1 Clustering

Clustering is a fundamental method of data analysis used to create subsets in data, so-called clusters, according to their mutual similarity. The aim of clustering is to arrange a set of objects so that objects within one cluster are as similar to each other as possible, while objects from different clusters are markedly different from one another. Similarity is usually expressed by means of a distance or another similarity metric that quantifies the extent to which two objects are "close" or "distant".

Clustering belongs to the methods of unstructured (undirected) learning (unsupervised learning) within machine learning and statistics. This means that in clustering we do not have predefined categories or classes, as is the case, for example, in classification. The task of the algorithm is to discover, on the basis of the data, their internal structure – that is, the natural groups (subsets) present in the data. Clustering thus enables us to discover hidden patterns, relationships and structures in data that would otherwise remain invisible.

In a practical context, clustering can be understood as a tool for knowledge discovery. In the social sciences or educational research, it can be used, for example, to identify groups of pupils with similar behavioural patterns, learning strategies or performance profiles. In the field of natural language processing (NLP), clustering is used to group similar texts or concepts according to their meaning. In technical applications such as image processing, clustering helps to recognize regions with similar visual properties.

At the theoretical level, clustering also represents an interesting optimization problem – the algorithm seeks to find a partitioning of the data that minimizes the intra-cluster distance (i.e. the differences between objects within one cluster) and at the same time maximizes the inter-cluster distance (the differences between individual clusters). This principle can be expressed mathematically, and different algorithms implement it in different ways.

The best-known approaches include:

- **K-means algorithm**, which partitions data into a predetermined number k of clusters on the basis of their distance from so-called centroids.
- **Hierarchical clustering**, which creates a gradual hierarchy of clusters represented by means of a dendrogram.
- **Density-based methods** (e.g. DBSCAN), which search for regions with a high density of points and separate them from sparse regions.
- **Probabilistic models** (e.g. Gaussian Mixture Models), which assume that the data originate from a combination of several probability distributions.

Clustering thus forms an important bridge between the theory of statistical modelling and the practical discovery of patterns and relationships in data.

1.1 Use of clustering in phishing detection

Clustering, as a method of unstructured learning, is also applied in the field of cybersecurity, especially in the detection of phishing attacks. Phishing is a type of social engineering in which the attacker uses fake e-mails, websites or other communication channels with the aim of obtaining sensitive information (e.g. login credentials, financial data).

When systems are monitored, extensive logs are generated – records of user activities, system access, e-mail communication or network traffic behaviour. These data are often very heterogeneous and do not contain explicitly labelled records of phishing attacks. This is precisely where clustering is applied:

- **Discovery of abnormalities**
  Clustering makes it possible to group similar logs, whereby records that differ markedly from the majority may indicate suspicious or malicious behaviour. For example, e-mails with a similar structure, content or sender may form a natural cluster, whereas phishing messages often differ in their unusual syntax, sender domain or temporal activity.

- **Reduction of the space for analysis**
  Logs often contain millions of records. Clustering makes it possible to group large data sets into a smaller number of representative groups, thereby making it easier for analysts or automated systems to identify potentially malicious activities without the need to search through each record individually.

- **Identification of attack patterns**
  By using algorithms such as DBSCAN or K-means, it is possible to identify patterns of similar behaviour that are typical of phishing. For example, multiple login attempts from unknown IP addresses within a short time or e-mails with the same template can be clustered into a single "suspicious" cluster.

- **Integration with other methodologies**
  Clustering is often used as preprocessing or as a detection filter before classifiers based on supervised learning (*supervised learning*). Clustering can, for example, flag potentially phishing records, which subsequently undergo detailed analysis or a classification model, which increases the accuracy and efficiency of the entire system.

Overall, clustering makes it possible to work efficiently with large, heterogeneous logs and to discover hidden anomalies, which is crucial in detecting phishing attacks, as these often change their patterns in order to evade traditional rule-based or signature-based detection systems.

The most effective clustering methods for detecting phishing attacks on the basis of logs and network activity include density-based methods (Density-based clustering), such as DBSCAN (Density-Based Spatial Clustering of Applications with Noise) or its improved version HDBSCAN. This approach is suitable for several reasons:

- **Detection of anomalies and outliers:** Phishing attacks usually occur only sporadically and in small quantities compared with normal user traffic. Density-based methods naturally label points that do not fall into any dense cluster as "noise" or an anomaly. In this way, phishing activities, which are atypical, are automatically highlighted without the need for prior classification.

