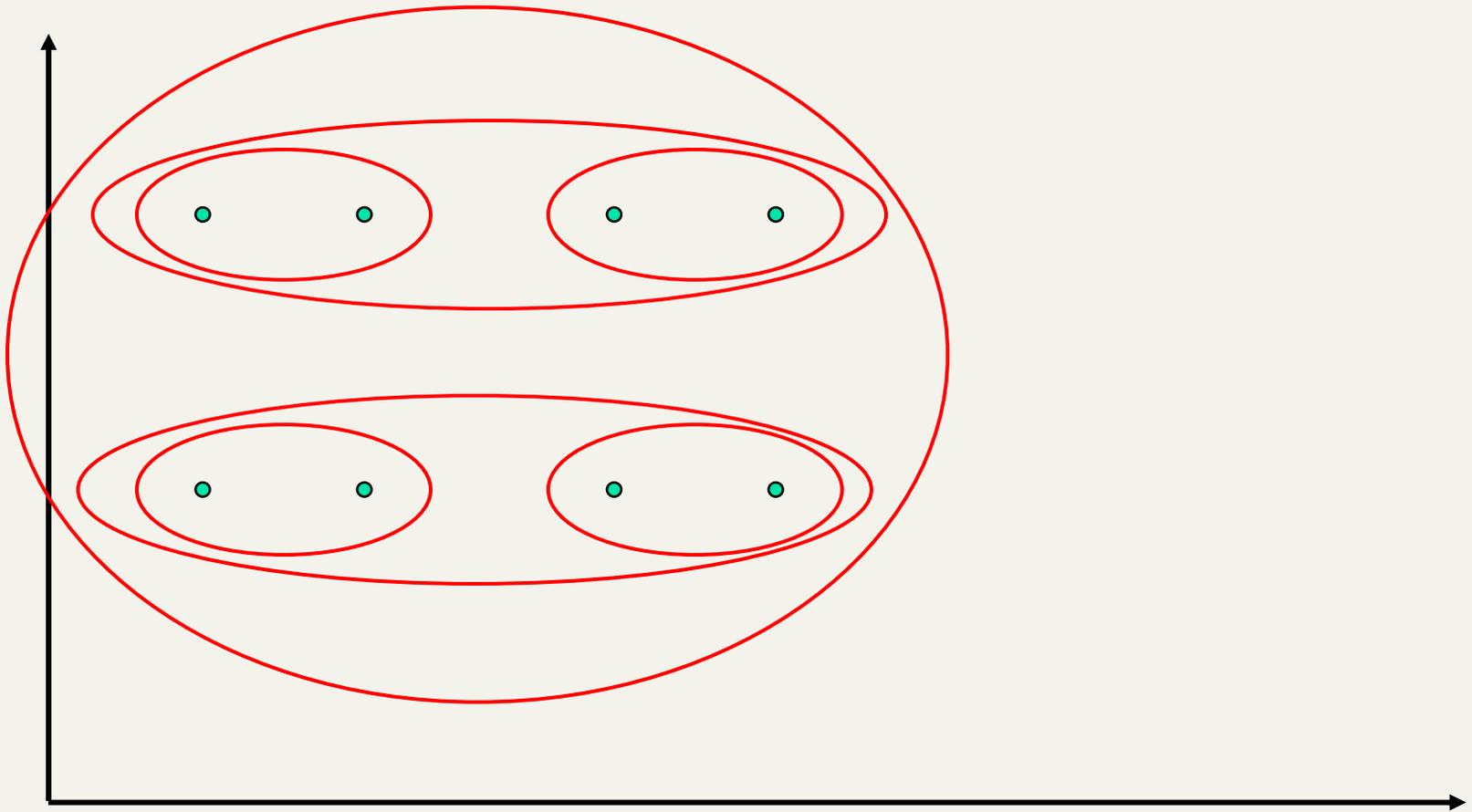
Complete Link Agglomerative Clustering

- Use minimum similarity of pairs:

$$sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} sim(x, y)$$

- Makes "tighter," spherical clusters that are typically preferable.

