UNIVERSIDAD DE CHILE

Minería de Datos

Welcome to the Machine Learning class

Valentin Barriere

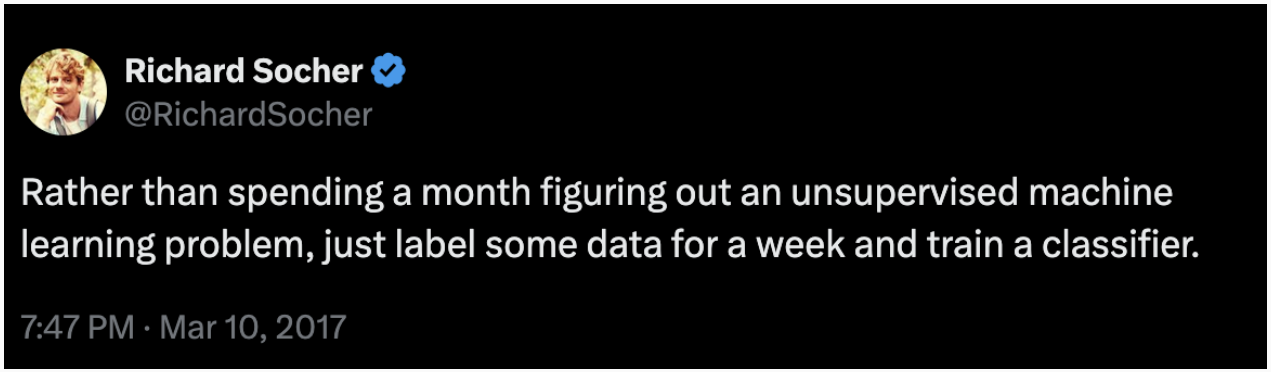
Universidad de Chile – DCC

CC5205, Fall 2025

Clustering

Supervised and Unsupervised Learning

- Supervised learning:
  - Objective: Learn a function f that predicts a variable Y from an instance $\mathbf{X}$.
  - Data: Training set $(\mathbf{X}_i, Y_i)$
- Unsupervised learning:
  - Objective: Discover a structure within a set of instances $(\mathbf{X}_i)$.
  - Data: Training set $(\mathbf{X}_i)$
- The first case is the simplest:



Clustering

Outline : Clustering

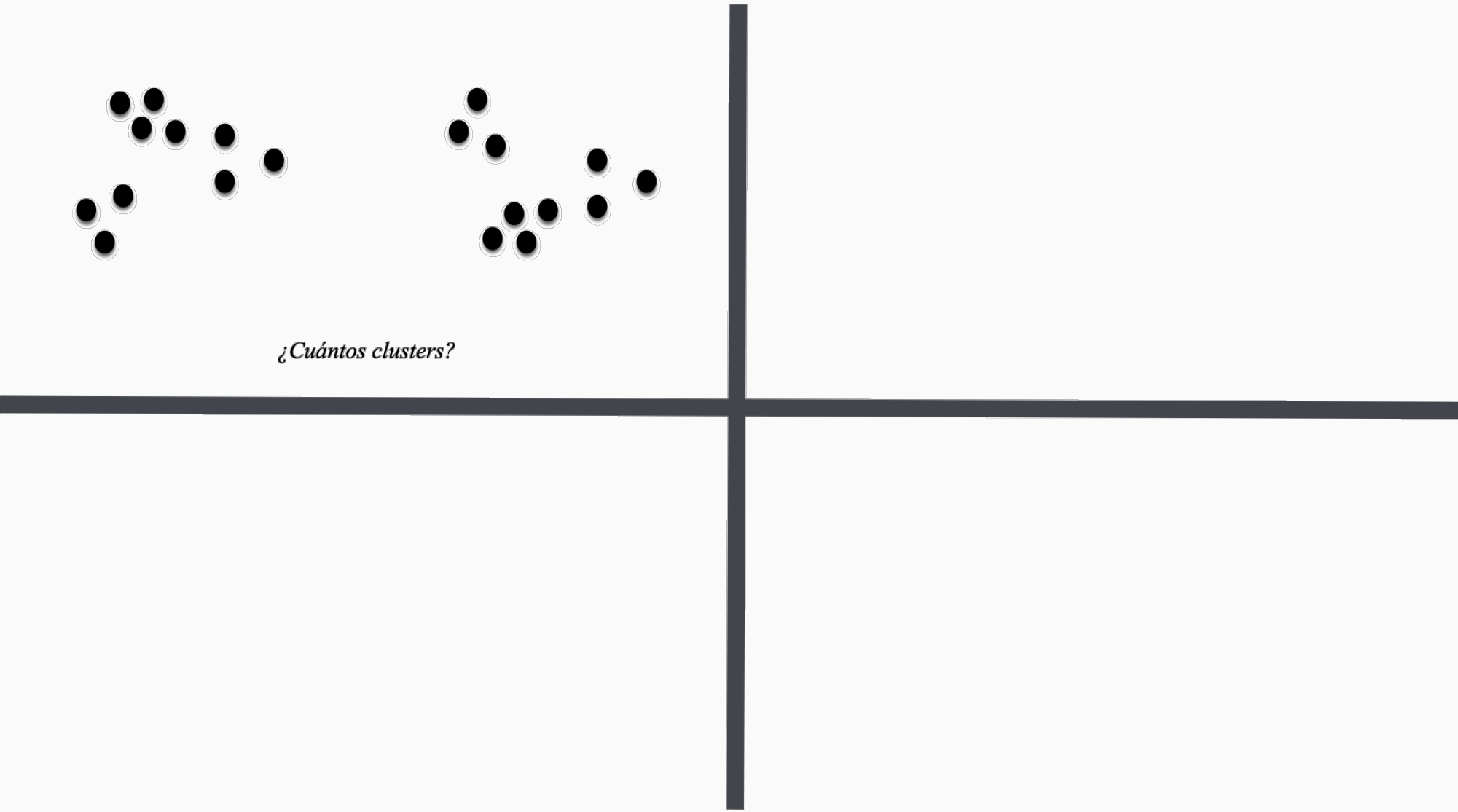
Outline : Principle

Why?

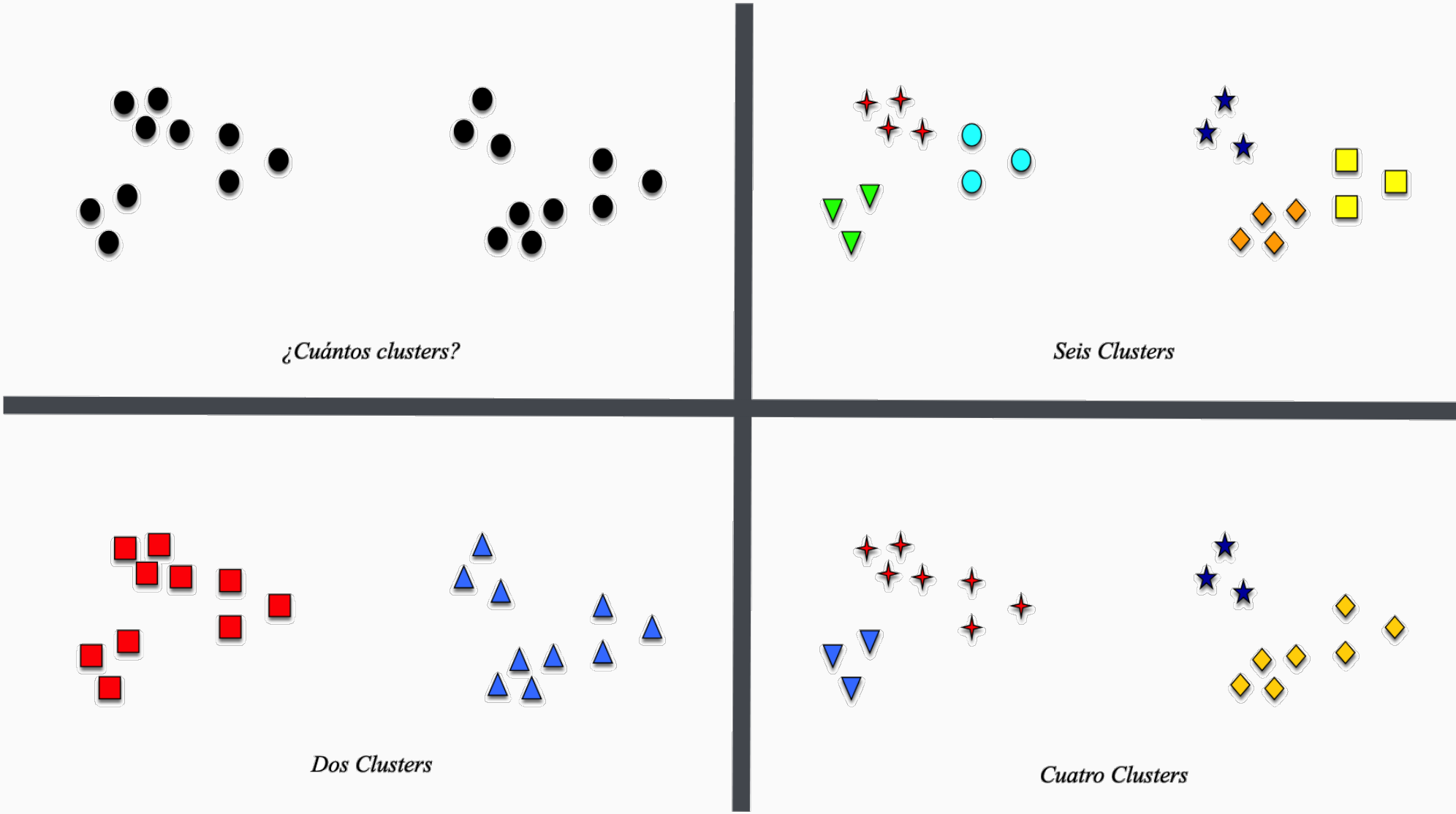
Because it can be useful to create coherent groups of documents:

- **Topic modeling**: to identify different topics of interest in documents or comments
- **Praise/criticism discovery**: clustering user comments by sentiment
- **Customer grouping**: grouping customers together
- **Efficiency**: train sub-models on different clusters

Why Isn't It That Simple? Ambiguity



Why Isn't It That Simple? Ambiguity



How? Minimize and Maximize Distances

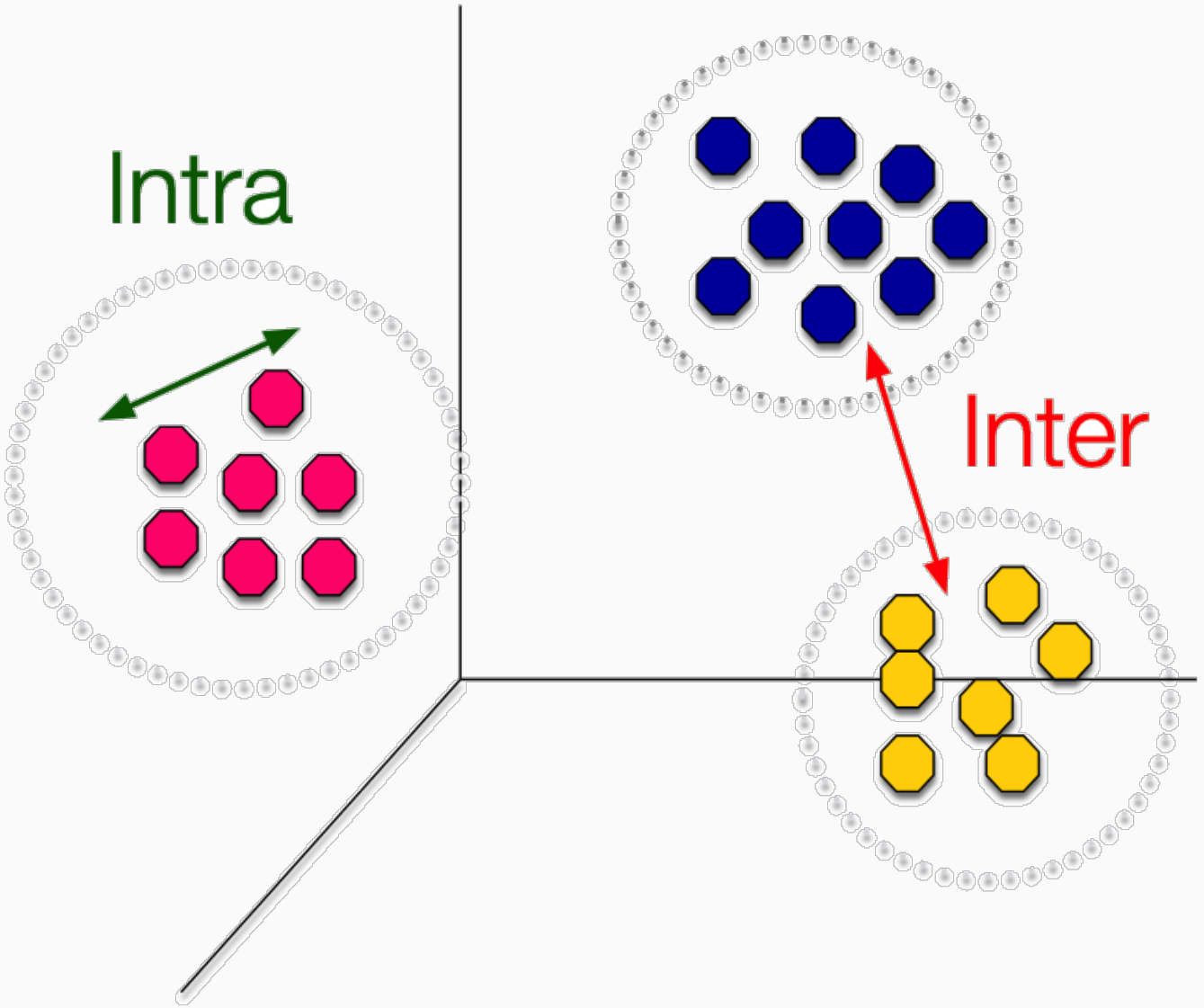


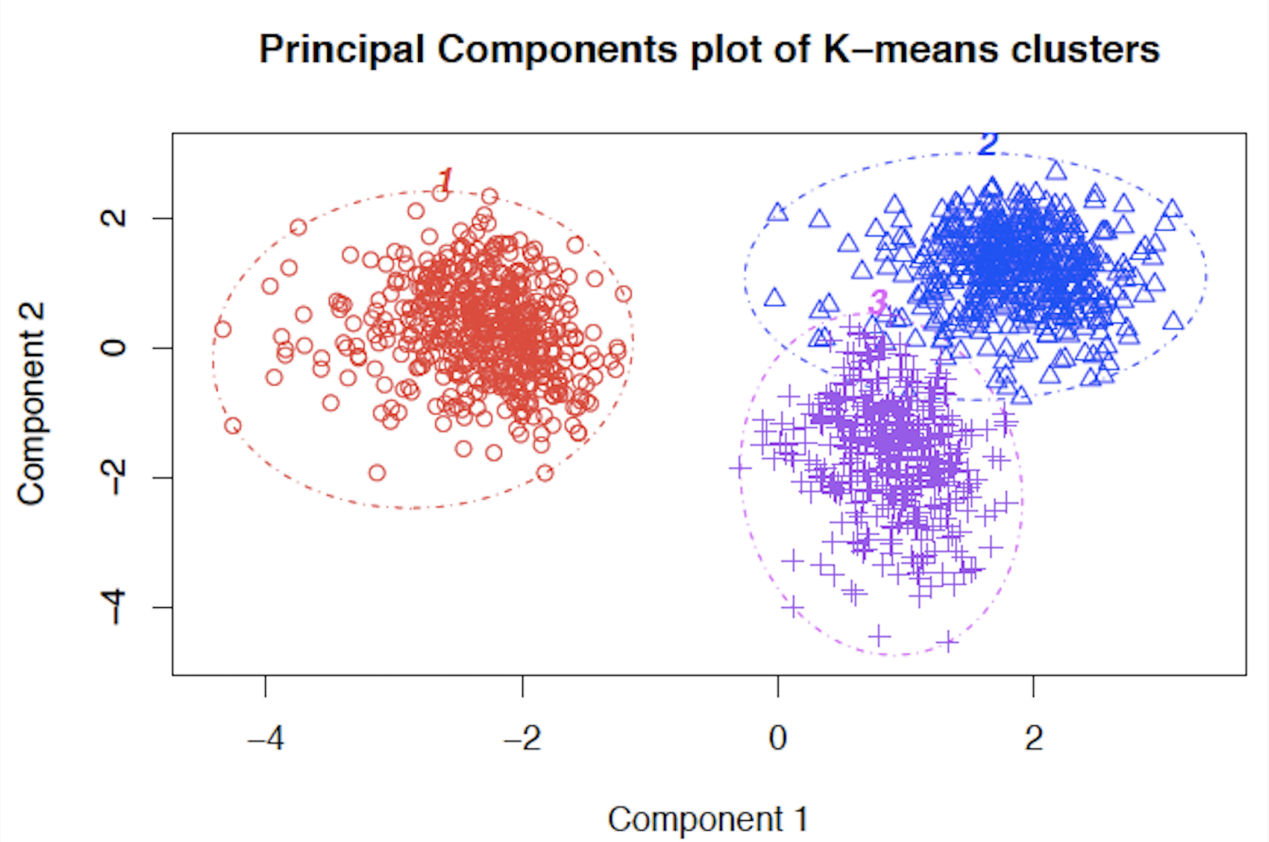
Figure 2: Reduce intra-cluster distance and increase inter-cluster distance

Outline : K-means

K-means

Principle

Partition the data into K *clusters* using their centroids.



