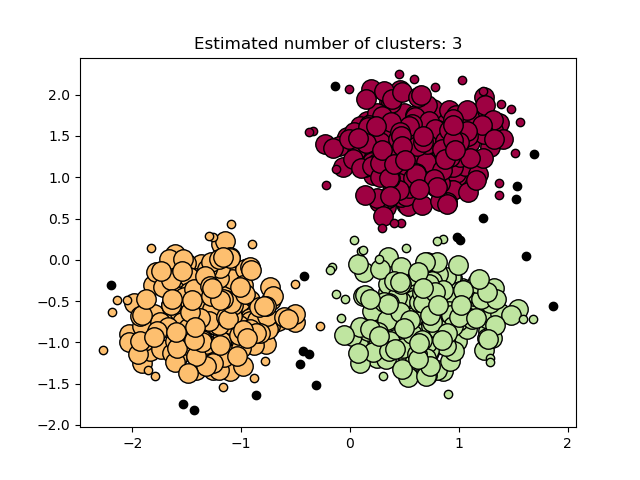
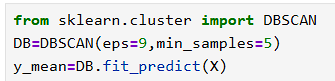
**DBSCAN:**

* DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
* It finds core samples of high density and expands clusters from them.
* Good for data which contains clusters of similar density.
* when we say dense, there are two parameter, **epa** and **min\_samples.**
* When the **eps** param is large and **min\_samples** is low, while the original DBSCAN only uses linear memory.
* There two parameters help to find the core sample is in a dense area of the vector space. A cluster is a set of core samples.
* DBSCAN is opposed to k-means which assumes that clusters are convex shaped. The central component to the DBSCAN is the concept of core samples, which have high density.
* When chosen too small, then most of data will be labelled as -1 for “noise”.
* When chosen too large it causes close clusters to be merged into one cluster, or possibly in single clusters.



****

**Pros:**

**No need to specify number of clusters**

Unlike KMeans, DBSCAN does not require pre-defining no of clusters.

**Can detect non - shaped clusters**Works well with non-spherical clusters

**Handles noise well**

Can identify and separate outliers or noise points.

**No randomness**

Produces the same results each time

**Cons:**

Sensitive to parameter selection

Choosing good values for eps and min\_samples is not always easy.

**Struggles with varying density**

Can’t cluster data well when cluster densities vary significantly.

**Performance on high-dimensional data**

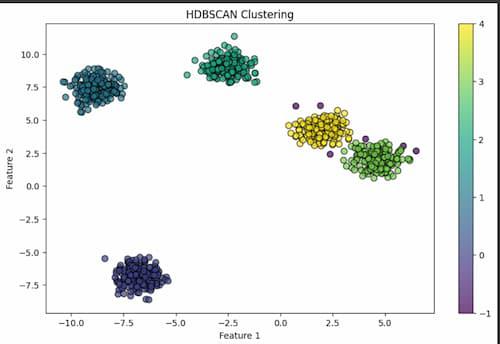
Not efficient or effective in high dimensions.

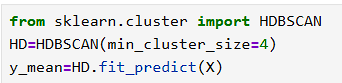
**HDBSCAN:**

* HDBSCAN - Hierarchical Density-Based Spatial Clustering of Applications with Noise.
* This algorithm can be seen as an extension of DBSCAN and OPTICS.
* DBSCAN assumes that the clustering criterion like density requirement is globally same.
* DBSCAN may struggle to successfully capture clusters with different densities.
* HDBSCAN explores all possible density scales by building an alternative representation

HDBSCAN working process points:

1. Builds a **minimum spanning tree** of the distance graph.
2. Converts it into a **hierarchy of clusters**.
3. Uses **cluster stability** to extract the best flat clustering.
4. Can label outliers as **noise**.



****

**Pros:**

**No need to specify number of clusters**

Automatically finds the optimal number of clusters.

**Handles clusters of varying densities**

Works well with data where cluster density is not uniform.

**More flexible than DBSCAN**

Doesn’t require a global density threshold like eps in DBSCAN.

**Cons:**

**Slower on large datasets**

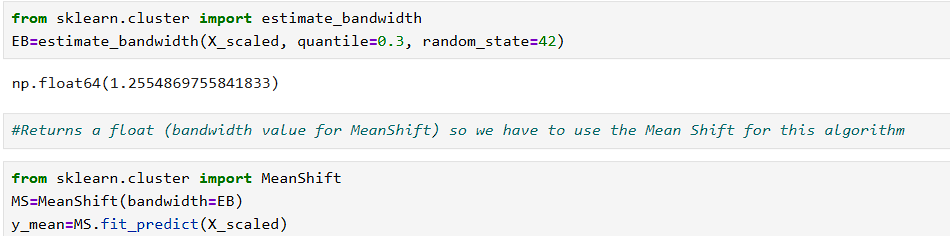
Can be computationally intensive compared to KMeans or DBSCAN.

**Harder to explain to non-technical teams**

Based on complex graph and stability concepts.

**Estimate\_bandwidth:**

* Estimate the bandwidth to use with the mean-shift algorithm.
* It’s used for find the bandwidth values, we can use the bandwidth values in MeanShift parameter.
* Used to find high-density areas (clusters).
* Too small → too many micro-clusters. Too large → meaningful clusters get merged.



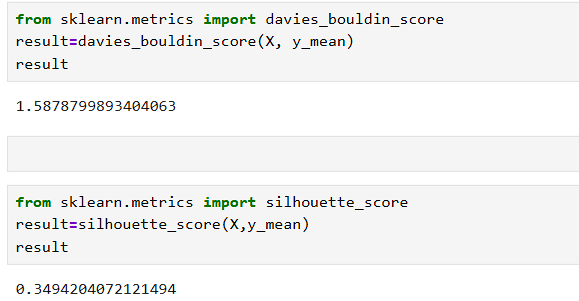
**Clustering evaluation method:**

For checking the clustering evaluation value I’m using **Davies bouldin scores and Silhouette score.**

**Davies bouldin:**

The score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances.

The minimum score is zero, with lower values indicating better clustering.



**Silhouette score:**

The Silhouette Coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample.

The best value is 1 and the worst value is -1. Values near 0 indicate overlapping clusters.

Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

