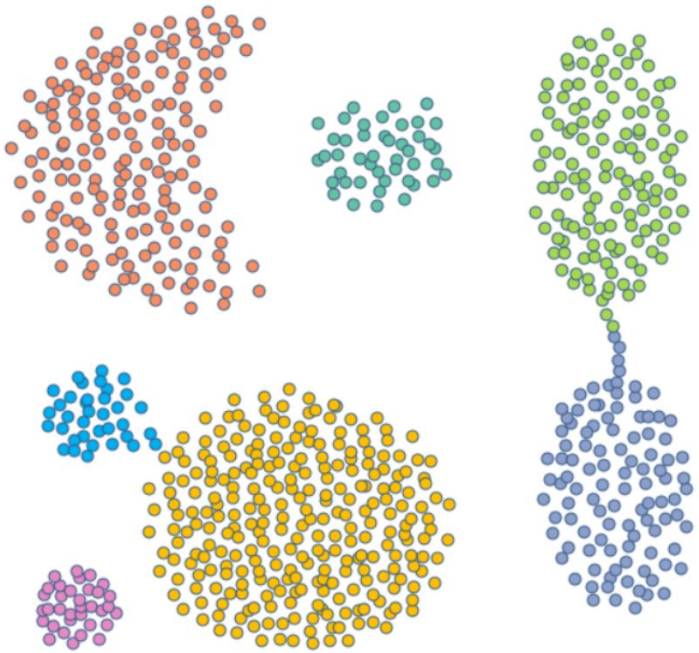
A Practical Intro
To Density Based Clustering
(E.g. some stuff that are not K-means)

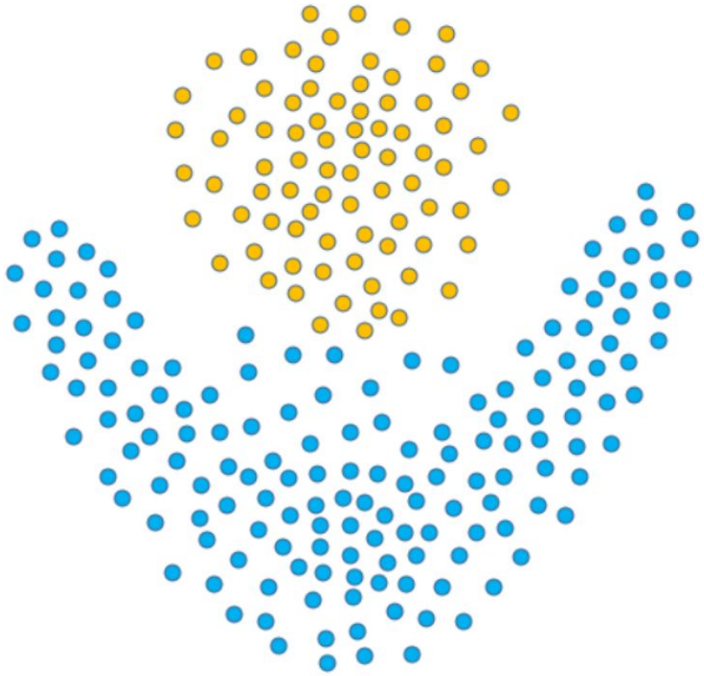
Nadav Bar

Python Noteook: https://github.com/nadavbar/density-based-clustering/blob/master/clustering_example.ipynb

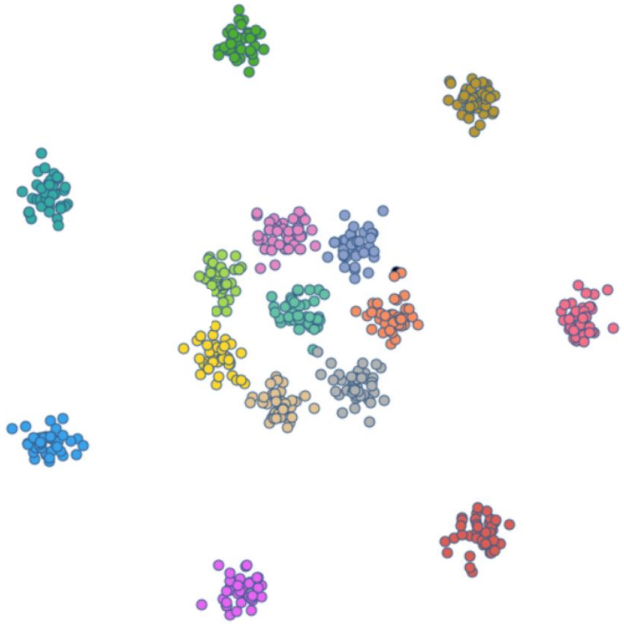
Clusters can come in many different shapes and sizes!



A. Gionis, H. Mannila, and P. Tsaparas, Clustering aggregation. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007. 1(1): p. 1-30.

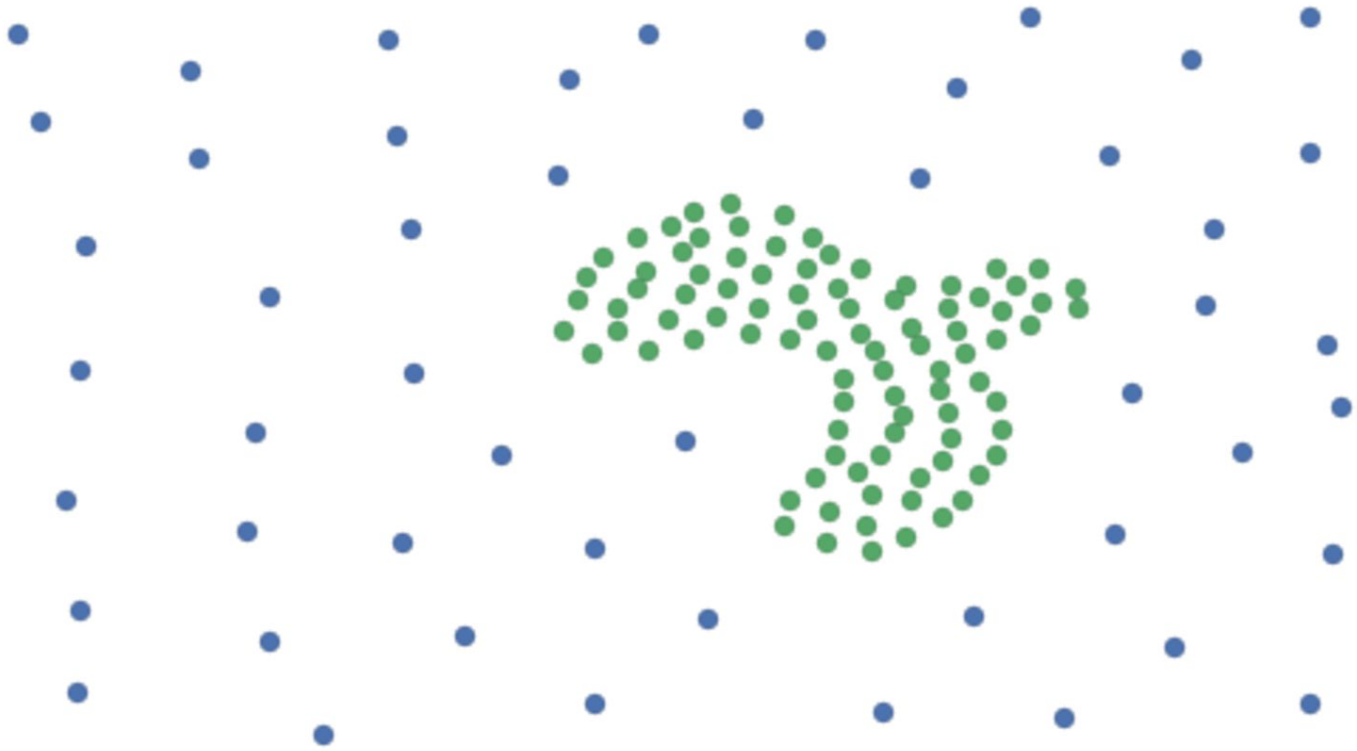


L. Fu and E. Medico, FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data. BMC bioinformatics, 2007. 8(1): p. 3.



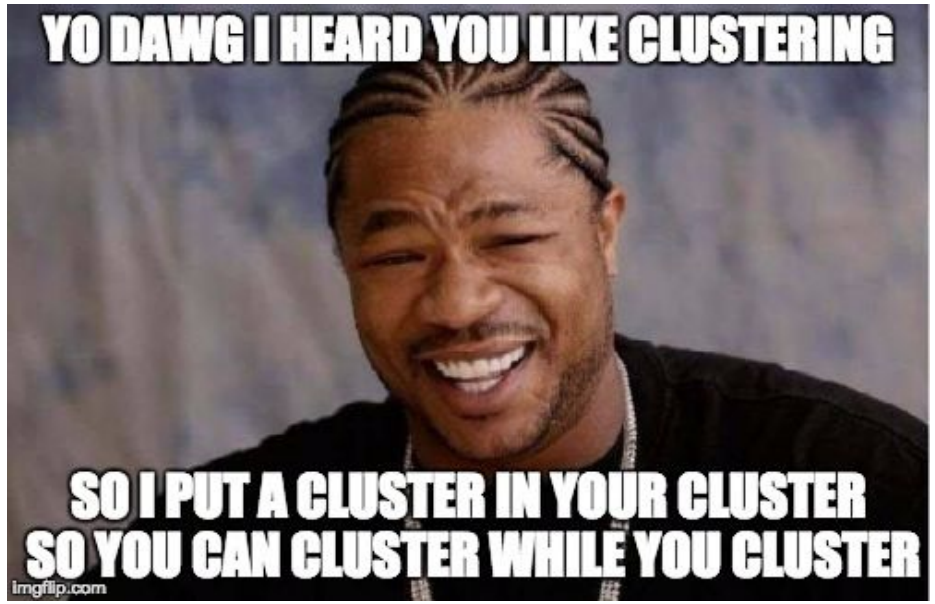
C.J. Veenman, M.J.T. Reinders, and E. Backer, A maximum variance cluster algorithm. IEEE Trans. Pattern Analysis and Machine Intelligence, 2002. 24(9): p. 1273-1280.

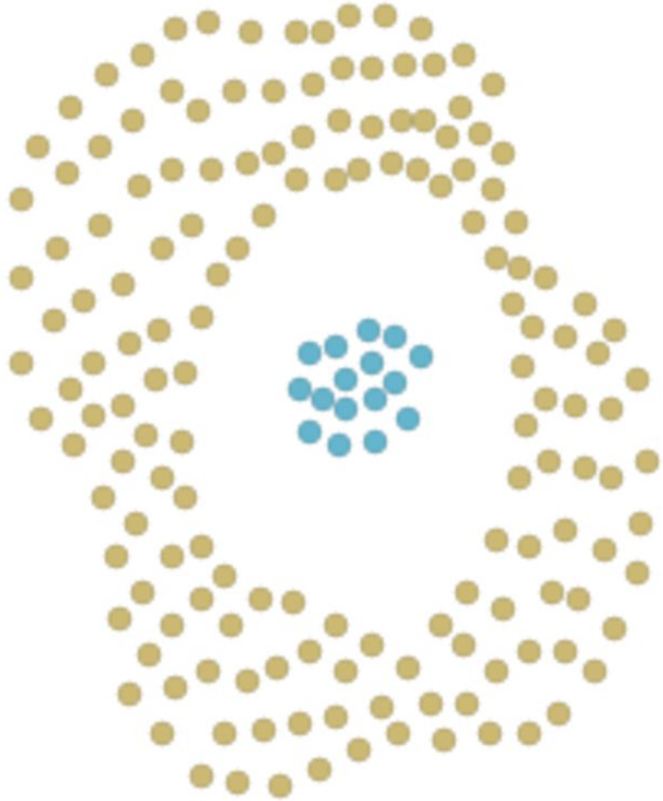
Sometimes, the clusters are surrounded by noise ☹



C.T. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 1971. 100(1): p. 68-86.

Oh yes, and what about those times when you have clusters inside other clusters???





C.T. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 1971. 100(1): p. 68-86.

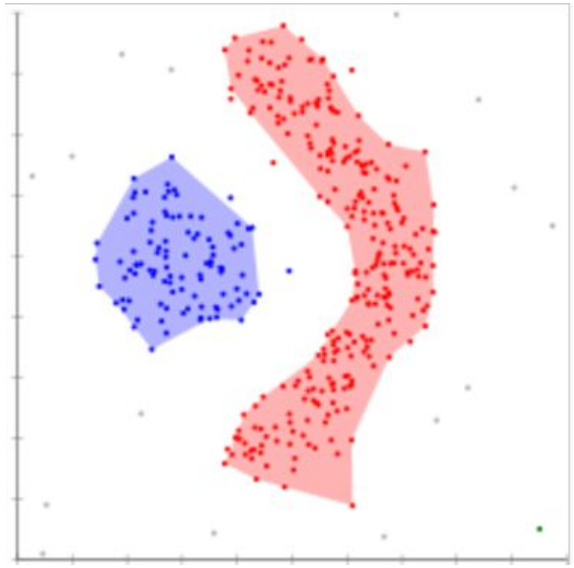
K?

Density based clustering- motivation

- GOTO: Python notebook

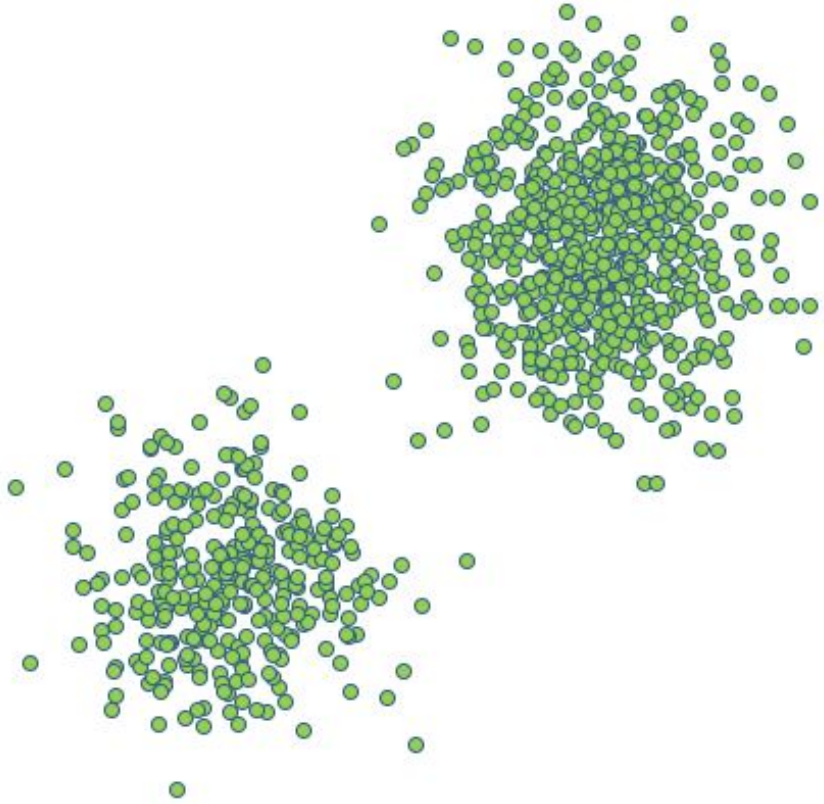
Density based clustering

- Automatically discover the number of clusters
- Clusters are defined as areas of higher density than the remainder of the data set
- (Some methods) have notion of noise: objects in sparse areas are usually considered to be noise and border points
- Can find clusters of arbitrary shapes and sizes