K-means: Theory — Minimizing the SSE

The K-Means algorithm partitions the points into K disjoint groups $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ by minimizing the within-cluster variance. The criterion G being minimized is called the **inertia** or SSE (Sum of Squared Errors):

$$G_X(\mathcal{C}_1, \ldots, \mathcal{C}_K) = \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} \|\mathbf{X}_i - \mu_k\|_2^2$$

where the $\mu_k \in \mathbb{R}^d$ are the centroids of the K clusters ($|\mathcal{C}_k|$ is the cardinality of each cluster):

$$\mu_k = \frac{1}{|\mathcal{C}_k|} \sum_{i \in \mathcal{C}_k} \mathbf{X}_i, \quad \forall k \in [K],$$

and where the i-th observation is assigned to the class of its nearest centroid, i.e. $i \in \mathcal{C}_j$ if and only if:

$$j = \arg\min_{k \in [K]} \|\mathbf{X}_i - \mu_k\|_2 = \arg\min_{k \in [K]} \|\mathbf{X}_i - \mu_k\|_2^2.$$

Note: ties are broken at random if necessary.

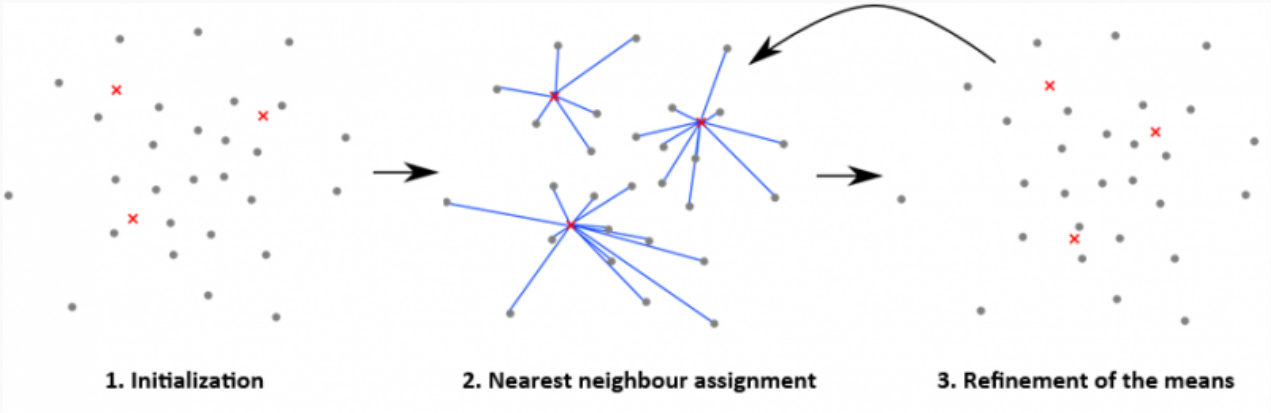
K-means: Learning

Cluster learning is done by iteratively alternating the following two steps:

I) *Assignment step*: given the current centroids $(\mu_k)_{k=1,\dots,K}$, assign each point to the nearest centroid.

II) *Centroid update step*: given the cluster assignments, recompute each cluster's centroid.

The algorithm stops when the inertia no longer decreases significantly.



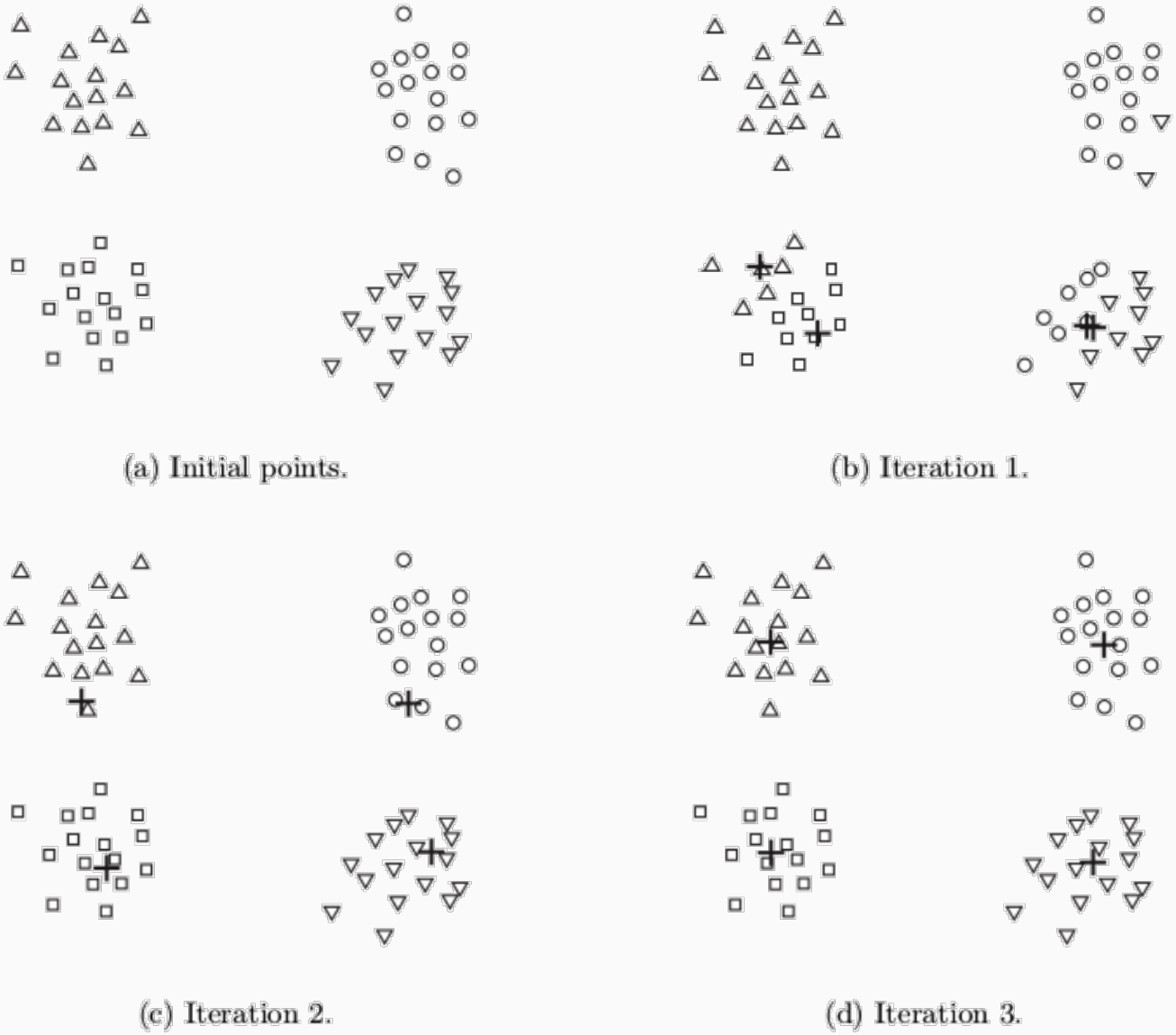
K-means: Initialization

Inertia is a non-convex criterion, so the solution found depends on the initialization, as is often the case with such problems.

Solution

Run the algorithm multiple times with different initializations, then select the solution with the lowest inertia.

K-means: Initialization

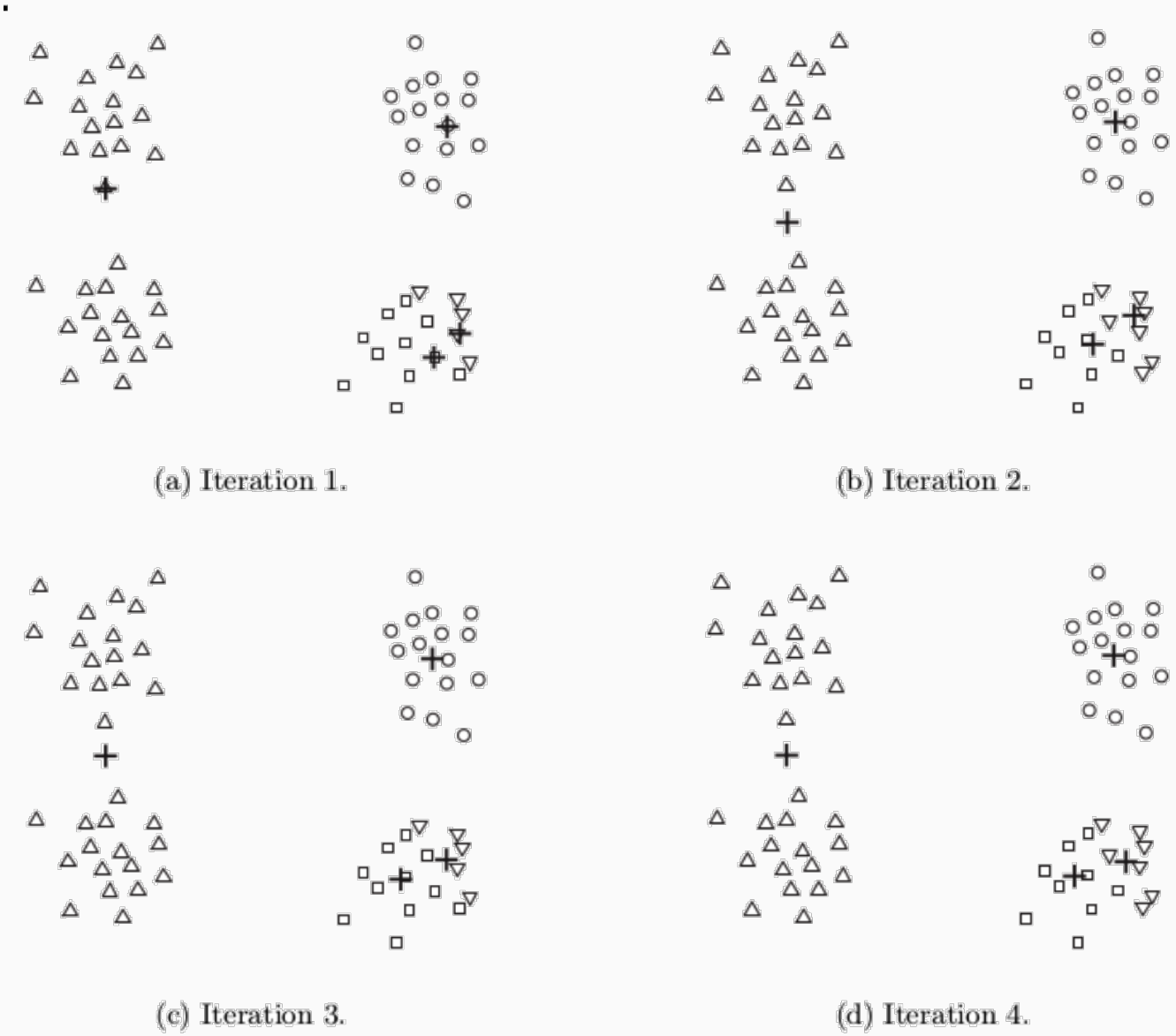


(a) Initial points. (b) Iteration 1.

(c) Iteration 2. (d) Iteration 3.

Figure 3: Initialization impacts the final clustering

K-means: Initialization



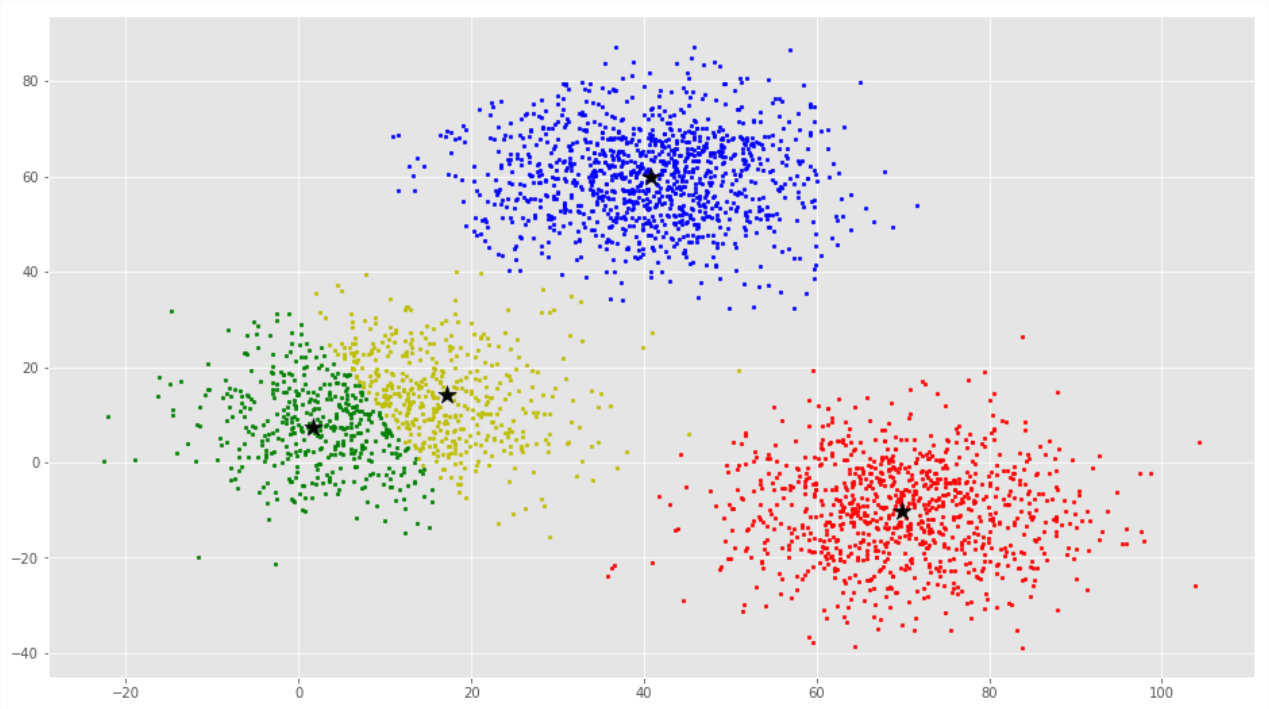
(a) Iteration 1. (b) Iteration 2.

(c) Iteration 3. (d) Iteration 4.