- After merging c_i and c_j, the similarity of the resulting cluster to another cluster, c_k, is:

$$sim((c_i \cup c_j), c_k) = \min(sim(c_i, c_k), sim(c_j, c_k))$$
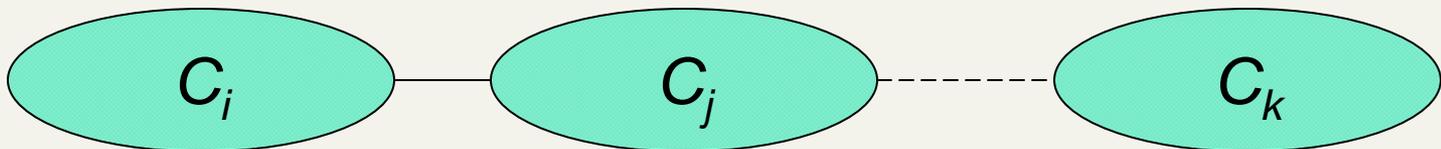
Complete Link Example
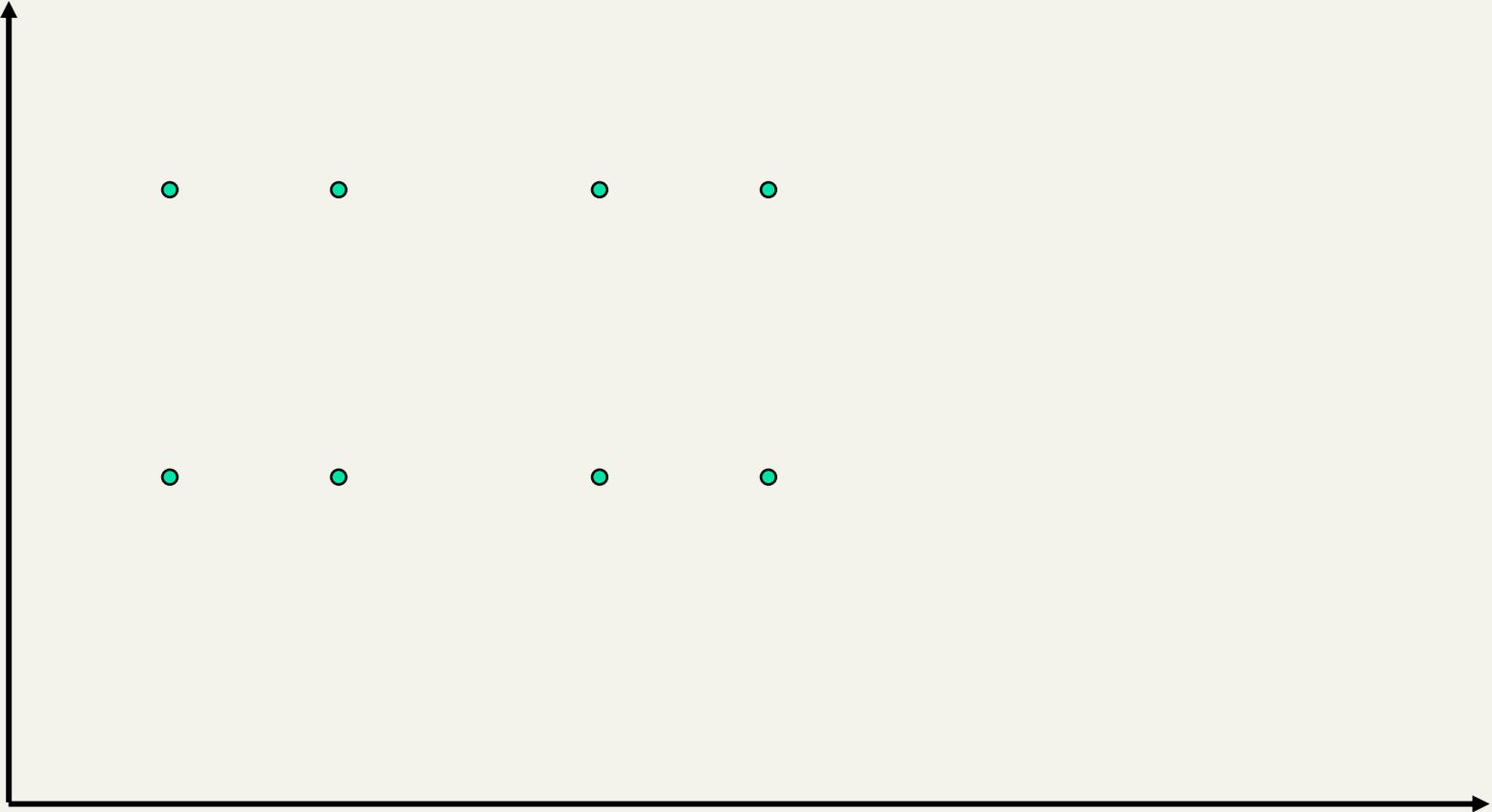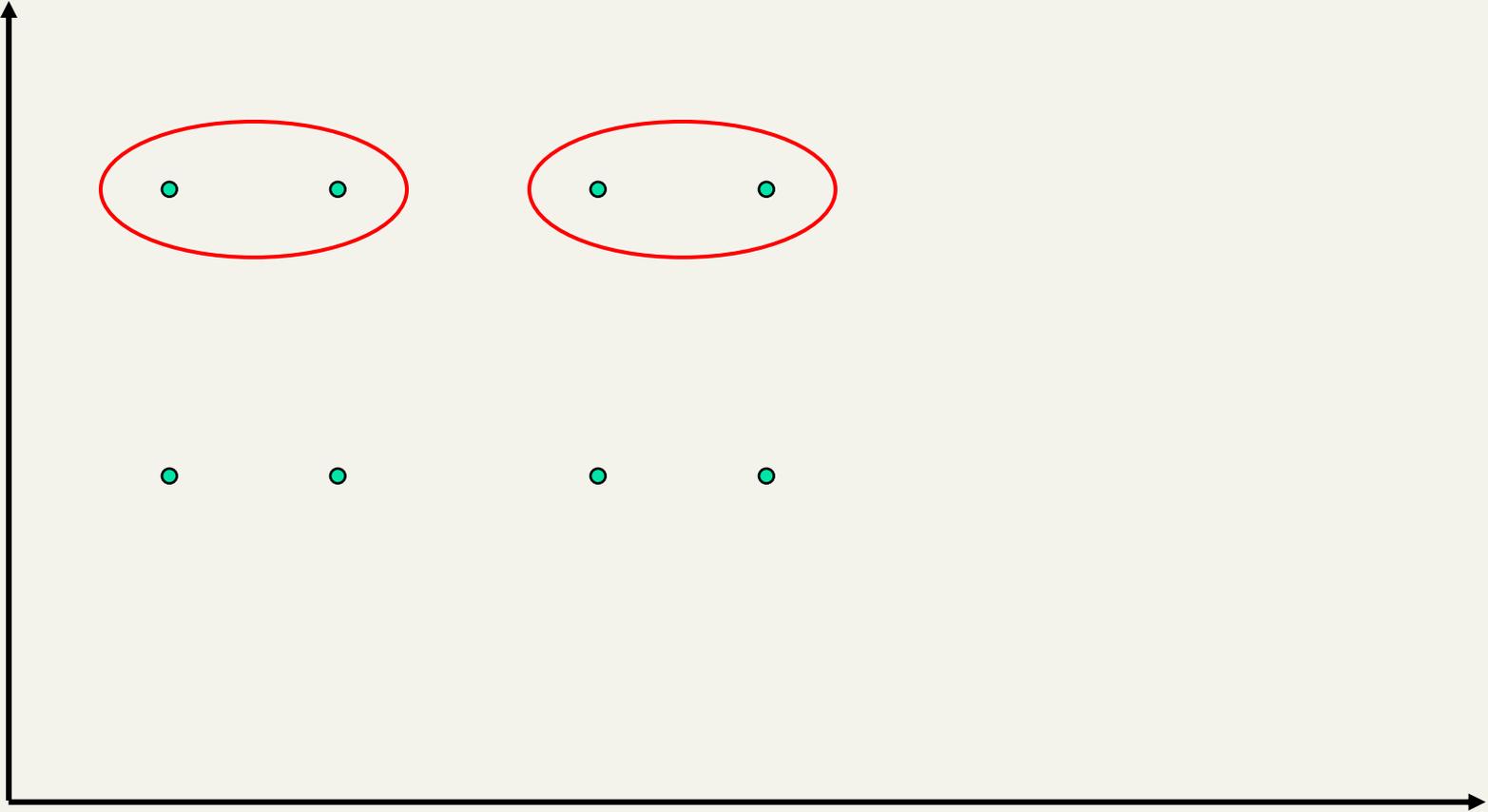
Complete Link Example