- **Independence from assumptions about cluster shape:** Algorithms such as K-means assume that clusters have a spherical or elliptical shape, which need not correspond to actual anomalies. DBSCAN and HDBSCAN, however, can detect clusters of arbitrary shape, which is important in phishing, where suspicious behaviour may manifest itself in various combinations of attributes (e.g. IP address, time of sending, message content).

- **Does not require a predetermined number of clusters:** In phishing detection, we cannot determine in advance how many types of attacks or suspicious activities are present in the logs. Density-based algorithms identify clusters automatically on the basis of the local density of points, thereby removing the need to specify the number of clusters k, which is a major advantage over K-means or GMM.

- **Robustness to noise and data variability:** System logs and e-mail data are often incomplete, heterogeneous and noisy. Density-based methods can handle these properties because they focus on concentrations of points, while sparse or unusual behaviour is naturally flagged as a deviation.
- **HDBSCAN – hierarchical robustness:** HDBSCAN extends DBSCAN with hierarchical density analysis, thereby making it possible to identify clusters of varying density. This is particularly useful in phishing, where some suspicious behaviour occurs frequently (e.g. mass spam campaigns), while other behaviour occurs only sporadically (targeted attacks).

It follows from the above that density-based clustering is ideal for phishing detection, because it can effectively distinguish normal behaviour from anomalies, does not require a precisely defined number of clusters, can work with irregular cluster shapes and is resistant to noise and data variability, all of which are key properties of logs containing phishing activities.

2 The DBSCAN method – density-based clustering

2.1 Introduction

The **DBSCAN** method (*Density-Based Spatial Clustering of Applications with Noise*), proposed by Ester, Kriegel, Sander and Xu (1996), is one of the most influential density-based clustering algorithms. Its main advantage is the ability to identify clusters of arbitrary shape and, at the same time, to recognize noise (*noise*) – objects that do not belong to any cluster. Unlike methods such as *k-means*, it does not require the number of clusters to be determined in advance.

2.2 Basic idea

The idea of *density-based clustering* is that, for each object belonging to a cluster, at least a minimum number of other objects must exist in its neighbourhood. The cardinality of the neighbourhood must therefore reach a certain threshold value, denoted as **MinPts**.

In this way, objects located in denser regions of the multidimensional space can form clusters that are separated by regions of lower density (so-called noise). Density is therefore determined by means of two parameters:

- ε – **neighbourhood radius (radius)**, which determines the distance within which objects are considered to be neighbours,
- **MinPts – minimum number of points** within a cluster (ε-neighbourhood) for a region to be considered sufficiently dense.

2.3 Formal definitions

Let $O = \{o_1, o_2, \ldots, o_n\} \subseteq \mathbb{R}^d$ be a set of objects in a d-dimensional space and $\mathrm{dist}(o_i, o_j)$ a distance metric (most often the Euclidean distance).

Definition 2.1: ε-neighbourhood

For a given point $o \in O$ and a parameter $\varepsilon > 0$, we define the **ε-neighbourhood** of the point o as the set:

$$N_\varepsilon(o) = \{o' \in O \mid \text{dist}(o, o') \leq \varepsilon\}.$$

This means that the ε-neighbourhood of an object $o \in O$ includes all objects $o' \in O$ located within the radius ε around the point o; the distance $\text{dist}(o, o')$ between the points o and o' must not be greater than ε.

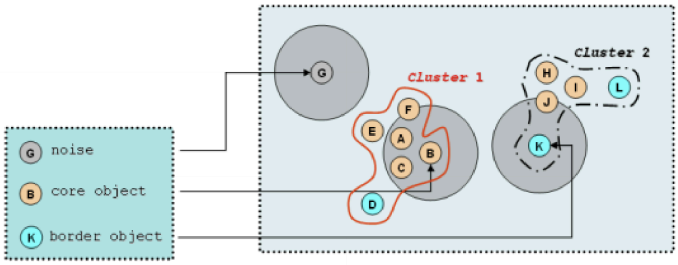


Fig. 1: Density-Based clustering

Figure 1 shows that at the end of the process three classes of objects arise:

- **class of noise (noise)** – objects that cannot be assigned to any cluster,
- **class of core objects (core objects)**,
- **class of border objects (border objects)** – they belong to a cluster but are not themselves core objects.

Definition 2.2: Core point (*core point*)

A point $o \in O$ is a **core point** if the number of points in its ε-neighbourhood satisfies the condition:

$$|N_\varepsilon(o)| \geq \text{MinPts}.$$

In the DBSCAN algorithm, a cluster does not arise from a single core object but from several core objects that are mutually "density-reachable" (*density-reachable*). Although one core point often "triggers" the process of cluster formation (when traversing the data), the resulting cluster consists of several core objects and possibly also of border objects lying within their ε-radius.

