DBSCAN: Density-Based Clustering

A 5-slide visual companion to Lesson 4 (Cluster Analysis)

Prof. Dr. Jörg Osterrieder

School of Engineering and Computer Sciences

1. Density, Not Distance to a Centre

k-Means asks *which centre is nearest?* DBSCAN (Density-Based Spatial Clustering of Applications with Noise) asks a different question: *is this point sitting in a crowded region?*

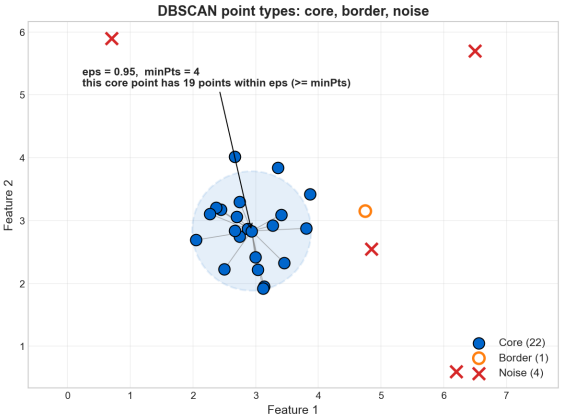
Two parameters define "crowded":

- **eps** (the neighbourhood radius): how close counts as nearby.
- **minPts** (minimum points): how many neighbours within eps make a region dense.

Every point gets one of three labels:

- **Core**: at least minPts points within eps (itself included).
- **Border**: inside a core point's eps, but not itself dense.
- **Noise**: neither core nor border (an outlier).

Worked setting: with eps $= 0.95$ and minPts $= 4$, the picture has **22 core**, **1 border**, **4 noise** points.



What three labels can DBSCAN give a point, and what exactly makes a point a core point?

2. The DBSCAN Algorithm, Step by Step

The procedure grows one cluster at a time from a dense seed:

1. Pick an unvisited point, count its neighbours within eps.
2. If the count is at least minPts it is a *core* point: open a cluster and absorb everything *density-reachable* (a chain of core-to-core hops, collecting border points).
3. Otherwise tag it *noise* for now (a later core may still claim it as a border point).
4. Repeat until every point has been visited.

Predict before the reveal: two dense blobs plus scattered points, how many scattered points stay noise?

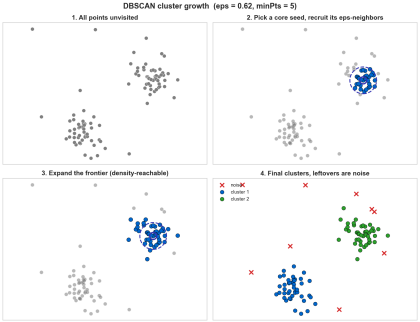
2. The DBSCAN Algorithm, Step by Step

The procedure grows one cluster at a time from a dense seed:

1. Pick an unvisited point, count its neighbours within eps.
2. If the count is at least minPts it is a *core* point: open a cluster and absorb everything *density-reachable* (a chain of core-to-core hops, collecting border points).
3. Otherwise tag it *noise* for now (a later core may still claim it as a border point).
4. Repeat until every point has been visited.

Predict before the reveal: two dense blobs plus scattered points, how many scattered points stay noise?

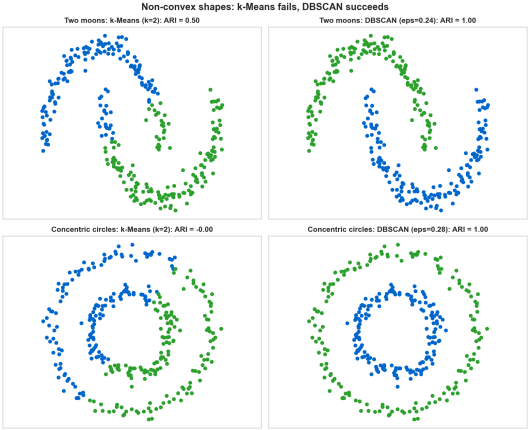
With eps = 0.62 and minPts = 5, DBSCAN finds **2 clusters** and leaves **9 points as noise**. No k was ever specified: the data decided.



