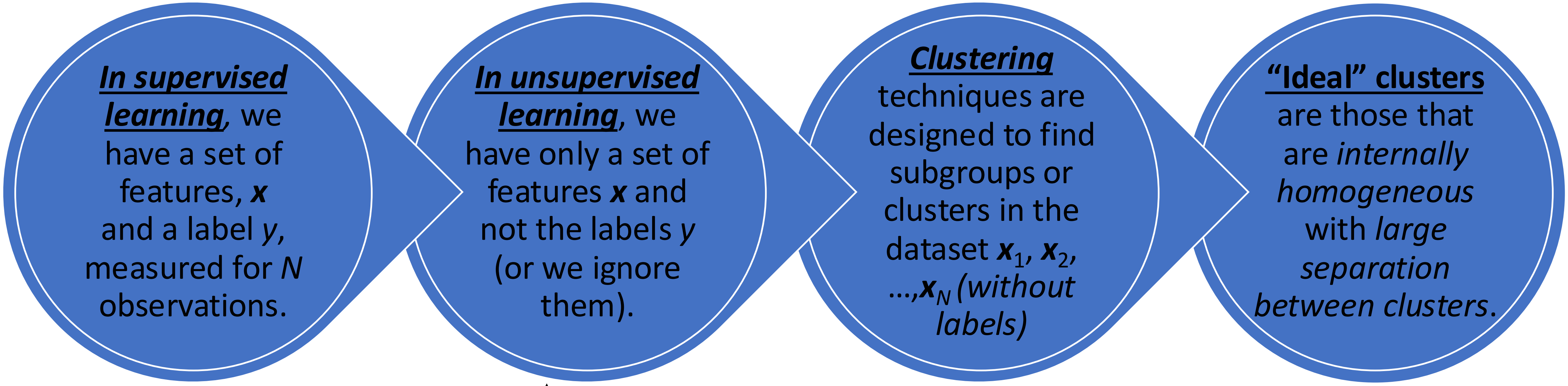
Clustering: a first foray into unsupervised learning land

Clustering as Unsupervised Learning

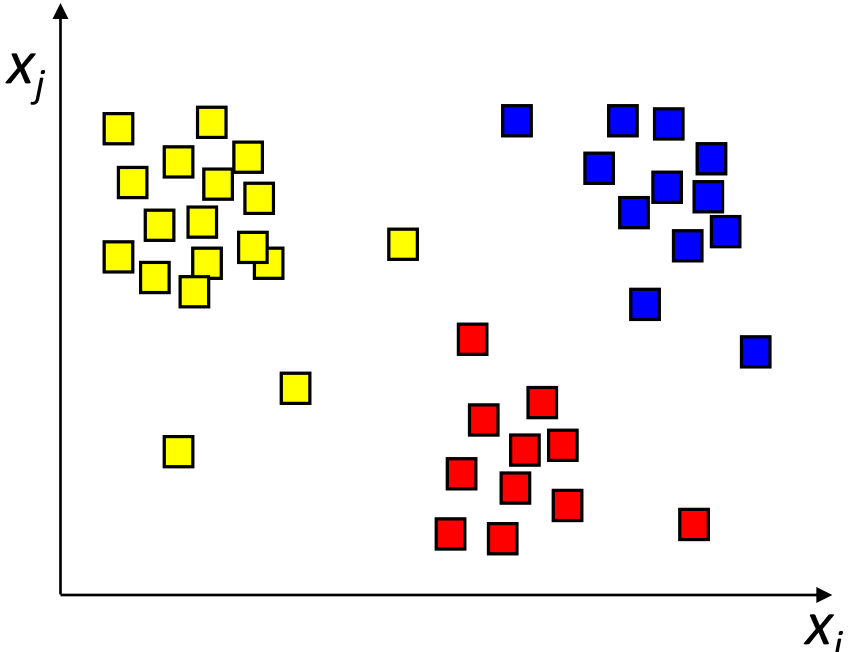


In supervised learning, we have a set of features, ***x*** and a label *y*, measured for *N* observations.

In unsupervised learning, we have only a set of features ***x*** and not the labels *y* (or we ignore them).

Clustering techniques are designed to find subgroups or clusters in the dataset $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, …,$\boldsymbol{x}_N$ *(without labels)*

"Ideal" clusters are those that are *internally homogeneous* with *large separation between clusters*.



What are common goals of clustering?

Discover Groupings: Identify clusters where data points within a group are more similar to each other than to those in other groups.

Reduce Dimensionality: Simplify complex datasets, aiding in visualization, interpretation, and data compression.

Detect Anomalies: Reveal outliers that don’t fit well within any cluster, useful in fraud detection and quality control.

Recognize Patterns: Uncover hidden patterns for applications like image segmentation, customer segmentation, and biological analysis.

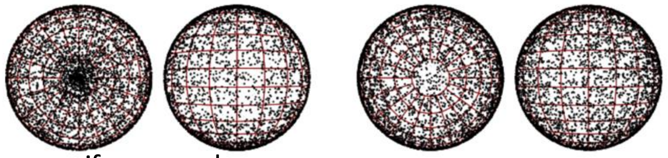
Support Feature Engineering: Highlight key characteristics and dimensions to enhance supervised learning models.

How should we approach clustering?