Figure 3: Initialization impacts the final clustering

K-means: Choosing K

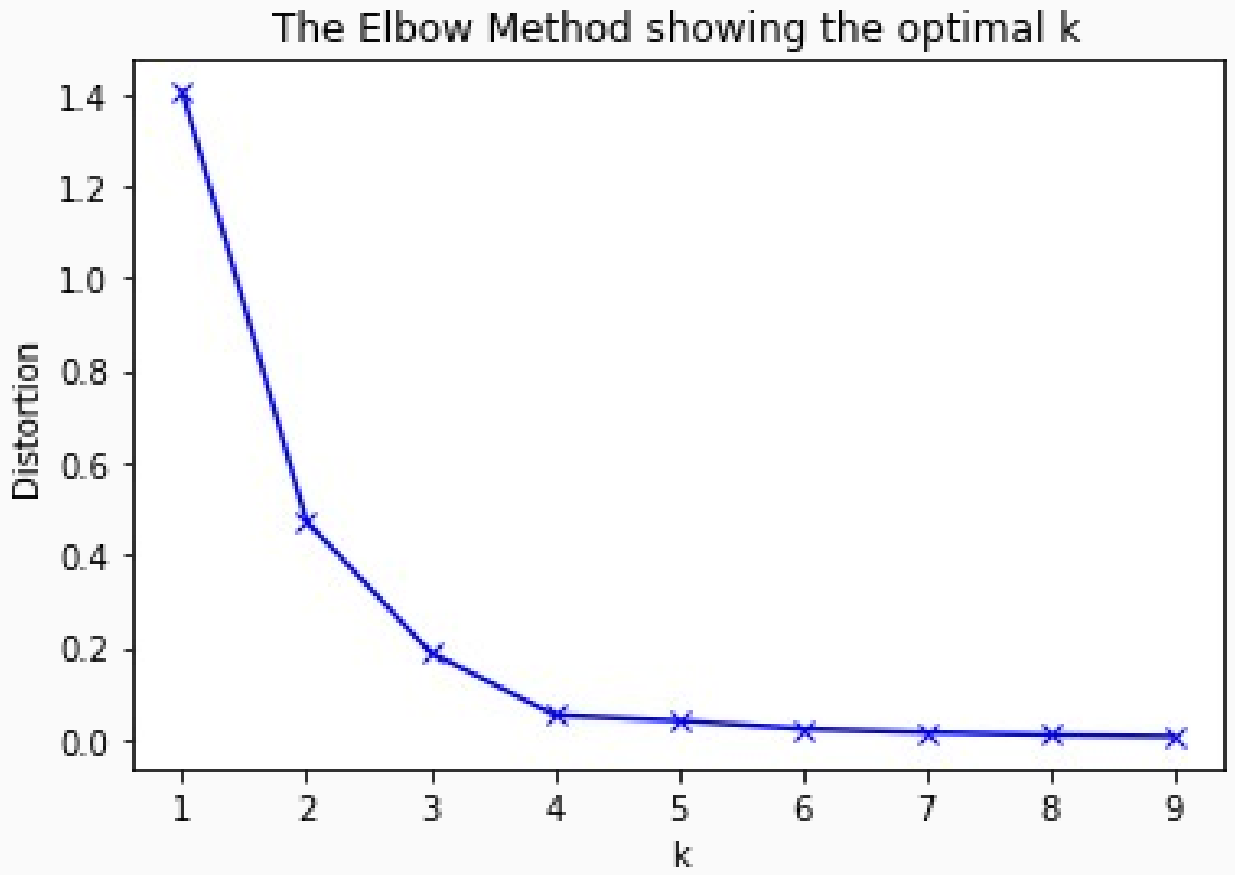
The results vary greatly depending on the initial choice of K.



K-means: Elbow Method

Most common method to find the optimal K

Run K-Means with different values of K and compute the cluster variances. As K increases, the variance decreases. Stop when the variance stops decreasing significantly.



K-means: Another Method

We still consider a dataset X of n points. Let $X_t, t \in [T]$, be T independent samples of n random points drawn uniformly over the bounding box of X. From the inertias G_X and G_{X_t} of K-means run on X and X_t, we define the gap statistic difference:

$$\delta(k) = \mathrm{Gap}(k) - \big(\mathrm{Gap}(k+1) - \sigma(k+1)\big),$$

where $\mathrm{Gap}(k) \in \mathbb{R}$ is the difference between the expected log inertia of G_{X_t} and the log inertia of G_X:

$$\mathrm{Gap}(k) = \mathbb{E}[\log G_{X_t}] - \log G_X,$$

and where

$$\sigma(k) = \sqrt{\frac{T+1}{T}\,\mathbb{E}\Big[\big(\log G_{X_t} - \mathbb{E}[\log G_{X_t}]\big)^2\Big]}.$$

Outline : DBSCAN

DBSCAN: Example

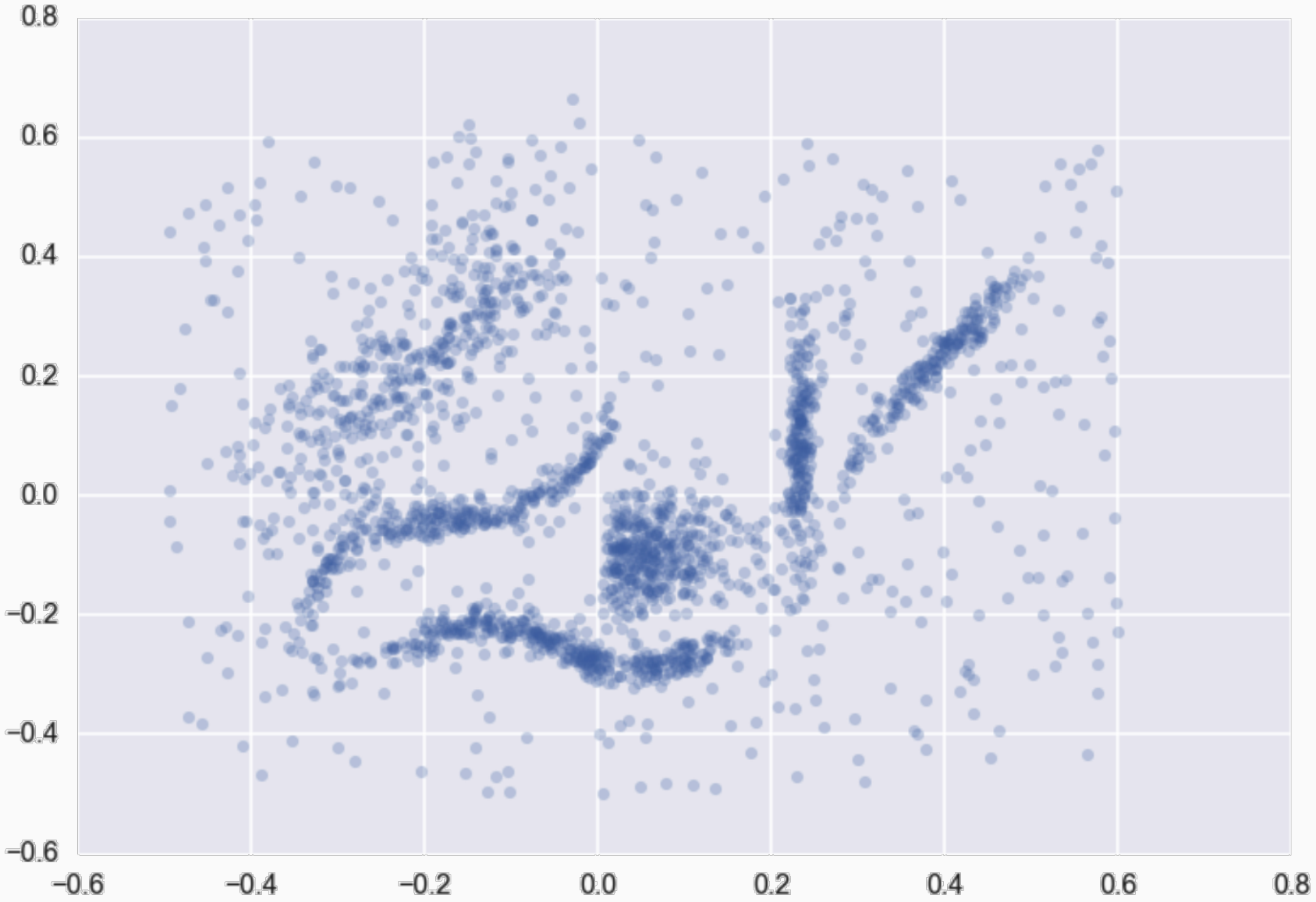


Figure 4: DBSCAN allows you to discard noise

DBSCAN: Example

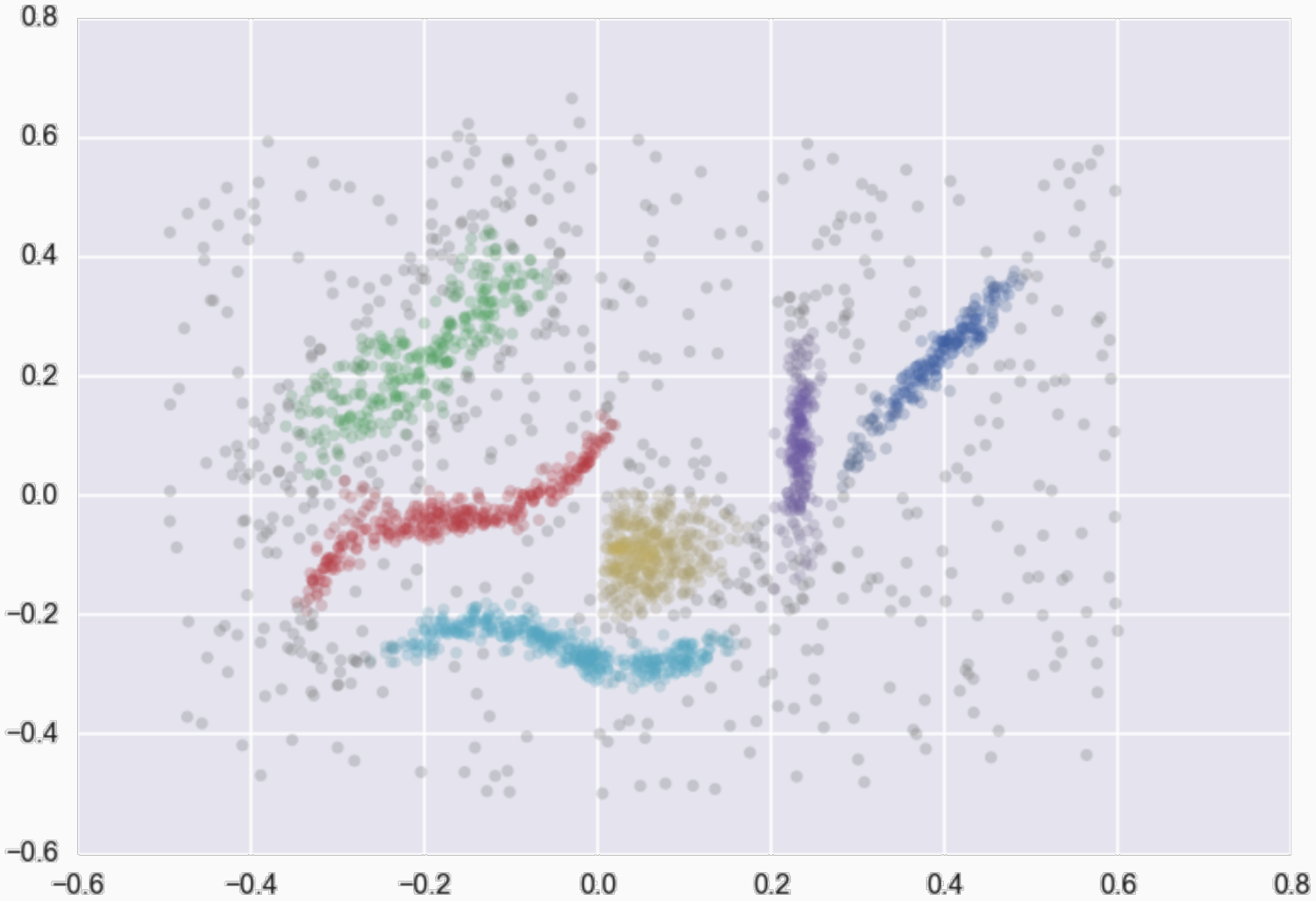


Figure 4: DBSCAN allows you to discard noise

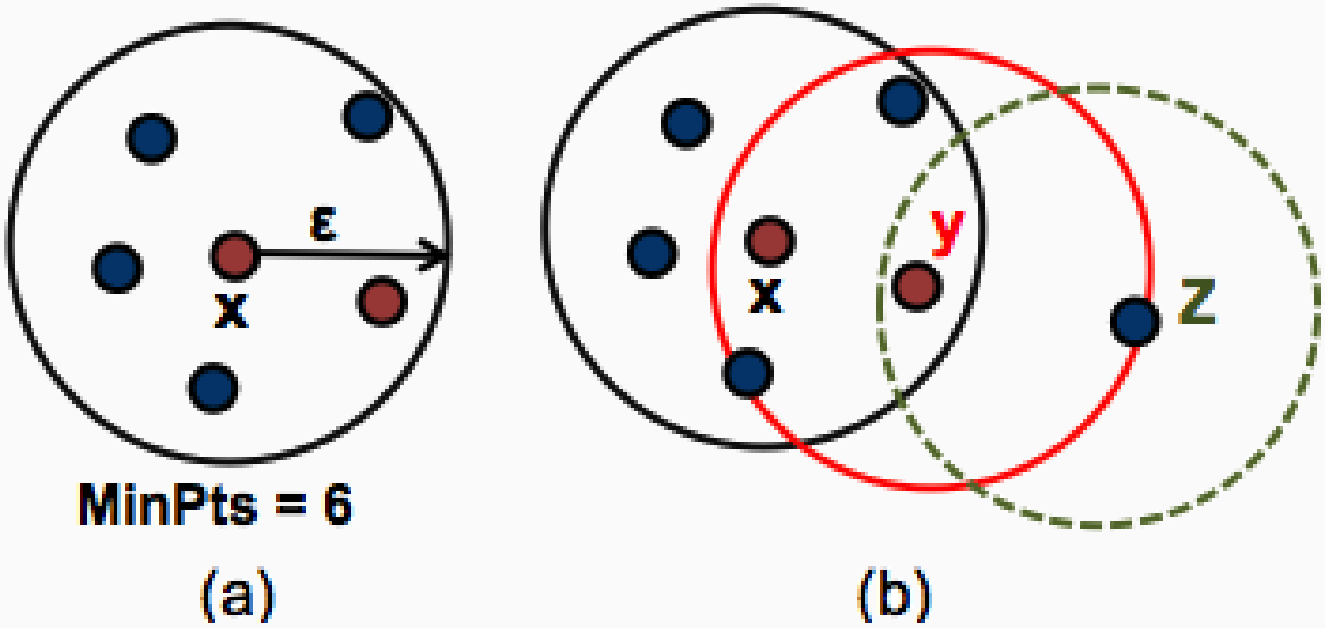
DBSCAN: Principles I

- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** forms clusters in dense regions of points.
- Two important parameters:
  - `eps`: neighborhood radius.
  - `min_samples`: minimum number of samples within that radius for a point to be considered a core point.

Classifies points as core, border, or noise

- **Core points**: have at least `min_samples` neighbors within `eps`.
- **Border points**: are assigned to the same cluster as a core point if they are within its `eps` neighborhood.
- **Noise points**: do not meet either condition above.

DBSCAN: Principles II



- Each point has neighbors within a circle of radius `eps` (ε).
- Point x is a core point because it has more than `MinPts` neighbors.
- Point y is a border point as it does not have `MinPts` neighbors.
- Point z is a noise point as it is not neighbor (within `eps`) of any core point.
- Border points are assigned to the neighboring cluster.

Soft clustering can also be performed with DBSCAN.

DBSCAN: Advantages and Disadvantages

Advantages

- Detect **arbitrarily-shaped clusters**; not assuming spherical shapes
- Distinguishes **noise** (outliers) from the rest of the data.
- Usually requires specifying fewer parameters than hierarchical methods.

DBSCAN: Advantages and Disadvantages

Advantages

- Detect **arbitrarily-shaped clusters**; not assuming spherical shapes
- Distinguishes **noise** (outliers) from the rest of the data.
- Usually requires specifying fewer parameters than hierarchical methods.

Disadvantages

- **Sensitive** to the choice of `eps` and `min_samples`.
- If the density varies greatly within the dataset, a single `eps` may not be appropriate for all clusters.

