Aglomerative Clustering and Clustering Evaluation

David Mareček

December 10, 2024

EUROPEAN UNION
European Structural and Investment Fund
Operational Programme Research,
Development and Education

Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

Some of the slides in this presentation were taken from the Alexandra Chouldechova's presentations (Carnegie Mellon University)

Hierarchical Clustering

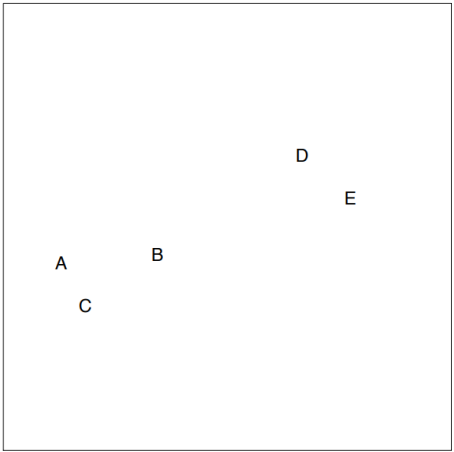
Hierarchical clustering

- **K-means** and **GMM** are objective-based approaches that require us to pre-specify the number of clusters K.
- The answer they give is somewhat random. It depends on the random initialization it started with.

- **Hierarchical clustering** is an alternative approach that does not require a pre-specified choice of K, and which provides a deterministic answer (no randomness).

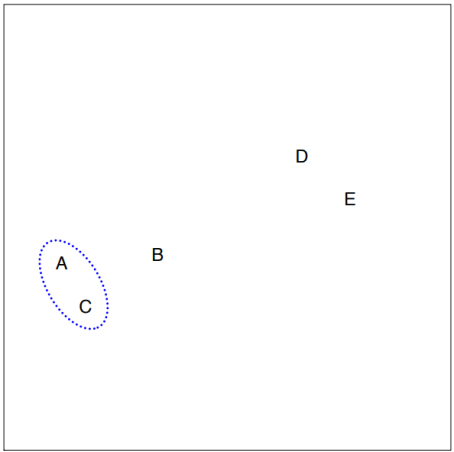
- We'll focus on **bottom-up** or **agglomerative** hierarchical clustering
- **top-down** or **divisive** clustering is also good to know about, but we won't directly cover it here

Hierarchical clustering – Example



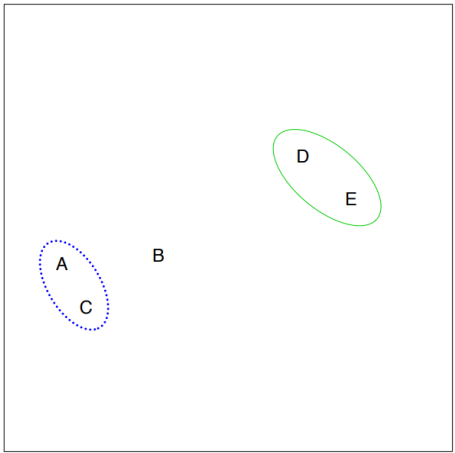
Each point starts as its own cluster

Hierarchical clustering – Example



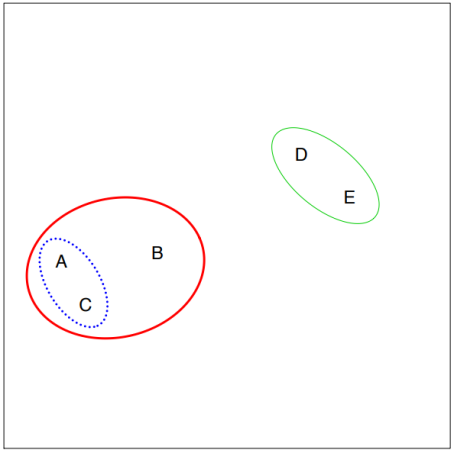
We merge the two clusters (points) that are closest to each other.

Hierarchical clustering – Example



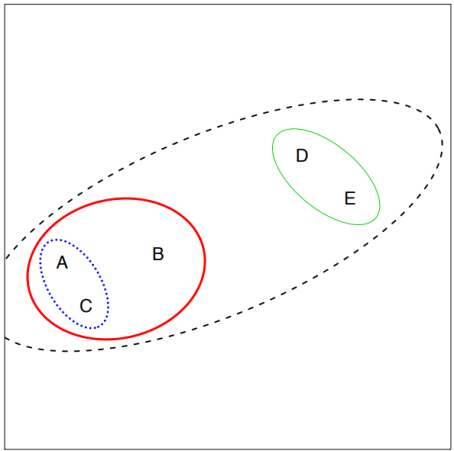
Then we merge the next two closest clusters.

Hierarchical clustering – Example



Then the next two closest clusters...