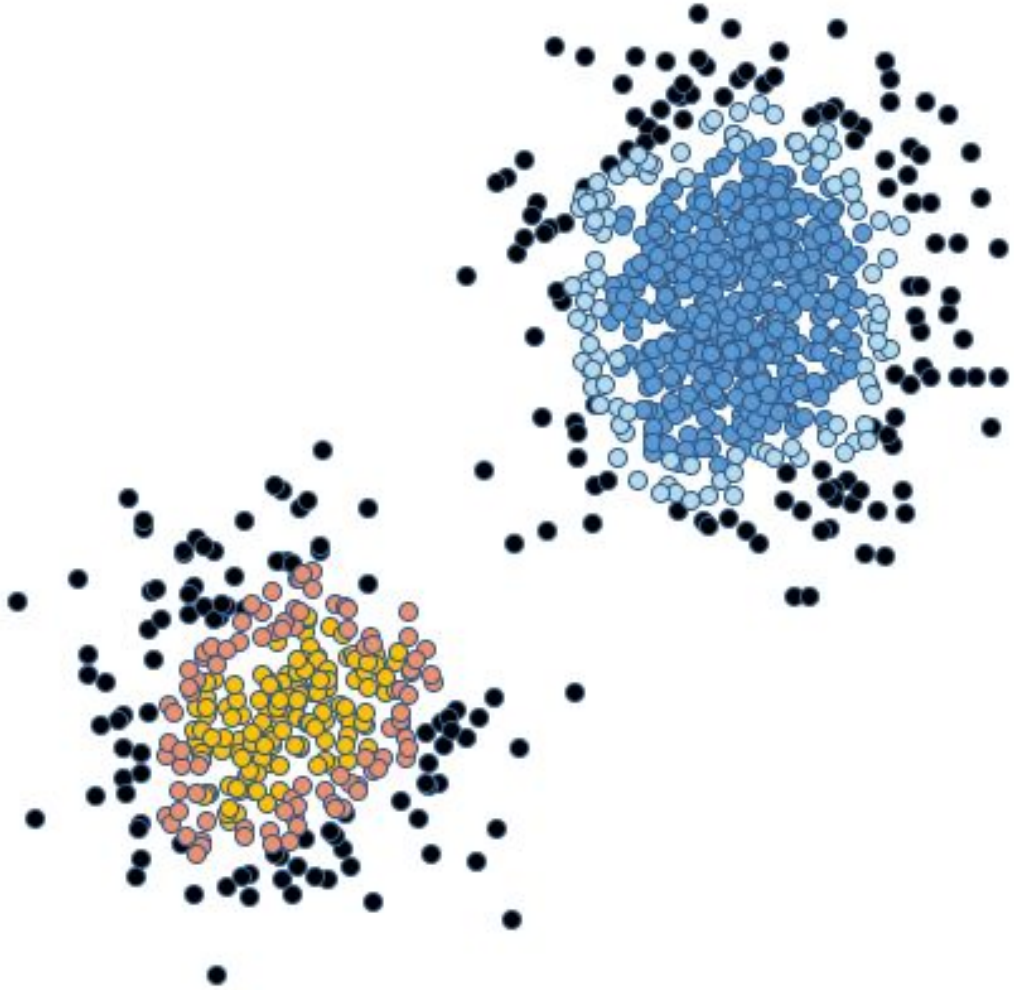


https://en.wikipedia.org/wiki/DBSCAN#/media/File:DBSCAN-density-data.svg

DBSCAN (Ester et al. 1996):
"Density-based spatial clustering of applications with noise"

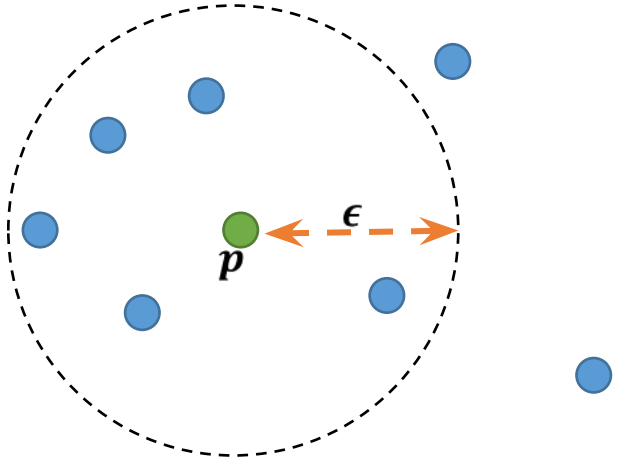


DBSCAN: Core and non-core points



DBSCAN *(Density-based spatial clustering of applications with noise)*

- Input parameters – **minPts** (integer), ϵ (distance)
- **Core points** – A point in is a core point if there are at least **minPts** points with distance which is less than ϵ from it:

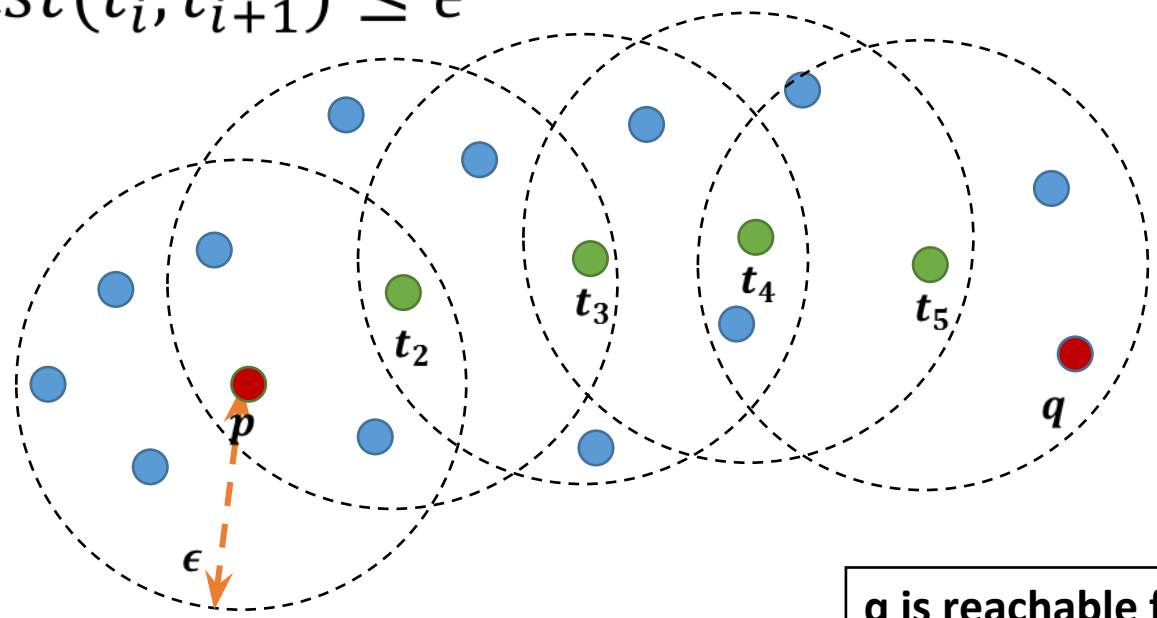


For example, point p here is a core point for **minPts<=5**

DBSCAN *(Density-based spatial clustering of applications with noise)*

The main idea: connect core points that are in distance $\leq \epsilon$ from each other + their neighborhoods into clusters.

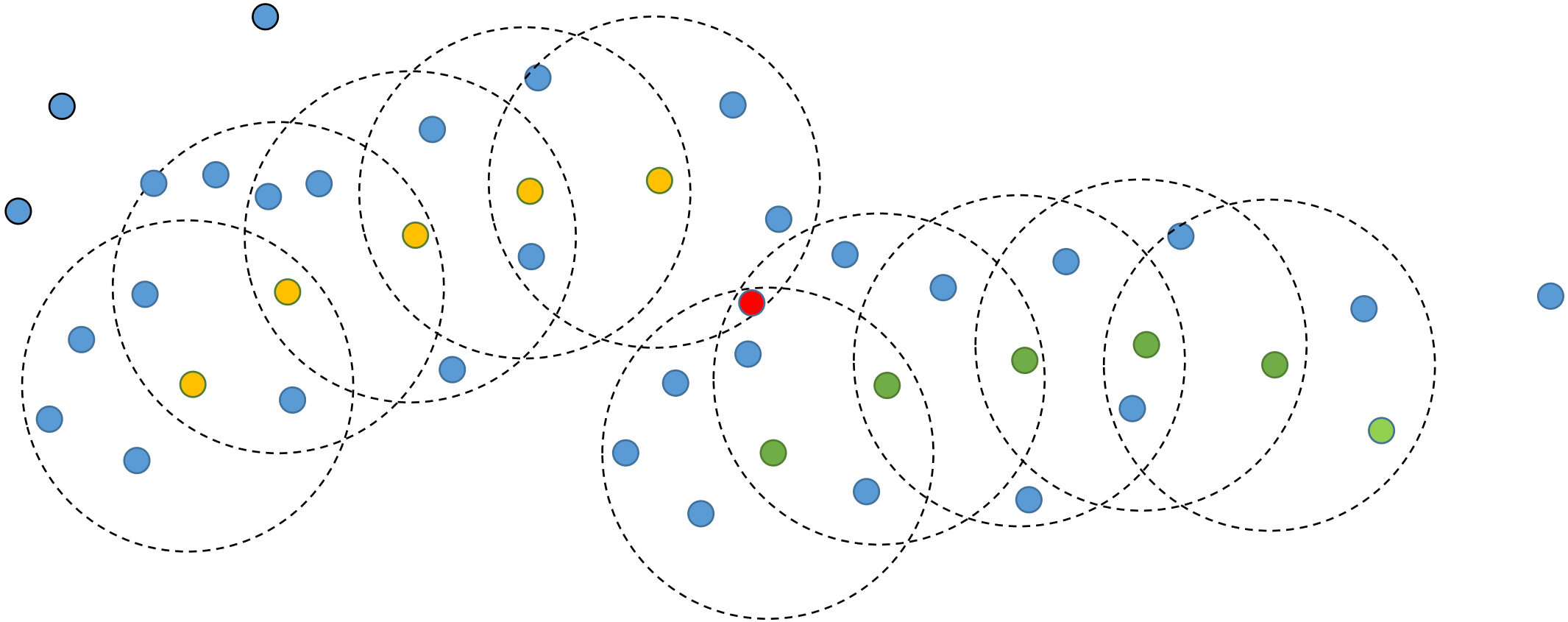
We say that a point **q** is **<u>reachable</u>** from point **p** if there is a series of core points $t_1, t_2, \ldots t_n$ and $t_1 = p, and\ dist(q, t_n) \leq \epsilon$, and also for every *i it holds that:* $dist(t_i, t_{i+1}) \leq \epsilon$



q is reachable from p

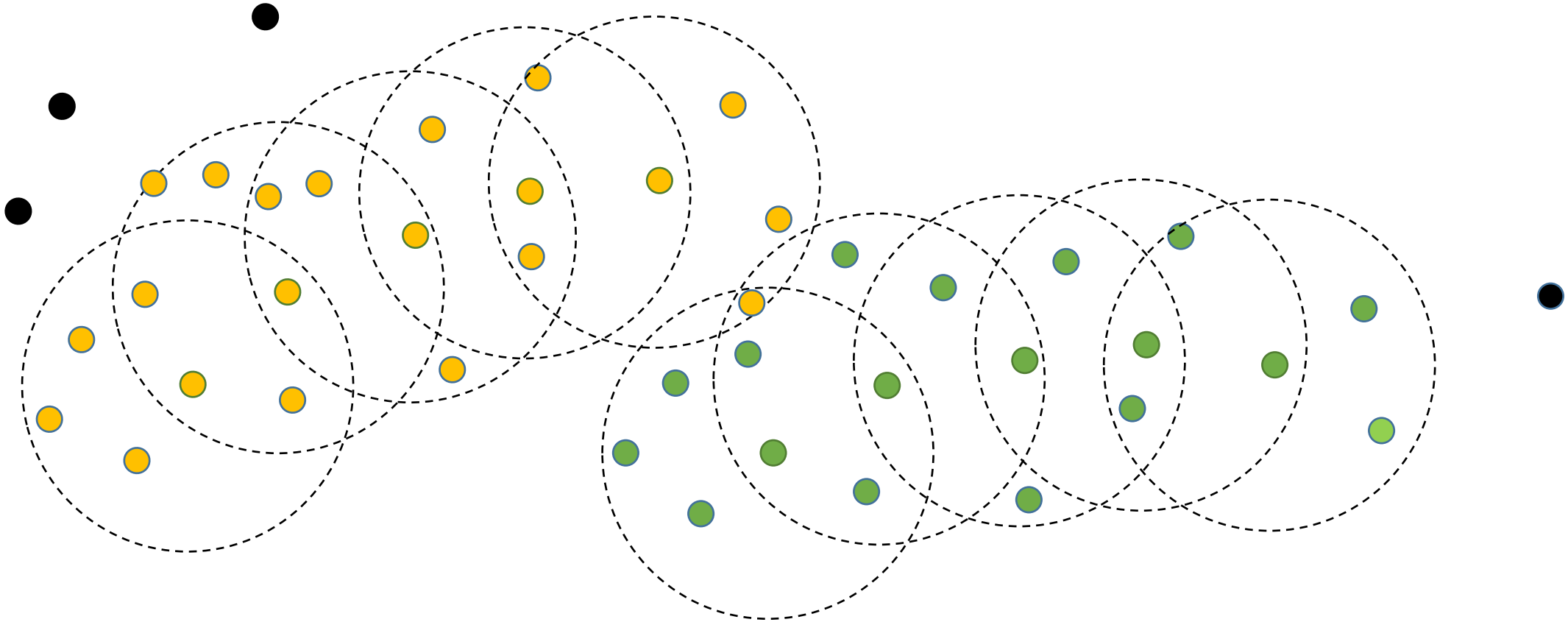
DBSCAN *(Density-based spatial clustering of applications with noise)*

A core point, p - form a cluster with all points that are reachable from it. And this cluster will contain all of the points that are reachable from the core points that are reachable from p, and so on..



DBSCAN *(Density-based spatial clustering of applications with noise)*

A core point, p - form a cluster with all points that are reachable from it. And this cluster will contain all of the points that are reachable from the core points that are reachable from p, and so on..



```
DBSCAN(D, eps, MinPts) {
   C = 0
   for each point P in dataset D {
      if P is visited
         continue next point
      mark P as visited
      NeighborPts = regionQuery(P, eps)
      if sizeof(NeighborPts) < MinPts
         mark P as NOISE
      else {
         C = next cluster
         expandCluster(P, NeighborPts, C, eps, MinPts)
      }
   }
}

expandCluster(P, NeighborPts, C, eps, MinPts) {
   add P to cluster C
   for each point P' in NeighborPts {
      if P' is not visited {
         mark P' as visited
         NeighborPts' = regionQuery(P', eps)
         if sizeof(NeighborPts') >= MinPts
            NeighborPts = NeighborPts joined with NeighborPts'
      }
      if P' is not yet member of any cluster
         add P' to cluster C
   }
}

regionQuery(P, eps)
   return all points within P's eps-neighborhood (including P)
```

https://en.wikipedia.org/wiki/DBSCAN

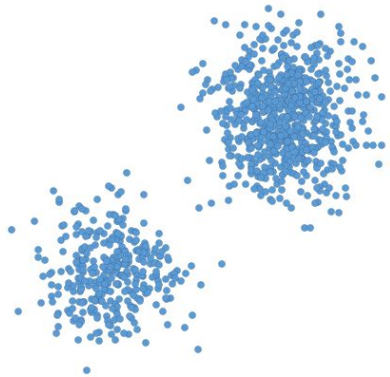
DBSCAN’s parameters

Somewhat sensitive to values of minPts and ϵ.
How to choose good value for ϵ?