Definition 2.3: Border object

An object $o \in O$ is called a *border object* if:

1. $|N_\varepsilon(o)| < \text{MinPts}$ (the core-object condition is not satisfied),
2. $\exists o' \in \{\text{core objects}\}$, for which $o \in N_\varepsilon(o')$.

An object is therefore a *border object* if fewer than MinPts objects are located within its ε-radius (neighbourhood). A border object belongs to a cluster because it is located within the ε-radius (neighbourhood) of a core object, but it cannot form a cluster itself (cf. Figure 1).

All objects in the cluster region must be directly density-reachable with respect to the parameters MinPts and ε.

Definition 2.4: Direct density reachability (*directly density-reachable*)

A point p is **directly density-reachable** from a point q if:

$$p \in N_\varepsilon(q) \quad \text{and at the same time} \quad |N_\varepsilon(q)| \geq \text{MinPts}.$$

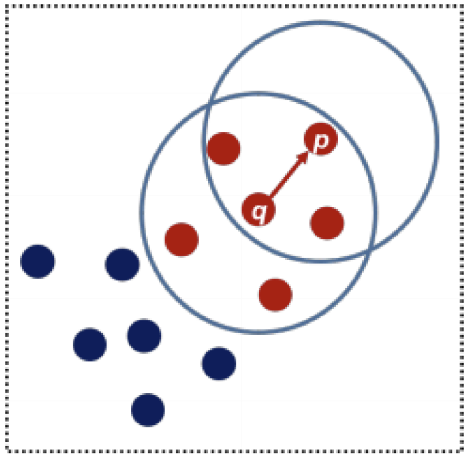


Fig. 2: Direct density reachability

Definition 2.5: Density reachability (*density-reachable*)

A point p is **density-reachable** from a point q if there exists a finite sequence of points

$$q = o_1, o_2, \ \ldots, o_k = p,$$

such that each point o_{i+1} is *directly density-reachable* from the point o_i.

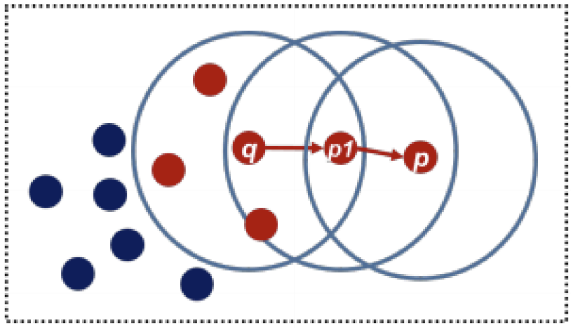


Fig. 3: Density reachability

Density reachability is an extension of direct density reachability. This relation is transitive but not symmetric (cf. Figure 3). Symmetry generally does not hold for all objects within a cluster, but it may hold for a particular object o for which $|N_\varepsilon(o)| \geq \text{MinPts}$.

Two border objects of the same cluster need not be mutually density-reachable, because there is not a sufficient number of objects in their ε-region. Another core object is therefore needed from which both border points are density-reachable. In this way, the concept of *density connection* is introduced.

Definition 2.6: Density connection (*density-connected*)

Two points $p, q \in O$ are **density-connected** if there exists a point $o \in O$ from which both points are density-reachable:

$$\exists o \in O: \; p \text{ is density-reachable from } o \; \wedge \; q \text{ is density-reachable from } o\,.$$

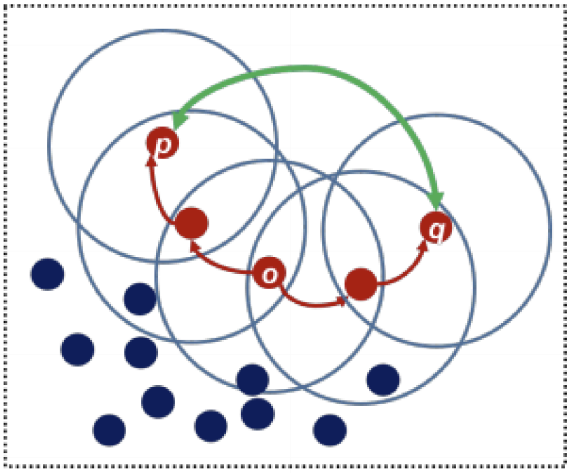


Fig. 4: Density connection

Density connection is a symmetric relation. The objects p and q are thus mutually density-connected via the object o. The object o may also be one of the objects p or q (cf. Figure 4).