Before pursuing data clustering, you should carefully consider

- *What do you intend as a cluster?*
- *How do you define similarity between data points?*
- *Should you "condition"/transform the data prior to clustering?*
- *Do you need a precise number of clusters?*
- *Does the data have any clustering tendency?*

sometimes data generation can naturally and/or unintentionally lead to clustering

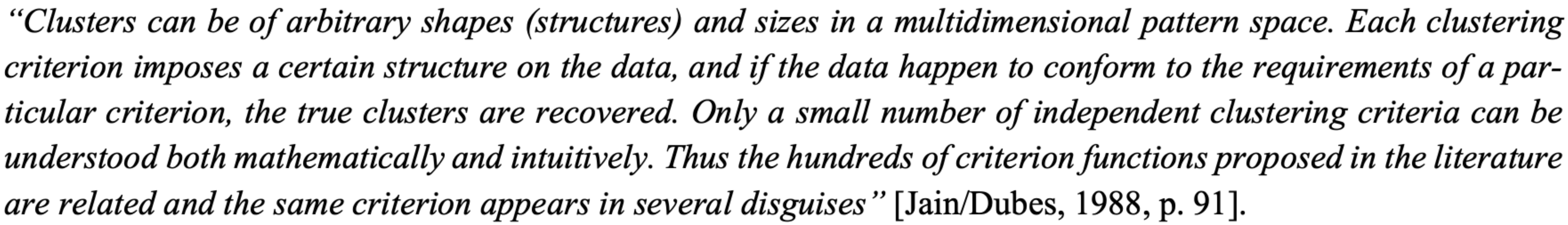


uniform random polar coords

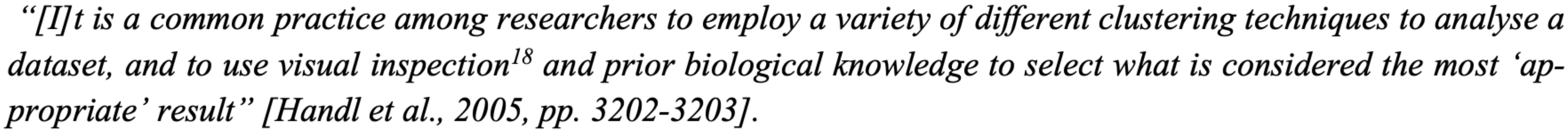
Marsaglia

Answering these questions should facilitate identification of the "best" clustering method for your task.

How should we approach clustering?



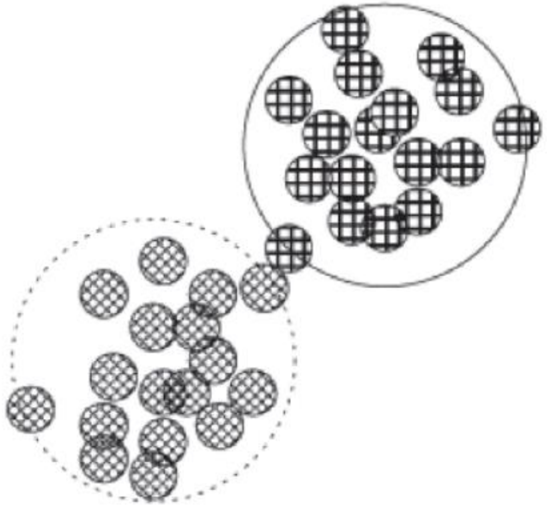
“Clusters can be of arbitrary shapes (structures) and sizes in a multidimensional pattern space. Each clustering criterion imposes a certain structure on the data, and if the data happen to conform to the requirements of a particular criterion, the true clusters are recovered. Only a small number of independent clustering criteria can be understood both mathematically and intuitively. Thus the hundreds of criterion functions proposed in the literature are related and the same criterion appears in several disguises” [Jain/Dubes, 1988, p. 91].



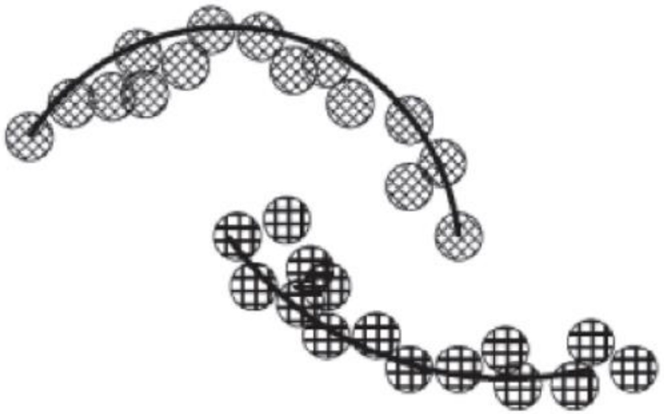
“[I]t is a common practice among researchers to employ a variety of different clustering techniques to analyse a dataset, and to use visual inspection[18] and prior biological knowledge to select what is considered the most ‘appropriate’ result” [Handl et al., 2005, pp. 3202-3203].

Terms for defining clusters

Compact Clusters
→ *within-cluster distance < between-cluster distance*



Connected Clusters
→ *determined based on neighborhoods of points*
→ high between-cluster connectivity



Different algorithms may tend towards different cluster structures

Defining Similarity

To perform clustering, we must have a measure of similarity or distance between data points

e.g.,

Inner products (in general)

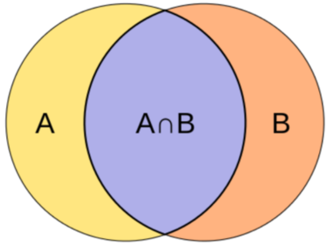
Minkowski distance(s)

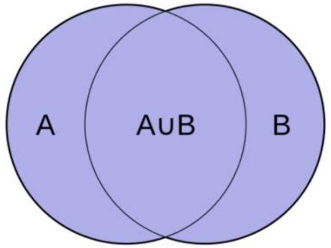
$$d(\boldsymbol{x}_1, \boldsymbol{x}_2) = \left(\sum_{i=1}^{m} |\boldsymbol{x}_1[i] - \boldsymbol{x}_2[i]|^r \right)^{1/r} \quad ; \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^m$$

r = 1 → "Manhattan"
r = 2 → "Euclidean"
r → ∞ → "Chebyshev"