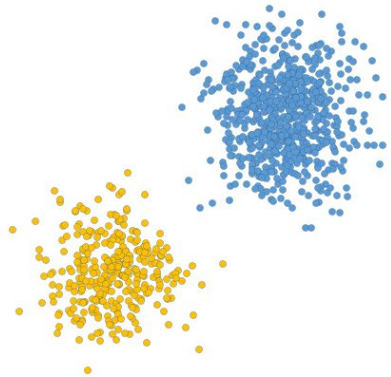
minPts=5, ϵ=0.6

minPts=,5 ϵ=0.5

minPts=15, ϵ=0.7



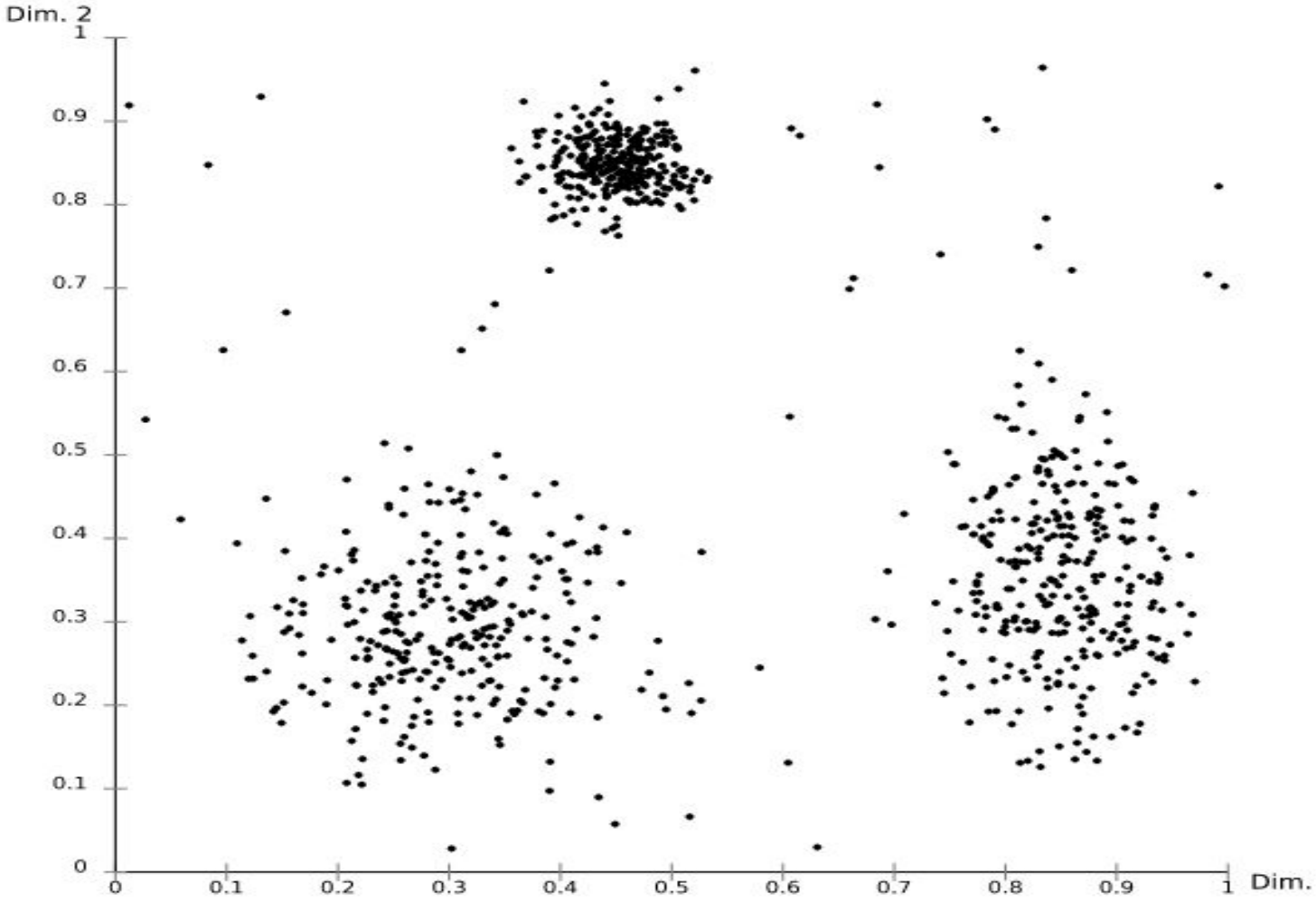
minPts=15, ϵ=0.6



DBSCAN clustering example

- GOTO: python notebook

The choice of a single ϵ and **minPts** values imply uniform density levels between the clusters ➔ DBSCAN Does not deal well with clusters of various densities.



https://upload.wikimedia.org/wikipedia/commons/f/f9/OPTICS.svg

OPTICS (Ankerst et al. 1999): *"Ordring points to identify clustering structure"*

- Similar to DBSCAN, input parameters are still ϵ and minPts, but outputs an ordered list of points instead of clustering.
- Introduces a notion of "reachability distance" between every two points:

$$\text{reachability-dist}_{\varepsilon,MinPts}(o,p) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(p)| < MinPts \\ \max(\text{core-dist}_{\varepsilon,MinPts}(p), \text{dist}(p,o)) & \text{otherwise} \end{cases}$$

- Where:

$$\text{core-dist}_{\varepsilon,MinPts}(p) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(p)| < MinPts \\ MinPts\text{-th smallest distance to } N_\varepsilon(p) & \text{otherwise} \end{cases}$$

- core-dist – The distance threshold for which p is considered a core point

Example: core-dist and reachability-dist

In this example:
$\mathbf{minPts} = 5$, and $\boldsymbol{\epsilon} \geq \boldsymbol{dist}(\boldsymbol{p}, \boldsymbol{q})$

Hence:

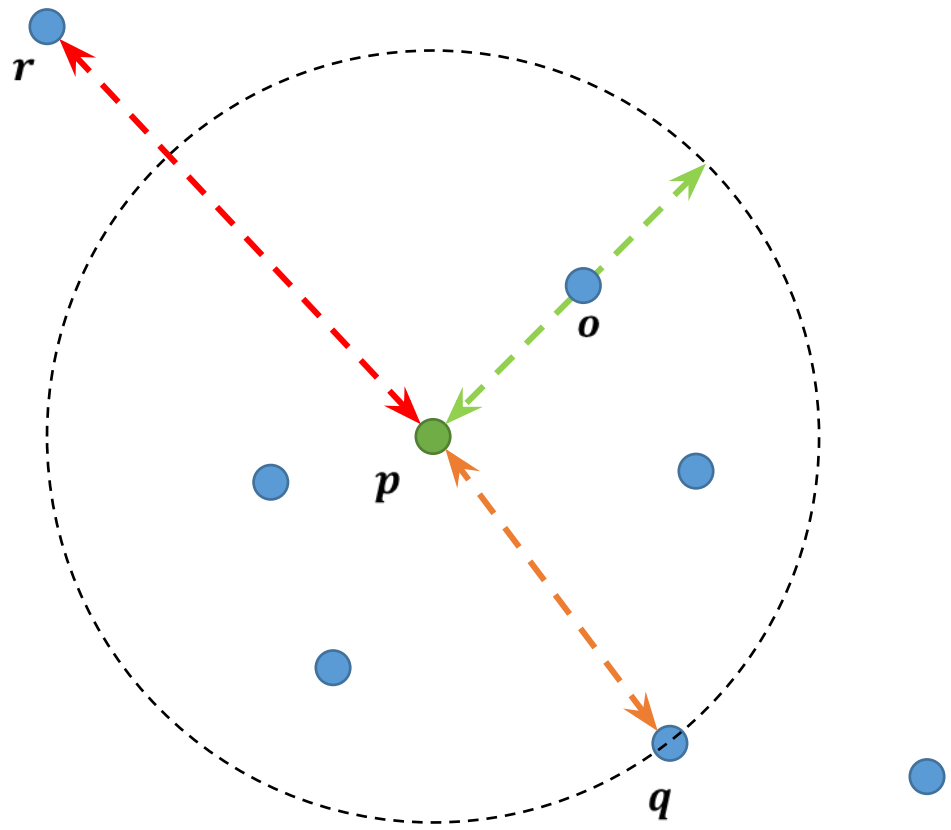
$$\boldsymbol{core} - \boldsymbol{dist}_{\epsilon.minPts}(\mathbf{p}) = \mathbf{dist}(\mathbf{p}, \mathbf{q})$$

And:

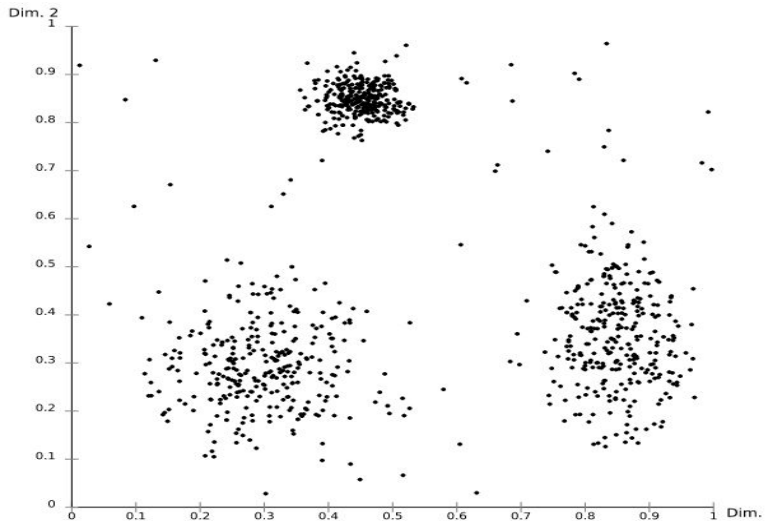
$$\boldsymbol{reachability} - \boldsymbol{dist}_{\epsilon.minPts}(\mathbf{o}, \mathbf{p}) = \boldsymbol{core} - \boldsymbol{dist}_{\epsilon.minPts}(\mathbf{p})$$

But:

$$\boldsymbol{reachability} - \boldsymbol{dist}_{\epsilon.minPts}(\mathbf{r}, \mathbf{p}) = \boldsymbol{dist}(\boldsymbol{p}, \boldsymbol{r})$$



OPTICS algorithm output

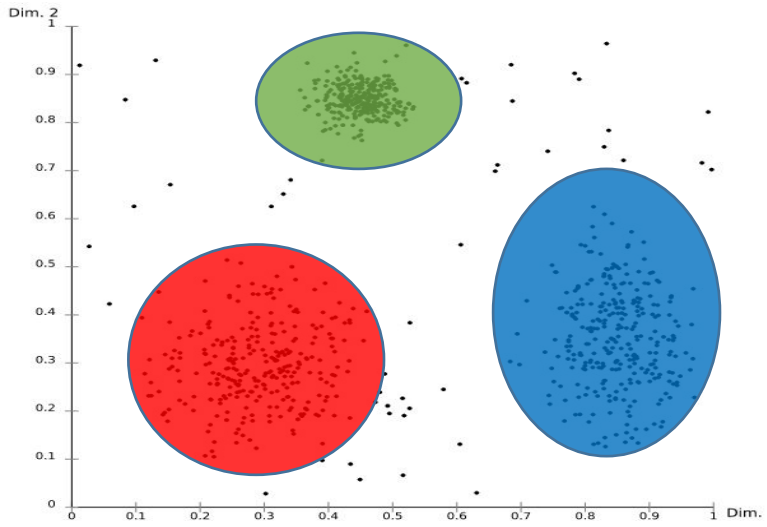


Minimum reachability-dist for a point

Points, ordered according to their order in the ordered-list

https://upload.wikimedia.org/wikipedia/commons/f/f9/OPTICS.svg

OPTICS algorithm output



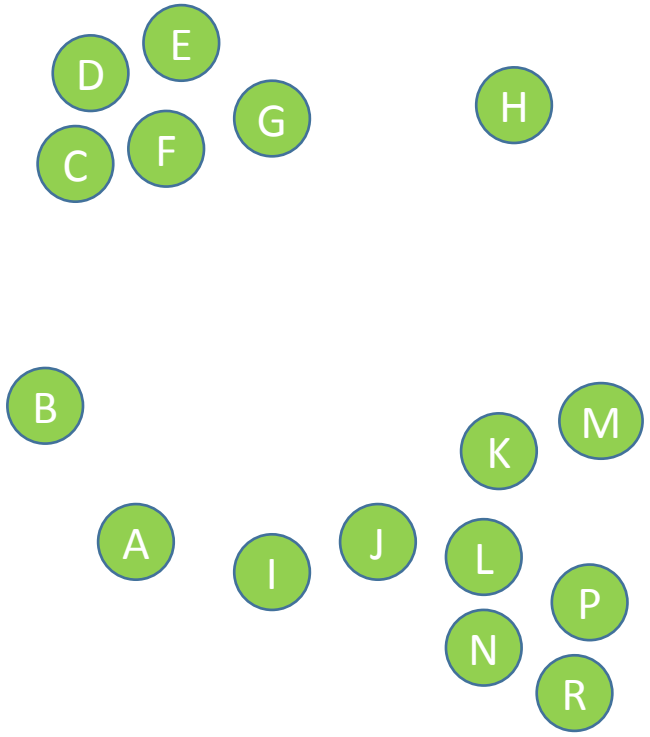
Minimum reachability-dist for a point

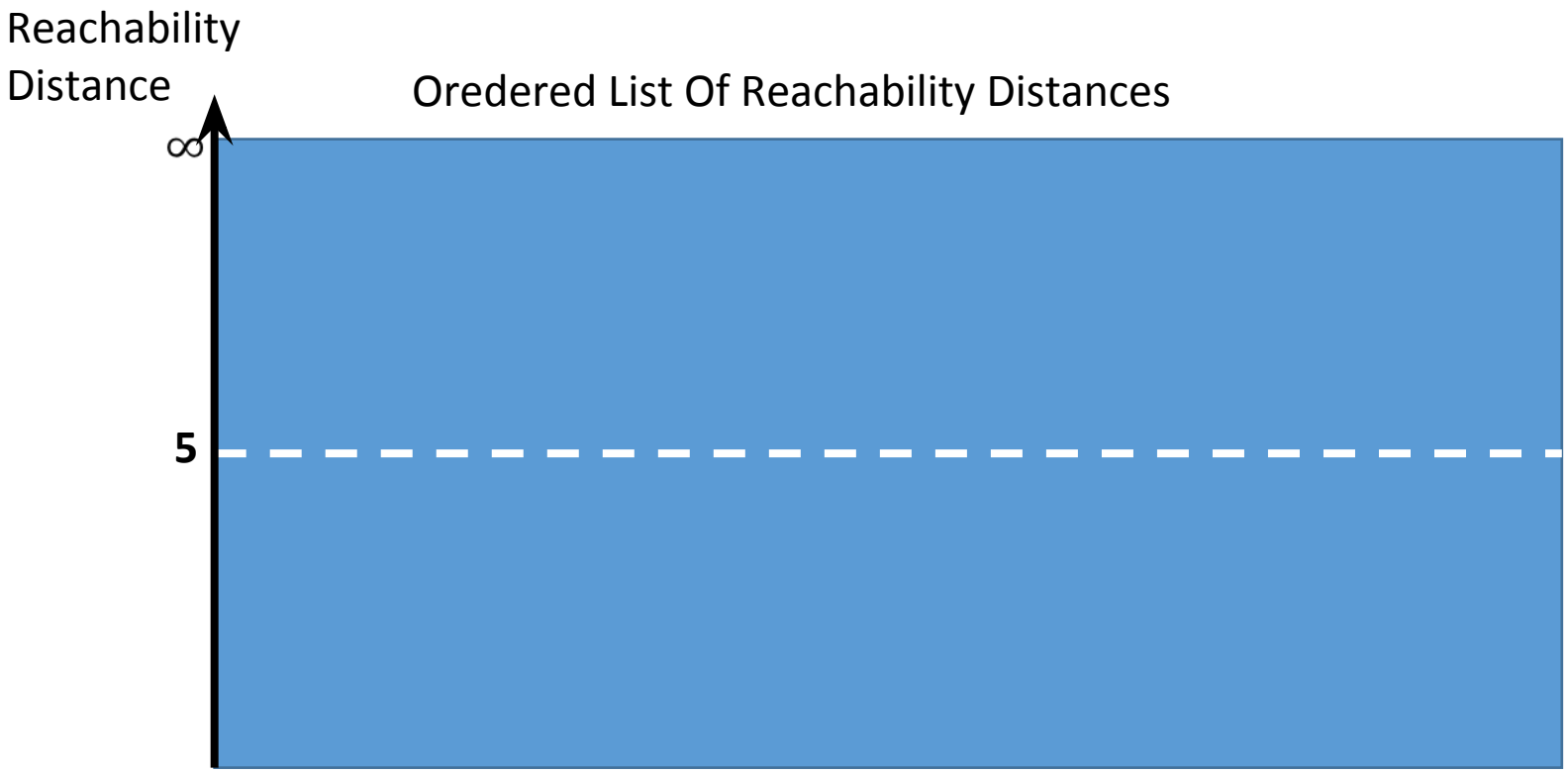
Points, ordered according to their order in the ordered-list