Why can a point start out as noise and still end up as a border point of a cluster?

3. Why DBSCAN Beats k-Means on Shapes

k-Means draws straight boundaries because it assumes round, equal-size blobs. DBSCAN follows the density trail, so it can wrap around crescents and rings.



Score: Adjusted Rand Index (ARI), where 1.0 is a perfect match to the true groups and 0 is no better than random.

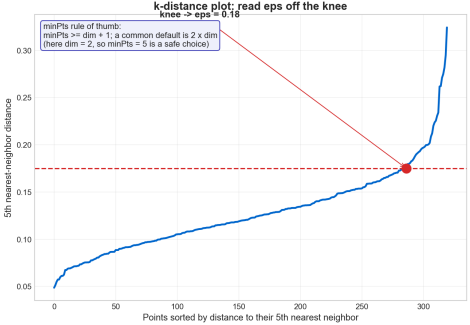
- Two moons: k-Means ARI = 0.50, DBSCAN ARI = **1.00**.
- Concentric circles: k-Means ARI $\approx$ 0.00, DBSCAN ARI = **1.00**.

k-Means slices each shape in half (a straight cut cannot separate a ring from its centre). DBSCAN recovers both shapes exactly because the points stay density-connected along the curve.

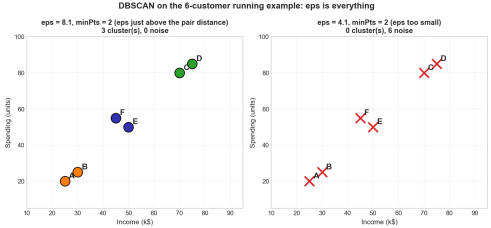
On the concentric-circles data, why does k-Means score about 0 while DBSCAN scores 1.0?

4. Choosing eps and minPts (and a Reality Check)

Pick eps from the k-distance plot. For every point measure the distance to its k-th nearest neighbour (with k = minPts), sort those distances, and read eps off the *knee* (where the curve turns sharply upward). Here the knee gives eps ≈ 0.18. **Pick minPts by a rule of thumb:** minPts $\geq$ dimensions $+ 1$; a common default is $2\times$ dimensions (so 4 in 2D, and 5 is a safe choice here).



Reality check on our 6 customers. With only 6 customers DBSCAN has almost no density to exploit. With minPts $= 2$ and eps just above the within-pair distance (≈ 8) it recovers the 3 known groups (Budget, Premium, Middle); shrink eps to ≈ 4 and all 6 customers become noise.



DBSCAN earns its keep on large, noisy, non-convex data, not on 6 tidy points.

How do you read a starting eps off the k-distance plot, and why is DBSCAN a poor fit for only 6 customers?

5. Strengths, Limits, and DBSCAN in R

Strengths

- Finds arbitrarily shaped clusters (crescents, rings).
- No need to choose k in advance.
- Labels outliers explicitly as noise.

Limits

- Struggles when clusters have very different densities.
- eps is hard to set in high dimensions (distances concentrate).
- Results are sensitive to eps and minPts.

Use it for spatial or geographic data, anomaly detection, and non-convex structure. Prefer k-Means when clusters are round, similar in size, and k is roughly known.

DBSCAN in R

```r
library(dbscan)

# X: numeric matrix, scaled
X <- scale(my_data)

# eps from the k-distance knee,
# minPts = 2 * ncol(X)
db <- dbscan(X, eps = 0.18, minPts = 5)

# cluster 0 = noise points
table(db$cluster)
```

Scale features first: eps is a single radius, so columns must share a comparable range.

Name one task where DBSCAN is the right tool and one where k-Means is the safer choice.