Hierarchical clustering – Example

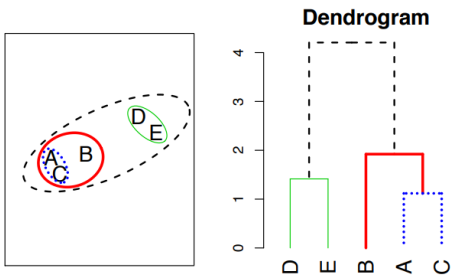


Until at last all of the points are all in a single cluster.

Aglomerative Hierarchical clustering

- Start with each point in its own cluster.
- Identify the two closest clusters and merge them.
- Repeat until all points are in a single cluster.

To visualize the results, we can look at the resulting **dendrogram**.



y-axis on dendrogram is (proportional to) the distance between the clusters that got merged at that step.

Hierarchical clustering - Linkages

- Let $d_{ij} = d(x_i, x_j)$ denote the **dissimilarity** (distance) between points x_i and x_j.
- At our first step, each cluster is a single point, so we start by merging the two points that have the lowest dissimilarity.
- But after that, we need to think about distances not between points, but between sets of points (clusters).
- The dissimilarity between two clusters is called the **linkage**.
- i.e., given two sets of points, G and H, a linkage is a dissimilarity measure $d(G, H)$ telling us how different the points in these sets are.

Common linkage types

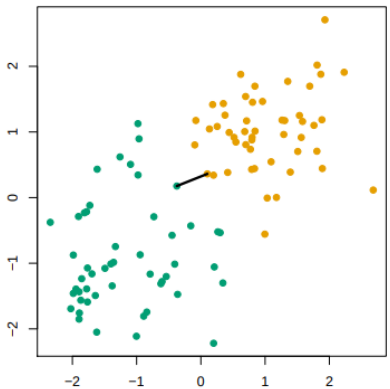
- **Single** – Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster G and the observations in cluster H, and record the **smallest** of these dissimilarities.
- **Complete** – Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster G and the observations in cluster H, and record the **largest** of these dissimilarities.
- **Average** – Mean inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster G and the observations in cluster H, and record the **average** of these dissimilarities.
- **Centroid** – Dissimilarity between the centroid for cluster G (a mean vector of length p) and the centroid for cluster H. Centroid linkage can result in undesirable **inversions**.
- **Ward** – Minimizes the variance, similar to **k-means** objective.

Single linkage

In **single linkage** (i.e., nearest-neighbor linkage), the dissimilarity between G, H is the smallest dissimilarity between two points in different groups:

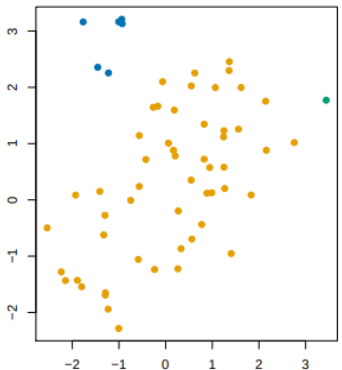
$$d_{single}(G, H) = \min\{d(x_i, x_j),\ i \in G, j \in H\}$$

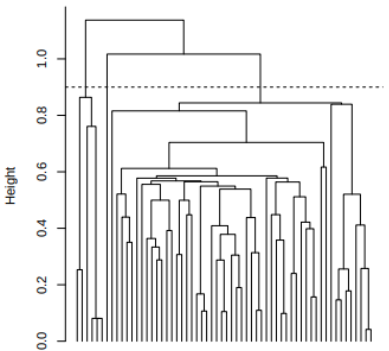
Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{single}(G, H)$ is the distance of the closest pair.



Single linkage – Example

Here $n = 60$, $x_i \in \mathbb{R}^2, d_{ij} = \| x_i - x_j \|_2$. Cutting the tree at $h = 0.9$ gives the clustering assignments marked by colors.





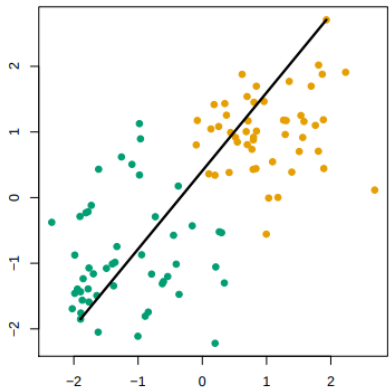
Cut interpretation: for each point x_i, there is another point x_j in its cluster such that $d(x_i, x_j) \leq 0.9$.