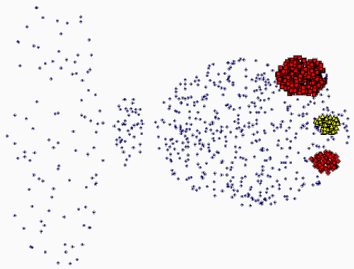


Figure 5: $\varepsilon = 9.92$

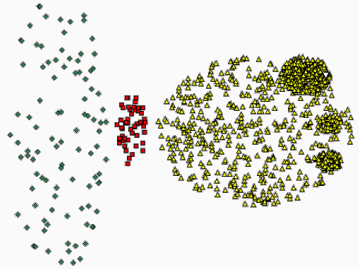
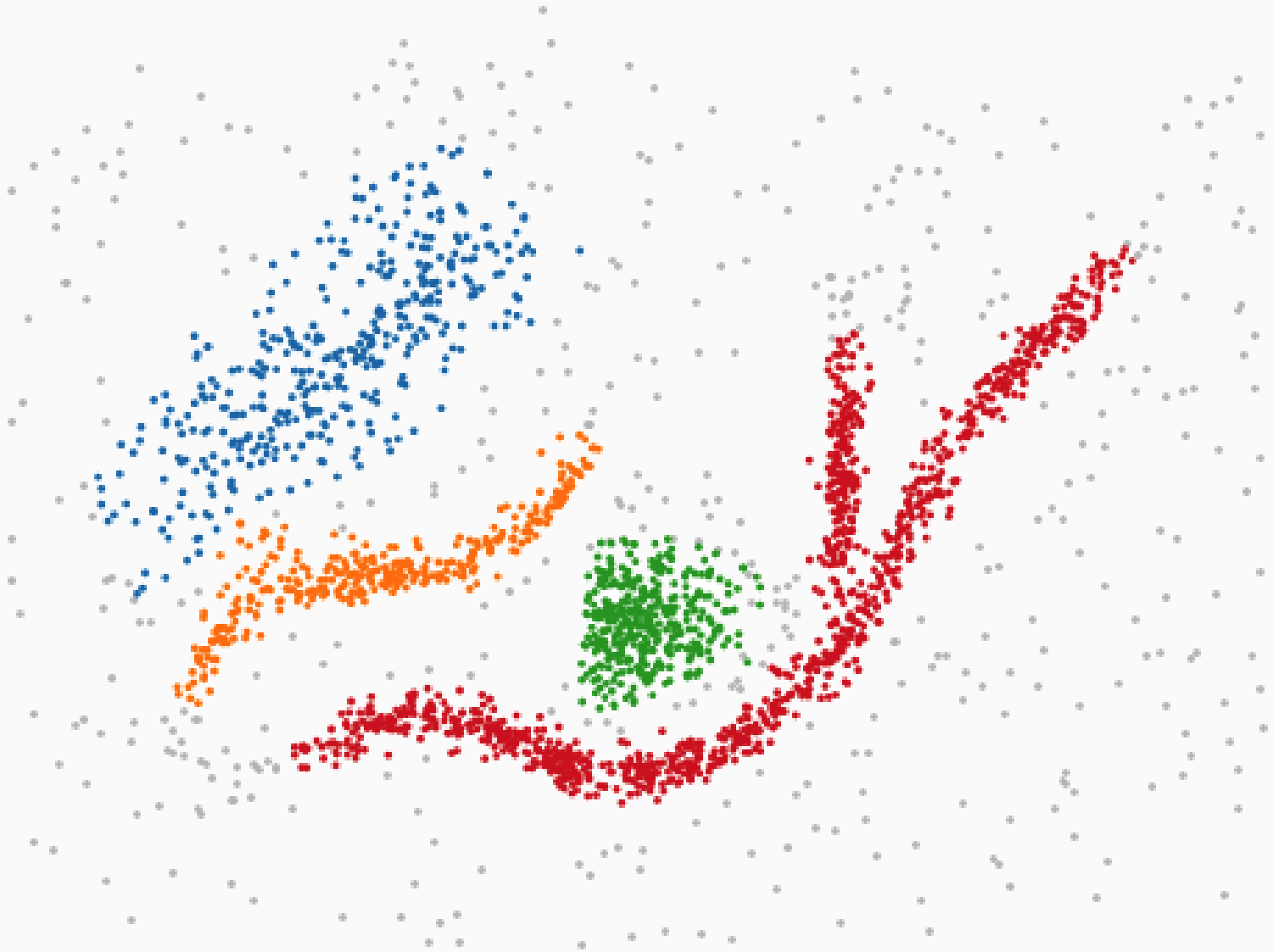
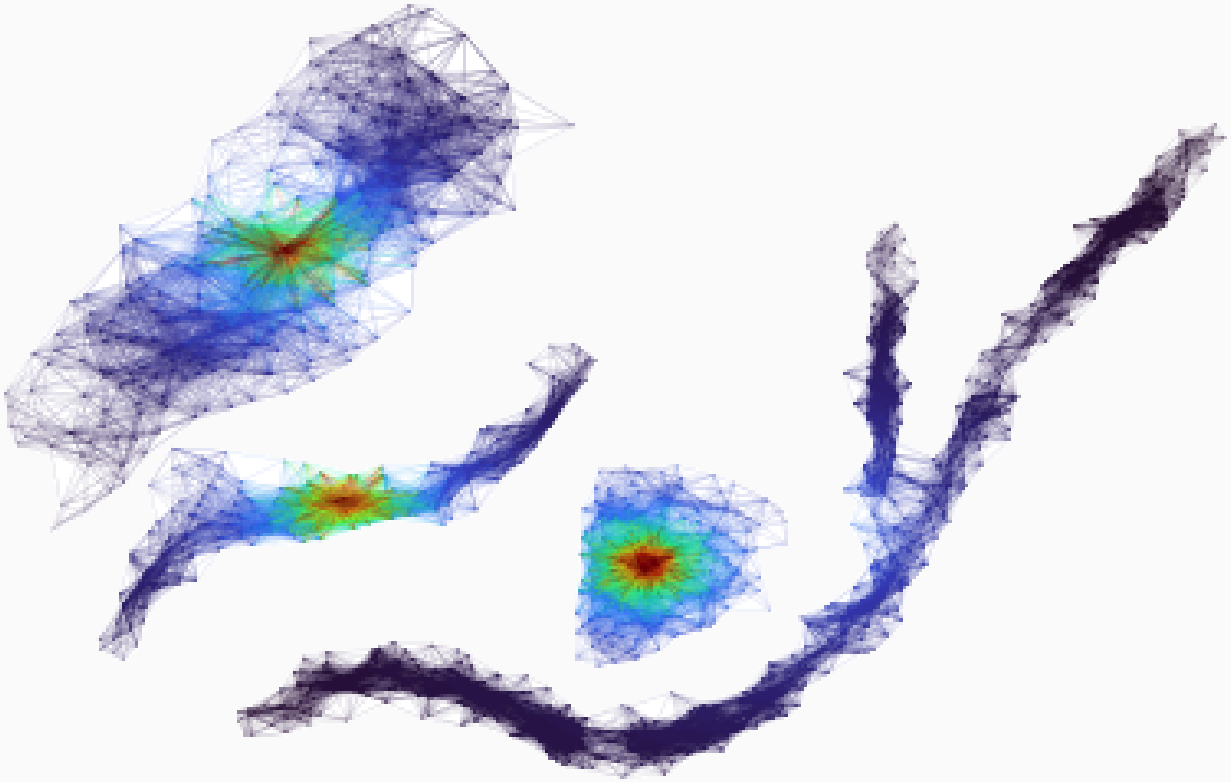


Figure 6: $\varepsilon = 9.75$

DBSCAN: Graph-like Visualization



DBSCAN: Graph-like Visualization



DBSCAN: Choosing `minPts` and ε

Rule of Thumb for `minPts`

As a general rule, set `minPts` at least to the dimensionality D of the data plus one, i.e. $\texttt{minPts} \geq D + 1$.

ε: k-Distance Graph and Elbow

- To choose ε, plot the **k-distance** graph (with $k = \texttt{minPts} - 1$), sorting distances in descending order.
- A "good" ε often appears at the **elbow** of this curve.
- Too small ε: many points remain unassigned as noise.
- Too large ε: clusters merge into one large cluster.
- **Practical rule**: choose ε so that only a small fraction of points lie within that distance.

One can also use the **OPTICS** algorithm to select ε.

DBSCAN: Elbow Method

- Compute the k-distance for all points with a fixed k.
- Sort the k-distance values in ascending order.
- Plot k-distance vs. number of points.
- Identify the "jump" in the k-distance plot (e.g. 10 in the Figure).
- `minPts` can be set to the k at that jump.

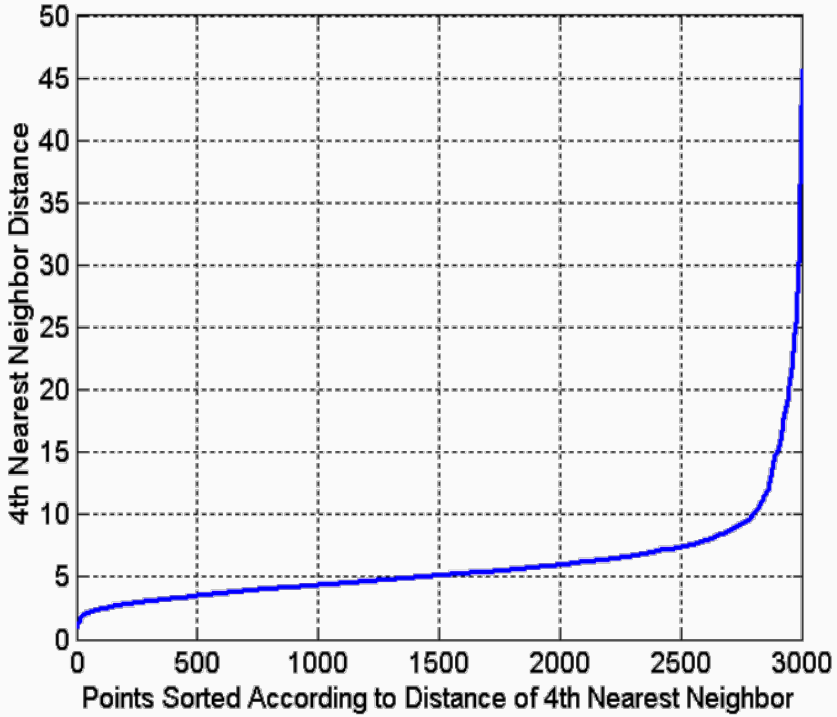


Figure 7: A k-distance plot

DBSCAN: Disadvantages

- Does not handle variable densities well $\rightarrow$ variable-density regions may be marked as noise.
- High dimensionality: density estimation becomes difficult.

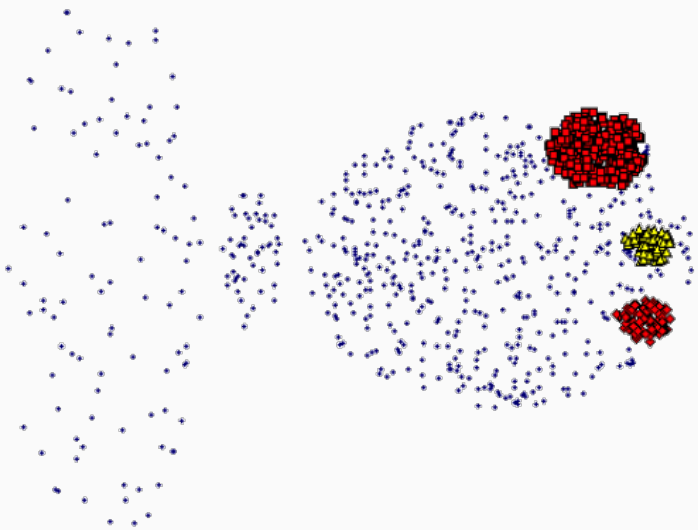


Figure 8: $\varepsilon = 9.92$

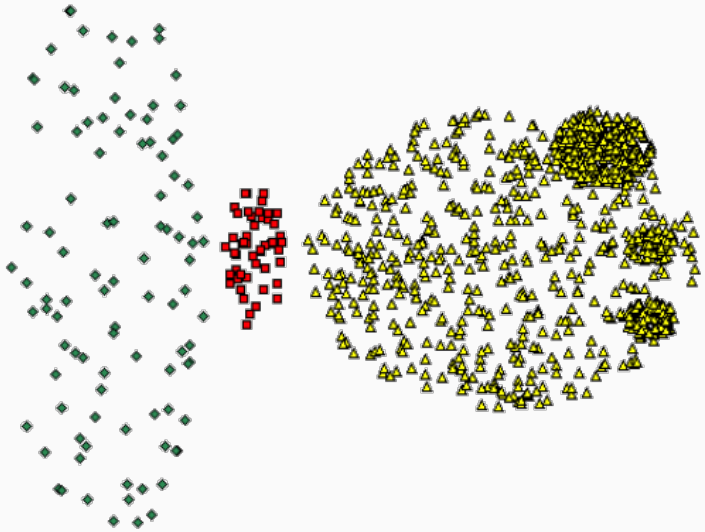
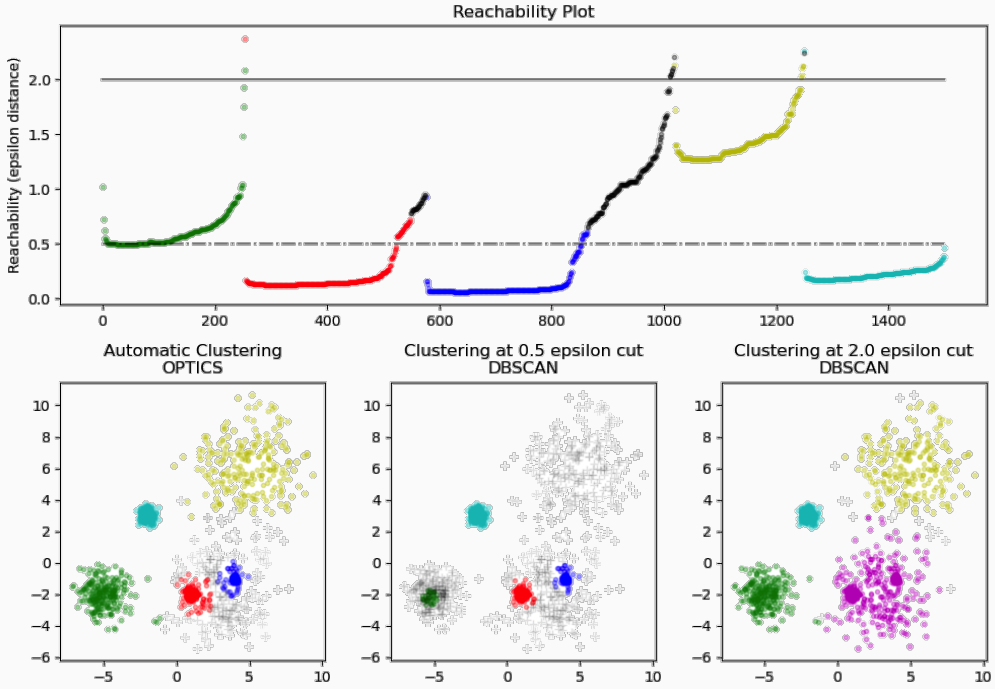


Figure 9: $\varepsilon = 9.75$

OPTICS: A Generalization of DBSCAN

- Allows **variable densities** by adjusting a range of ε, instead of a fixed value.
- Builds a **reachability graph**, assigning each sample a `reachability_distance` and an `ordering_`.
- The **reachability plot** helps identify clusters by "cutting" the plot and detecting abrupt changes (parameter ξ).
- Provides a **hierarchical clustering structure** and can emulate DBSCAN for different ε values without full re-computation.



