Chapter 5

Density-Based Clustering

Martin Ester
Simon Fraser University
British Columbia, Canada
ester@cs.sfu.ca

5.1 Introduction 111
5.2 DBSCAN 113
5.3 DENCLUE 115
5.4 OPTICS 116
5.5 Other Algorithms 116
5.6 Subspace Clustering 118
5.7 Clustering Networks 120
5.8 Other Directions 123
5.9 Conclusion 124
Bibliography 125

5.1 Introduction

Many of the well-known clustering algorithms make, implicitly or explicitly, the assumption that data are generated from a probability distribution of a given type, e.g., from a mixture of k Gaussian distributions. This is the case in particular for EM (Expectation Maximization) clustering and for k-means. Due to this assumption, these algorithms produce spherical clusters and cannot deal well with datasets in which the actual clusters have nonspherical shapes. Nonspherical clusters occur naturally in spatial data, i.e., data with a reference to some two- or three-dimensional concrete space corresponding to our real world. Spatial data include points, lines, and polygons and support a broad range of applications. Clusters in spatial data may have arbitrary shape, i.e., they are often drawn-out, linear, elongated etc., because of the constraints imposed by geographic entities such as mountains and rivers. In geo-marketing, one may want to find clusters of homes with a given characteristic, e.g., high-income homes, while in crime analysis one of the goals is to detect crime hot-spots, i.e., clusters of certain types of crimes. Even in higher-dimensional data the assumption of a certain number of clusters of a given shape is very strong and may often be violated. In this case, algorithms such as k-means will break up or merge the actual clusters, leading to inaccurate results. The objective to minimize the average squared distances of points from their corresponding cluster center leads to a partitioning of the dataset that is equivalent to the Voronoi diagram of the cluster centers, irrespective of the shape of the actual clusters. Figure 5.1 illustrates this weakness on a small 2-dimensional dataset, showing that k-means with $k = 3$ breaks up and merges the three horizontal clusters.

This observation motivates the requirement to discover clusters of arbitrary shape. The increasingly large sizes of real-life databases require scalability to large databases, i.e., efficiency on databases of up to millions of points or more. Finally, the clustering of large databases requires the

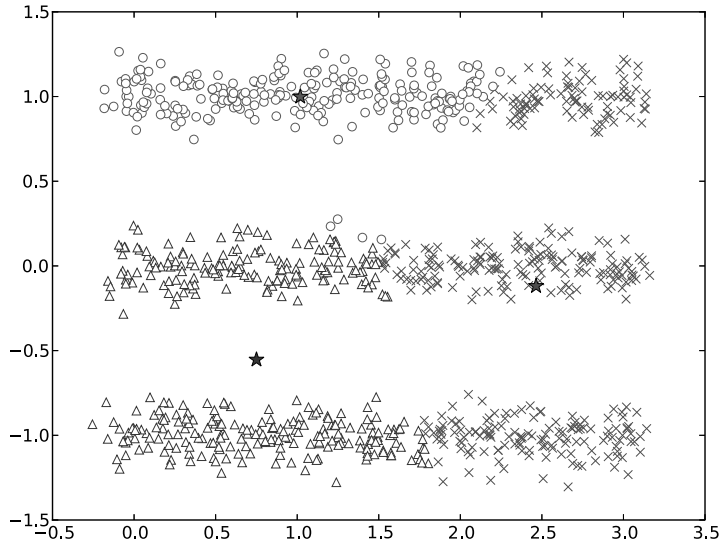

FIGURE 5.1: k-means with $k = 3$ on a sample 2-dimensional dataset.

ability to detect and remove noise and outliers. The paradigm of density-based clustering has been proposed to address all of these requirements. Density-based clustering can be considered as a non-parametric method, as it makes no assumptions about the number of clusters or their distribution.

Density-based clusters are connected, dense areas in the data space separated from each other by sparser areas. Furthermore, the density within the areas of noise is assumed to be lower than the density in any of the clusters. Due to their local nature, dense connected areas in the data space can have arbitrary shape. Given an index structure that supports region queries, density-based clusters can be efficiently computed by performing at most one region query per database object. Sparse areas in the data space are treated as noise and are not assigned to any cluster.