Complete linkage

In **complete linkage** (i.e., furthest-neighbor linkage), dissimilarity between G, H is the largest dissimilarity between two points in different groups:

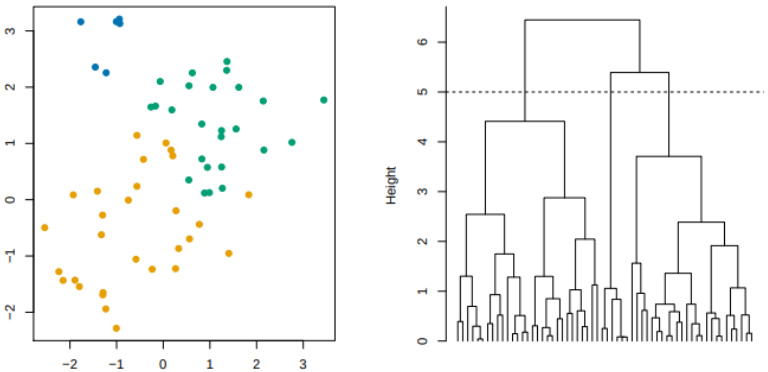
$$d_{complete}(G, H) = \max\{d(x_i, x_j),\ i \in G, j \in G\}$$

Example (dissimilarities d_{ij} are distances, groups are marked by colors): complete linkage score $d_{complete}(G, H)$ is the distance of the furthest pair.



Complete linkage – Example

Same data as before. Cutting the tree at $h = 5$ gives the clustering assignments marked by colors.



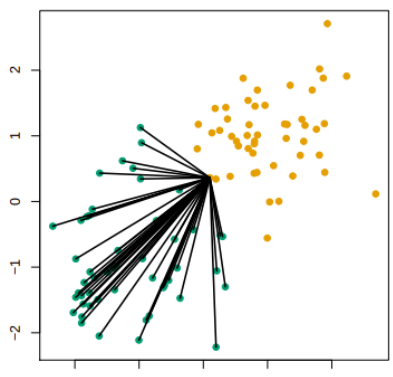
Cut interpretation: for each point x_i, every other point x_j in its cluster satisfies $d(x_i, x_j) \leq 5$.

Average linkage

In **average linkage**, the dissimilarity between G, H is the average dissimilarity over all points in opposite groups:

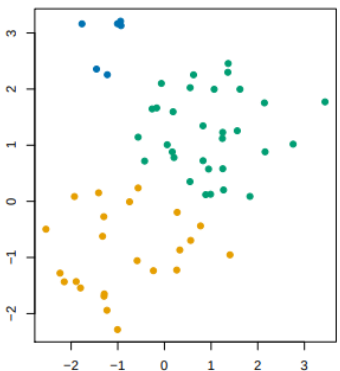
$$d_{average}(G, H) = \frac{1}{|G| \cdot |H|} \sum_{i \in G} \sum_{j \in H} d(x_i, x_j)$$

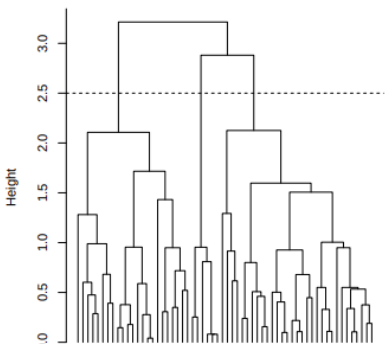
Example (dissimilarities $d_i j$ are distances, groups are marked by colors): average linkage score $d_{average(G,H)}$ is the average distance across all pairs (Plot here only shows distances between the green points and one orange point).



Average linkage – Example

Same data as before. Cutting the tree at $h = 2.5$ gives the clustering assignments marked by colors.





Cut interpretation: there really is not a good one! :(

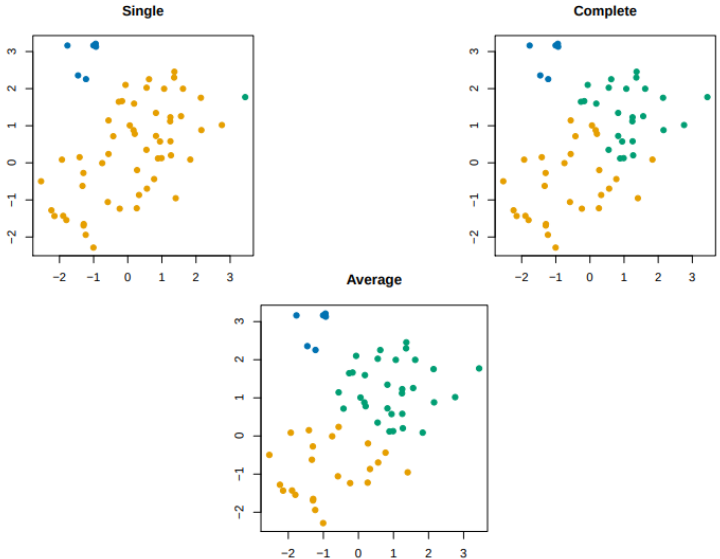
Shortcomings of Single and Complete linkage

Single and complete linkage have some practical problems:

- Single linkage suffers from **chaining**.
  - In order to merge two groups, only need one pair of points to be close, irrespective of all others. Therefore *clusters can be too spread out*, and not compact enough.
- Complete linkage avoids chaining, but suffers from **crowding**.
  - Because its score is based on the worst-case dissimilarity between pairs, *a point can be closer to points in other clusters than to points in its own cluster*. Clusters are compact, but not far enough apart.