Density-based clusters can generally be understood as *density-connected sets*. Density-connected sets are defined in the same way as density-based clusters.

Definition 2.7: Density-based cluster (*density-based cluster*)

Let O be a set of objects. A subset $C \subseteq O$ is a **cluster** with respect to the parameters ε and MinPts if C is not an empty set and the following conditions hold:

- **Maximality:** $\forall p, q \in O$: if $p \in C$ and q is density-reachable from p with respect to ε and MinPts, then $q \in C$.
- **Connectivity:** $\forall p, q \in C$: the points p and q are density-connected in O with respect to ε and MinPts.

Definition 2.8: Noise (*noise*)

Points that do not belong to any cluster are labelled as **noise**:

$$\text{Noise} = \{o \in O \mid \forall C_i : o \notin C_i\}\,.$$

2.4 Interpretation of density

Density-based clustering can also be interpreted through the **local density of points**. For a point o, we define the local density as:

$$\rho(o) = \frac{|N_\varepsilon(o)|}{V_\varepsilon},$$

where V_ε is the volume of the ε-ball in d-dimensional space:

$$V_\varepsilon = \frac{\pi^{d/2}}{\Gamma\left(\frac{d}{2}+1\right)}\varepsilon^d .$$

However, DBSCAN does not compute this density explicitly; instead, it estimates it implicitly by comparing the number of points with the threshold *MinPts*.

2.5 DBSCAN algorithm

Input:

- set of points $O \subseteq \mathbb{R}^d$,
- parameters ε, MinPts.

Output:

- set of clusters $C_1, C_2, \ldots, C_k$,
- set of noise points N.

Algorithm procedure:

1. For each point $o \in O$ that has not yet been visited:
   (a) Mark o as visited and compute $N_\varepsilon(o)$.
   (b) If $|N_\varepsilon(o)| <$ MinPts, mark the point as noise.
   (c) Otherwise, start a new cluster and expand it by all points that are density-reachable from o.
2. Repeat the process until all points are labelled as:
   - part of a cluster, or
   - noise.

2.6 Properties of the algorithm

- It identifies clusters of arbitrary shape.
- It can distinguish noise and border points.
- It does not require the number of clusters to be determined in advance.
- It is sensitive to the choice of the parameters ε and MinPts.

2.7 Choice of parameters

- **MinPts:** MinPts $\geq d + 1$ is recommended, where d is the dimension of the space.
- **ε:** it is determined on the basis of the *k-distance graph*, in which a point of sharp bend (the "knee") is sought.

2.8 Conclusion

The DBSCAN method provides a robust and theoretically well-founded approach to identifying clusters on the basis of the local density of points. Its mathematical core lies in the concepts of the ε-neighbourhood and density connectivity, which make it possible to model the structure of data without assumptions about the shape or number of clusters.

2.9 HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)

The **HDBSCAN** algorithm is an extension of the DBSCAN method that removes its main limitations, in particular the need to determine the parameter ε in advance. HDBSCAN makes it possible to detect **clusters of varying density** and to create their **hierarchical structure**, while automatically extracting stable and significant clusters without the need for manual threshold selection.

2.9.1 Basic idea

Whereas DBSCAN defines clusters as regions with a point density higher than a given threshold ε, HDBSCAN models density continuously and creates a *hierarchy of density levels*, from which it subsequently extracts stable clusters. It thereby avoids the problem of choosing a single suitable ε for data with non-homogeneous density.

2.9.2 Mathematical foundations

Let $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d$ be a set of data points and $d(x_i, x_j)$ a distance metric.

1. **Core-distance:** For a point x_i, we define its *core distance* as the distance to its k-th nearest neighbour:

   $$\text{core_dist}(x_i) = d_k(x_i),$$

   where $k = \text{MinPts}$ is the minimum number of points in the neighbourhood (a parameter of the algorithm).

2. **Mutual reachability distance:** For a pair of points x_i, x_j, we define the *mutual reachability* as:

   $$d_{\text{mreach}}(x_i, x_j) = \max\{\text{core_dist}(x_i),\ \text{core_dist}(x_j),\ d(x_i, x_j)\}.$$

   This metric smooths out differences in local density and ensures that points from sparse regions will not be artificially joined with dense regions.

3. **Transformation to the density scale:** The parameter

   $$\lambda = \frac{1}{d}$$

   is often used (for clustering boundaries), i.e. small distances correspond to large values of λ (high "density"). In the hierarchical representation, one works with an ordering by increasing distance d (or decreasing λ).

2.9.3 HDBSCAN algorithm

1. **We compute the core-distance:** For each point, we compute the *core-distance* (the k-NN distance for the parameter `MinPts`).