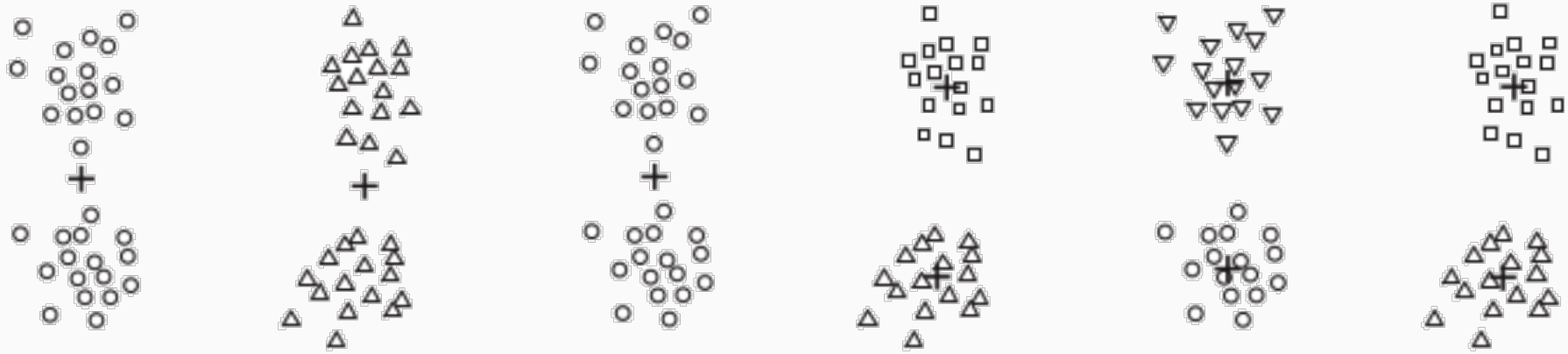
Hierarchical Clustering

Outline : Hierarchical Clustering

Outline : Bisecting K-means

Bisecting K-means: General Idea

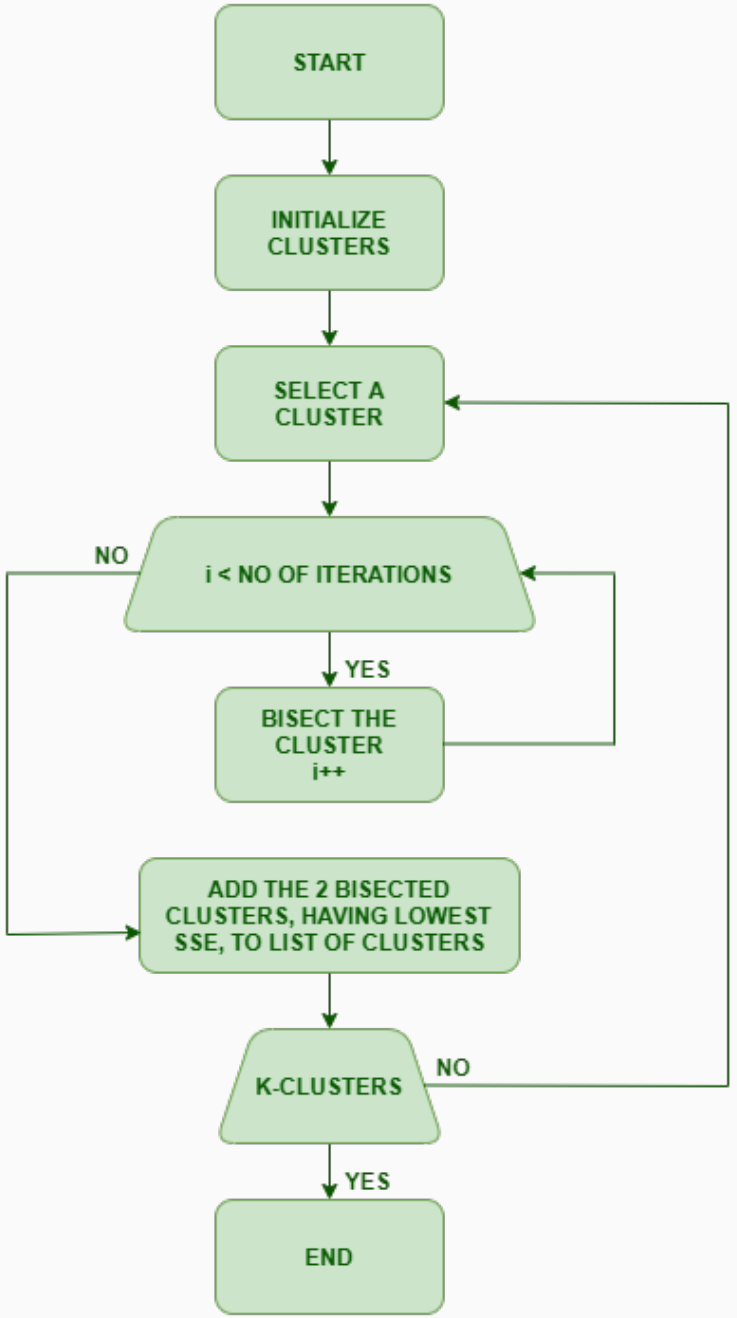
- **Bisecting K-means** is a hierarchical variant of K-means:
  - Start by treating all data as a single cluster.
  - Apply K-means (with $K = 2$) to split that cluster into two subgroups.
  - Select the subgroup with the highest inertia (SSE) and split it again.
  - Repeat until the desired number of clusters is reached.
- Combines the **advantages** of K-means with a hierarchical approach and can outperform simple K-means when K is large.
- Tends to produce clusters with a more **regular large-scale structure**.



(a) Iteration 1. (b) Iteration 2. (c) Iteration 3.

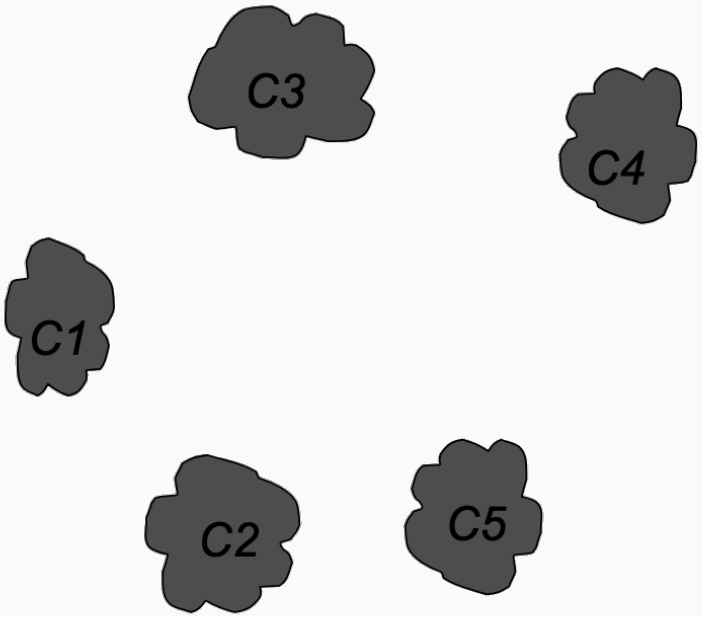
Bisecting K-means: Algorithm

- Simple extension of K-means
- Split the set of all points into two clusters, pick one to split further, and iterate until K clusters are produced.
- Each split is performed by running K-means (with $k = 2$).



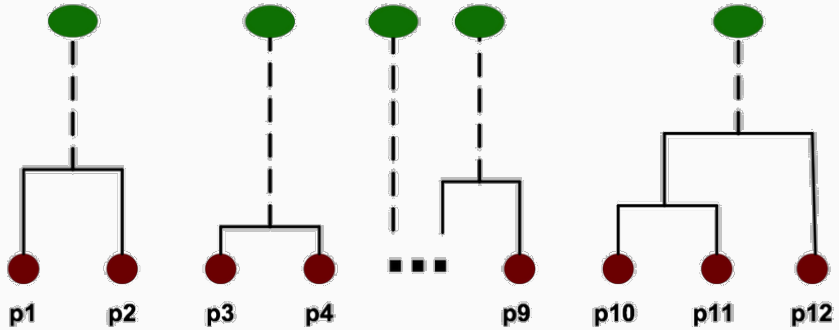
Outline : Agglomerative Hierarchical Clustering

Agglomerative Hierarchical Clustering: Principle



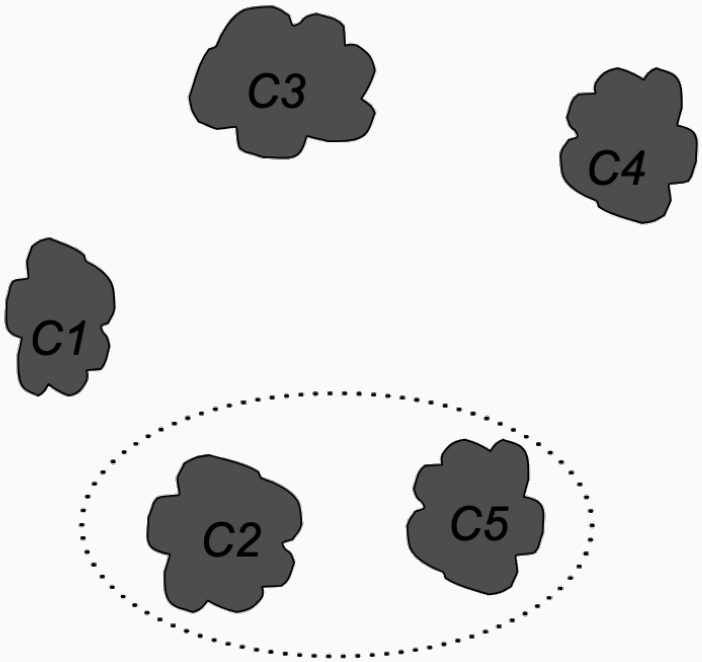
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

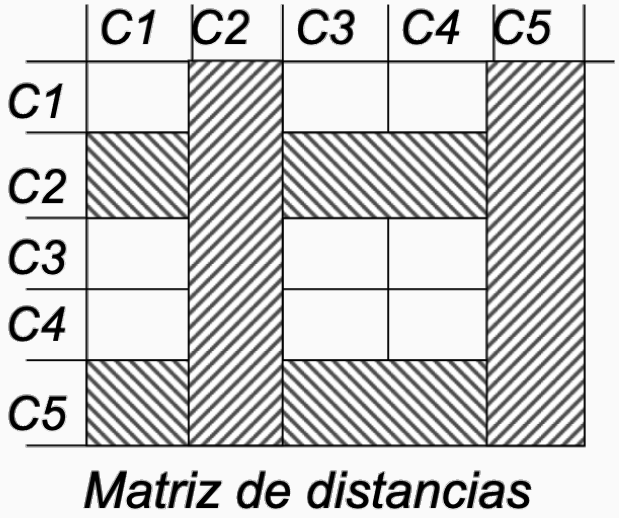
Matriz de distancias



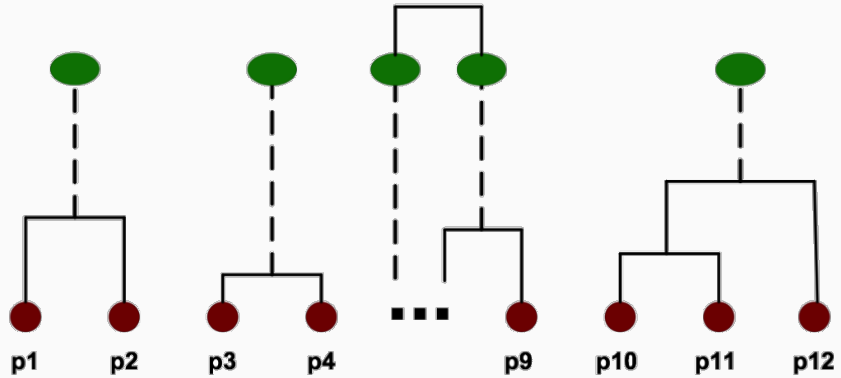
- Compute distances between clusters
- Merge the closest pair
- Recompute the distance matrix
- Represent the result as a tree (or dendrogram)

Agglomerative Hierarchical Clustering: Principle



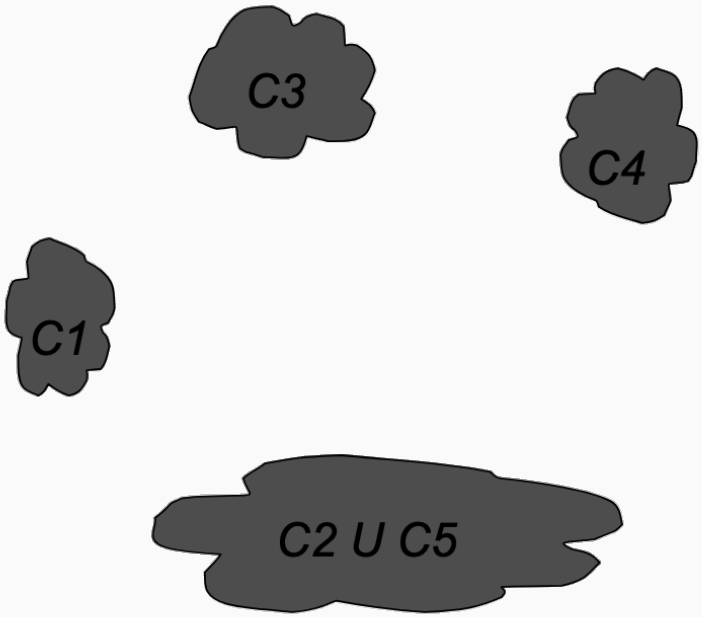


Matriz de distancias



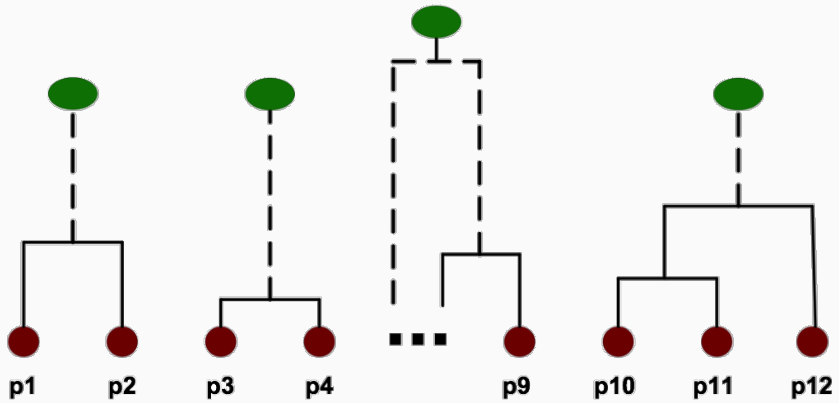
- Compute distances between clusters
- Merge the closest pair
- Recompute the distance matrix
- Represent the result as a tree (or dendrogram)

Agglomerative Hierarchical Clustering: Principle



	C1	C25	C3	C4
C1		?		
C25	?	?	?	?
C3		?		
C4		?		