Average linkage tries to **strike a balance**. It uses average pairwise dissimilarity, so clusters tend to be relatively compact and relatively far apart

Example of chaining and crowding

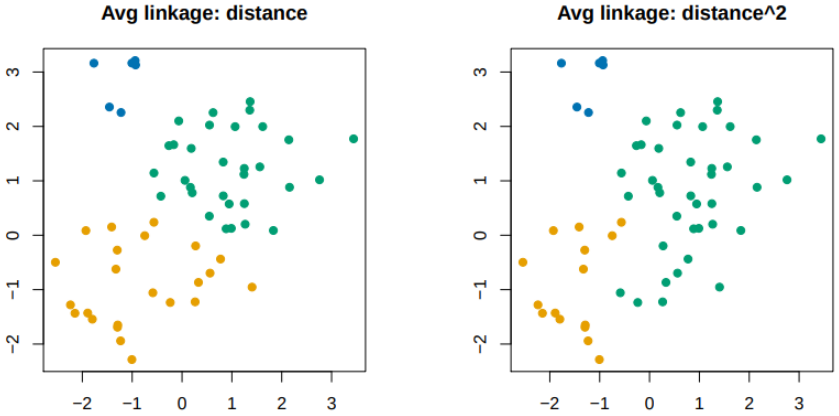


Shortcomings of the Average linkage

Average linkage has its own problems:

- Unlike single and complete linkage, average linkage doesn't give us a nice interpretation when we cut the dendrogram.
- Results of average linkage clustering **can change** if we simply apply a monotone increasing transformation to our dissimilarity measure, our results can change
  - e.g. $d \leftarrow d^2$ or $d \leftarrow \frac{e^d}{1+e^d}$.
  - This can be a big problem if we're not sure precisely what dissimilarity measure we want to use.
  - Single and Complete linkage **do not have this problem**.

Average linkage – monotone dissimilarity transformation



The left panel uses $d(x_i, x_j) = ||x_i - x_j||_2$ (Euclidean distance), while the right panel uses $||x_i - x_j||_2^2$. The left and right panels would be same as one another if we used single or complete linkage. For average linkage, we see that the results can be different.

Centroid linkage

Dissimilarity between the centroid for cluster G and the centroid for cluster H.

$$d_{centroid} = d\left(\frac{1}{|G|}\sum_{i\in G} x_i, \frac{1}{|H|}\sum_{i\in H} x_i\right)$$

Centroid linkage can result in undesirable **inversions**.

- The pair of merged clusters may have a lower distance than a previously merged pair.
- Example?

Centroid linkage

Dissimilarity between the centroid for cluster G and the centroid for cluster H.

$$d_{centroid} = d\left(\frac{1}{|G|}\sum_{i\in G} x_i, \frac{1}{|H|}\sum_{i\in H} x_i\right)$$

Centroid linkage can result in undesirable **inversions**.

- Consider three points forming almost an equilateral triangle.
- What will be the distances between clusters?

Ward linkage

Ward linkage is a variance minimizing approach. The distance between two clusters G and H is how much the sum of squares will increase when we merge them. It is similar to the k-means objective function but tackled with an agglomerative hierarchical approach.

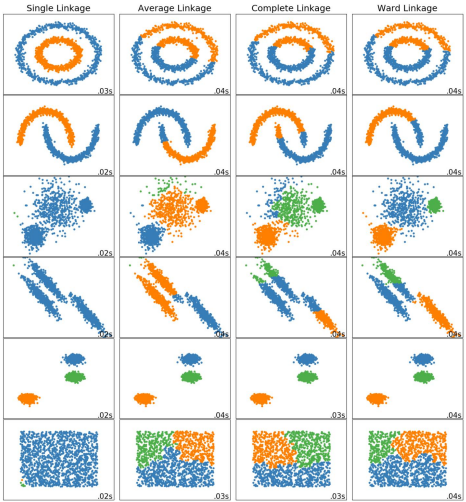
$$d_{Ward}(G,H) = \sum_{i \in G \cup H} ||x_i - m_{G \cup H}||^2 - \sum_{i \in G} ||x_i - m_G||^2 - \sum_{i \in H} ||x_i - m_H||^2,$$