Complete Link Example

Complete Link Example
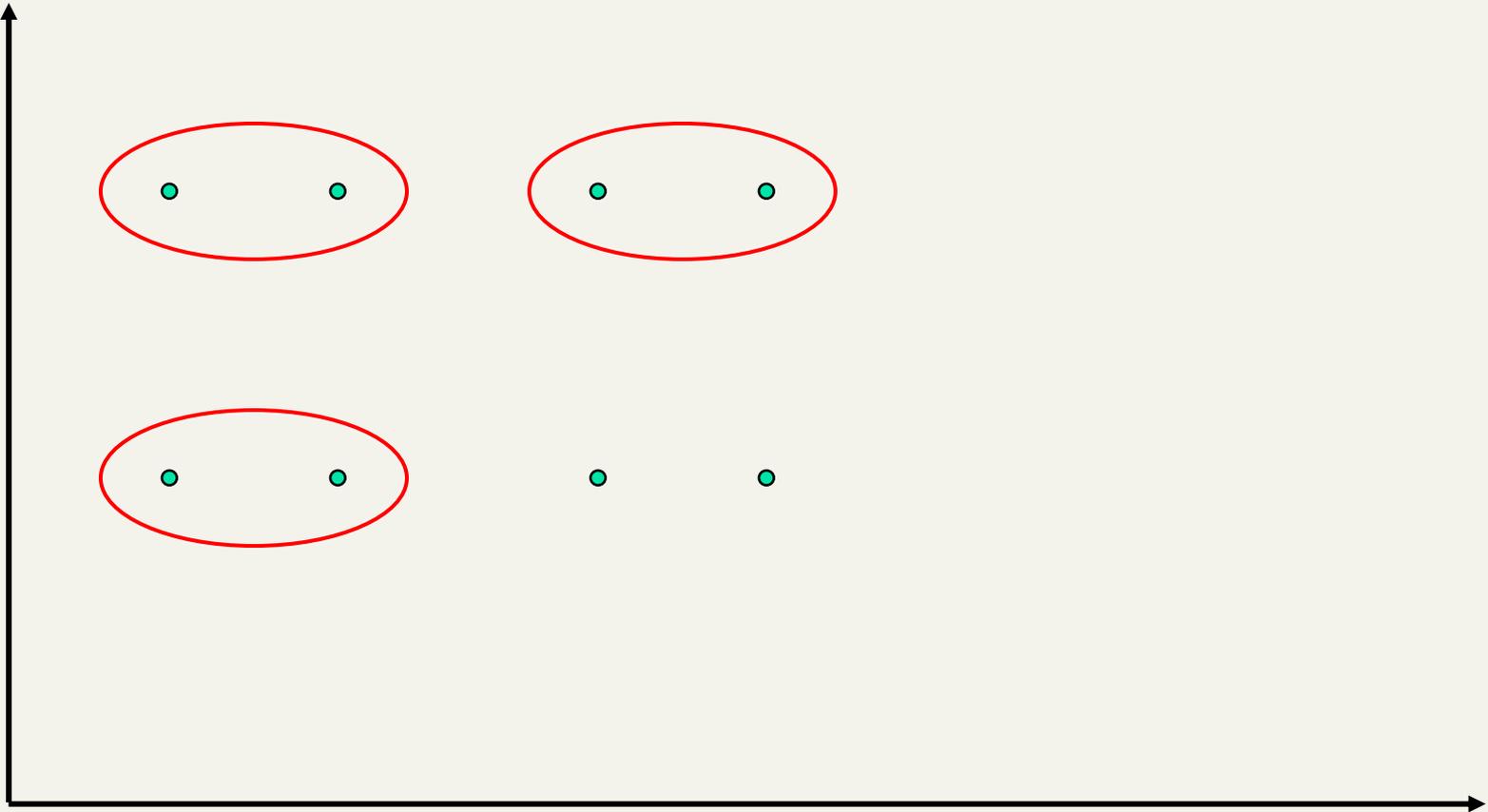