Jaccard Similarity/ Tanimoto Similarity

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$





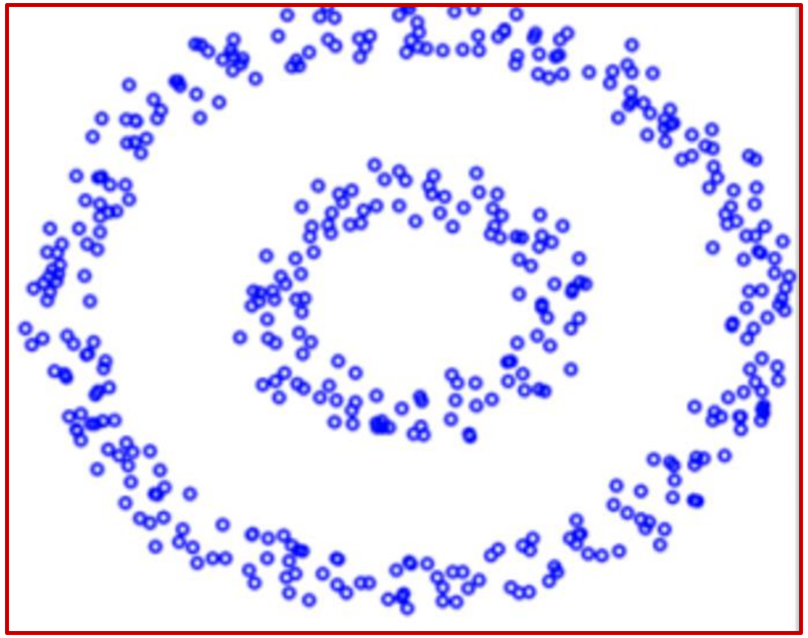
application to a ***vector***

$$J(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{\boldsymbol{x}_1 \cdot \boldsymbol{x}_2}{||\boldsymbol{x}_1||^2 + ||\boldsymbol{x}_2||^2 - \boldsymbol{x}_1 \cdot \boldsymbol{x}_2}$$

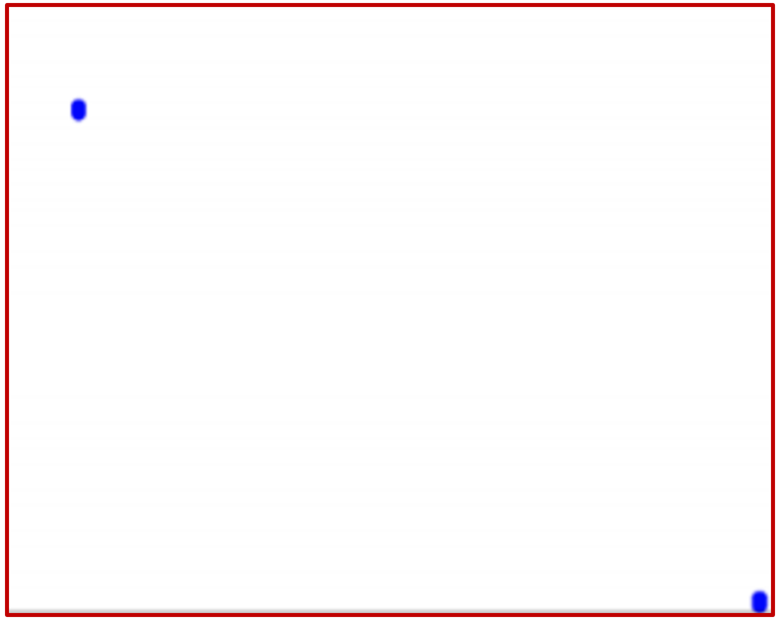
- **Jaccard Similarity**: Widely used for comparing text documents, binary attributes, or any sets where it's useful to know the proportion of shared elements, such as in recommendation systems or image segmentation.
- **Tanimoto Similarity**: Commonly applied in cheminformatics to compare molecular structures, where the similarity between molecular fingerprints can suggest similar chemical properties or biological activities.

Data Conditioning

Sometimes data may be artificially difficult to cluster; data conditioning is essentially a preprocessing/transformation step



transform via some kernel/mapping



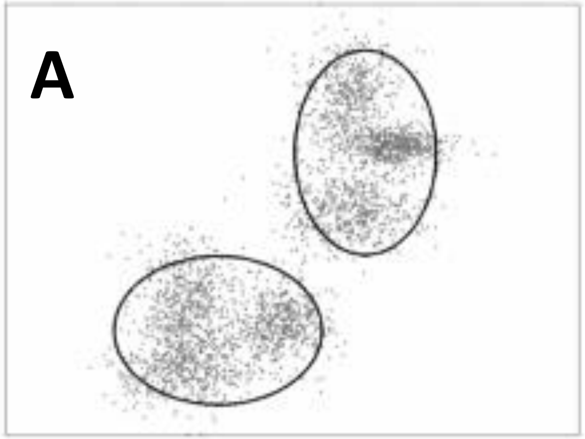
possibly difficult to cluster

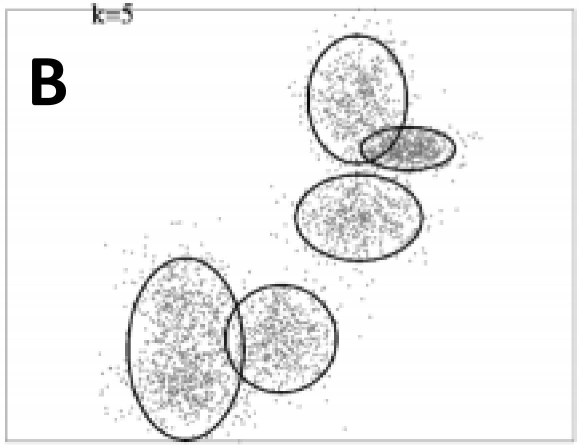
trivial to cluster

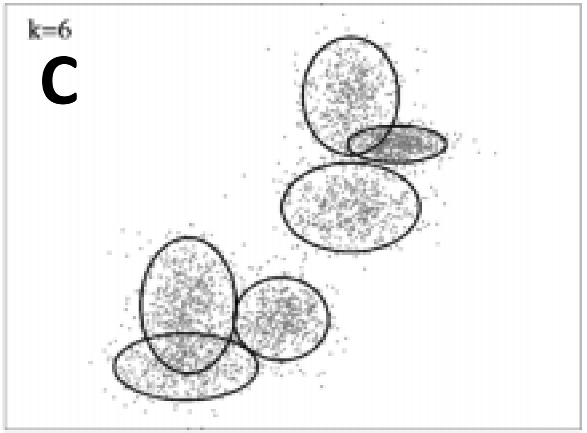
may be possible to use other unsupervised learning techniques as preprocessing steps prior to clustering