https://upload.wikimedia.org/wikipedia/commons/f/f9/OPTICS.svg

OPTICS RUN Example

Parameters: minPts=2, ϵ=5



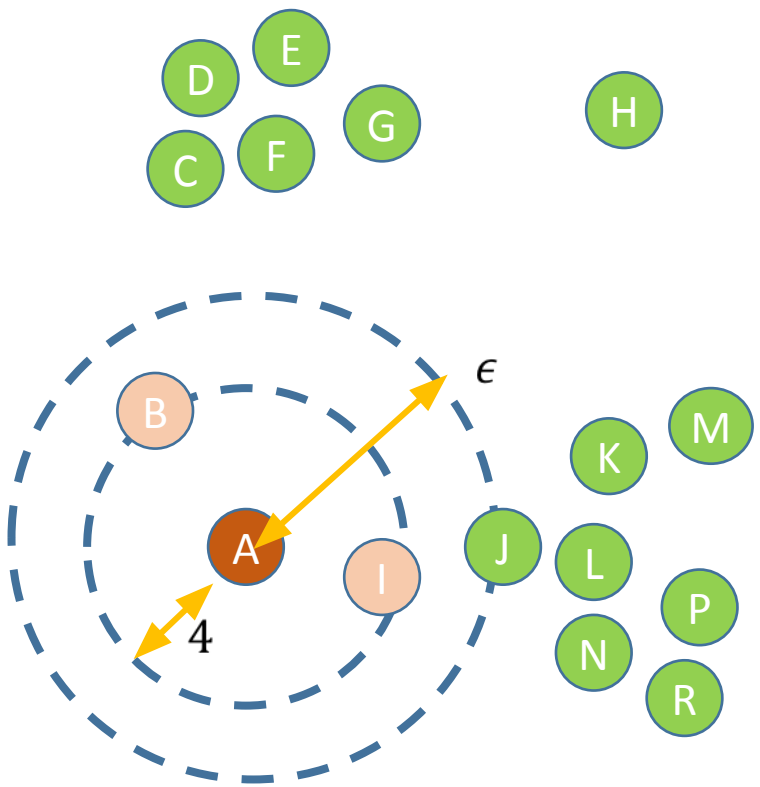


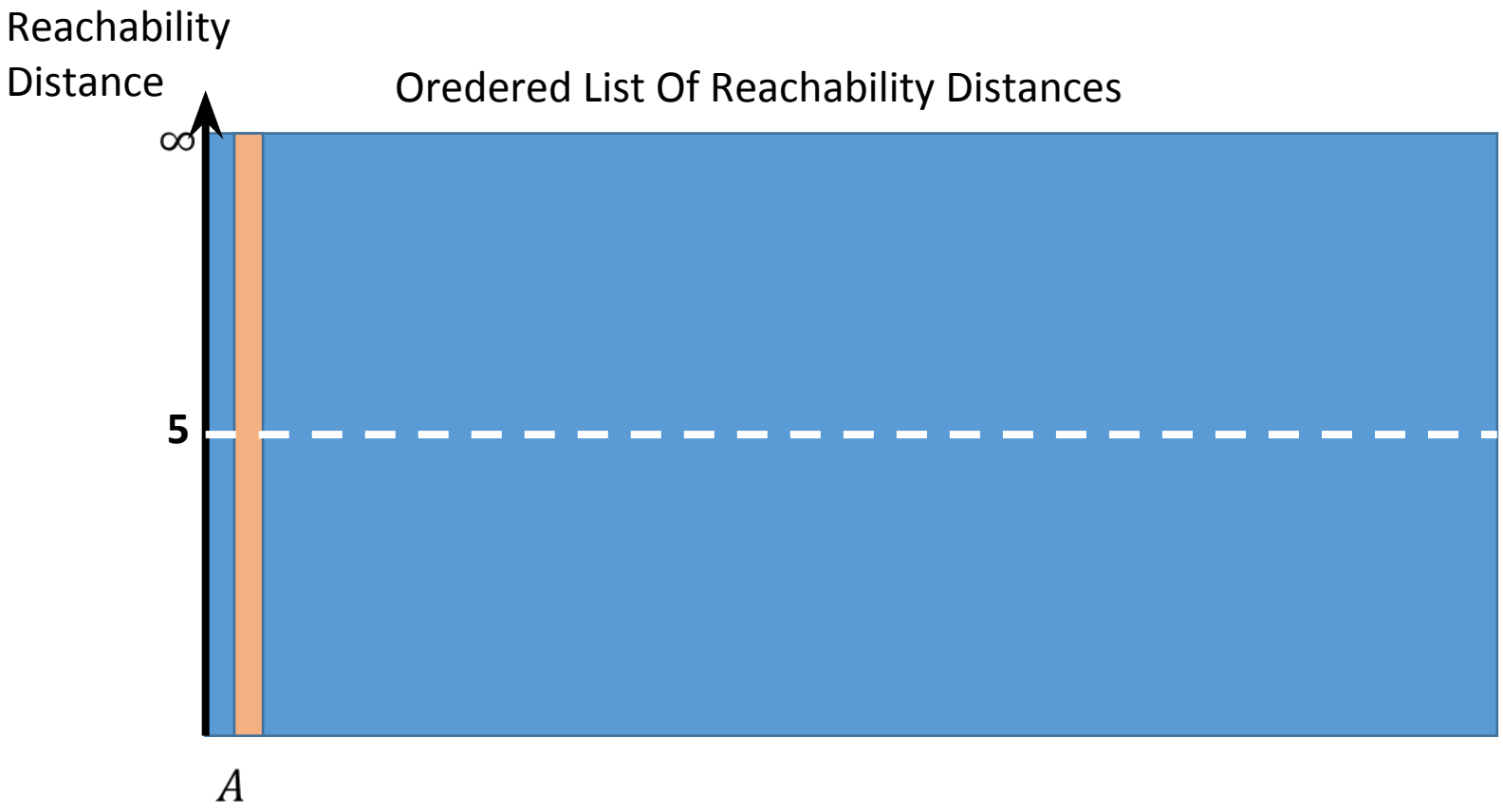
Priority Queue:

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5



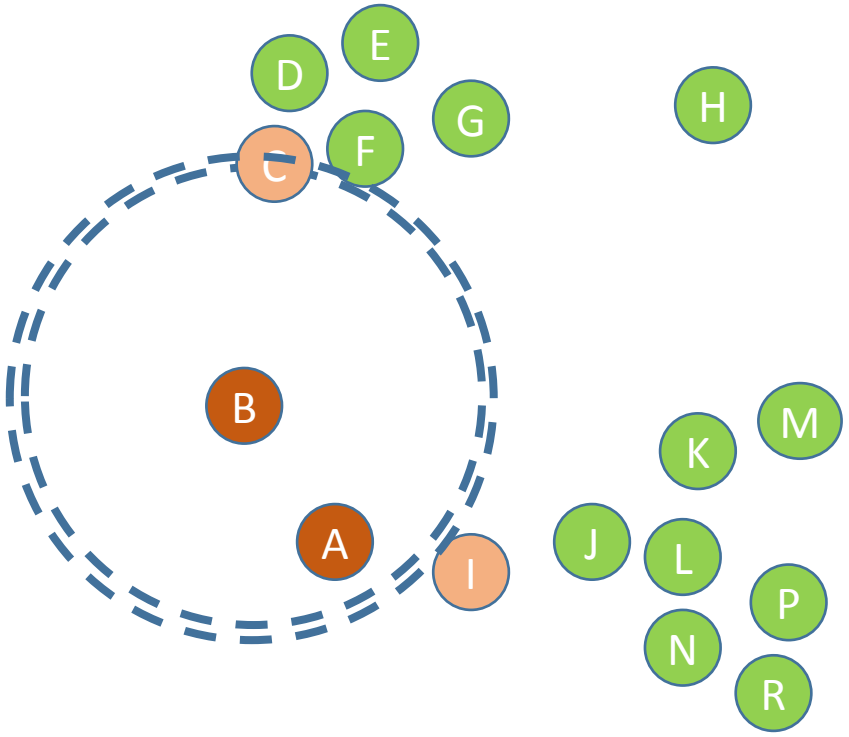


Priority Queue: (B,4), (I,4)

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5

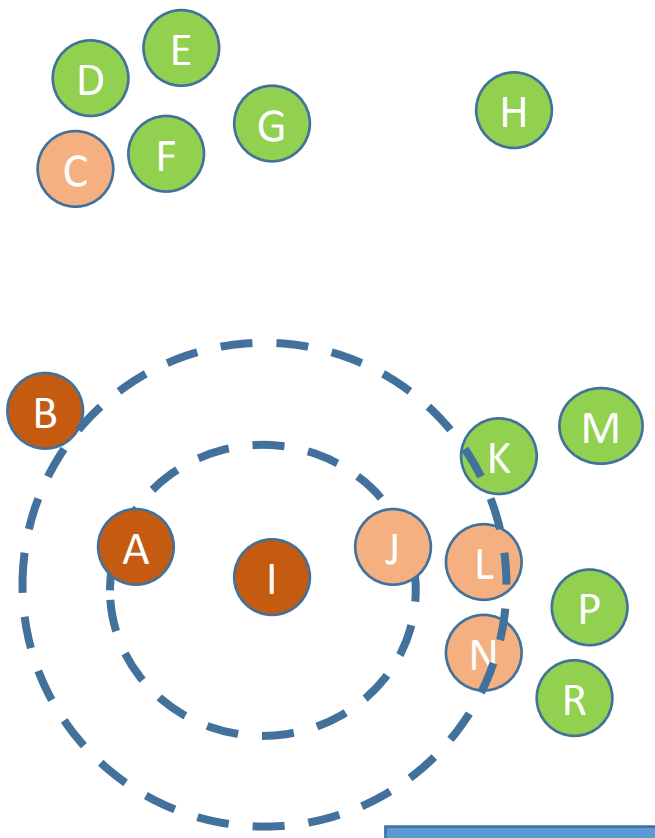


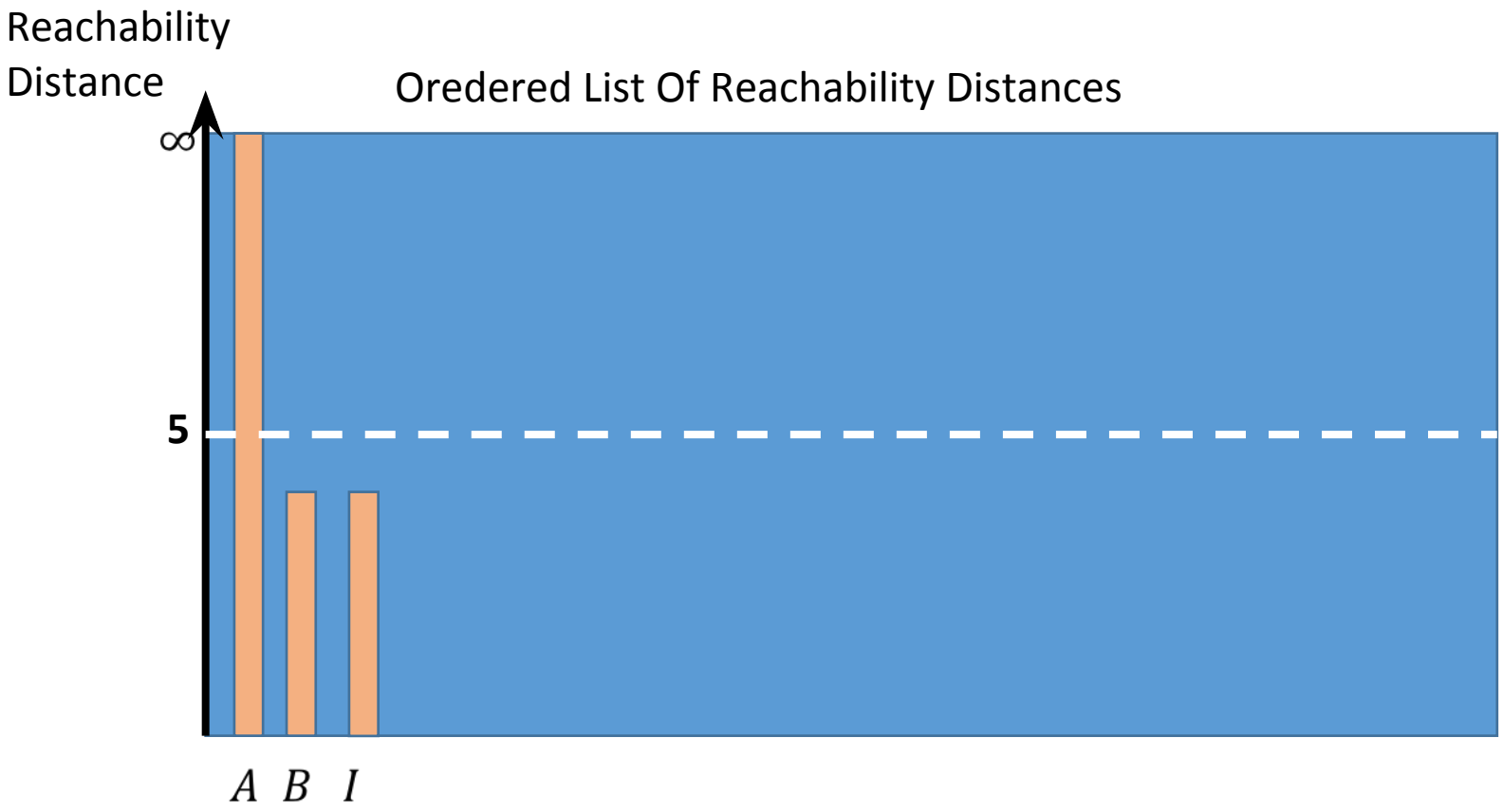
Priority Queue: (I,4), (C,4.8)

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5



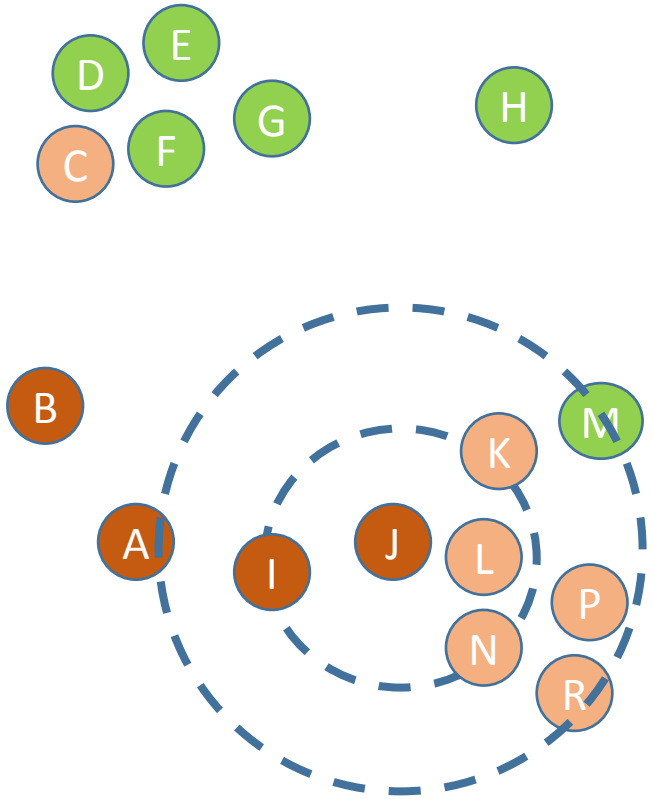


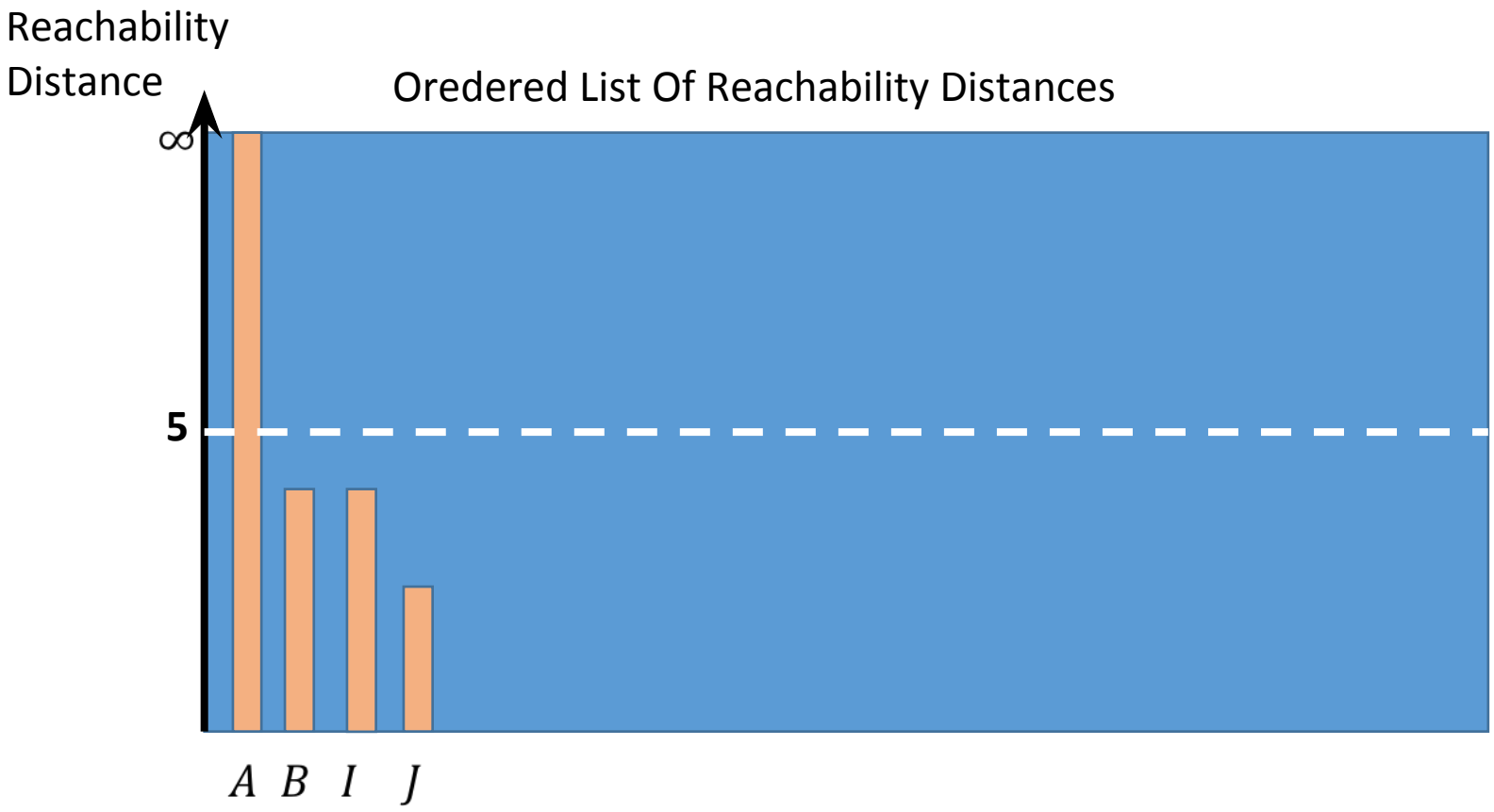
Priority Queue: (J, 2.5), (L,4), (N,4.1), (C,4.8)