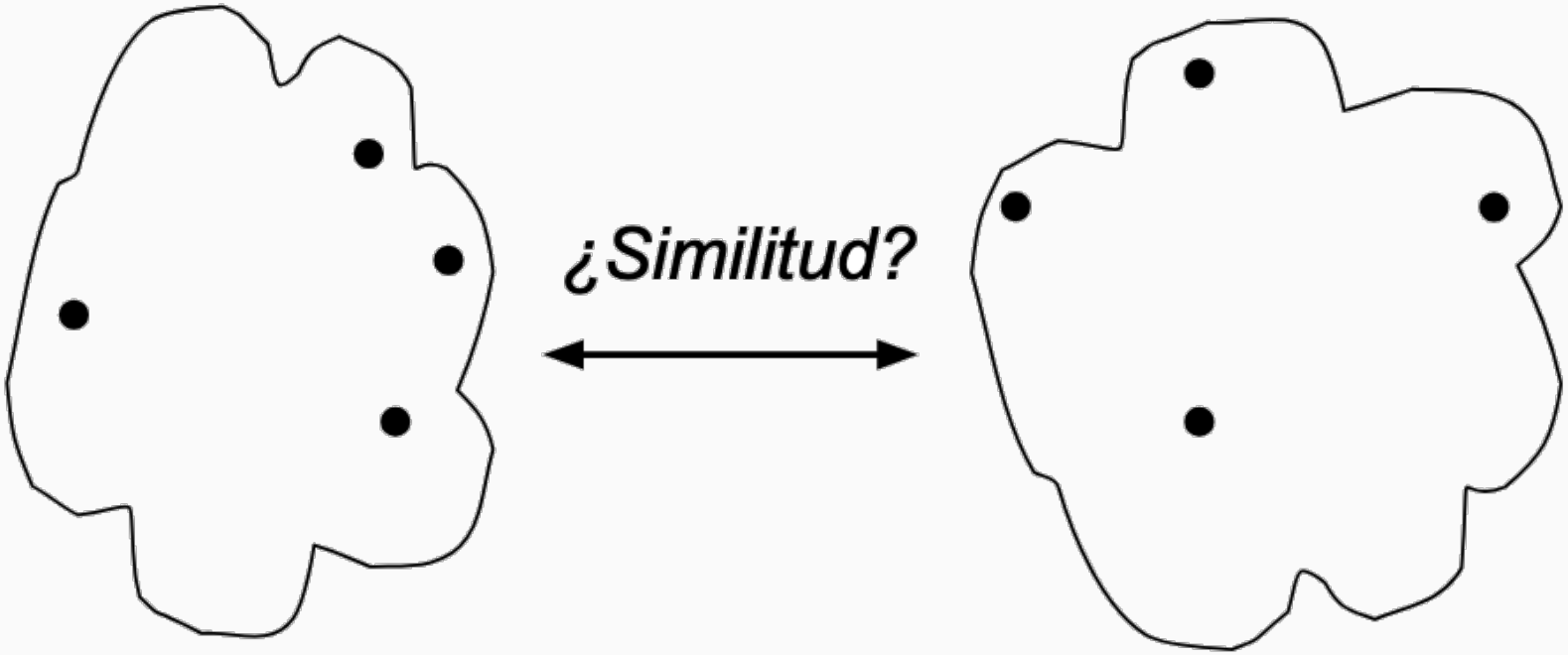
Matriz de distancias



- Compute distances between clusters
- Merge the closest pair
- Recompute the distance matrix
- Represent the result as a tree (or dendrogram)

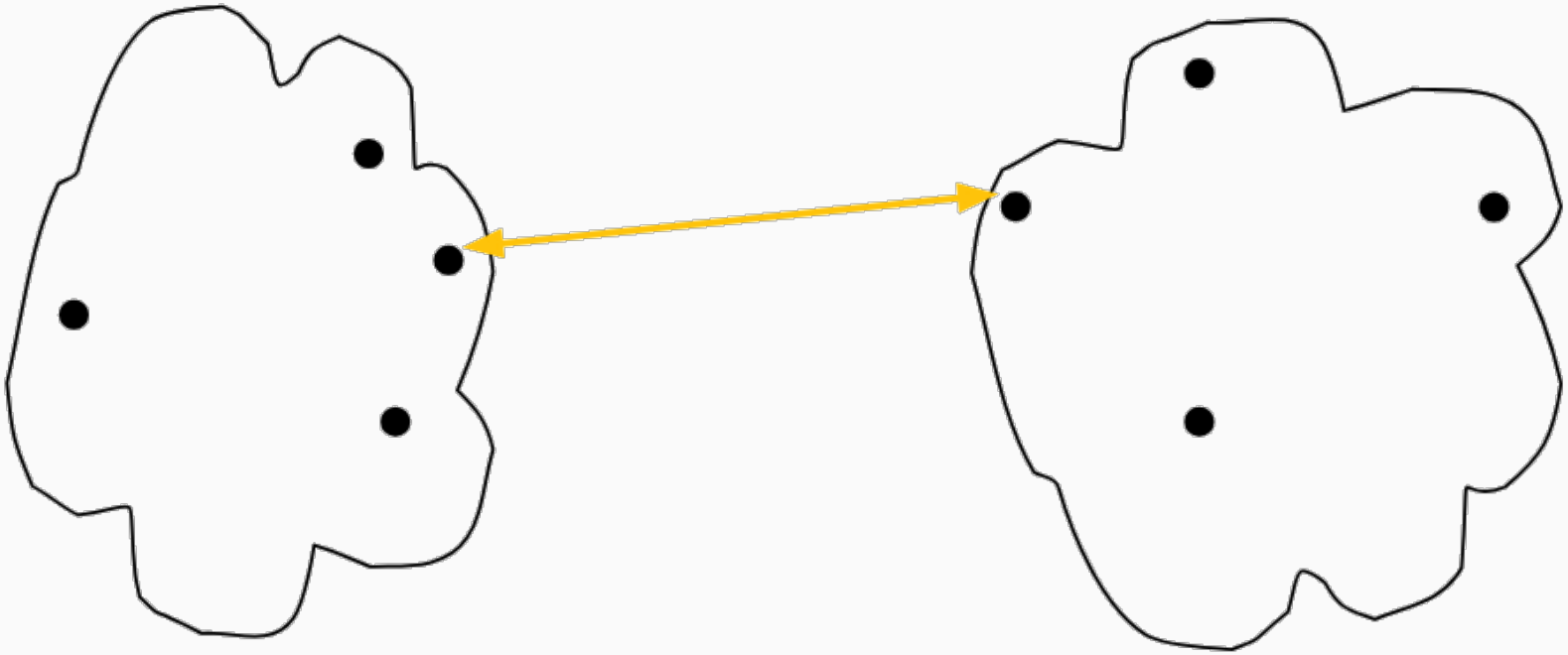
Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**



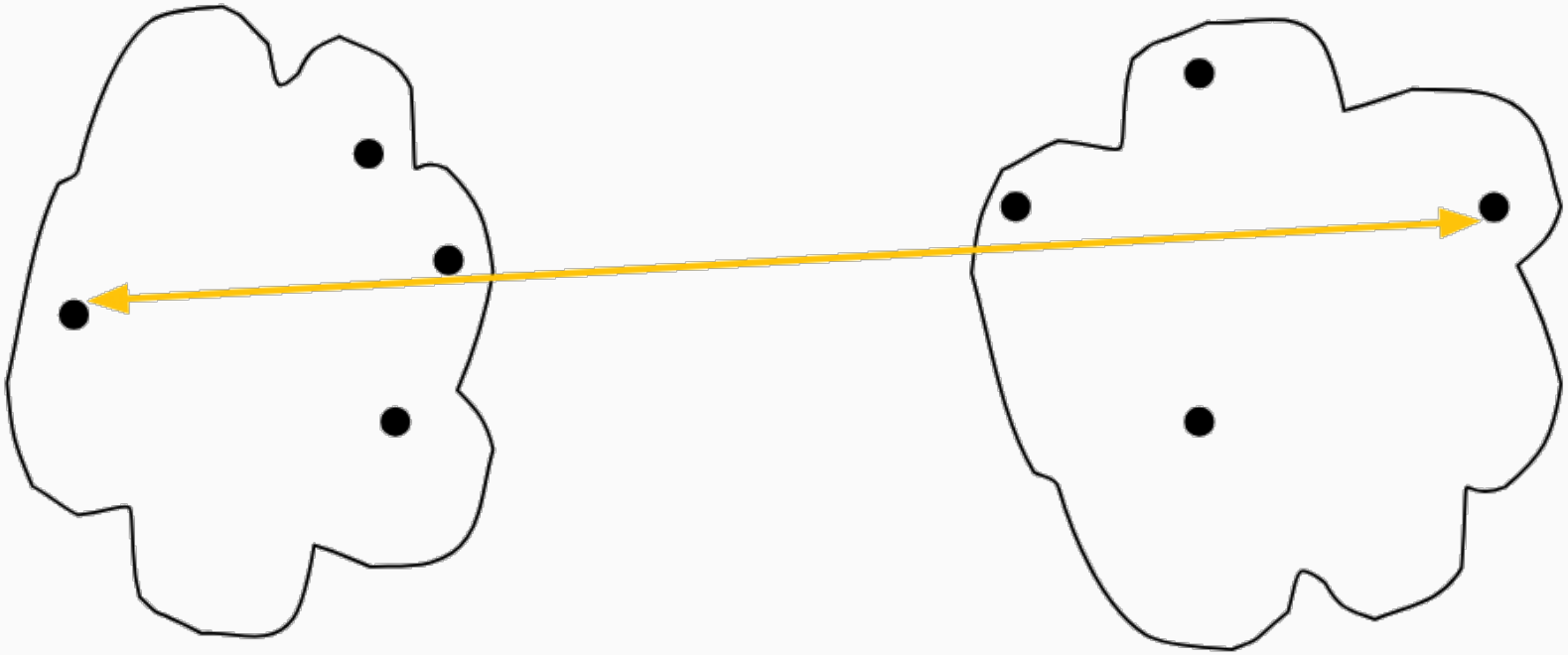
Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**
- *Single linkage*: distance between closest points — sensitive to noise/outliers



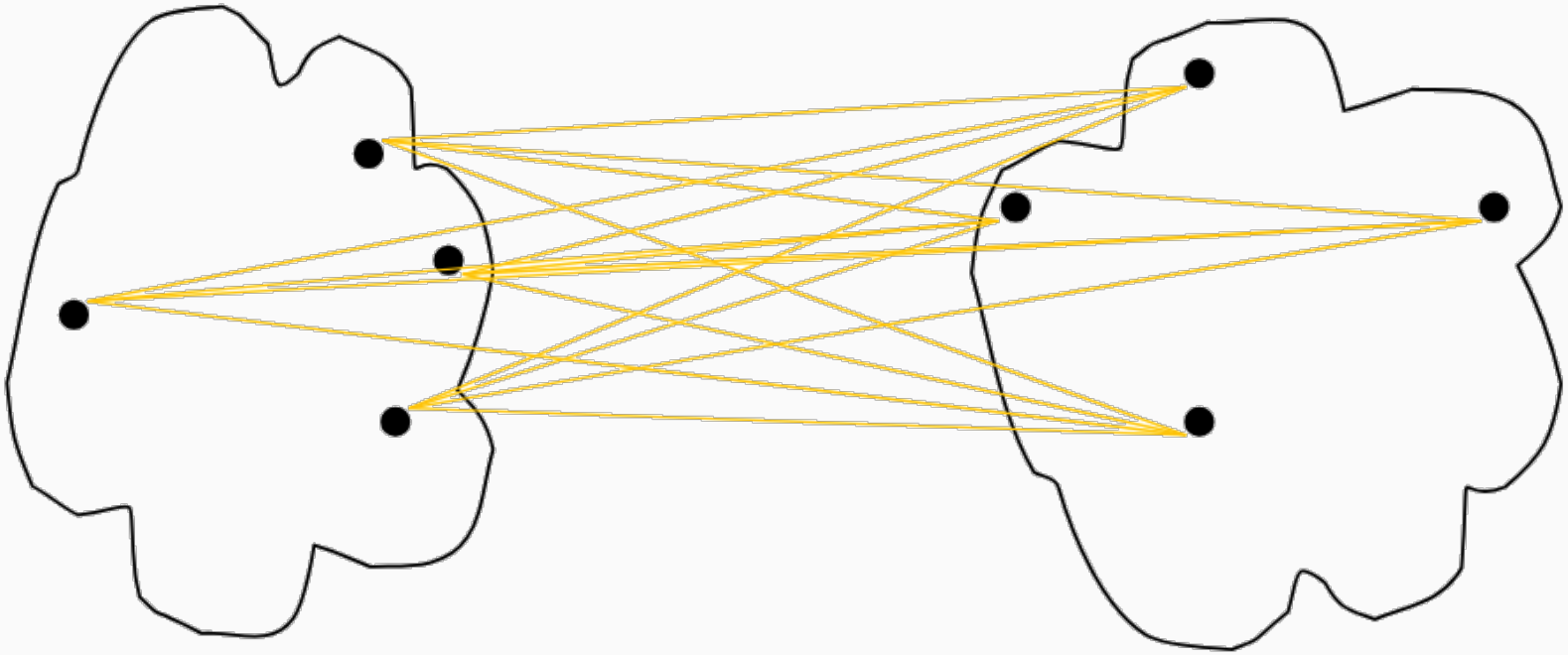
Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**
- *Single linkage*: distance between closest points — sensitive to noise/outliers
- *Complete linkage*: distance between farthest points — performs poorly on large, circular clusters



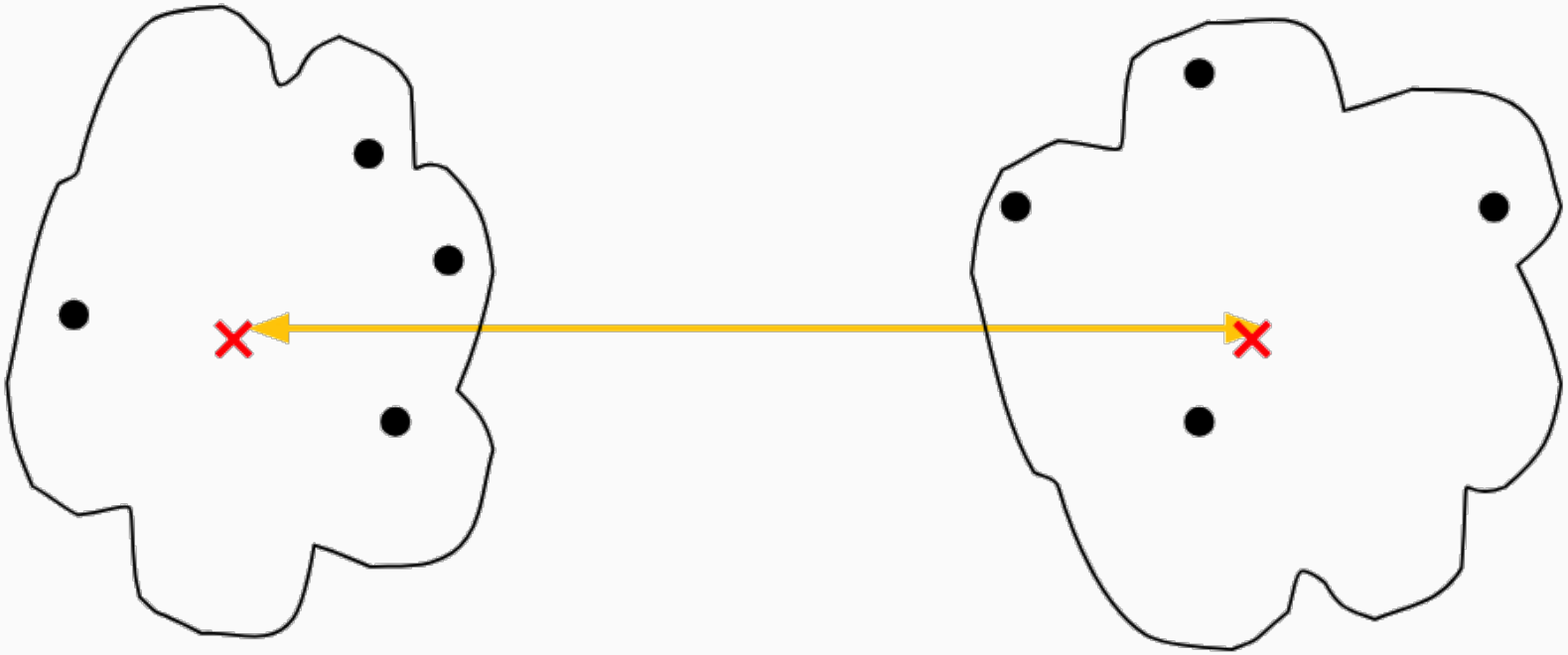
Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**
- *Single linkage*: distance between closest points — sensitive to noise/outliers
- *Complete linkage*: distance between farthest points — performs poorly on large, circular clusters
- *Average linkage*: compromise between single and complete



Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**
- *Single linkage*: distance between closest points — sensitive to noise/outliers
- *Complete linkage*: distance between farthest points — performs poorly on large, circular clusters
- *Average linkage*: compromise between single and complete
- *Centroid linkage*: using cluster centroids



Distances Between Clusters: Linkage Criteria

- **How to estimate distance/similarity between clusters?**
- *Single linkage*: distance between closest points — sensitive to noise/outliers
- *Complete linkage*: distance between farthest points — performs poorly on large, circular clusters
- *Average linkage*: compromise between single and complete
- *Centroid linkage*: using cluster centroids
- *Ward's method*: based on the increase in SSE when merging two clusters. Similar to the K-means objective but approached hierarchically.

Outline : Hierarchical DBSCAN

HDBSCAN

- **HDBSCAN** is a hierarchical extension of DBSCAN:
  - Explores multiple density levels by varying ε, building a hierarchy of clusters where every level corresponds to a different criterion of density.
  - Does not require a fixed `eps`; instead, uses a metric of cluster stability to select the most relevant subsets of the hierarchy.
  - Produces **arbitrarily shaped** clusters and handles outliers effectively.
- **Key advantage**: avoids sensitivity to a single `eps` and adapts to variable-density data.