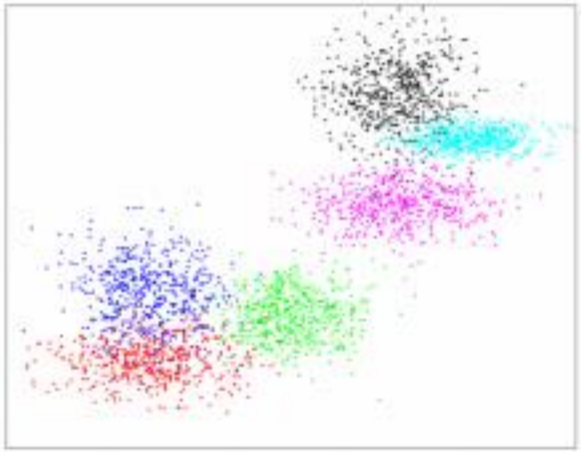
How many clusters?

Many clustering methods use the *number of clusters* as a *hyperparameter*, although this many not be clear *a priori*





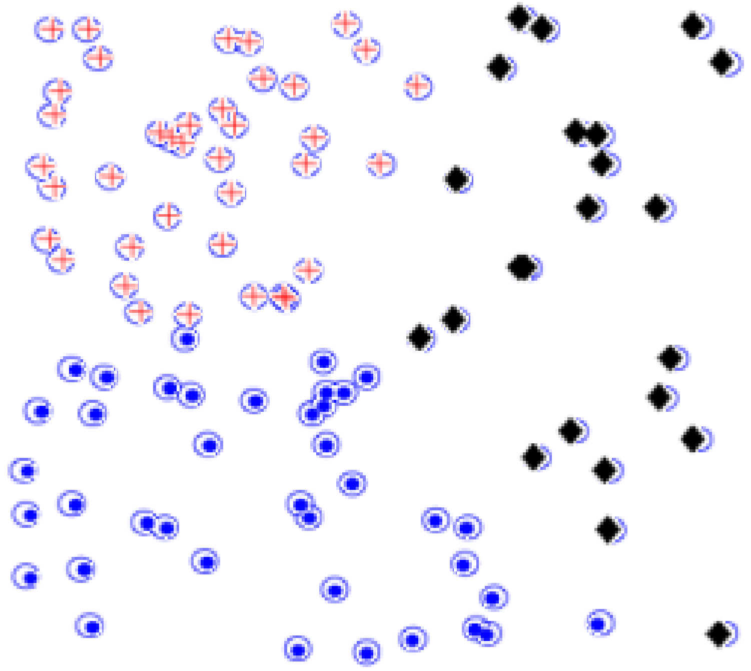




clusterin’ algorithms gon’ cluster

Like many algorithms, clustering schemes will simply do your bidding, for better or for worse

100 random points drawn from uniform 2D distribution



***k*-Means with *k* = 3**

clustering/unsupervised ML may detect patterns where there are none

Silhouette Score for Evaluating Clusters

The **silhouette score** is a metric used to evaluate the *quality of clusters* created by a clustering algorithm. It quantifies how well each data point fits within its assigned cluster compared to other clusters. This score provides insight into both the **cohesion** (*how similar data points in the same cluster are to each other*) and the **separation** (*how distinct each cluster is from others*).

For a given point *i*:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

Avg. distance from point i to points in the same cluster

Avg. distance from point i to points in the nearest neighboring cluster

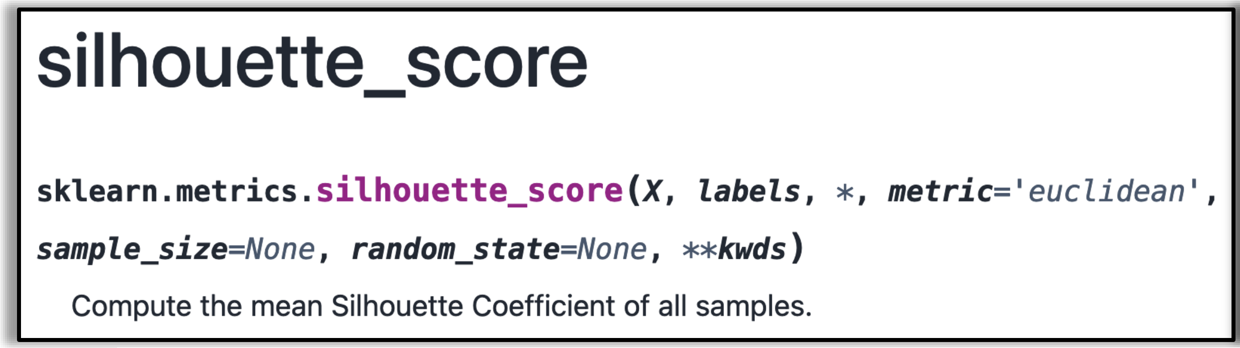
The average silhouette score of all points provides an overall measure of clustering quality:

Positive average SS (towards +1): indicates well-defined clusters, with distinct and tight groupings.

Low average SS (close to 0): suggests overlapping clusters or a boundary-like configuration.

Negative average SS (towards -1): suggests overlapping clusters or a boundary-like configuration.