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5



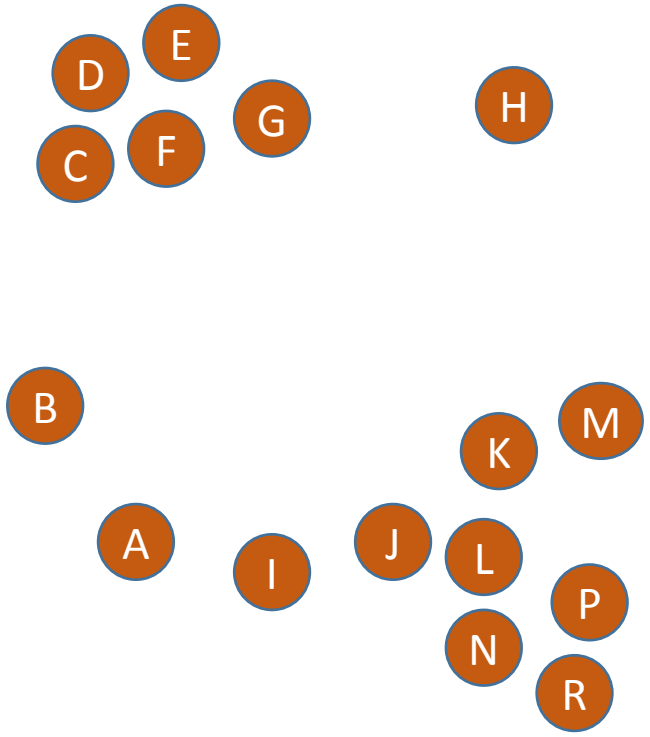


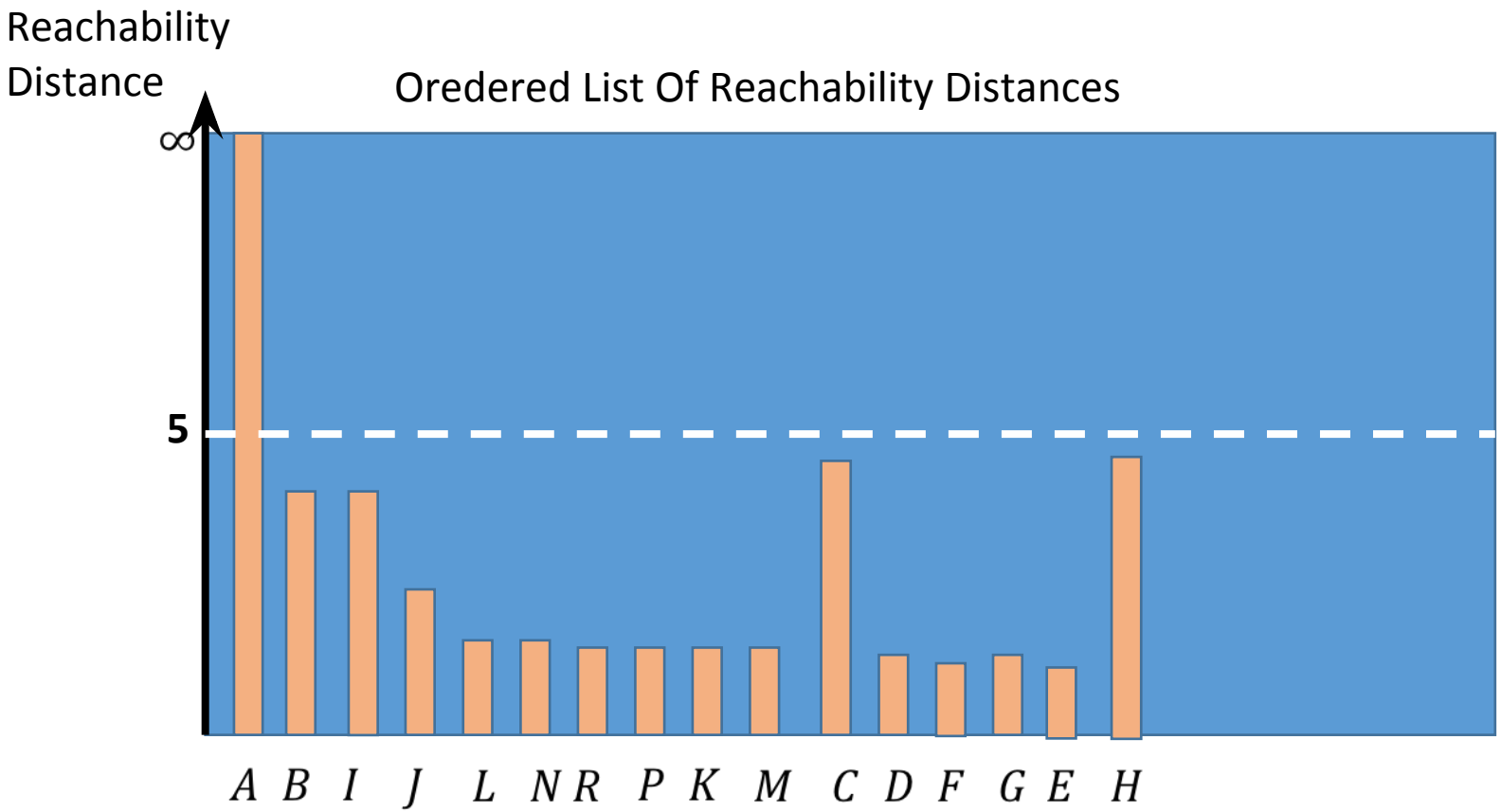
Priority Queue: (L,2.1), (N,2.1), (K,2.2), (R, 3), (M,3.2), (C,4.8)

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5



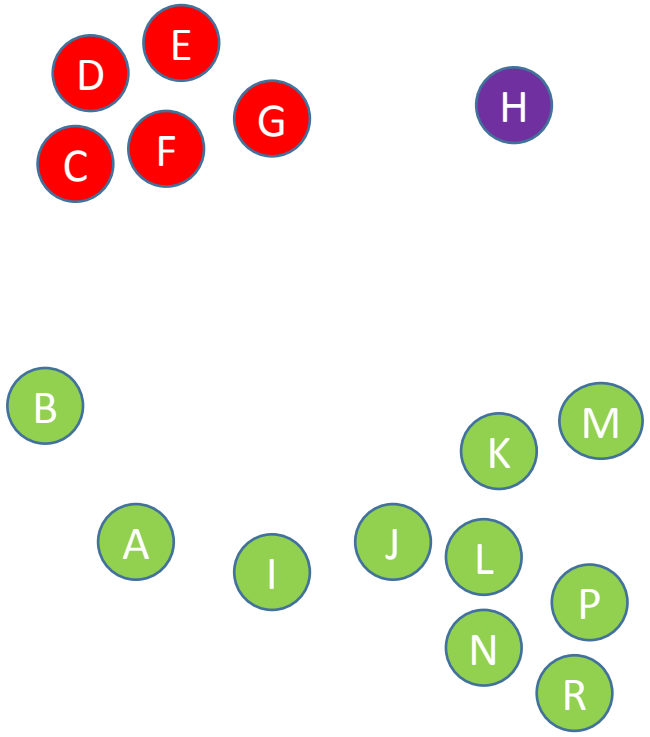


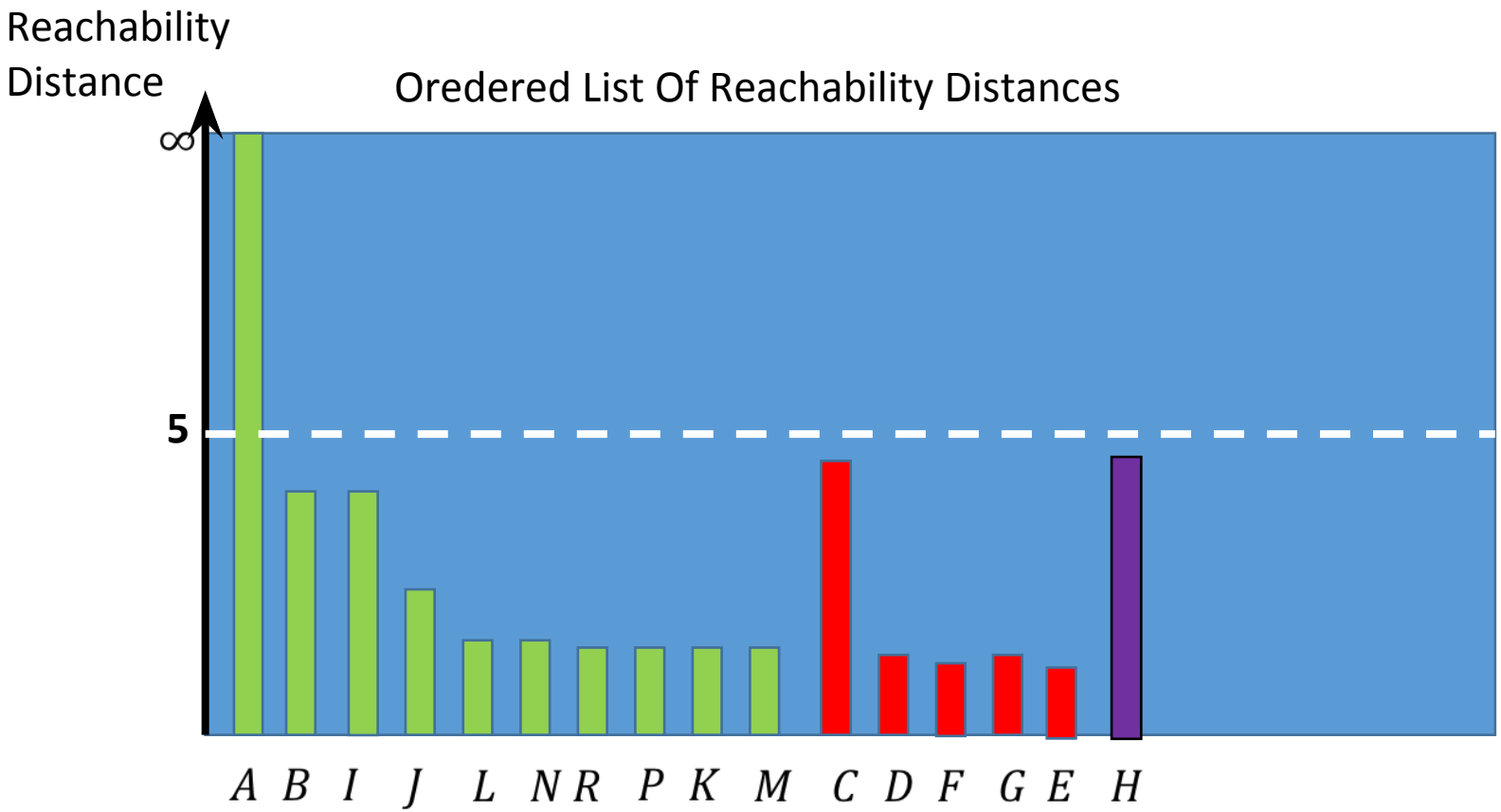
Priority Queue:

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS RUN Example

Parameters: minPts=2, ϵ=5





Priority Queue:

Example adapted from: https://www.slideshare.net/rpiryani/optics-ordering-points-to-identify-the-clustering-structure

OPTICS algorithm

```
OPTICS(DB, eps, MinPts)
   for each point p of DB
      p.reachability-distance = UNDEFINED
   for each unprocessed point p of DB
      N = getNeighbors(p, eps)
      mark p as processed
      output p to the ordered list
      if (core-distance(p, eps, Minpts) != UNDEFINED)
         Seeds = empty priority queue
         update(N, p, Seeds, eps, Minpts)
         for each next q in Seeds
            N' = getNeighbors(q, eps)
            mark q as processed
            output q to the ordered list
            if (core-distance(q, eps, Minpts) != UNDEFINED)
               update(N', q, Seeds, eps, Minpts)
```

https://en.wikipedia.org/wiki/OPTICS_algorithm

OPTICS algorithm

```
update(N, p, Seeds, eps, Minpts)
   coredist = core-distance(p, eps, MinPts)
   for each o in N
      if (o is not processed)
         new-reach-dist = max(coredist, dist(p,o))
         if (o.reachability-distance == UNDEFINED) // o is not in Seeds
             o.reachability-distance = new-reach-dist
             Seeds.insert(o, new-reach-dist)
         else                   // o in Seeds, check for improvement
             if (new-reach-dist < o.reachability-distance)
                o.reachability-distance = new-reach-dist
                Seeds.move-up(o, new-reach-dist)
```

https://en.wikipedia.org/wiki/OPTICS_algorithm

OPTICS clustering example

- GOTO: Python notebook

OPTICS algorithm

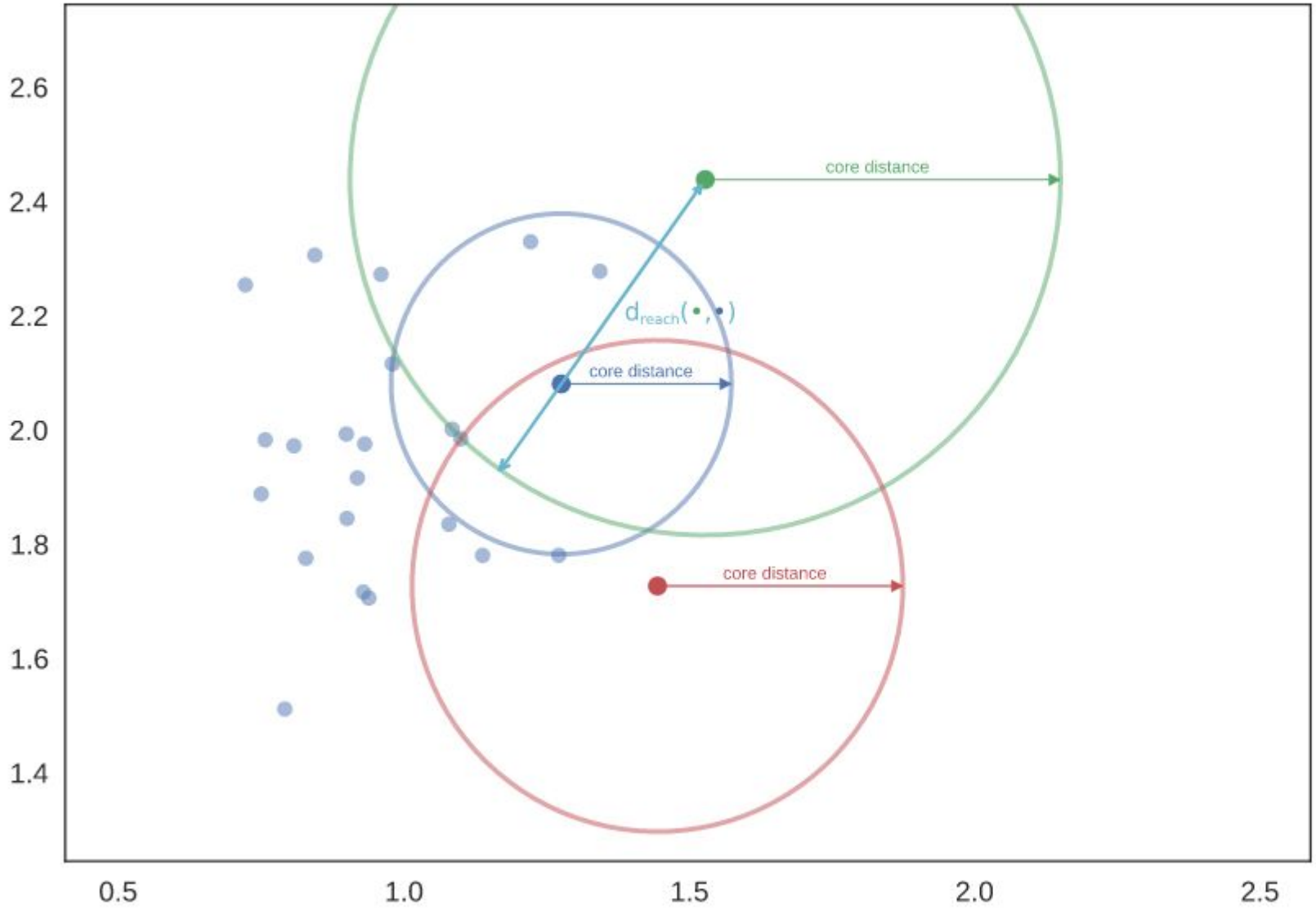
- The value of ϵ is less critical than in DBSCAN, and serves as a “threshold” for the maximum possible inter-cluster distance.
- The output is not limited to one global parameter setting (like DBSCAN), but can be viewed as containing information which corresponds to several different parameter values.
- Clusters extraction requires manual interaction, or a an automatic heuristic that might result to similar issues as with DBSCAN