Complete Link Example
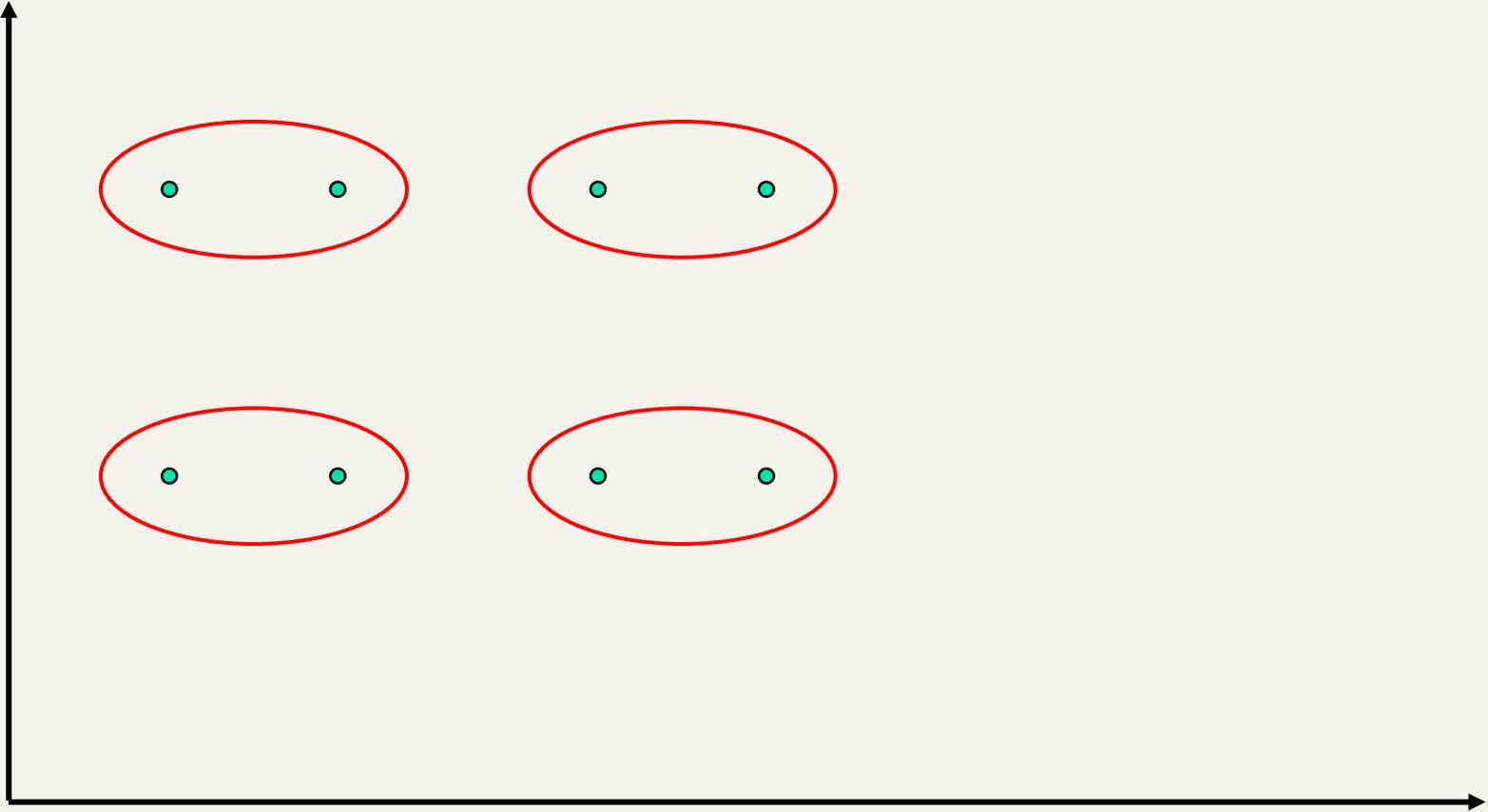
Complete Link Example