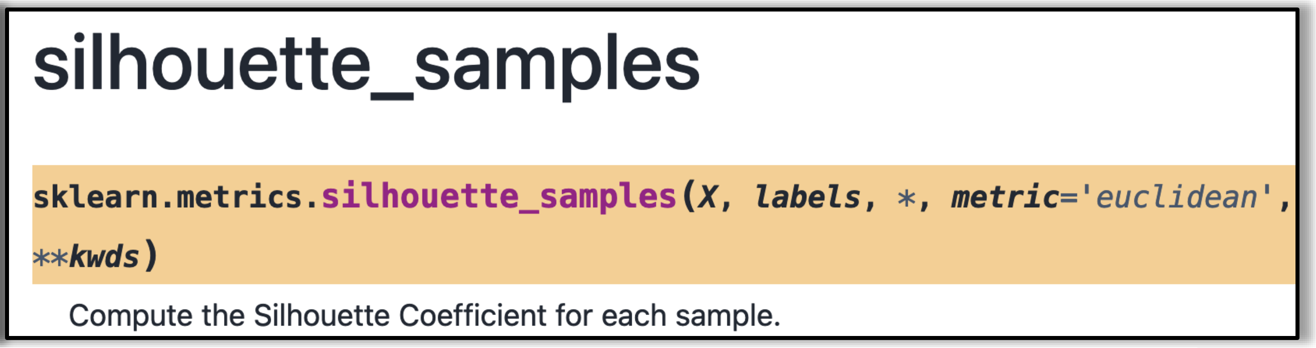
Silhouette Score in Scikit-learn



silhouette_score

```
sklearn.metrics.silhouette_score(X, labels, *, metric='euclidean', sample_size=None, random_state=None, **kwds)
```

Compute the mean Silhouette Coefficient of all samples.

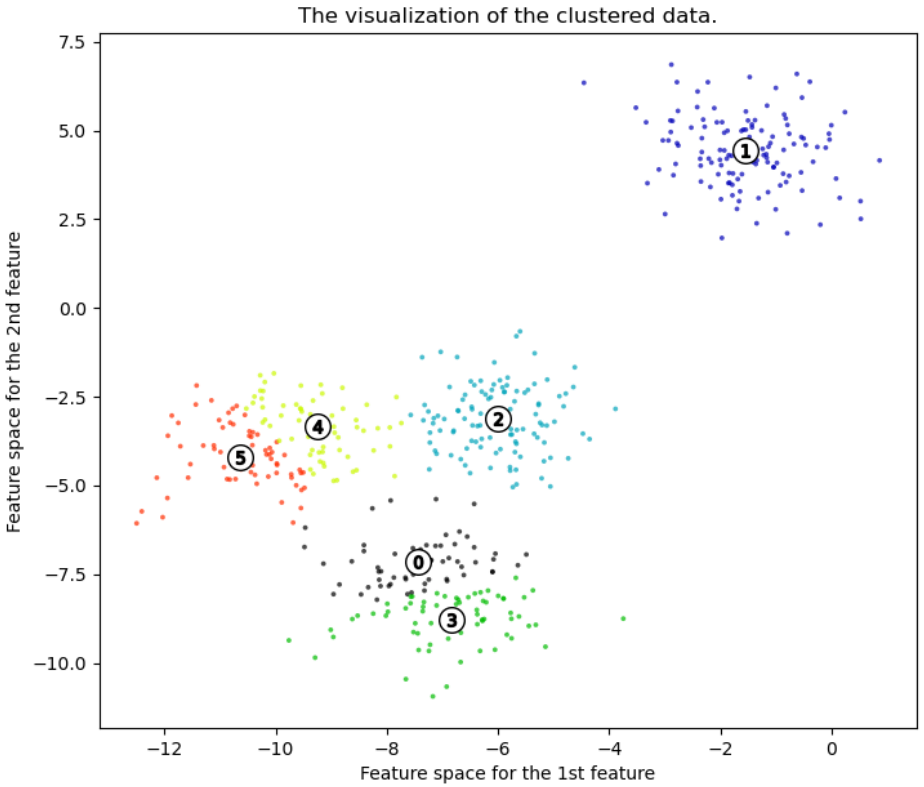


silhouette_samples

```
sklearn.metrics.silhouette_samples(X, labels, *, metric='euclidean', **kwds)
```

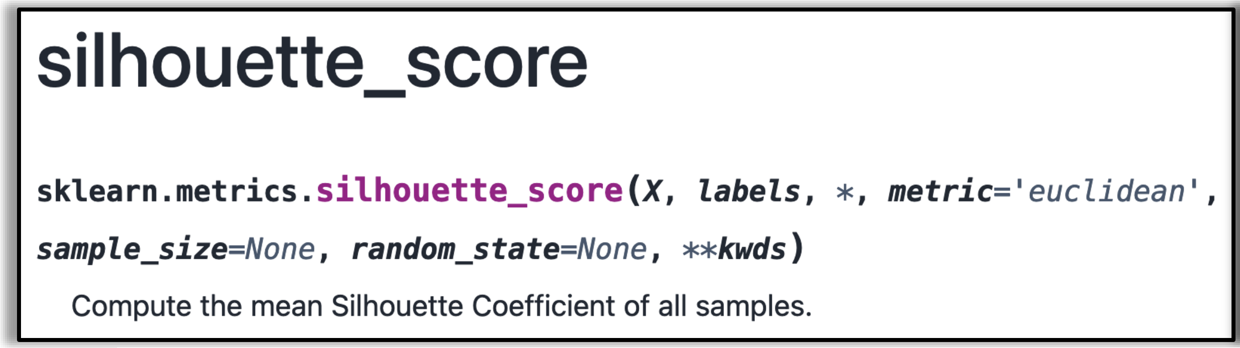
Compute the Silhouette Coefficient for each sample.

Silhouette analysis for KMeans clustering on sample data with n_clusters = 6



The visualization of the clustered data.