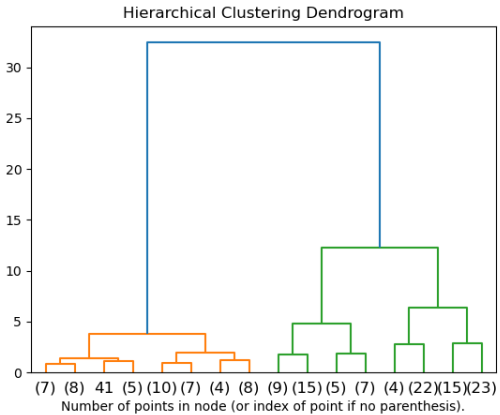
where m_X is the mean (center) of cluster X. It also corresponds to the squared distance between the centers of the clusters

$$d_{Ward}(G,H) = \frac{n_G n_H}{n_G + n_H} ||m_G - m_H||^2,$$

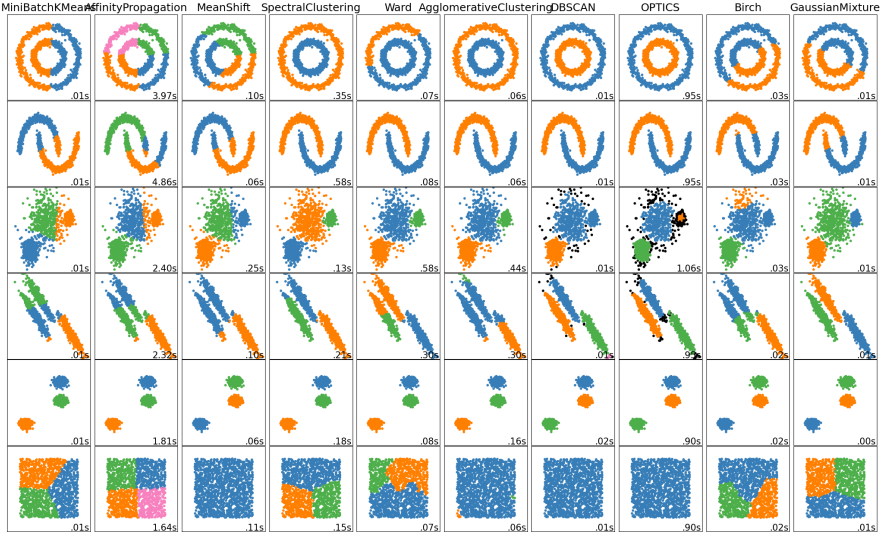
where n_G and n_H are number of points in clusters G and H, respectively.

Hierarchical clustering





Clustering Methods Comparison



Where should we place cell towers?

Suppose we wanted to place cell towers in a way that ensures that no building is more than 3000ft away from a cell tower. What linkage should we use to cluster buildings, and where should we cut the dendrogram, to solve this problem?

Dissimilarity measures

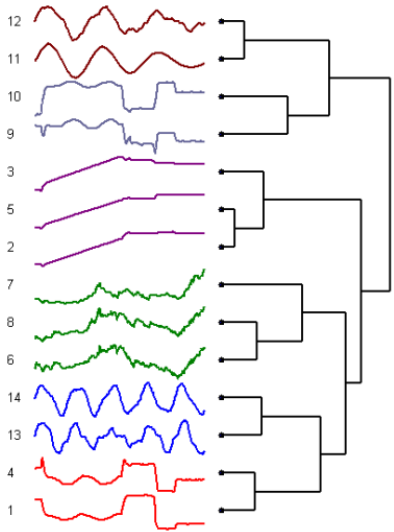
- The choice of **linkage** can greatly affect the structure and quality of the resulting clusters
- The choice of **dissimilarity** (equivalently, similarity) measure is arguably even more important.
- To come up with a **similarity measure**, you may need to think carefully and use your intuition about what it means for two observations to be similar. E.g.,
  - What does it mean for two people to have similar *purchasing behaviour*?
  - What does it mean for two people to have similar music listening habits?
- You can apply hierarchical clustering to any similarity measure $s(x_i, x_j)$ you come up with. The difficult part is coming up with a good similarity measure in the first place.

Example: Clustering time series

Here is an example of using hierarchical clustering to cluster time series.

You can quantify the similarity between two time series by calculating the **correlation** between them. There are different kinds of correlations out there.

[source: A Scalable Method for Time Series Clustering, Wang et al]



K-means vs Hierarchical clustering

K-means and GMM

- Low memory usage
- Essentially $O(n)$ compute time.
- Results are sensitive to random initialization.
- Number of clusters is pre-defined.
- Awkward with categorical variables.

Hierarchical clustering

- Deterministic algorithm
- Dendrogram shows us clusterings for various choices of K
- Requires only a **distance matrix**, quantifying how dissimilar observations are from one another
  - We can use a dissimilarity measure that gracefully handles categorical variables, missing values, etc.
- Memory-heavy, more computationally intensive than K-means.