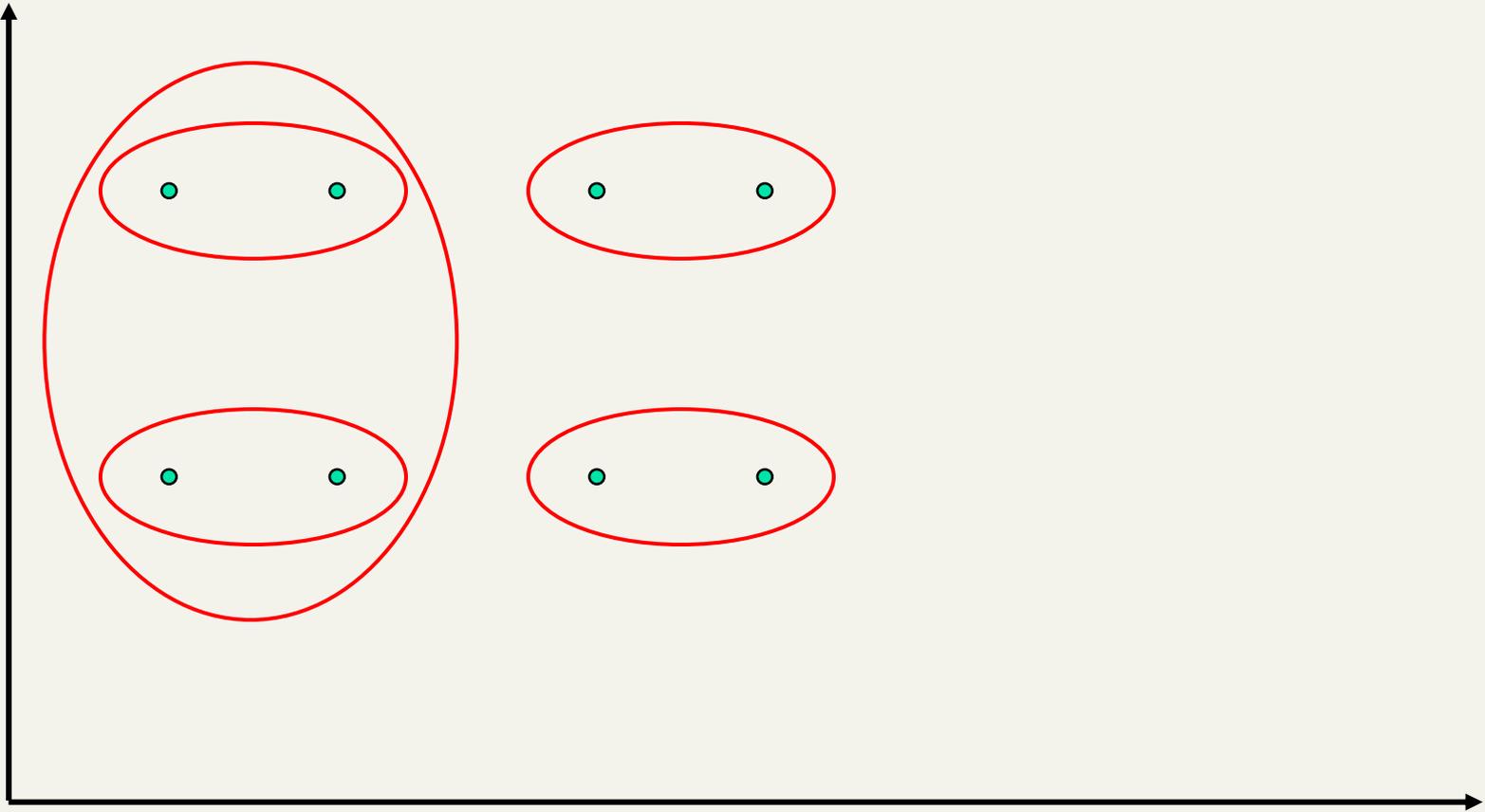
Complete Link Example
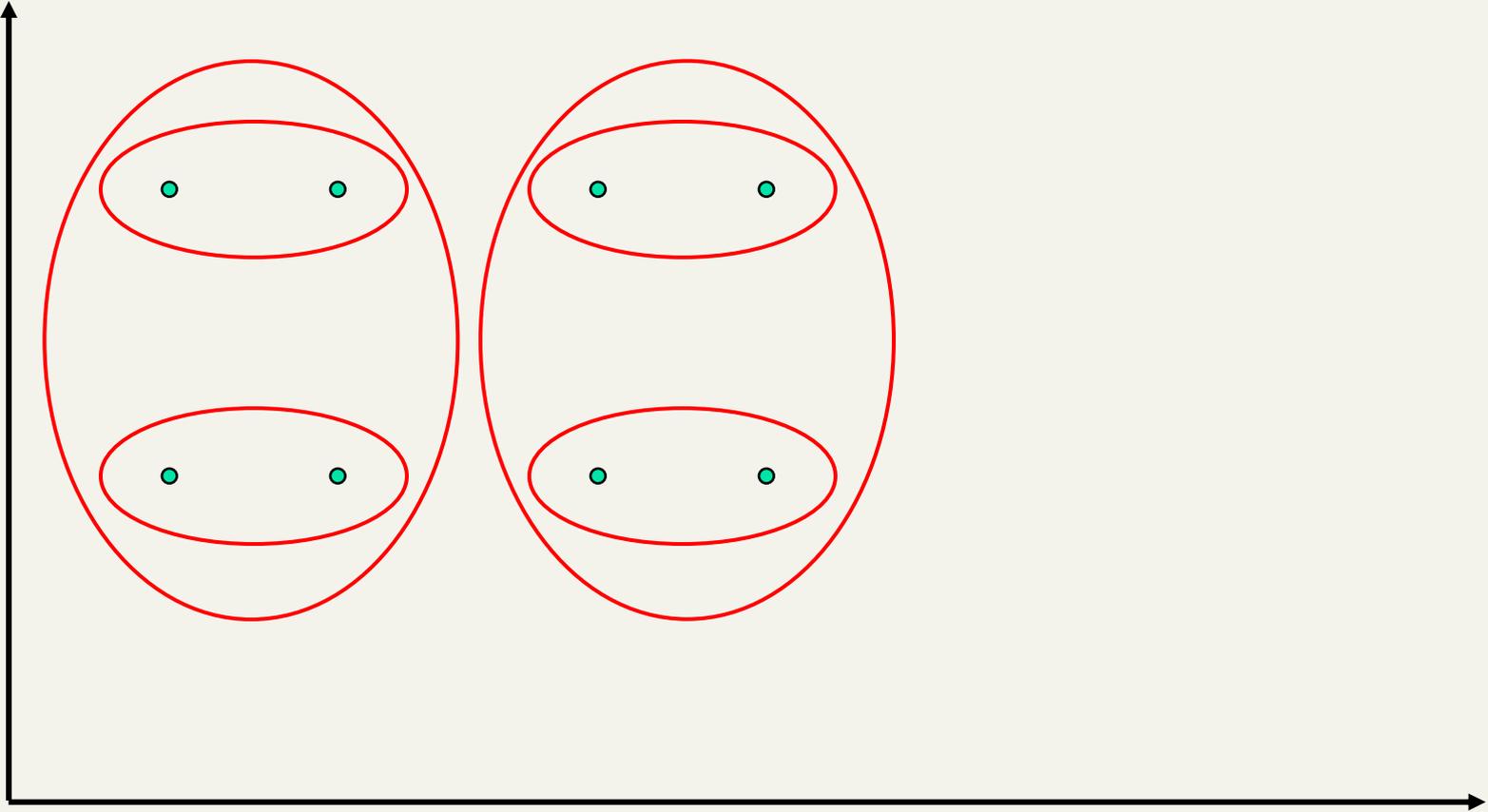