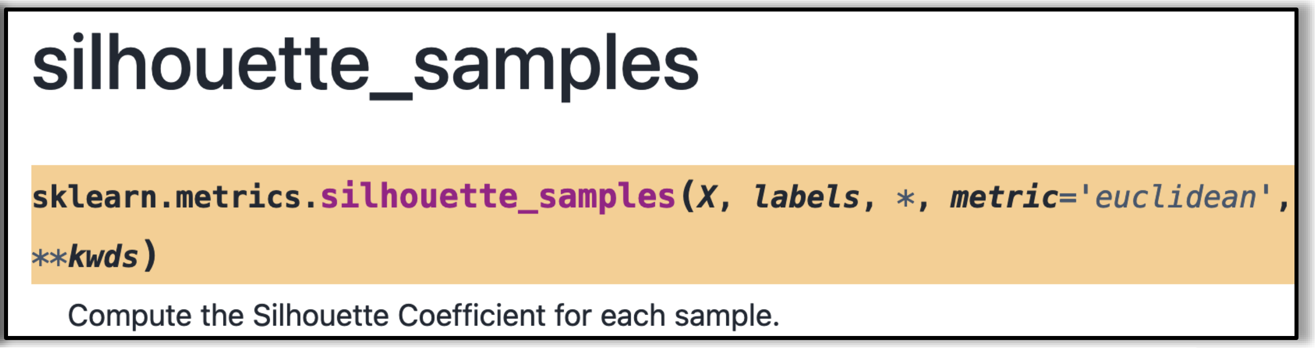
Silhouette Score in Scikit-learn



silhouette_score

```
sklearn.metrics.silhouette_score(X, labels, *, metric='euclidean', sample_size=None, random_state=None, **kwds)
```

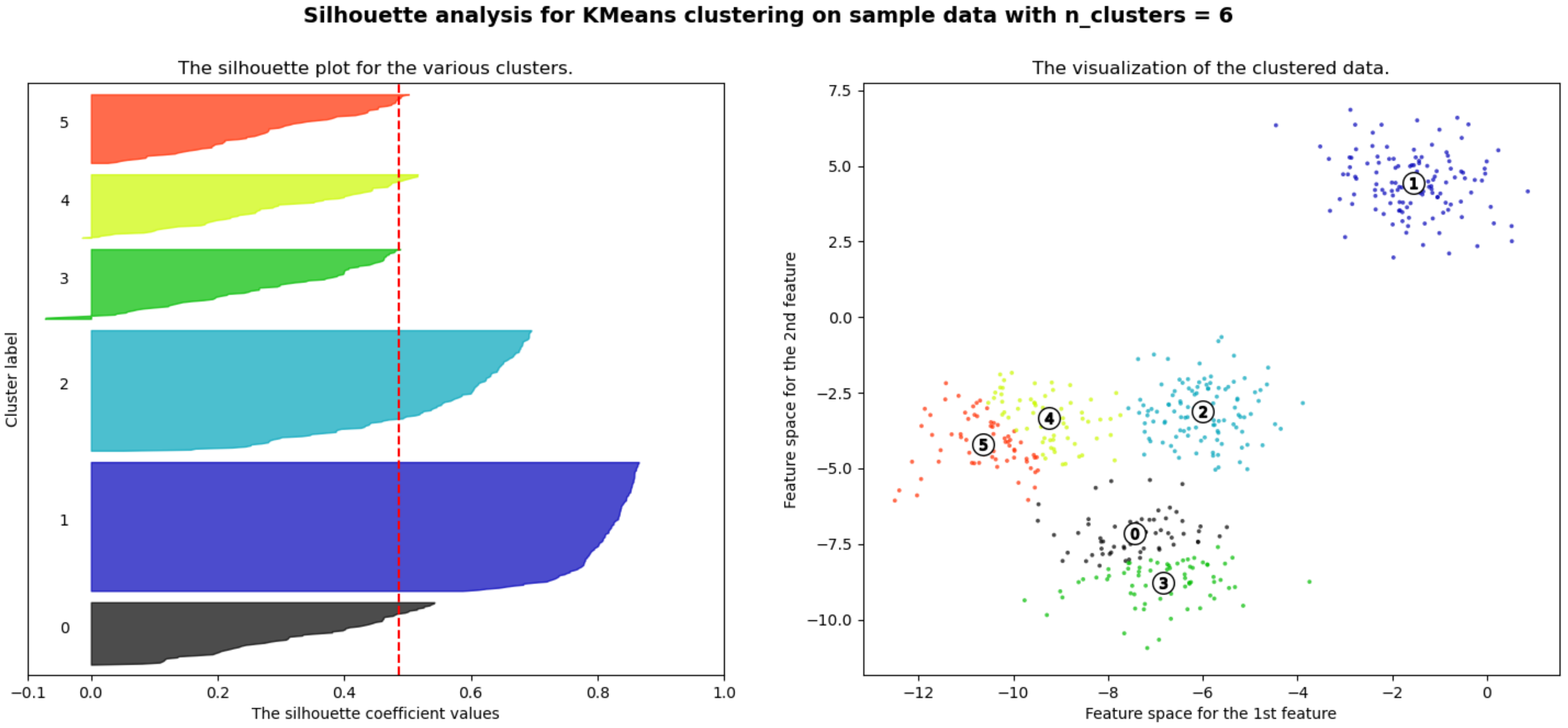
Compute the mean Silhouette Coefficient of all samples.



silhouette_samples

```
sklearn.metrics.silhouette_samples(X, labels, *, metric='euclidean', **kwds)
```

Compute the Silhouette Coefficient for each sample.



Silhouette analysis for KMeans clustering on sample data with n_clusters = 6

The silhouette plot for the various clusters.

The visualization of the clustered data.

Categorization of Clustering Models

Partition Methods

Given a fixed number of clusters, *k*, assign each observation of $x_1, \ldots, x_m$ to a unique cluster in a way that minimizes within-cluster variation and maximizes between cluster variation.