HDBSCAN – "Density-Based Clustering Based on Hierarchical Density Estimates" (Campello et al)

- Input parameters are **minPts** and **minClusterSize** "minimum cluster size", but there is no ϵ here(!) Outputs a clustering of the data set.
- Often, **minClusterSize=minPts** is used, resulting with only one input parameter for the algorithm.
- Can be seen as (somewhat of) a combination of DBSCAN with single linkage hierarchical clustering.
- Following OPTICS, it defines "mutual reachability distance" between two points:

$$\boldsymbol{mutual-reach-dist_{minPts}(p,o)}$$
$$= \mathbf{max}(\boldsymbol{core-dist_{minPts}(p)}, \boldsymbol{core-dist_{minPts}(o)}, \boldsymbol{dist(p,o)})$$

Mutual reachability distance – example (minPts=5)

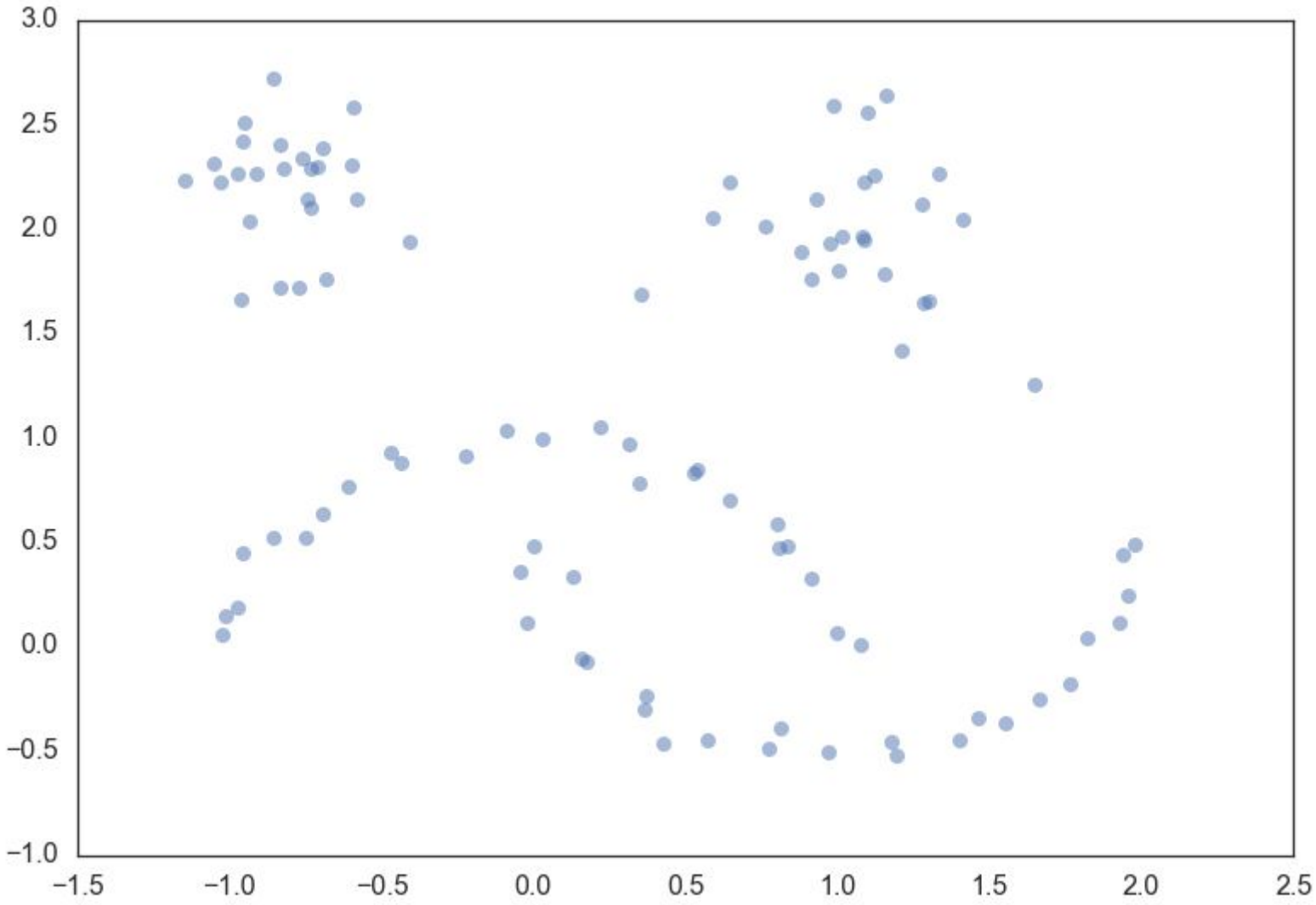


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Mutual reachability distances MST

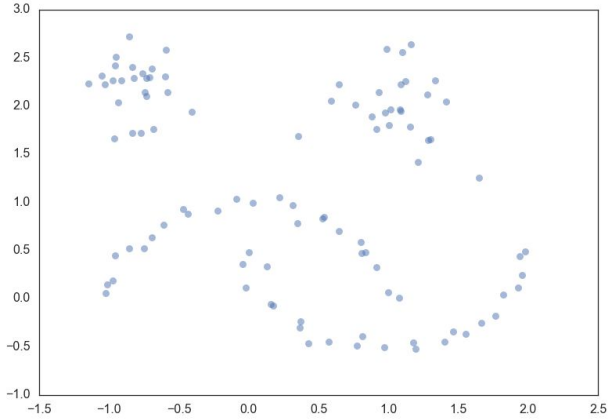
- Consider an undirected weighted graph where each data point is a node in the graph.
- The weight of each edge in the graph is the mutual reachability distance of it’s nodes.
- HDBSCAN constructs the Minimum Spanning Tree (MST) of that graph.
- The MST will be used to build a hierarchy of connected components according to the density.

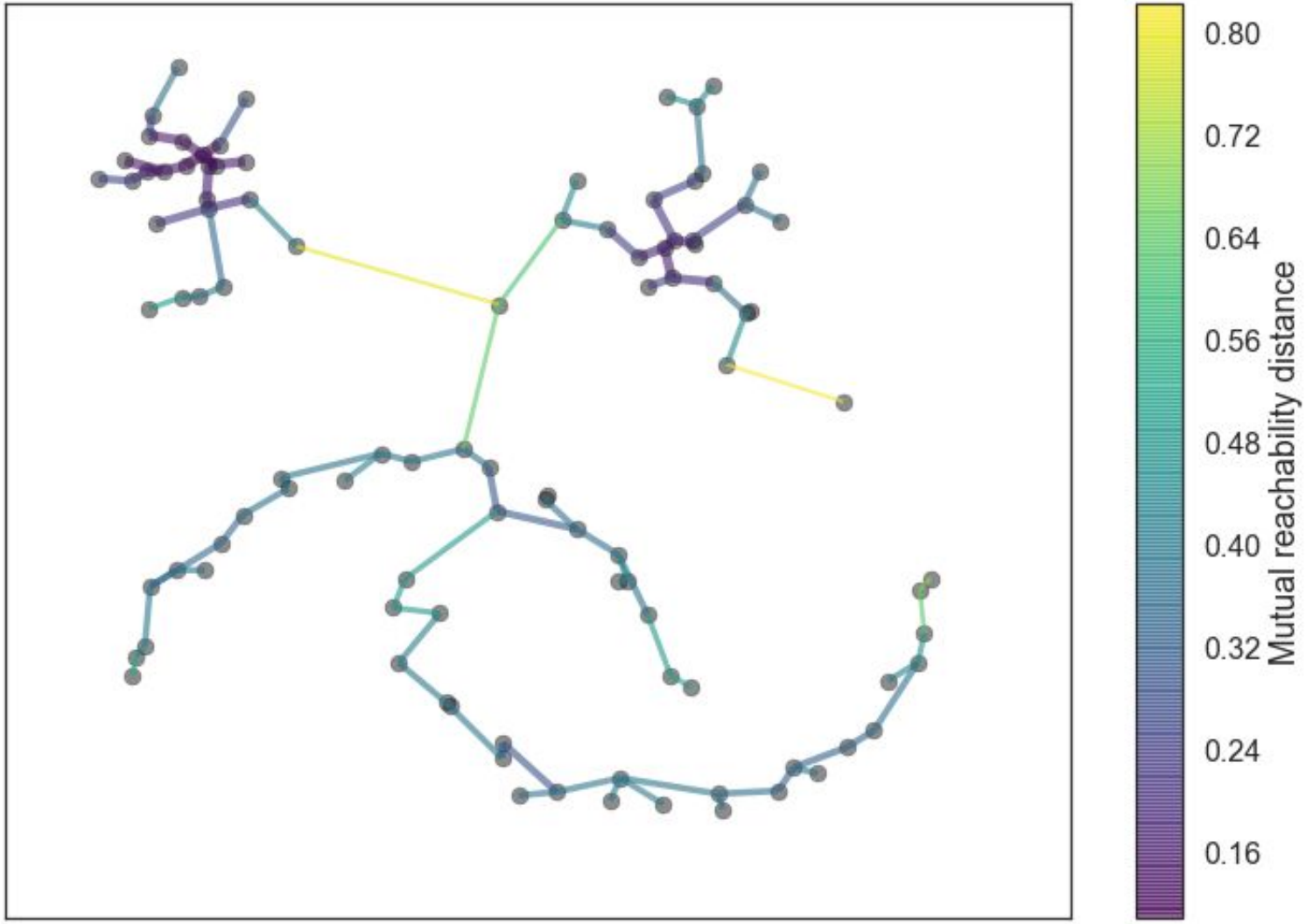
Example: Mutual reachability distances MST (minPts=5)



Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Example: Mutual reachability distances MST (minPts=5)



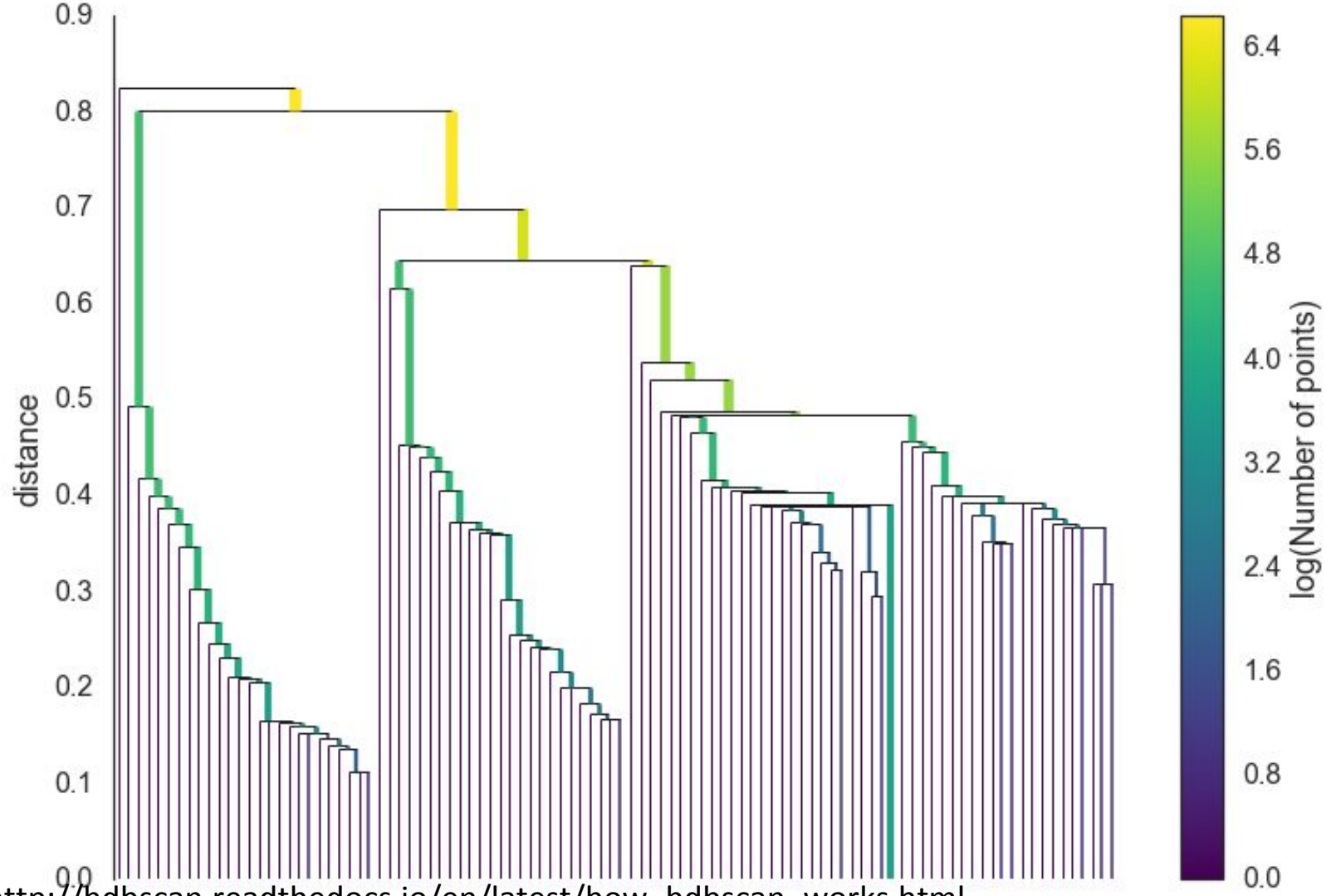


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Create a hierarchy of clusters

- Order the edges in the MST according to their weights.
- Build hierarchical clustering of the data points:
  - Initialize each point as a singleton cluster
  - Iterate over the edges according to their order
  - For each edge, merge the 2 cluster that it connected to

Create a hierarchy of clusters

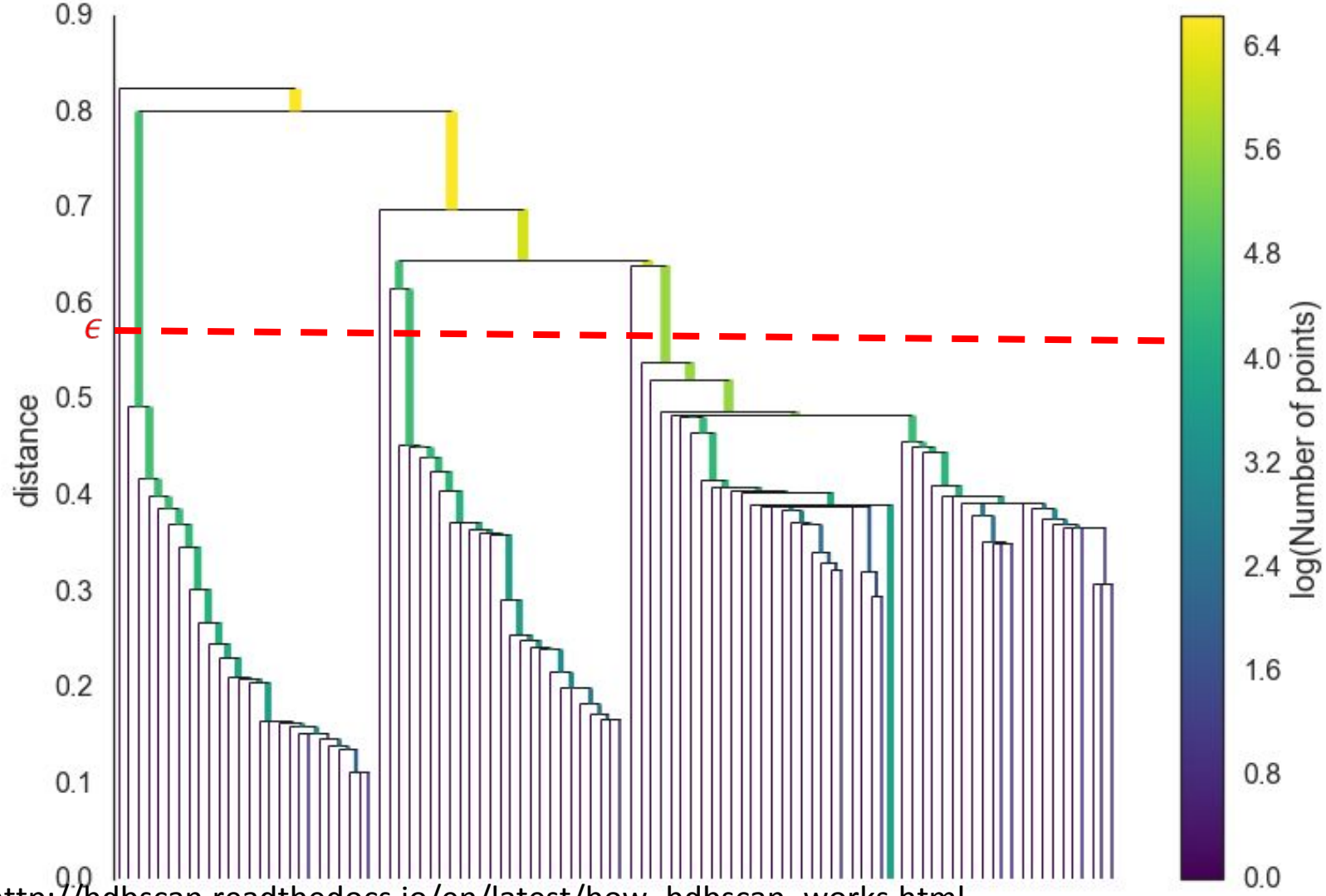


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

How to extract clusters from the hierarchy?