Complete Link Example
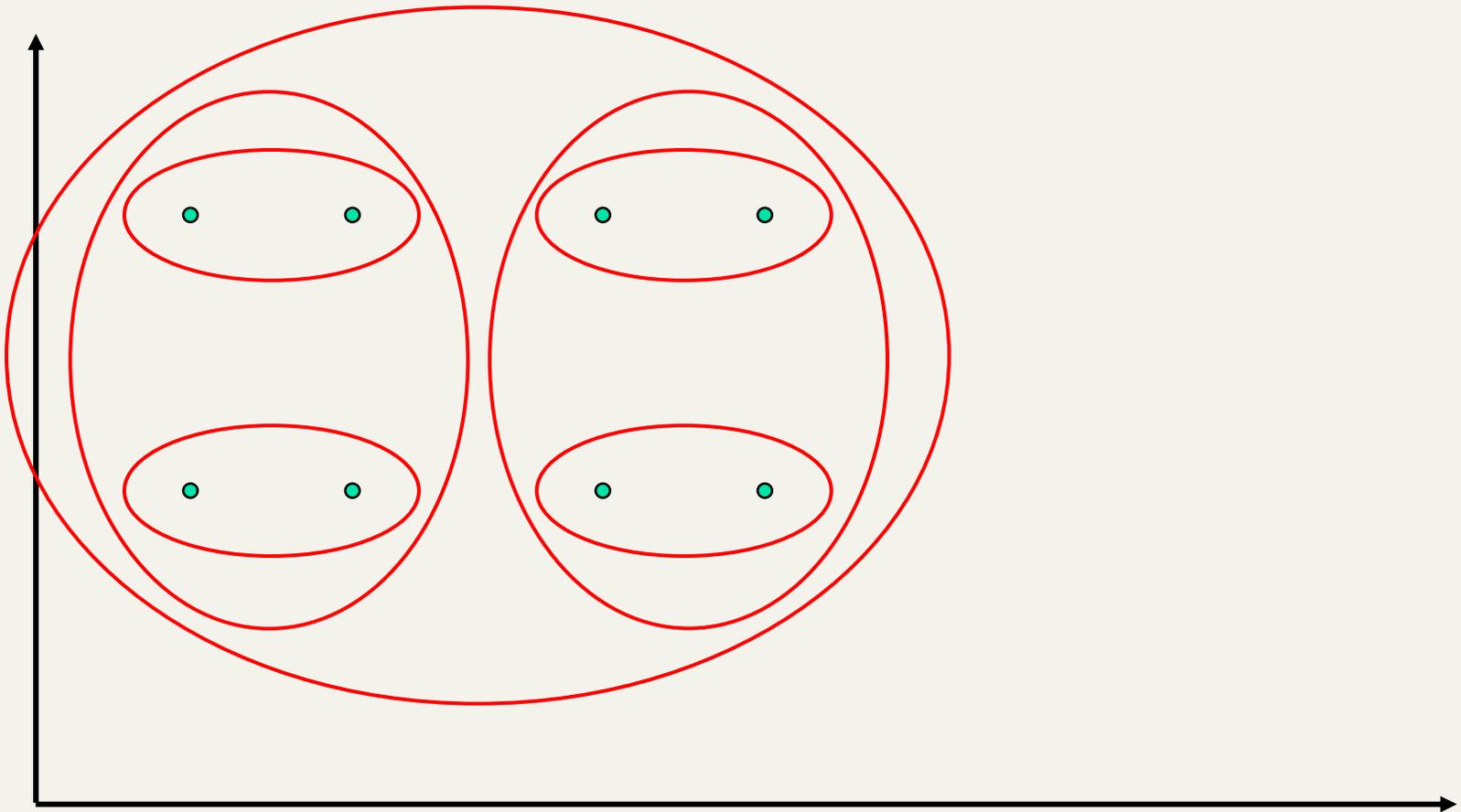
Computational Complexity