User guide here

Validation Metrics

Outline : Validation Metrics

Clustering Validation

- Unlike supervised learning, clustering validation usually requires:
  - Knowing **true labels** (if available) to use external metrics.
  - Or using **internal metrics** (silhouette, SSE, etc.) that rely only on cluster structure.
- **Examples of external metrics**:
  - Based on mutual information.
  - Homogeneity, Completeness, V-measure.
  - Rand Index, Fowlkes–Mallows.
- **Examples of internal metrics**:
  - Silhouette score.
  - SSE / inertia (used in K-means).

Metrics Based on MI, Homogeneity, Completeness, and V-measure

Mutual Information (MI)

- Measures the **dependence** between random variables.
- The higher the value, the more information is shared between the cluster assignments and the true labels.

Homogeneity and Completeness

- **Homogeneity**: all points in a cluster belong to the same true class.
- **Completeness**: all points in a true class are assigned to the same cluster.
- **V-measure**: the harmonic mean of Homogeneity and Completeness.

Fowlkes–Mallows and Silhouette Index

Fowlkes–Mallows

- Based on comparing **pairs** of points.
- Combines **precision** and **recall** of the clustering partition against the true labels.
- Ranges from 0 (worst) to 1 (best).

Silhouette

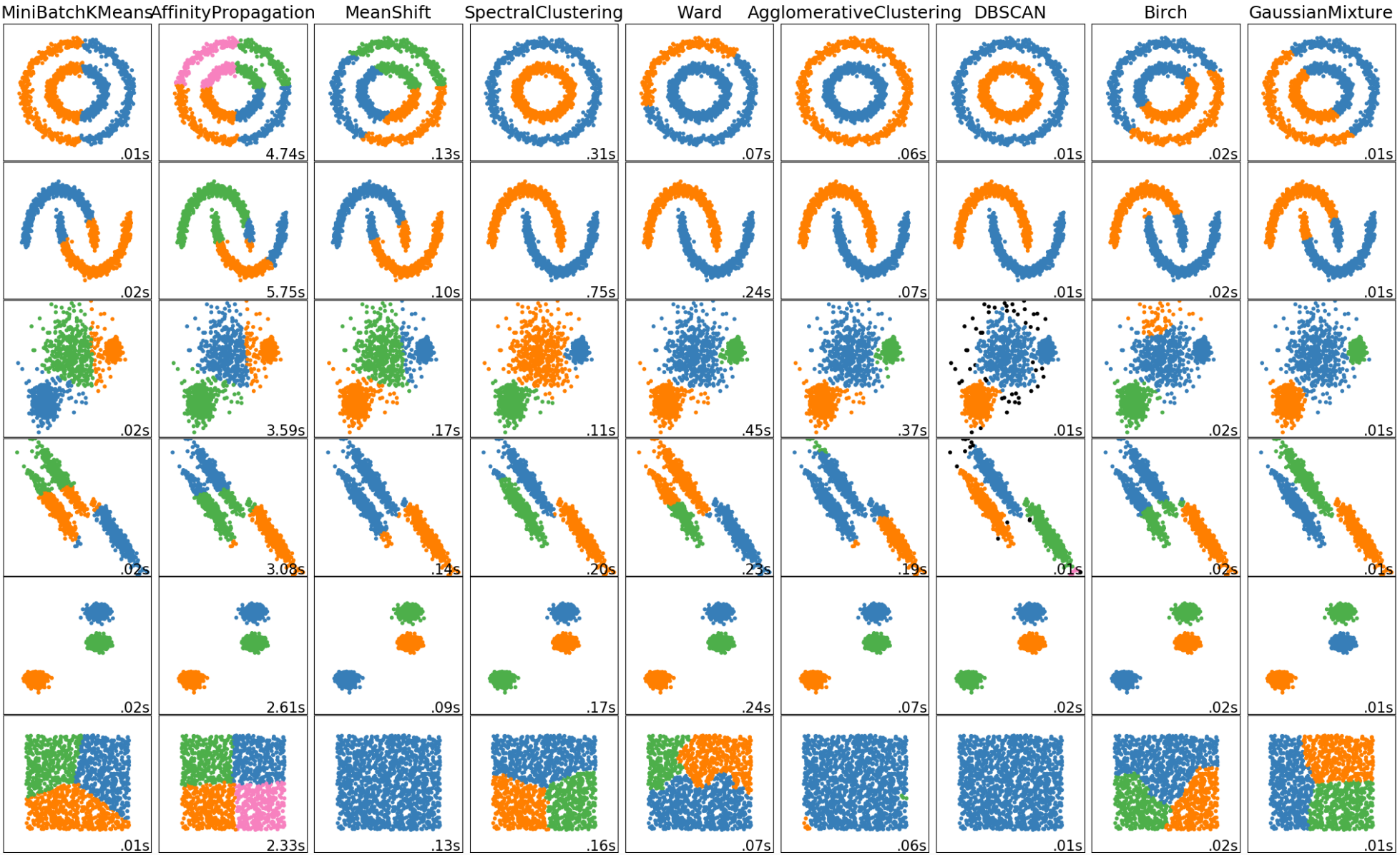
- Internal metric: does not require true labels.
- Computes **cohesion** and **separation** for each point and averages over the dataset.
- Varies between -1 and 1: higher values indicate dense, well-separated clusters.

Examples

Outline : Examples

Outline : Other Methods

Other Clustering Methods



Other Clustering Methods

Method Name	Use Case
K-Means	General-purpose, even cluster size, flat geometry, inductive
Affinity Propagation	Many clusters, uneven cluster size, non-flat geometry, inductive
Mean-Shift	Many clusters, uneven cluster size, non-flat geometry, inductive
Spectral Clustering	Few clusters, even cluster size, non-flat geometry, transductive
Ward Hierarchical Clustering	Many clusters, possibly connectivity constraints, transductive
Agglomerative Clustering	Many clusters, non-Euclidean distances, transductive
DBSCAN	Non-flat geometry, uneven cluster sizes, outlier removal, transductive
OPTICS	Non-flat geometry, uneven cluster sizes, variable density, outlier removal, transductive
Gaussian Mixtures	Flat geometry, good for density estimation, inductive
BIRCH	Large dataset, outlier removal, data reduction, inductive

More information [here]

Outline : Use-case: BERTopic

BERTopic

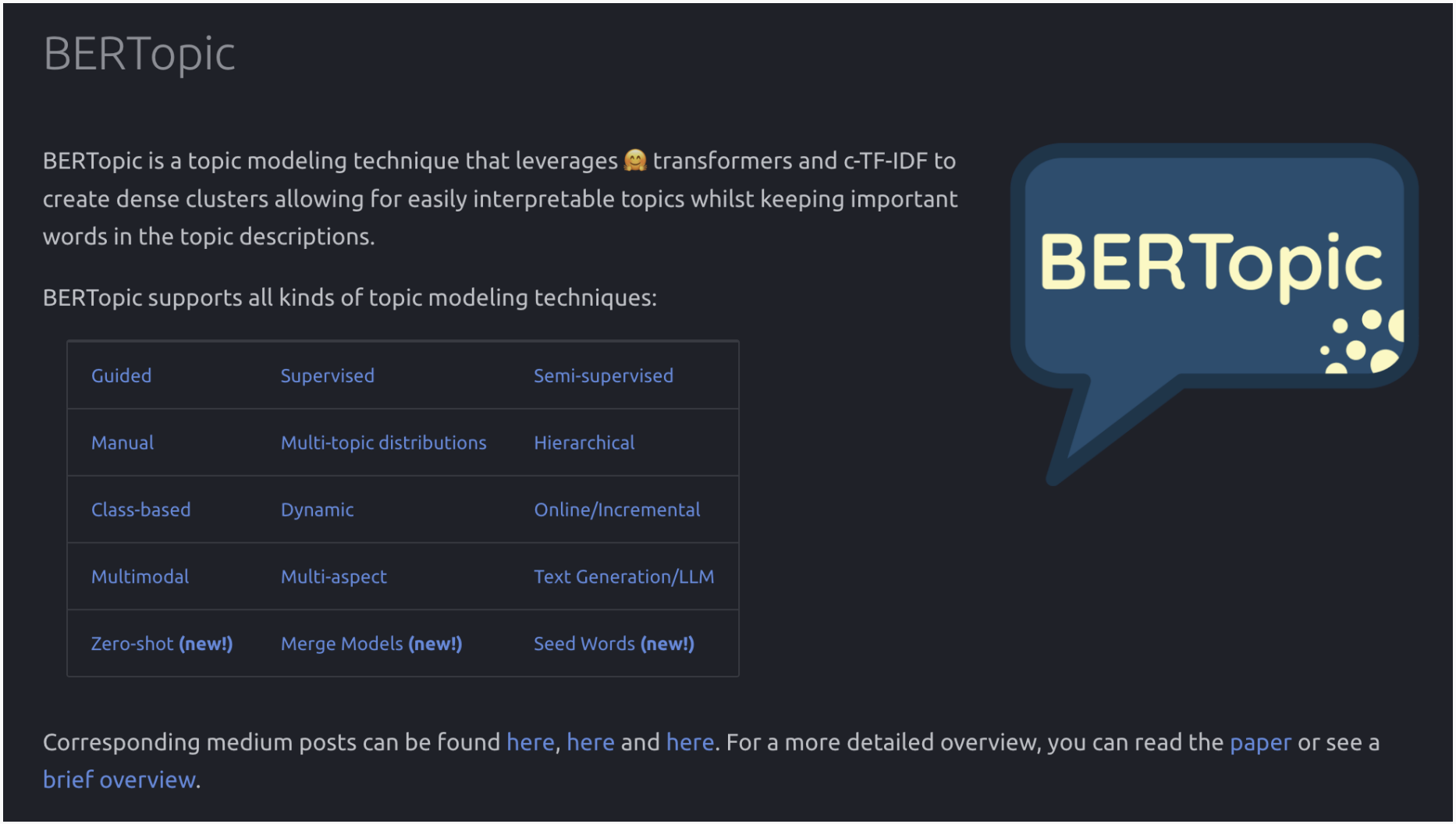
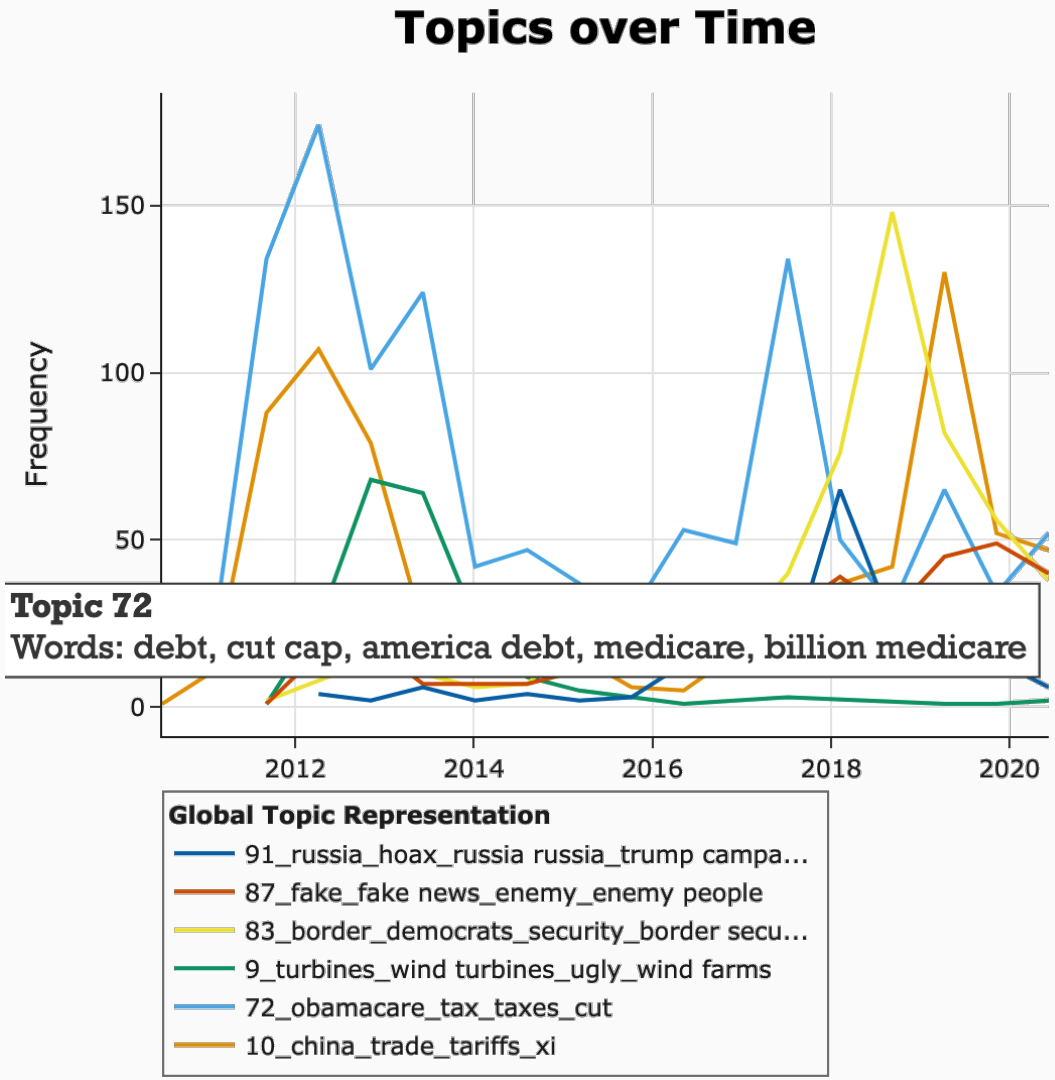


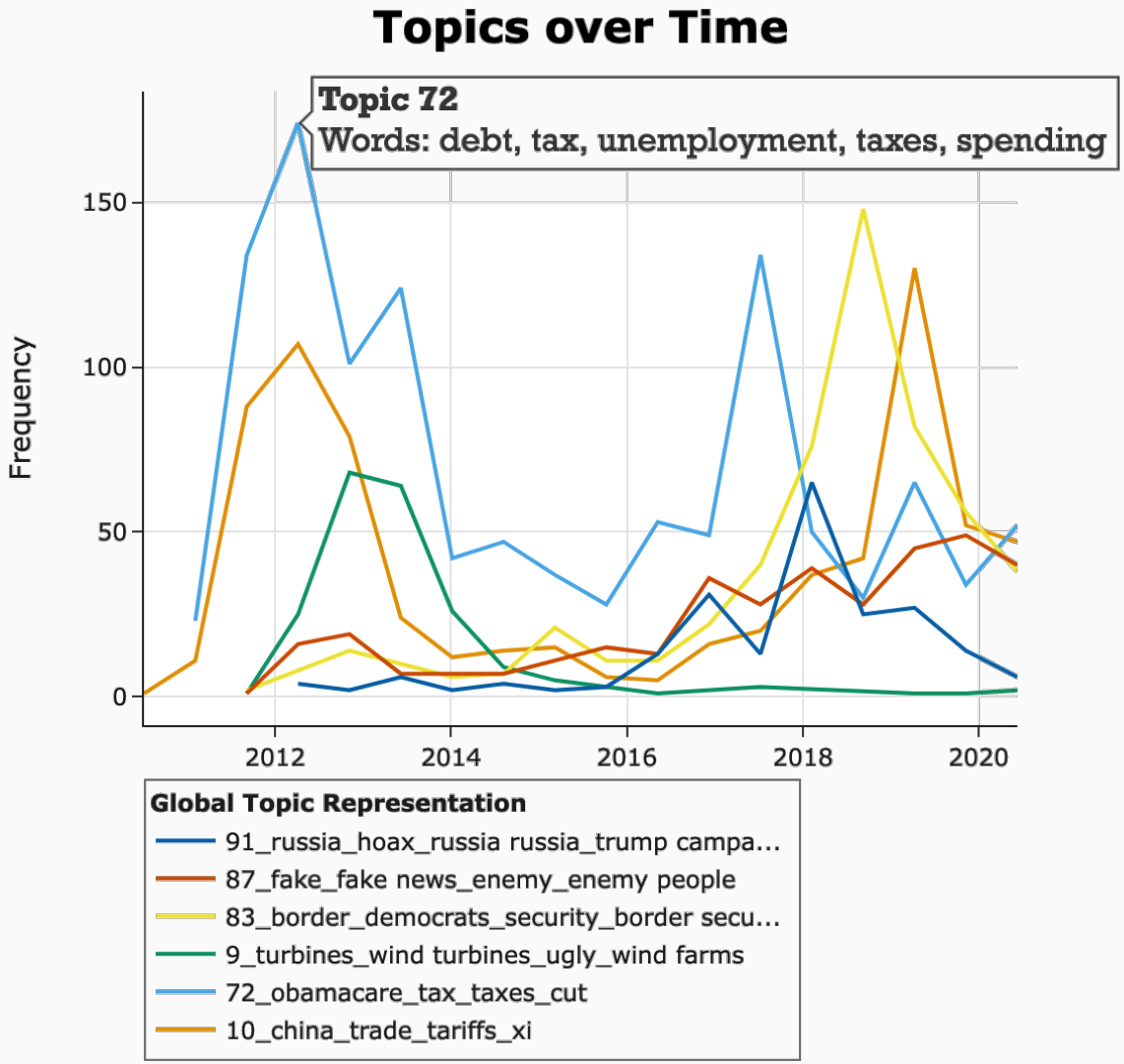
Figure 10: A simple library based on language models for topic modeling in text corpora