DBSCAN

DBSCAN

DBSCAN = Density-based spatial clustering of applications with noise

Martin Ester et al., 1996

NOT hierarchical

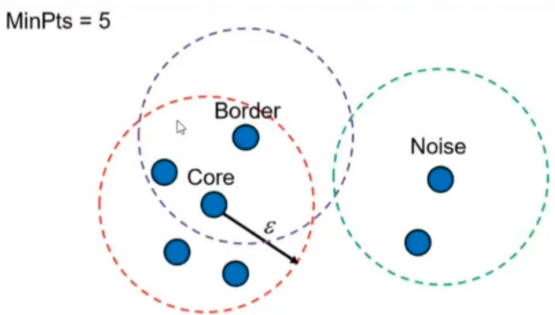
The size of clusters (or their count) is set indirectly by setting the following 2 hyperlarameters:

- ϵ – the maximum distance between two points for one to be considered as in the neighborhood of the other. Any distance function, e.g. Euclidean distance
- *minPts* – the minimum number of points in a neighborhood for a point to be considered as a core point

DBSCAN

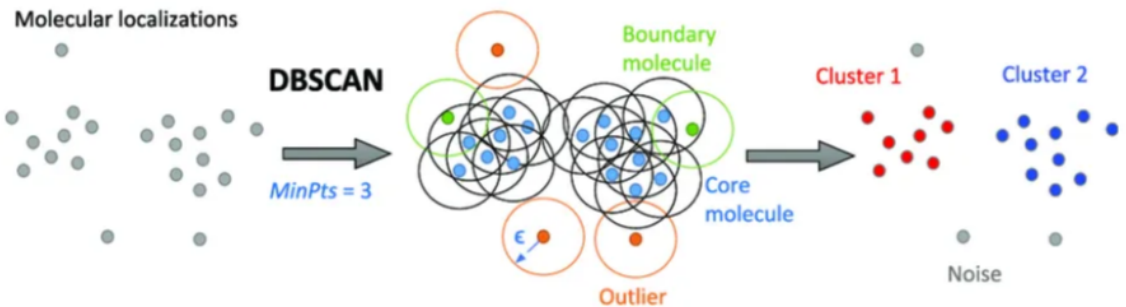
Based on these two parameters, we define:

- **core points** have at least *MinPts* other points within a given radius ϵ of itself.
- **border points** have less that *MinPts* points within the ϵ distance, but at least one *core point* is closer than ϵ.
- **noise points** are all other points



Density Connected Points

A point P is density connected to Q given ϵ and $MinPts$, if there is a chain of *core points* $P_1, P_2, P_3 \dots P_n$, where $P = P_1$ and $Q = P_2$ such that P_{i+1} is in the ϵ-neigborhood of P_i.



DBSCAN

Algorithm

1. Identify all points as either core point, border point or noise point.
2. For all of the unclustered core points:
   2.1 Create a new cluster.
   2.2 Add all the points that are unclustered and density connected to the current point into this cluster.
3. For each unclustered border point assign it to the cluster of nearest core point.
4. Leave all the noise points as it is.

DBSCAN – Advantages

- Robust to outliers – it isolates the noise points
- No need to specify the number clusters
- Can find arbitrary shaped clusters – suitable for data with irregular structures
- Only 2 hyperparameters to tune

DBSCAN – Disadvantages

- Sensitivity to hyperparameters
- Difficulty with varying density clusters
- Does not predict the cluster membership of new, unseen data points.

What clustering methods are able to predict clusters of new points?

Clustering evaluation

Clustering Evaluation

If you have a testing data available with annotated gold labels:

Rosenberg and Hirschberg (2007) define the following objectives for any cluster assignment:

- **Homogeneity** – each cluster contains only members of a single class
- **Completeness** – all members of a given class are assigned to the same cluster
- **V-measure** – their harmonic mean

If you do not have any labelled data:

- **Silhouette coefficient** – "unsupervised" consistency within clusters of data

Homogeneity

Homogeneity – To what extent each cluster contains only members of a single class?

$$h = 1 - \frac{H(C|K)}{H(C)}$$

$H(C|K)$ is the conditional entropy of the classes given the cluster assignments:

$$H(C|K) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{n_{c,k}}{n}\log\frac{n_{c,k}}{n_k}$$

$H(C)$ is the entropy of the classes:

$$H(C) = -\sum_{c=1}^{|C|}\frac{n_c}{n}\log\frac{n_c}{n}$$

Completeness

Completeness – To what extent all members of a given class are assigned to the same cluster?

$$c = 1 - \frac{H(K|C)}{H(K)}$$

$H(K|C)$ is the conditional entropy of the cluster assignments given the classes:

$$H(K|C) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{n_{c,k}}{n}\log\frac{n_{c,k}}{n_c}$$

$H(K)$ is the entropy of the clusters:

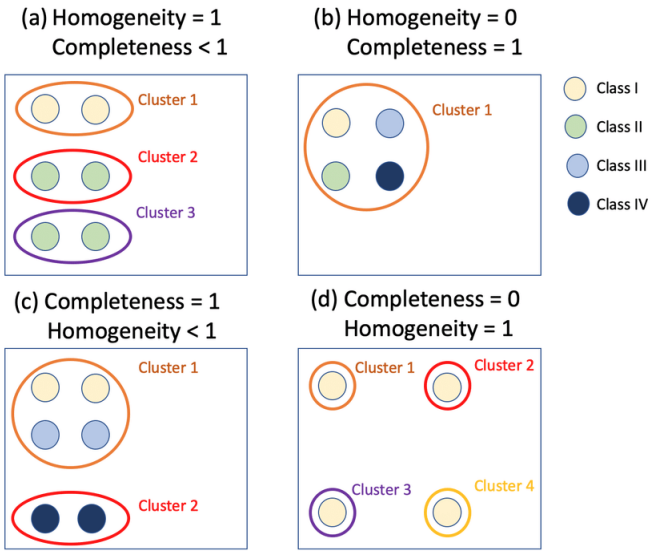
$$H(K) = -\sum_{k=1}^{|K|}\frac{n_k}{n}\log\frac{n_k}{n}$$

V-measure

V-measure – Harmonic mean of homogeneity and completeness:

$$v = \frac{2 \cdot h \cdot c}{h + c}$$

Homogeneity and Completeness



Silhouette coefficient

How similar an object is to its own cluster (cohesion) compared to other clusters (separation)

Values between -1 and 1

The Silhouette Coefficient s for a single sample is then given as:

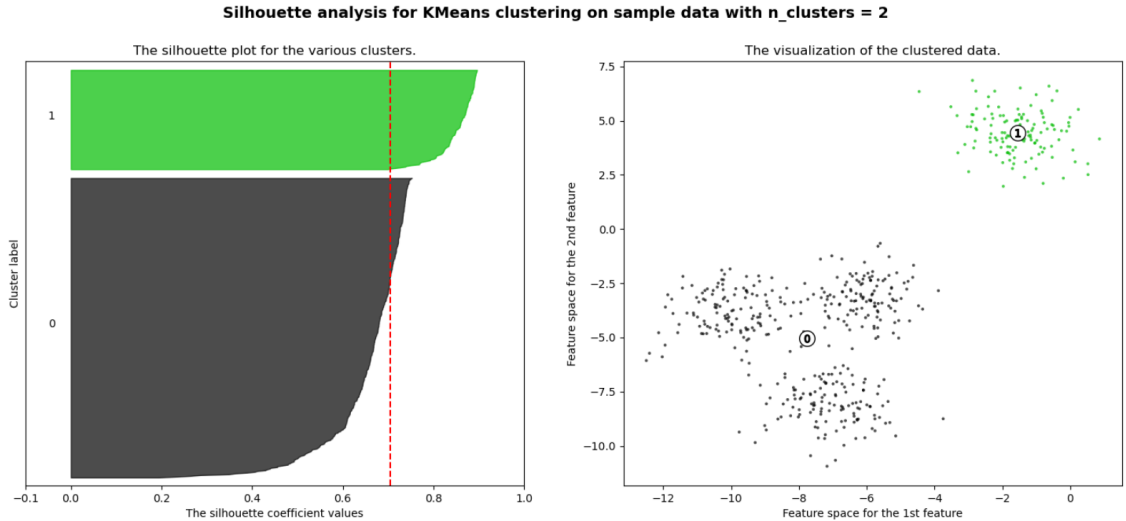
$$s_i = \frac{b_i - a_i}{max(a_i, b_i)}, \quad a_i = \frac{1}{C_I - 1} \sum_{j \in C_I, i \neq j} d(i,j), \quad b_i = \min_{J \neq I} \frac{1}{C_J} \sum_{j \in C_J} d(i,j)$$

- a is the mean distance between a sample and all other points in the same cluster
- b is the mean distance between a sample and all other points in the next nearest cluster

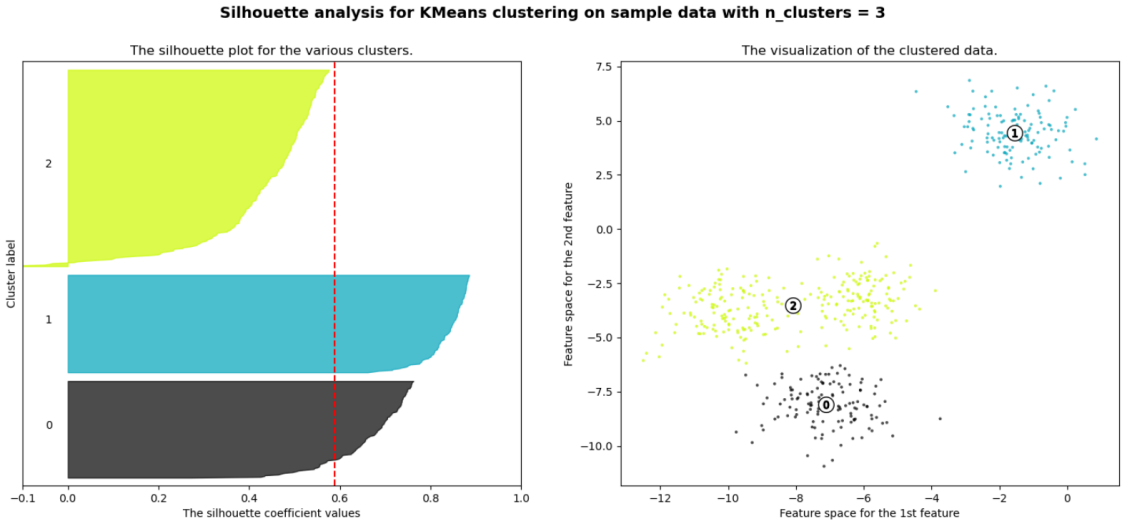
The mean over all points of a cluster is a measure of how tightly grouped all the points in the cluster are. Thus the mean over all data of the entire dataset is a measure of how appropriately the data have been clustered.