- In the first iteration, all HAC methods need to compute similarity of all pairs of n individual instances which is $O(n^2)$.

- In each of the subsequent $n–2$ merging iterations, compute the distance between the most recently created cluster and all other existing clusters.

- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.

  - Often $O(n^3)$ if done naively or $O(n^2 \log n)$ if done more cleverly

Group Average Agglomerative Clustering

- Similarity of two clusters = average similarity of all pairs within merged cluster.

$$sim(c_i, c_j) = \frac{1}{|c_i \cup c_j|(|c_i \cup c_j| - 1)} \sum_{\vec{x} \in (c_i \cup c_j)} \sum_{\vec{y} \in (c_i \cup c_j): \vec{y} \neq \vec{x}} sim(\vec{x}, \vec{y})$$

- Compromise between single and complete link.
- Two options:
  - Averaged across all ordered pairs in the merged cluster
  - Averaged over all pairs *between* the two original clusters
- No clear difference in efficacy

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:

    - the **intra-class** (that is, intra-cluster) similarity is high

    - the **inter-class** similarity is low

    - The measured quality of a clustering depends on both the document representation and the similarity measure used

External criteria for clustering quality

- Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data

- Assesses a clustering with respect to **ground truth** … requires **labeled data**

- Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, ω_1, ω_2, …, ω_K with n_i members.

External Evaluation of Cluster Quality

- Simple measure: **purity**, the ratio between the dominant class in the cluster π_i and the size of cluster ω_i

$$Purity(\omega_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

- Biased because having *n* clusters maximizes purity
- Others are entropy of classes in clusters (or mutual information between classes and clusters)

Purity example
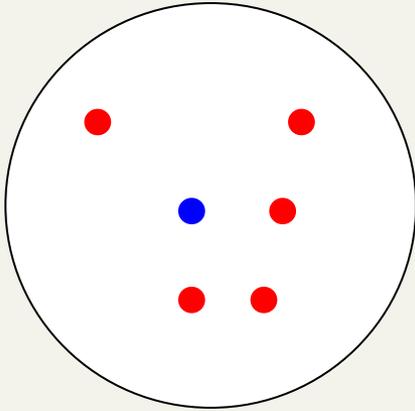
Cluster I
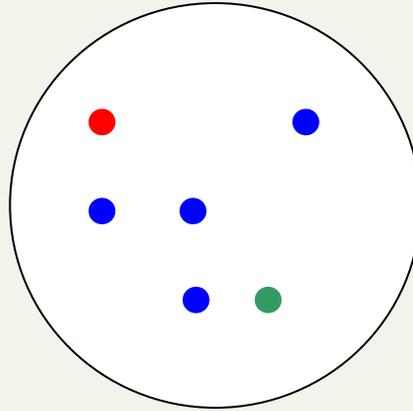
Cluster II
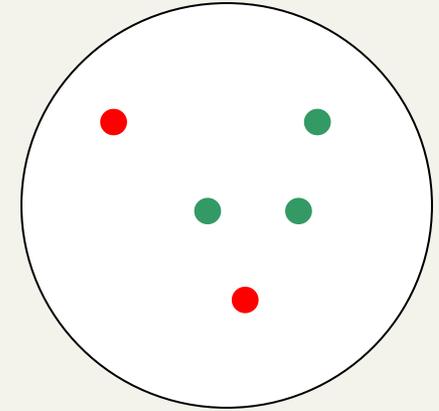
Cluster III

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

Final word and resources

- In clustering, clusters are inferred from the data without human input (unsupervised learning)

- However, in practice, it's a bit less clear: there are many ways of influencing the outcome of clustering: number of clusters, similarity measure, representation of documents, . . .

Resources

- IIR Chapters 16 – 16.4
- IIR Chapters 17 – 17.2, 17.6