2. **We define the mutual reachability distance:** For all pairs of points, we define the *mutual reachability distance*:

$$d_{\text{mreach}}(x_i, x_j) = \max\{\text{core_dist}(x_i), \text{core_dist}(x_j), d(x_i, x_j)\}.$$

3. **We build a weighted graph:** We create a graph with vertices X and edges weighted by d_{mreach}. (In practice, a k-NN graph or efficient approximations are often used so that a complete graph is not built.)

4. **We compute the minimum spanning tree MST:** From the graph, we compute the *minimum spanning (MST) tree* according to the weights d_{mreach}. The MST captures connections according to increasing "reachability" between points at increasing distances.

5. **We apply single-linkage hierarchical clustering:** Over the MST, we apply *single-linkage hierarchical clustering* according to the weights d_{mreach}. The result is a hierarchy of clusters (dendrogram) on the density scale.

6. **We condense the hierarchy:** We condense the tree and focus on components that have a size $\geq$ `min_cluster_size`. This eliminates very small, insignificant branches.

7. **We evaluate cluster stability:** For each candidate cluster in the condensed hierarchy, we compute its *stability* and extract the clusters that are the most stable. These clusters form the final, "flat" partitioning.

8. **We label noise:** Points that do not belong to any stable cluster are labelled as *noise* (noise).

2.9.4 Cluster stability measure

Each cluster C is *born* in the hierarchy at a certain density λ_{birth} and *dies* at the density λ_{death}. The stability of a cluster is defined as the integral or sum of the contributions of all points that leave the cluster:

$$\text{stability}(C) = \sum_{p \in C} (\lambda_p - \lambda_{\text{birth}}),$$

where λ_p is the density (or $1/d$) at which the point p ceases to belong to the cluster. Clusters with high stability persist across a wide range of densities and are therefore considered *robust and significant*.

Advantages of HDBSCAN over DBSCAN and other methods

- **Adaptability to local density:** it can find clusters of different densities in a single run (DBSCAN requires different ε for different densities).
- **Hierarchical output:** it provides information on the relationships between clusters at different density levels, which facilitates analysis and visualization.
- **Robust cluster extraction via the stability measure:** it depends less on a single highly sensitive threshold.
- **Better handling of noise:** points that are not part of any stable cluster are labelled as *outliers* (noise).

HDBSCAN parameters and their interpretation

- **min_cluster_size:** the minimum size of a cluster that we consider relevant. It determines the granularity of the condensed tree (typically chosen according to domain meaning, e.g. the expected minimum number of points that we consider a "real" cluster).
- **min_samples** (or `MinPts`): used in computing the *core-distance*. A higher value means a stricter assessment of core points, which makes the algorithm more robust to noise but may merge or remove small clusters.
- **Recommendations:** `min_samples = min_cluster_size` or `min_samples = min_cluster_size / 2` is often chosen, or alternatively `min_samples = dataset dimensionality`.
- **Sensitivity:** HDBSCAN is less sensitive to the exact setting of `min_cluster_size` than DBSCAN is to the parameter ε, but the parameters still affect the clustering result.

2.9.5 Use and applications

HDBSCAN is particularly suitable for tasks where regions with different densities and cluster shapes occur in the data – for example, in log analysis, anomaly detection or the identification of phishing attacks. Its ability to adapt to the local structure of the data and to separate noise automatically makes it a robust tool for detecting complex behavioural patterns in real data.

Practical recommendations (especially for log analysis / phishing detection)

- **Scaling and attribute selection:** before computing the metric, we normalize or scale the attributes (e.g. log-transform for time intervals, one-hot for categories or vector embeddings for e-mail text). We ensure correct scaling so that the different dimensions have appropriate importance and do not distort the core-distance.
- **Choice of metric:** for textual or combined data, we choose non-Euclidean metrics (cosine for TF-IDF embeddings, edit distance for domains, etc.) or construct a composite distance.
- **Setting min_cluster_size:** typically between 5–50; in phishing detection, we choose a smaller value if we expect small targeted campaigns, but not too small, so that a large amount of "noise" is not selected as clusters.
- **Setting min_samples:** we increase the value for a more conservative extraction of clusters; with a large amount of noise, we increase the value slightly.
- **Combination with other methods:** after HDBSCAN, we can apply a supervised classifier or perform a manual analysis of the candidate clusters (e.g. extracting typical domains or message templates in the cluster).
- **Interpretability:** HDBSCAN naturally creates a hierarchy, which we visualize (condensed tree, cluster persistence diagram) — very useful in the forensic analysis of phishing attacks.