It is worth noting that the algorithms that are being referred to in the literature as density-based clustering have various predecessors that have already explored some of the ideas. In particular, Wishart [33] explored ways to avoid the so-called chaining effect in single-link clustering, which is caused by a small number of noisy data points that connect sets of points that should in principle form separate clusters. First, nondense points, that have fewer than k neighbors within a distance of r, are removed. Second, single-link is employed to cluster the remaining points. Finally, nondense points may be allocated to one of the clusters according to some criterion. We would also like to point out the relationship of the paradigms of density-based clustering and mean-shift clustering [8]. The mean-shift procedure is an iterative procedure that replaces each point by the weighted mean of its neighboring points, where the neighborhood and weights are determined by the chosen kernel function, and it converges to the nearest stationary point of the underlying density function. Mean-shift clustering employs the mean-shift procedure as density-estimator. In mean-shift clustering, very narrow kernels create singleton clusters, very wide kernels create one cluster, and intermediate kernels create a natural number of clusters. As opposed to density-based clustering, in mean-shift clustering, the neighborhood membership is weighted (instead of Boolean), the minimum number of points does not need to be specified, and there is no guarantee that clusters are connected.

A density-based clustering algorithm needs to answer several key design questions:

- How is the density estimated?

- How is connectivity defined?
- Which data structures support the efficient implementation of the algorithm?

In the following, we will present the main density-based clustering algorithms and discuss the ways in which they answer these questions.

5.2 DBSCAN

DBSCAN [11] estimates the density by counting the number of points in a fixed-radius neighborhood and considers two points as connected if they lie within each other's neighborhood. A point is called *core point* if the neighborhood of radius *Eps* contains at least *MinPts* points, i.e., the density in the neighborhood has to exceed some threshold. A point q is directly density-reachable from a core point p if q is within the *Eps*-neighborhood of p, and density-reachability is given by the transitive closure of direct density-reachability. Two points p and q are called density-connected if there is a third point o from which both p and q are density-reachable. A cluster is then a set of density-connected points which is maximal with respect to density-reachability. Noise is defined as the set of points in the database not belonging to any of its clusters. The task of density-based clustering is to find all clusters with respect to parameters *Eps* and *MinPts* in a given database.

In the following we provide more formal definitions. Let D be a set (database) of data points. The definition of density-based clusters assumes a distance function $dist(p,q)$ for pairs of points. The Eps-neighborhood of a point p, denoted by $NEps(p)$, is defined by $NEps(p) = \{q \in D | dist(p,q) \leqq Eps\}$. A point p is *directly density-reachable* from a point q with respect to *Eps*, *MinPts* if (1) $p \in NEps(q)$ and (2) $|NEps(q)| \geq MinPts$. A point p is *density-reachable* from a point q with respect to *Eps* and *MinPts* if there is a chain of points $p_1, ..., p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i. Density-reachability is a canonical extension of direct density-reachability. Since this relation is not transitive, another relation is introduced. A point p is *density-connected* to a point q with respect to *Eps* and *MinPts* if there is a point o such that both p and q are density-reachable from o with respect to *Eps* and *MinPts*. Figure 5.2 illustrates these concepts. While p is density-reachable from q, q is not density-reachable from p. a and c are density-connected via b.

Intuitively, a density-based cluster is a maximal set of density-connected points. Formally, a cluster C with respect to *Eps* and *MinPts* is a nonempty subset of D satisfying the following two conditions:

1. $\forall p,q$ if $p \in C$ and q is density-reachable from p with respect to Eps and $MinPts$, then $q \in C$. (maximality)

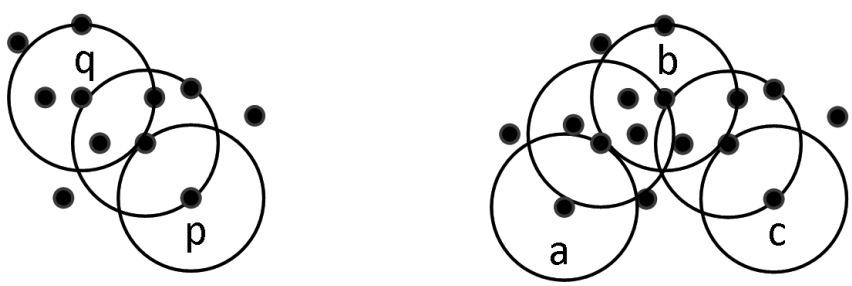

FIGURE 5.2: Density-reachability and connectivity.