Methods: K-means, k-medians, k-modes

Distribution Models

Tests how probable it is that all data points in the cluster belong to the same distribution (e.g. Normal, Poisson)

Methods: Gaussian Mixture, DBCLASD

Hierarchical Methods

Construct a hierarchy for the observations in terms of similarities. This results in a dendrogram which depicts the clustering of data at different levels ranging from one to *K* groups.

Methods: Ward, DIANA, AGNES, hclust

Density Models

Search the data space for areas of varied density of data points. It isolates various different density regions and assigns data points within these regions in the same cluster.

Methods: DBSCAN, HDBSCAN, OPTICS

Categorization of Clustering Models

Partition Methods

Advantages:

- Simple implementation
- Fast
- Applicable to large datasets
- Easy interpretation of results

Disadvantages:

- Must specify number of clusters *a priori*.
- Created clusters are of inconsistent sizes and densities
- Affected by noise and outliers

Distribution Models

Advantages:

- No need to specify number of clusters
- Metrics are easy to understand and tune

Disadvantages:

- Complex algorithm and slow
- Poor scaling to large data
- No guarantee data belongs to “nice” distribution.

Hierarchical Methods

Advantages:

- Simple implementation
- No need to specify number of clusters
- Easy interpretation of dendrograms

Disadvantages:

- High time complexity
- Doesn’t work for larger datasets
- Cluster assignment is strict

Density Models

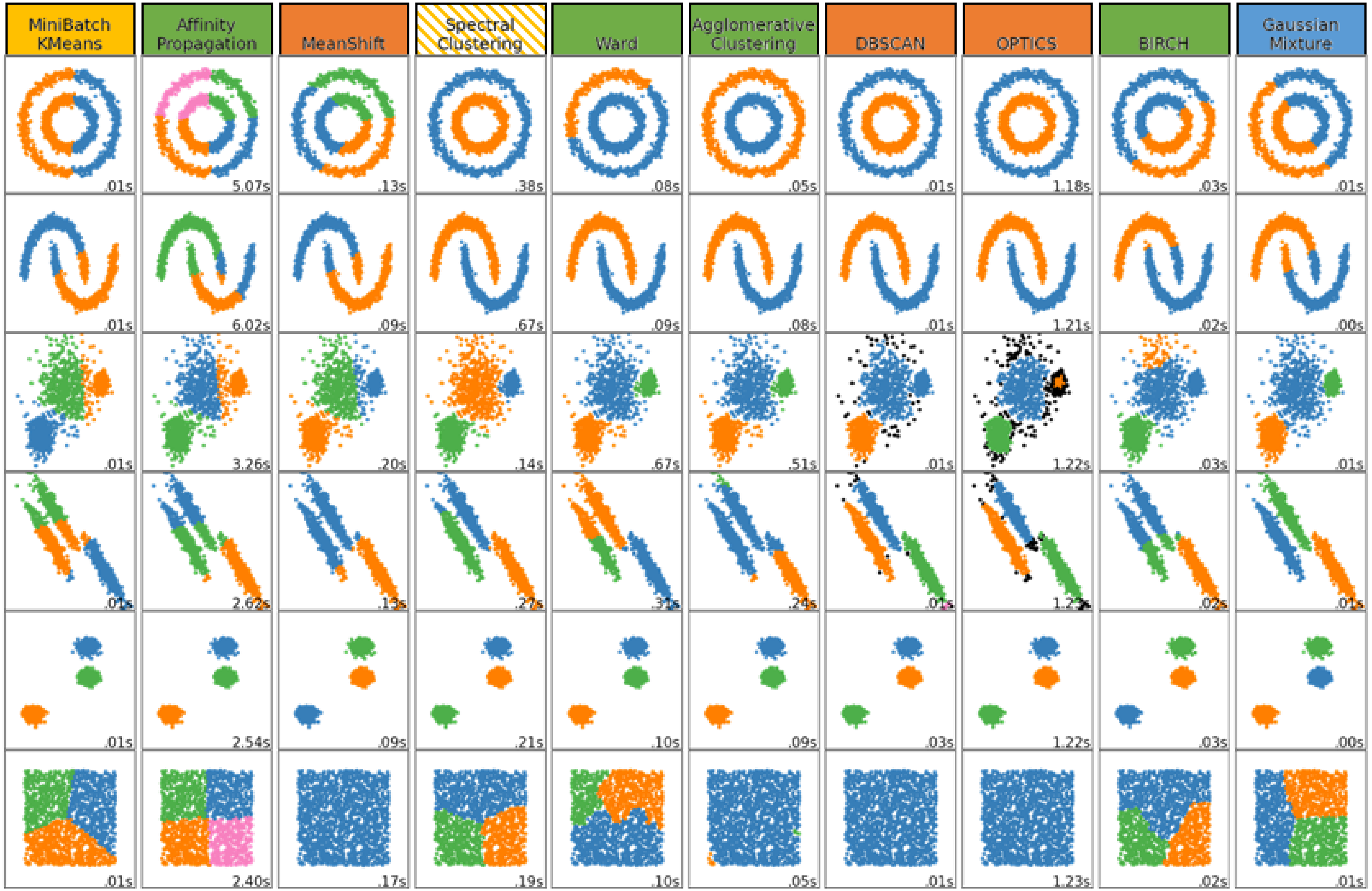
Advantages:

- Can handle noise and outliers
- No need to specify number of clusters
- Created clusters are homogeneous
- No restriction on cluster shapes

Disadvantages:

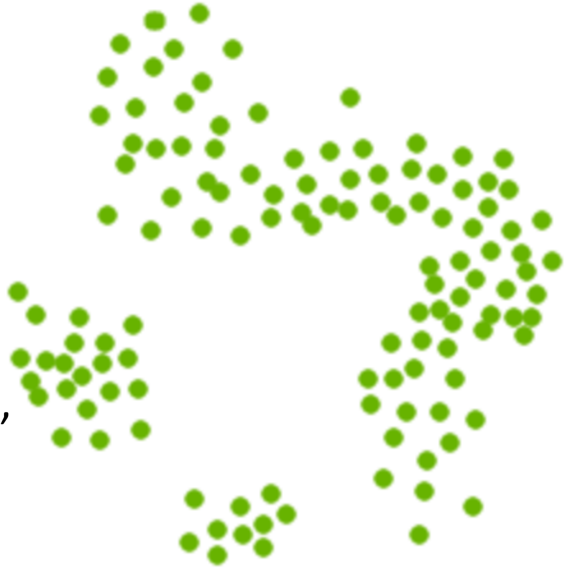
- Complex algorithm and slow
- Poor scaling to large data

There is not necessarily a “best” algorithm



https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

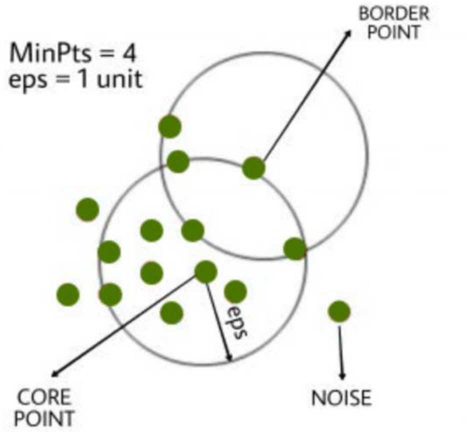


Premise:

- clusters are sample-dense regions in the feature-vector space
- clusters are distinguished from each other based on the presence of sample-lean (low-density) regions within the feature-vector space

Hyperparameters:

- **epsilon** – this defines a distance threshold to decide whether two points are "neighbors"
- ***N*$_{\text{min}}$ –** this sets the minimum number of points required to be considered a core point



"density connected" – two points are density-connected if they are both in the neighborhood of a set of points that are neighbors of each other

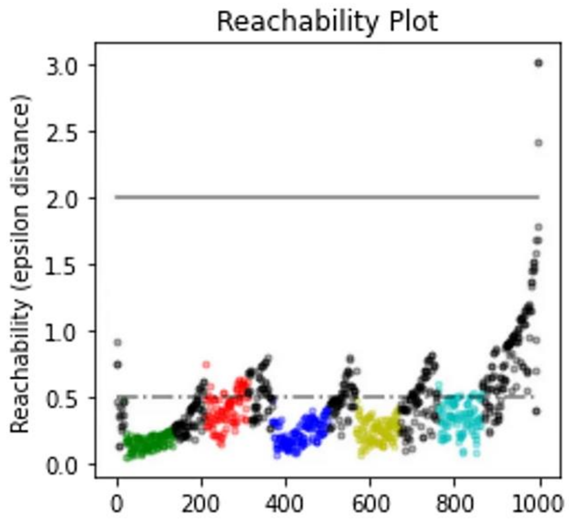
Basic algorithm:

1. For all points, find corresponding neighbors based on **epsilon**
2. For all core points, check if it is assigned to a cluster.
3. If core point is not in a cluster, create a cluster based on that point.
4. For newly created cluster, recursively identify all the density-connected points and assign them to the cluster.
5. Continue through all unvisited points in the dataset.
6. Points not assigned to clusters are noise.

Ordering Points to Identify cluster Structure (OPTICS)

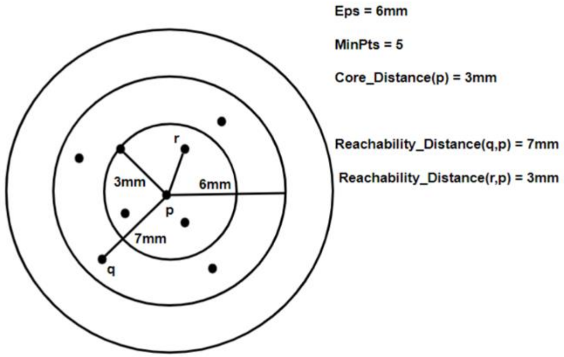
Premise:

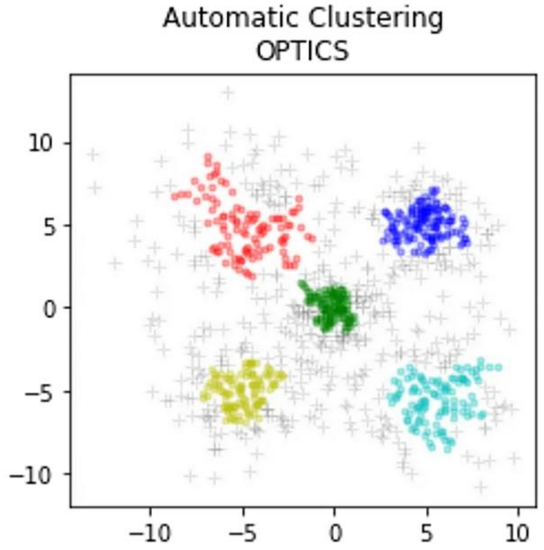
- similar in spirit to DBSCAN with some additional flexibility
- aims to reveal structure and assign clusters via the concept of **reachability**, which relates to how easily a given point could be visited from other points in the dataset
- this allows for clusters of differing density



Hyperparameters:

- **epsilon** – this defines a minimum distance around a point to be a core point
- ***N*$_{\text{min}}$ –** this sets the minimum number of points required to consider for core points


Eps = 6mm
MinPts = 5
Core_Distance(p) = 3mm
Reachability_Distance(q,p) = 7mm
Reachability_Distance(r,p) = 3mm
3mm
6mm
7mm
r
p
q



Basic idea of algorithm:

1. For each point, calculate the distance to its nearest N_{min} neighbors
2. Calculate the reachability distance based on the density of neighbors.
3. Order points according to their reachability distance.
4. Cluster points based on similarity of reachability distances.