`https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`

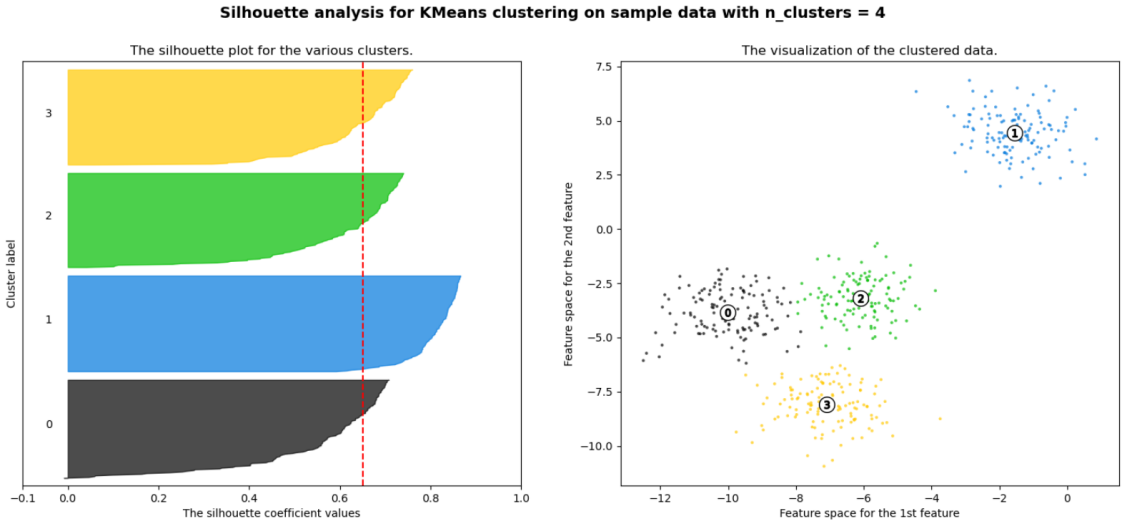
Silhouette coefficient



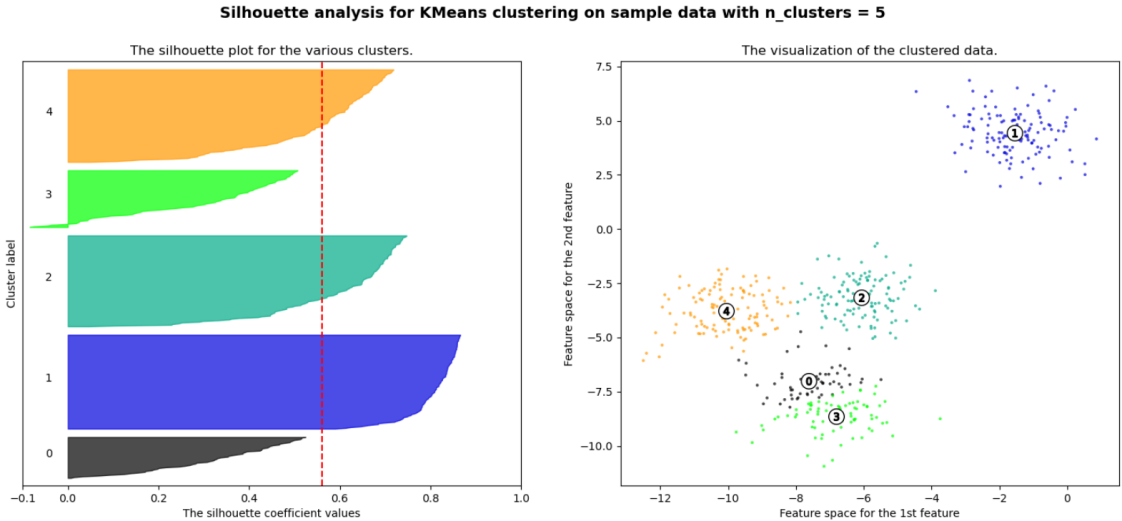
Silhouette coefficient



Silhouette coefficient



Silhouette coefficient



Silhouette coefficient