BERTopic



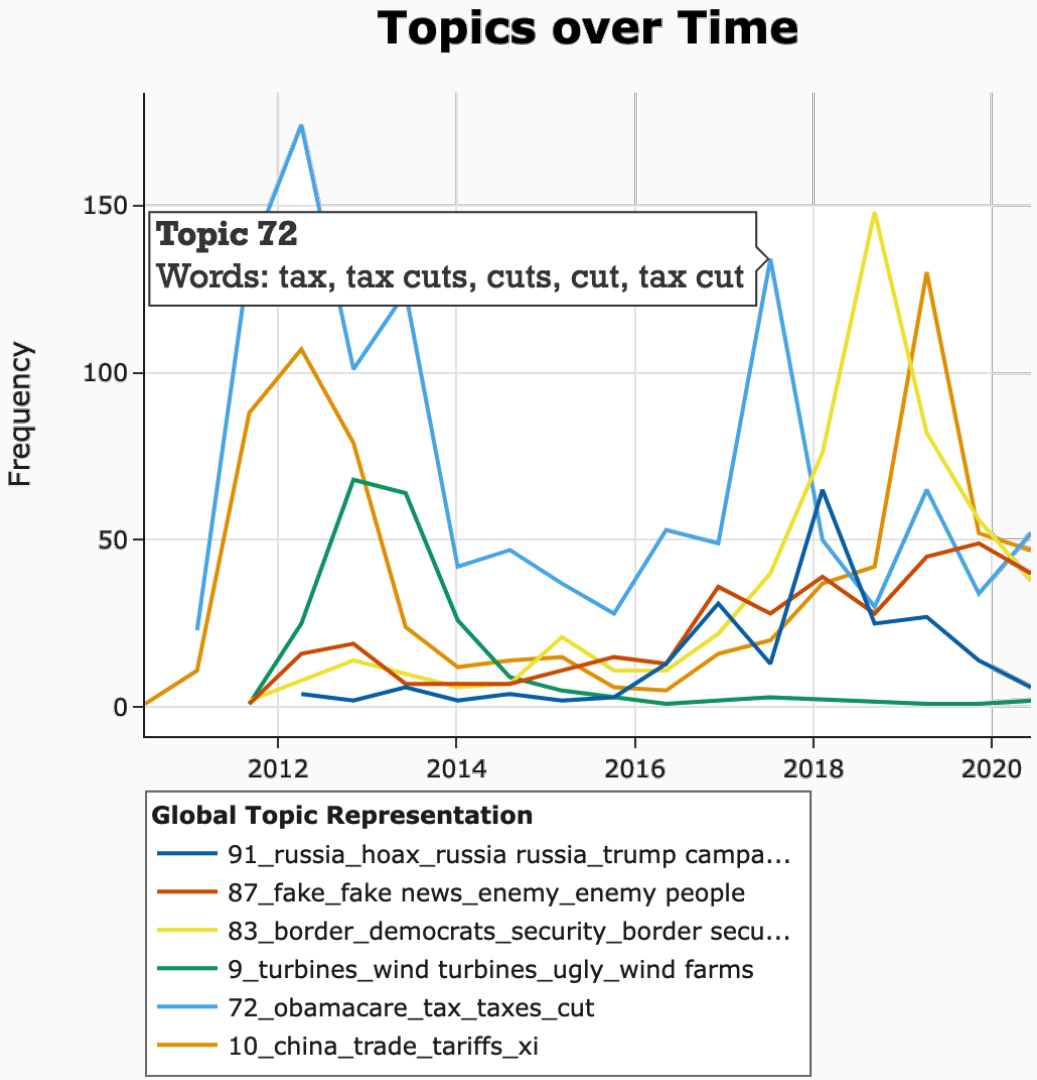
Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

BERTopic



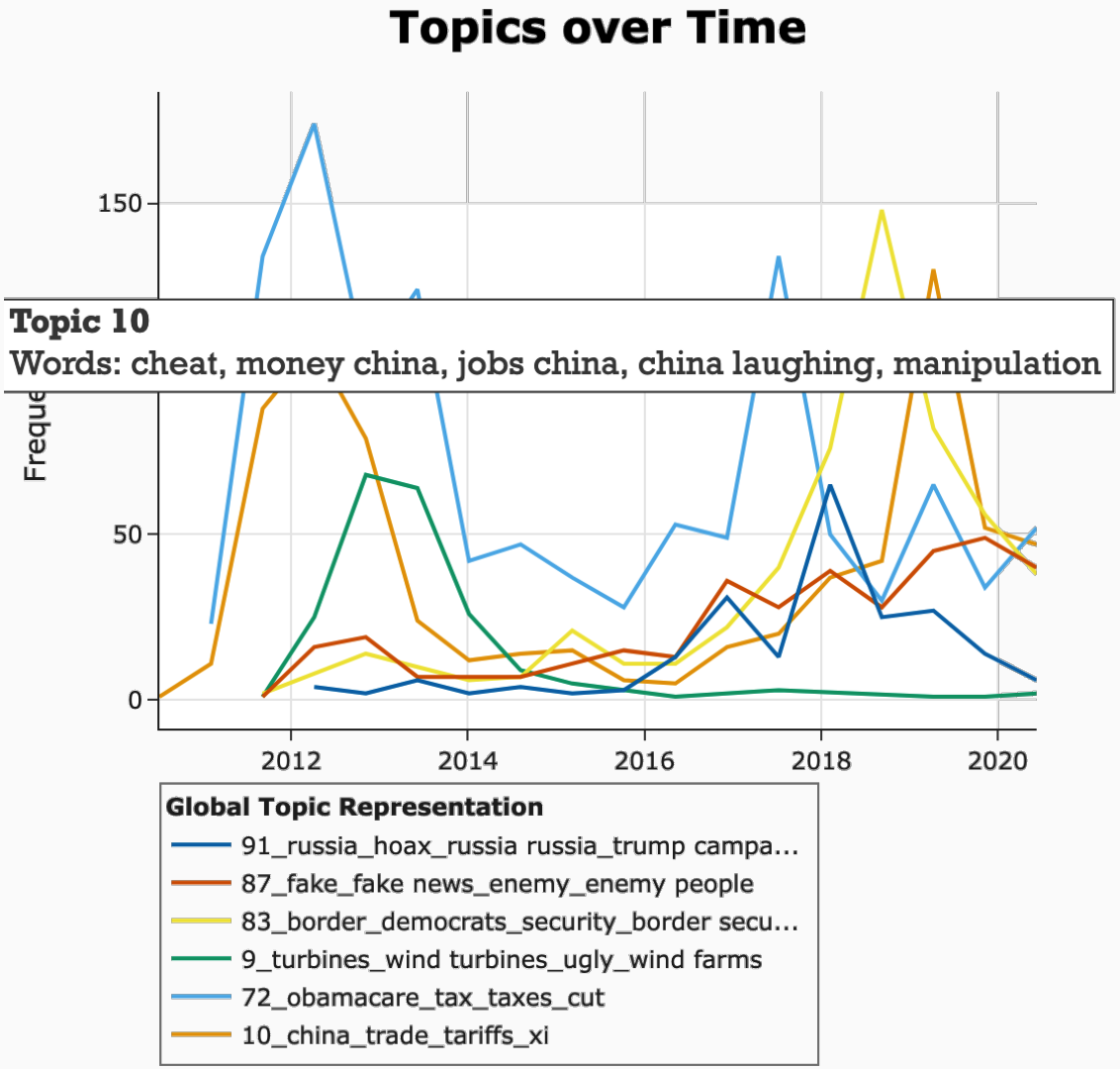
Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

BERTopic



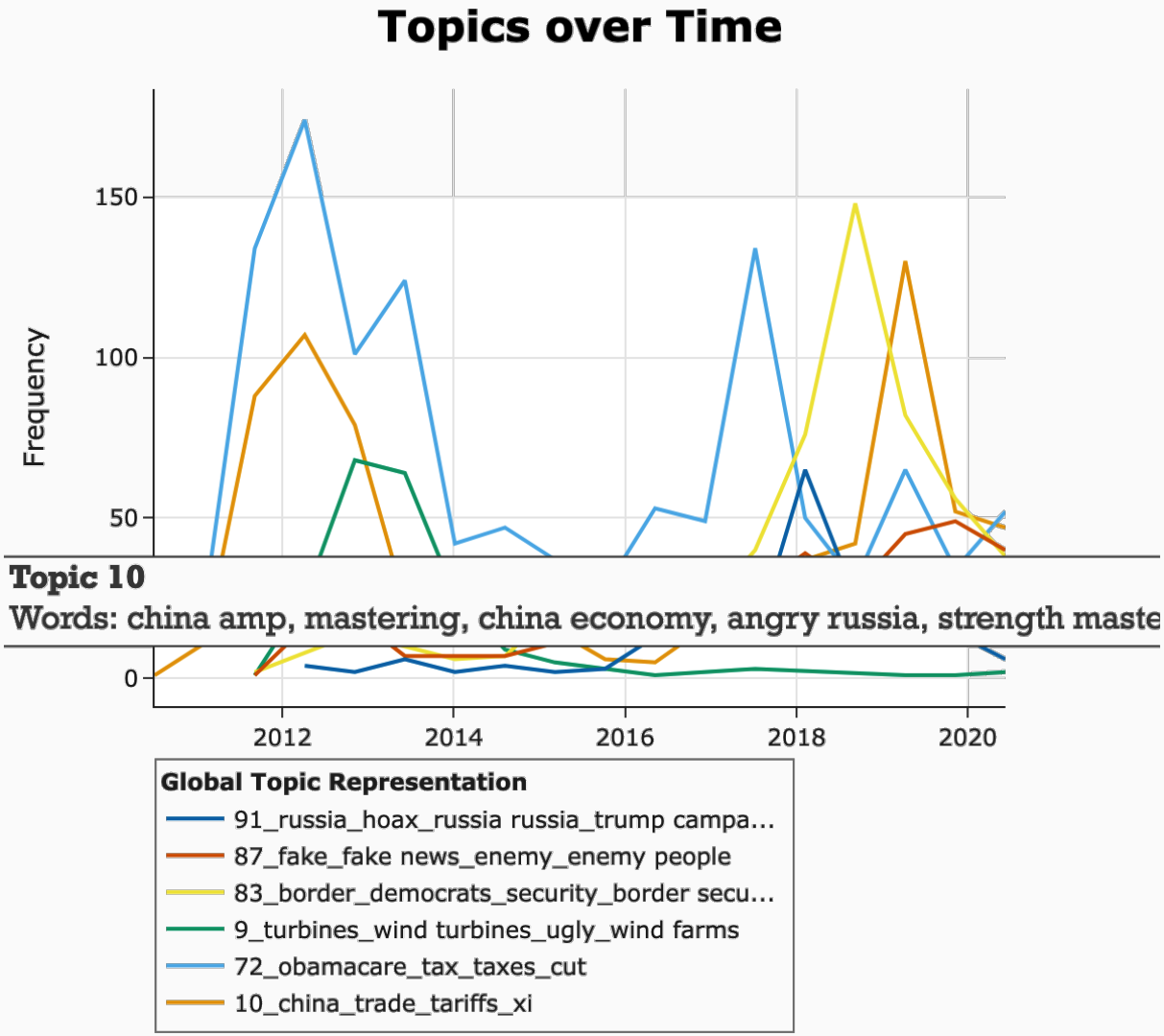
Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

BERTopic



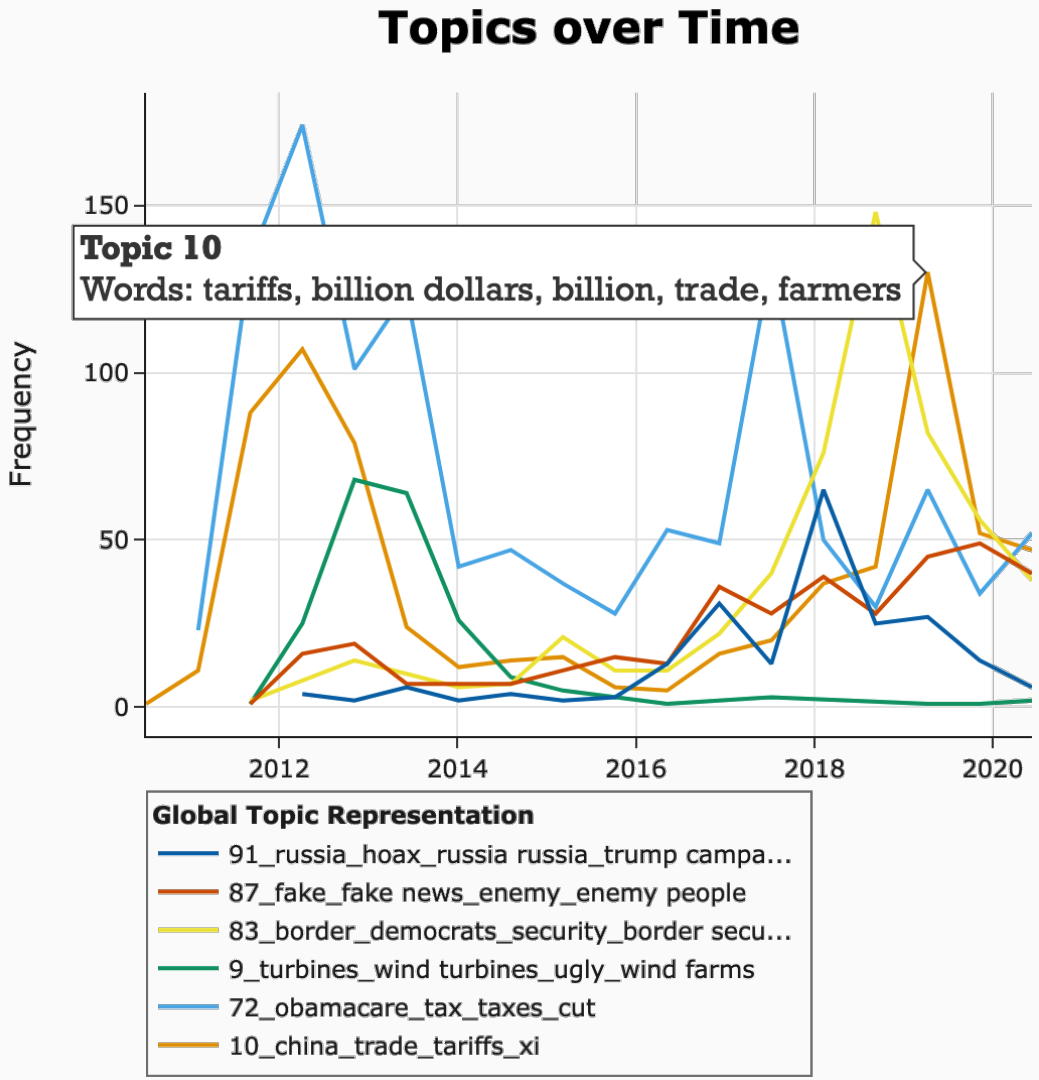
Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

BERTopic



Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

BERTopic



Can be used to cluster and easily visualize dynamic topic modeling or hierarchical clustering!

Questions?

References i