2. $\forall p,q \in C$: p is density-connected to q with respect to Eps and $MinPts$. (connectivity)

Let $C_1,\ldots,C_k$ be the clusters of the database D with respect to *Eps* and *MinPts*. Then the *noise* is defined as the set of points in D not belonging to any cluster C_i, i.e., $noise = \{p \in D | p \notin C_i \forall i\}$. Density-based clustering distinguishes three different types of points (see Figure 5.2):

- core points, i.e., points with a dense neighborhood ($|NEps(p)| \geq MinPts$),
- border points, i.e., points that belong to a cluster, but whose neighborhood is not dense, and
- noise points, i.e., points which do not belong to any cluster.

In Figure 5.2, e.g., q and b are core points, and p, a and c are border points.

Density-based clusters have two important properties that allow their efficient computation. Let p be a core point in D. Consider the set O of all points drawn from D, which are density-reachable from p with respect to Eps and $MinPts$. This set O is a cluster with respect to *Eps* and *MinPts*. Let C be a cluster in D. Each point in C is density-reachable from any of the core points of C and, therefore, a cluster C contains exactly the points which are density-reachable from an arbitrary core point of C. Thus, a cluster C with respect to *Eps* and *MinPts* is uniquely determined by any of its core points. This is the foundation of the DBSCAN algorithm.

To find a cluster, DBSCAN starts with an arbitrary database point p and retrieves all points density-reachable from p with respect to *Eps* and *MinPts*, performing region queries first for p and if necessary for p's direct and indirect neighbors. If p is a core point, this procedure yields a cluster with respect to *Eps* and *MinPts*. If p is not a core point, no points are density-reachable from p and DBSCAN assigns p to the noise and applies the same procedure to the next database point. If p is actually a border point of some cluster C, it will later be reached when collecting all the points density-reachable from some core point of C and will then be (re-)assigned to C. The algorithm terminates when all points have been assigned to a cluster or to the noise.

Standard DBSCAN implementations are based on a spatial index such as an R-tree [14] or X-tree [4], which provides efficient support of region queries that retrieve the *Eps*-neighborhood of a given point. In the worst case, DBSCAN performs one region query per database point. This leads to a runtime complexity of $O(nlogn)$ for DBSCAN, where n denotes the number of database points. Unfortunately, spatial indexes degenerate for high-dimensional data, i.e., the performance of region queries degenerates from $O(logn)$ to $O(n)$, and the runtime complexity of DBSCAN becomes $O(n^2)$ for such data. On the other hand, if a grid-based data structure is available that supports $O(1)$ region queries, the runtime complexity of DBSCAN decreases to $O(n)$. Note that a runtime complexity of $O(nlogn)$ is considered to be scalable to large datasets.

An incremental version of DBSCAN can further improve its efficiency in dynamic databases with insertions and deletions. [10] shows that a density-based clustering can be updated incrementally without having to rerun the DBSCAN algorithm on the updated database. It examines which part of an existing clustering is affected by an update of the database and presents algorithms for incremental updates of a clustering after insertions and deletions. Due to the local nature of density-based clusters, the portion of affected database objects tends to be small which makes the incremental algorithm very efficient.