We can theoretically use an "ϵ " like in DBSCAN and take the clusters that fall under that cut-off value.

Equivalent of running DBSCAN with a parameter of ϵ and treating the "border points" as noise.

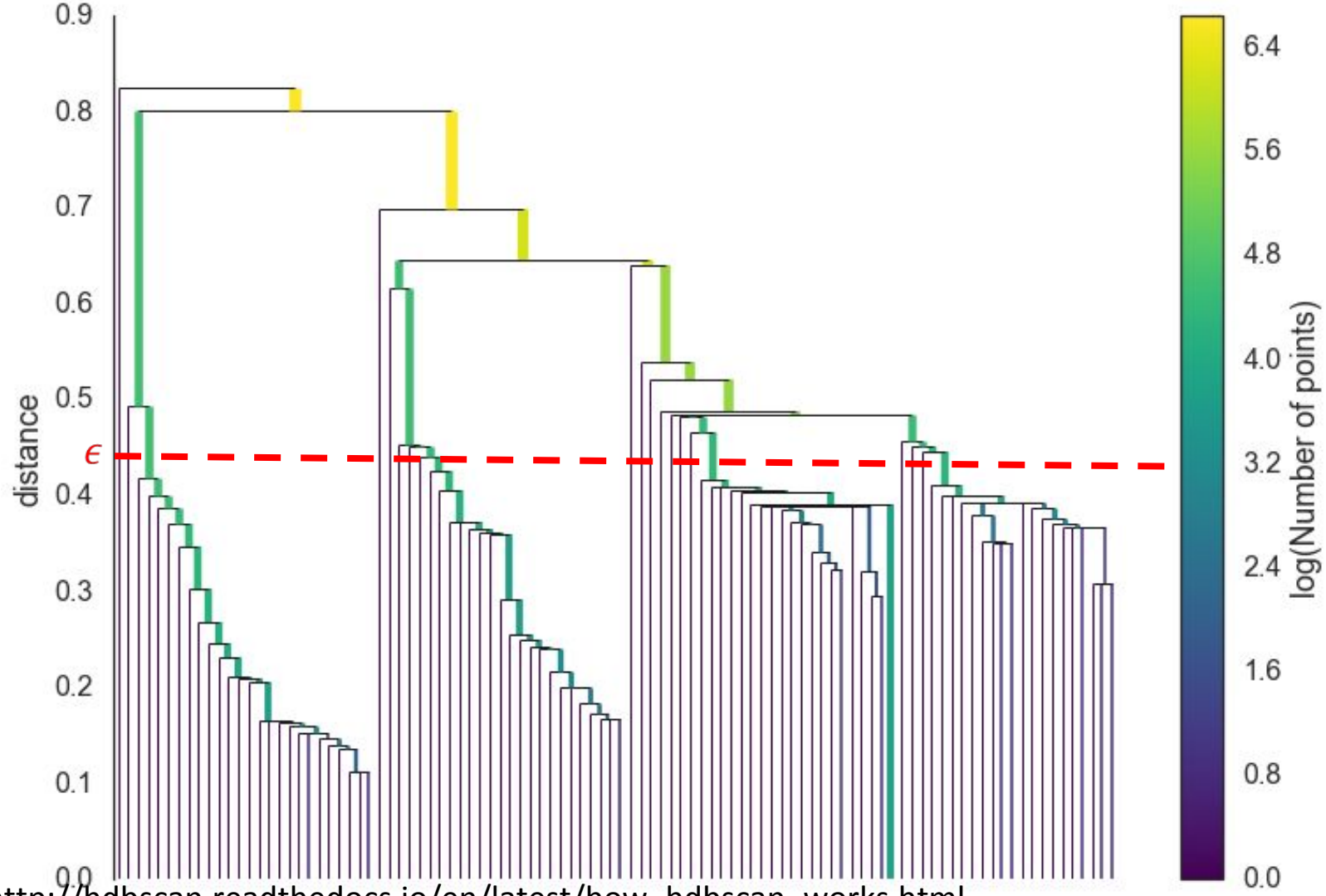


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

How to extract clusters from the hierarchy?

We can theoretically use an "ϵ " like in DBSCAN and take the clusters that fall under that cut-off value.

Equivalent of running DBSCAN with a parameter of ϵ and treating the "border points" as noise.

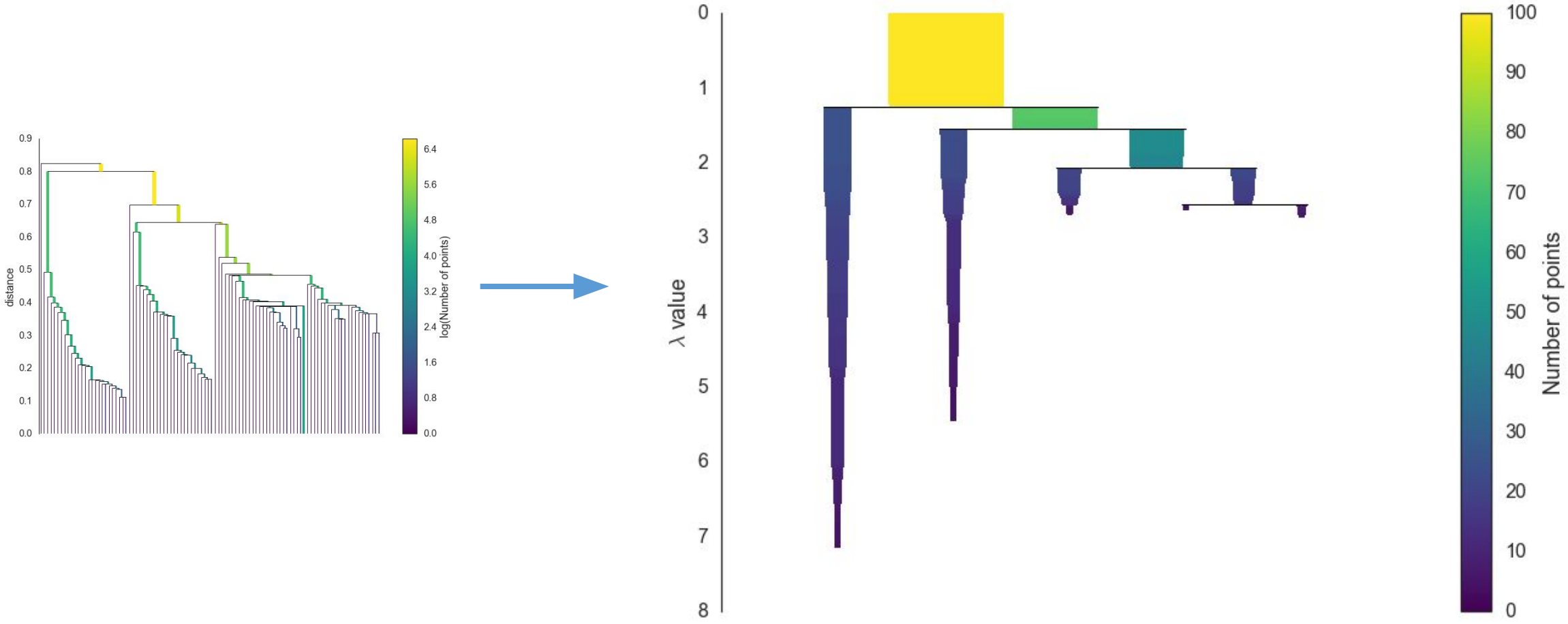


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Condense the cluster tree

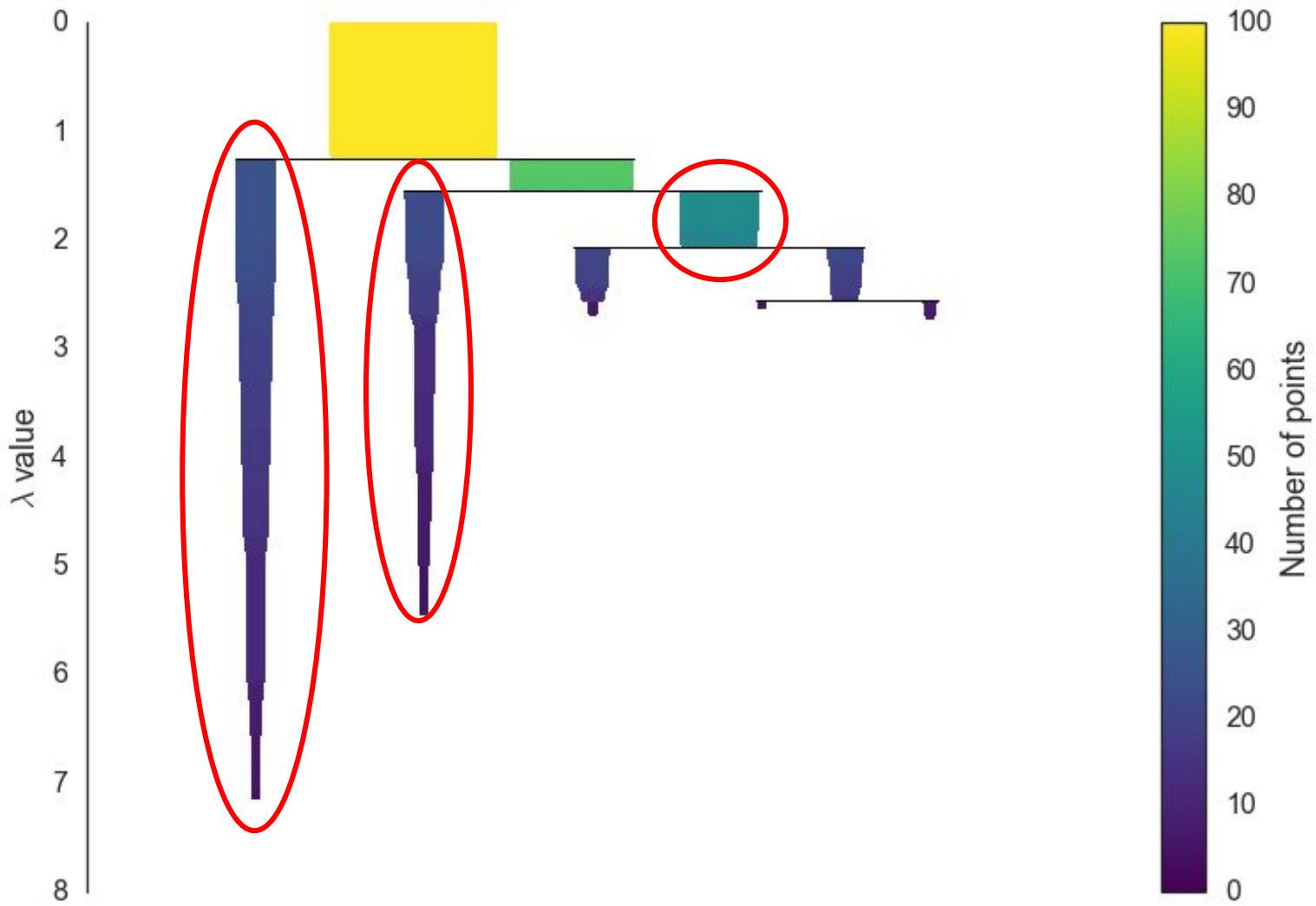
- Imagine we examine at a continues range of ϵ values and let $\lambda = 1/\epsilon$
- If we start with a very small value of λ and increase the value of λ (and lower ϵ), larger clusters disconnect, and smaller clusters are formed.
- When a split occurs, we check If the size of a newly formed cluster(s) is less then **minClusterSize**, we discard it and retain the "identity" of the parent cluster for the remaining points.

Condense the cluster tree (minClusterSize=5)



Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters - Intuition

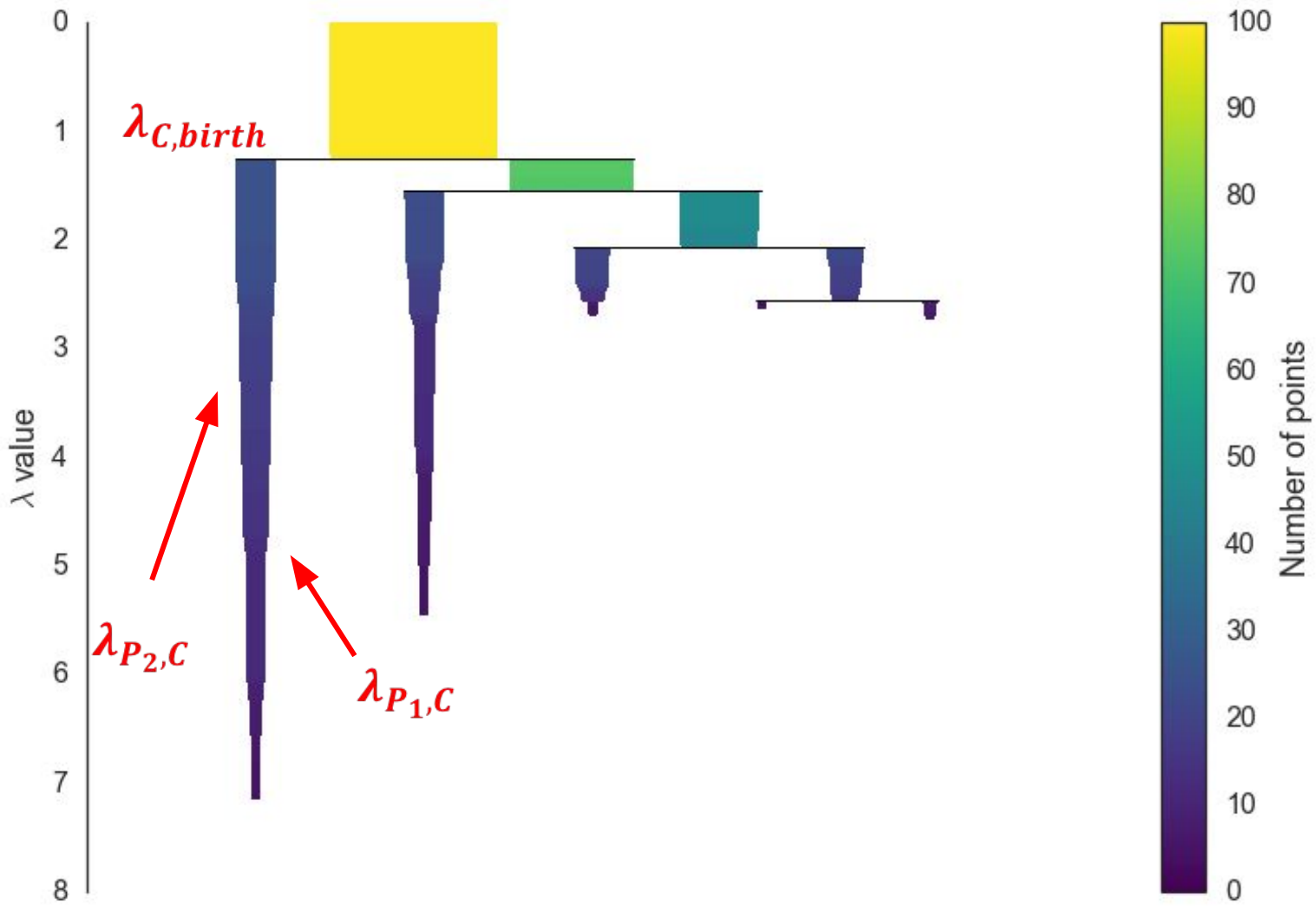


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters

- For each cluster in the condensed tree define: $\boldsymbol{\lambda_{C,birth}}$ - the value of λ for which the split that gave "birth" to that cluster ***C*** occurred
- For each point, p: $\boldsymbol{\lambda_{p,C}}$ - the value of λ in which the point was disconnected from the cluster ***C*** and discarded
- Define the "**Stability**" of a cluster by $S_C = \sum_{p \in C}(\lambda_{p,C} - \lambda_{birth,C})$
- Goal: find a "flat" clustering such that the sum of stabilities of clusters is **maximized**.