The basic idea of density-based clusters can be generalized in several ways [30]. First, any notion of a neighborhood can be employed instead of a distance-based *Eps*-neighborhood as long as the definition of the neighborhood is based on a predicate $NPred(p,q)$ which is symmetric and reflexive. The neighborhood N of p is then defined as the set of all points q satisfying $NPred(p,q)$. Second, instead of simply counting the elements in a neighborhood we can as well use a more general predicate $MinWeight(N)$ to determine whether the neighborhood N is dense, if *MinWeight* is monotone in N, i.e., if *MinWeight* is satisfied for all supersets of sets that satisfy N. Finally, not only point-like objects but also spatially extended objects such as polygons can be clustered. When clustering polygons, for example, the following predicates are more natural than

the *Eps*-neighborhood and the *MinPts* cardinality constraint: $NPred(X,Y)$ iff $intersect(X,Y)$ and $MinWeight(N)$ iff $\sum_{p\in N} population(p) \geq MinPop$. The GDBSCAN algorithm [30] for finding generalized density-based clusters is a straightforward extension of the DBSCAN algorithm.

5.3 DENCLUE

DENCLUE [16] takes another approach to generalize the notion of density-based clusters, based on the concept of *influence functions* that mathematically model the influence of a data point in its neighborhood. The density at some point is estimated by the sum of the influences of all data points. A point is said to be *density-attracted* to a so-called density-attractor, if they are connected through a path of high-density points. An influence function should be symmetric, continuous, and differentiable, and typical examples of influence functions are square wave functions or Gaussian functions. The *density function* at a point x is computed as the sum of the influence functions of all data points at point x. *Density-attractors* are points that correspond to local maxima of the density function. A point p is density-attracted to a density-attractor q if q can be reached from p through a path of points that lie within a distance of *Eps* from each other in the direction of the gradient. An *arbitrary-shape cluster* for a set of density-attractors X is then defined as the set of all points that are density-attracted to one of the density-attractors x from X where the density function at x exceeds a threshold ξ. In addition, all pairs of density-attractors need to be connected to each other via paths of points whose density meets the same threshold.

More precisely, the *influence function* of a data point $y \in F^d$ is a function $f_B^y(x)$ defined in terms of a basic influence function f_B, i.e., $f_B^y(x) = f_B(x,y)$. A simple example of an influence function is the Square Wave Influence Function:

$$f_B(x,y) = \begin{array}{ll} 1 & \text{if } d(x,y) \leq \sigma, \\ 0 & \text{otherwise.} \end{array}$$

Given a database of points $D = \{x_1,\ldots,x_N\} \subset F^d$, the *density function* is defined as $f_B^D(x) = \sum_{i=1}^N f_B^{x_i}(x)$. The gradient of the density function is defined as $\Delta f_B^D(x) = \sum_{i=1}^N (x_i - x) f_B^{x_i}(x)$. A point x^* is called a *density-attractor*, if x^* is a local maximum of the density function $f_B^D(x)$. A point x is *density-attracted* to density-attractor x^*, if there is a sequence of points $x^k, d(x^k,x^*) \leq \varepsilon$ for some distance function d, with $x^i = x^{i-1} + \delta \frac{\Delta f_B^D(x^{i-1})}{\|\Delta f_B^D(x^{i-1})\|}$.

Given a set of density-attractors X, an *arbitrary-shape cluster* with respect to σ and ξ is a subset $C \subseteq D$, where

1. $\forall x \in C\ \exists x^* \in X : f_B^D(x^*) \geq \xi$ and x is density-attracted to x^*, and
2. $\forall x_1^*, x_2^* \in X\ \exists$ path $P \subset F^d$ from x_1^* to x_2^* with $\forall p \in P : f_B^D(p) \geq \xi$.

The parameter σ, employed by the basic influence function, determines the reach of the influence of a point, while parameter ξ specifies when a density-attractor is significant. Note that the DENCLUE clusters become identical to the DBSCAN clusters when choosing the Square Wave Influence Function with $\sigma = Eps$ and $\xi = MinPts$.

The efficient implementation of the DENCLUE algorithm is based on the observation that most data points do not contribute to the density function at any given point of the data space. This can be exploited by computing only a local density function, while guaranteeing tight error bounds. To efficiently access neighboring points, a so-called *map* data structure is created, a d-dimensional grid structure of grid length 2σ. Only grid cells that actually contain points are determined and are