Extract the clusters

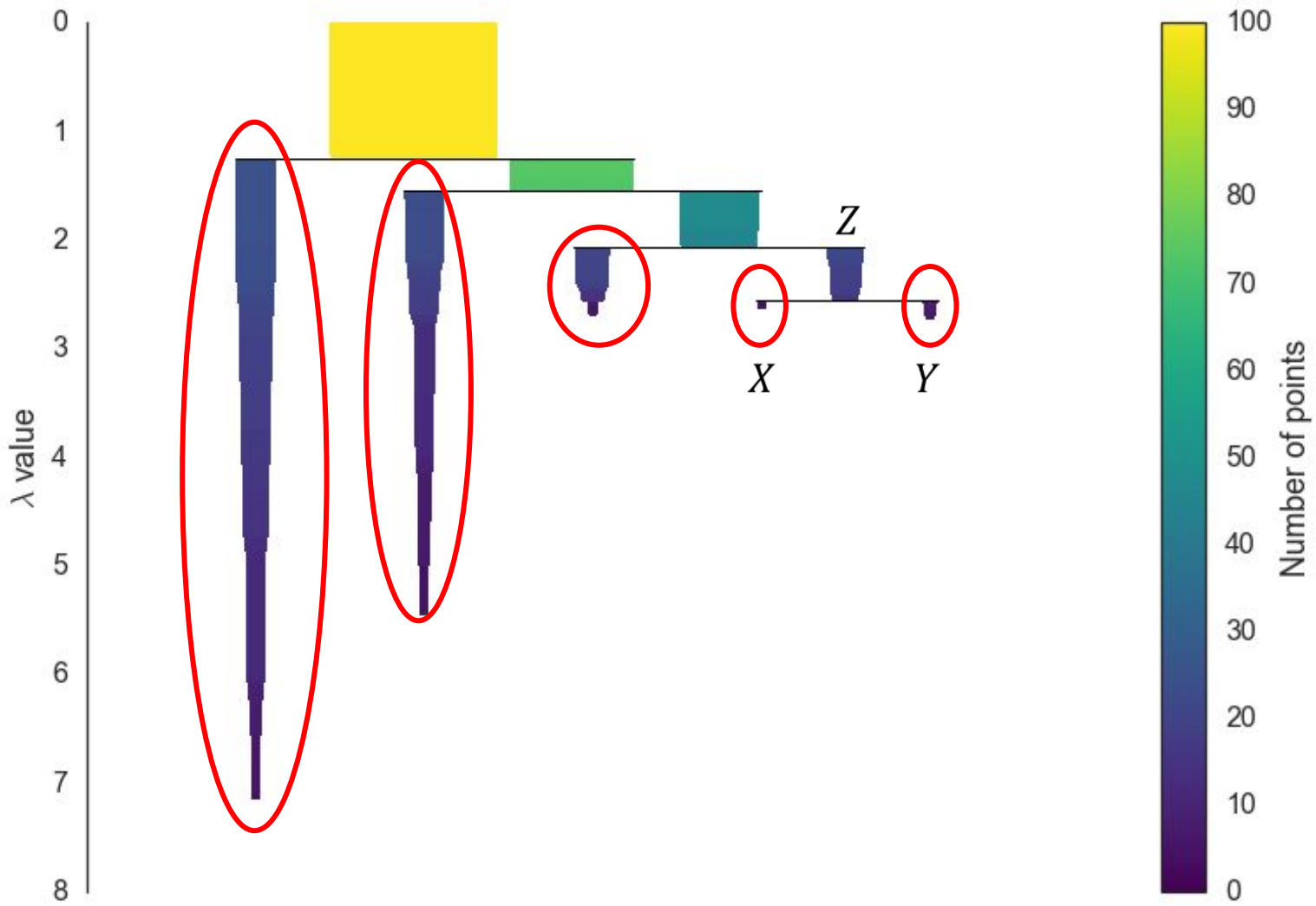


Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters

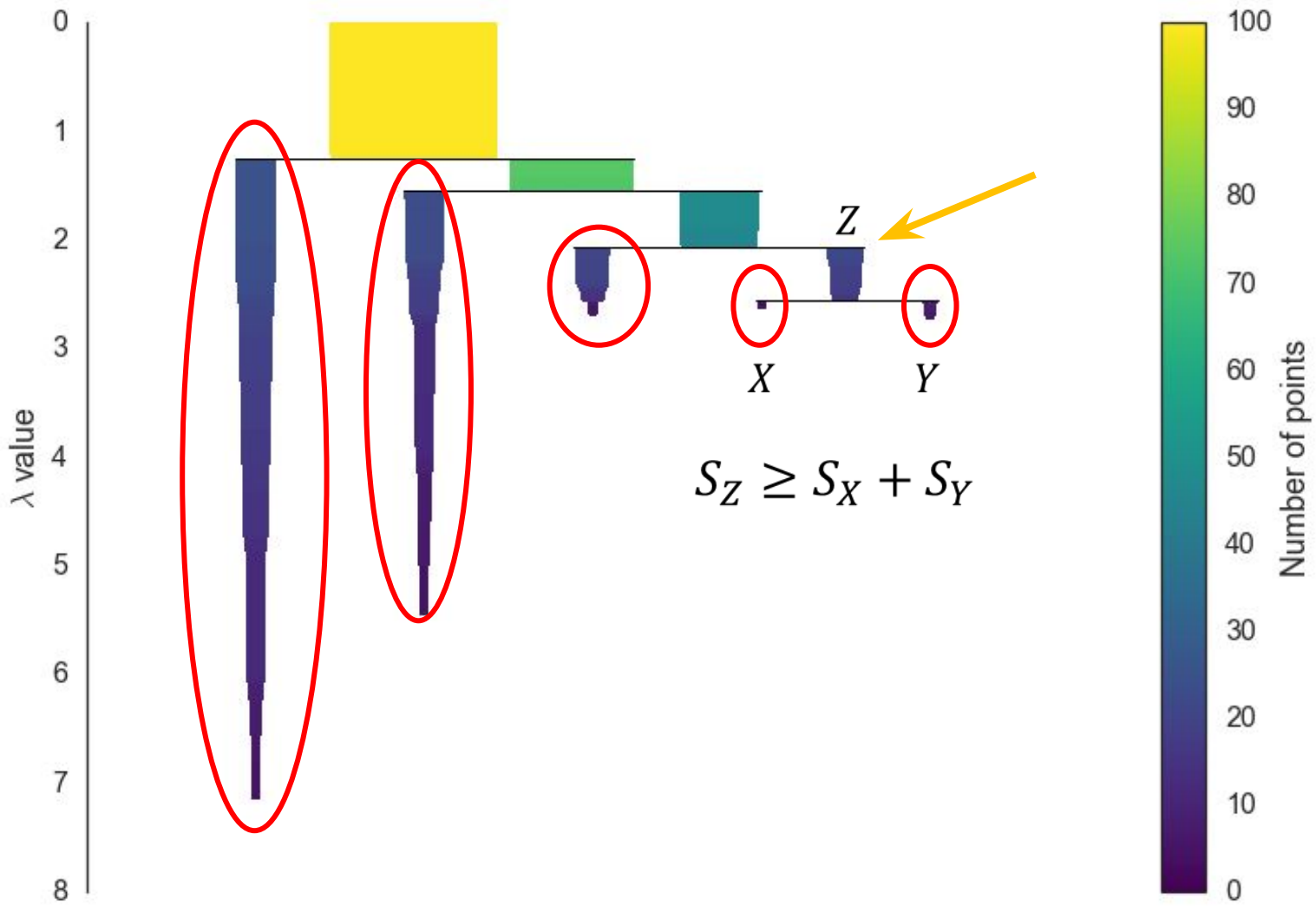
- Goal: find a “flat” clustering such that the sum of stabilities of clusters is maximized.
- Optimization algorithm:
  - Start from selecting the leaves of the condensed cluster tree and for each node split:
    - Check if the sum of stabilities of the left and right children are more then the stability of the parent.
    - If the sum is larger – “select” the left and right clusters and set the stability of the node to the sum of the stabilities of the left and right clusters.
    - If the sum is smaller or equal, “unselect” the left and right clusters and select the current node’s cluster.

Extract the clusters



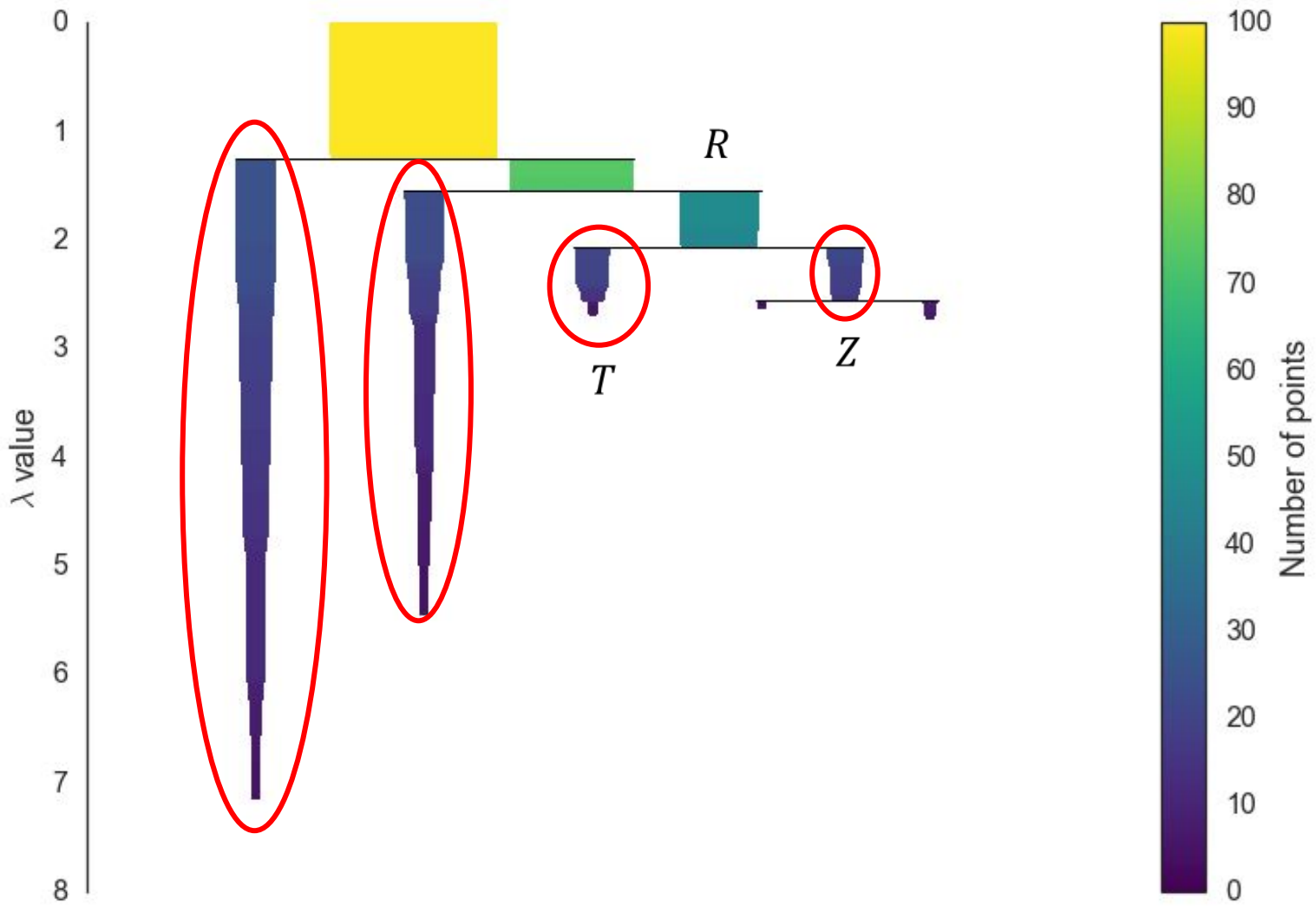
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



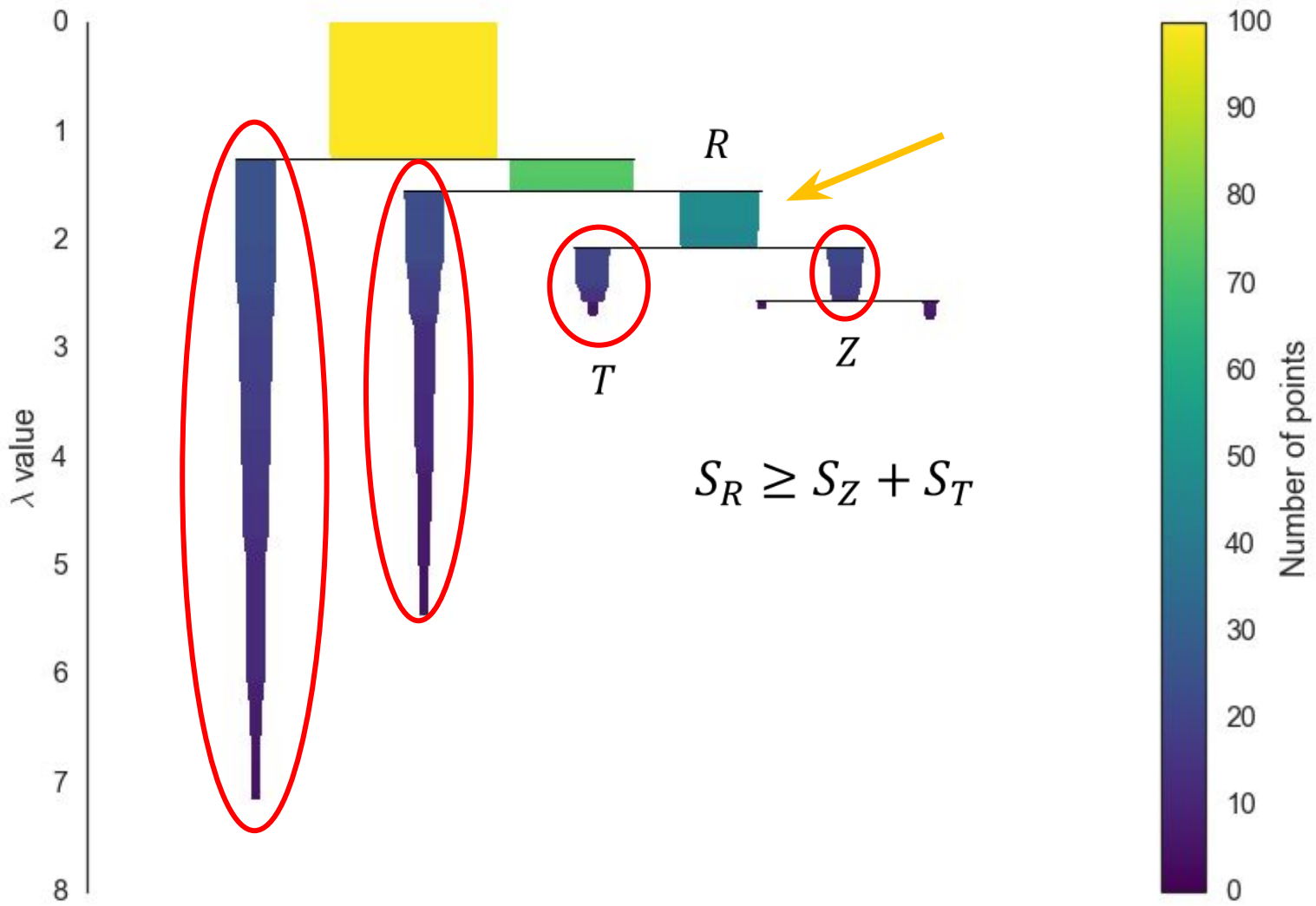
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



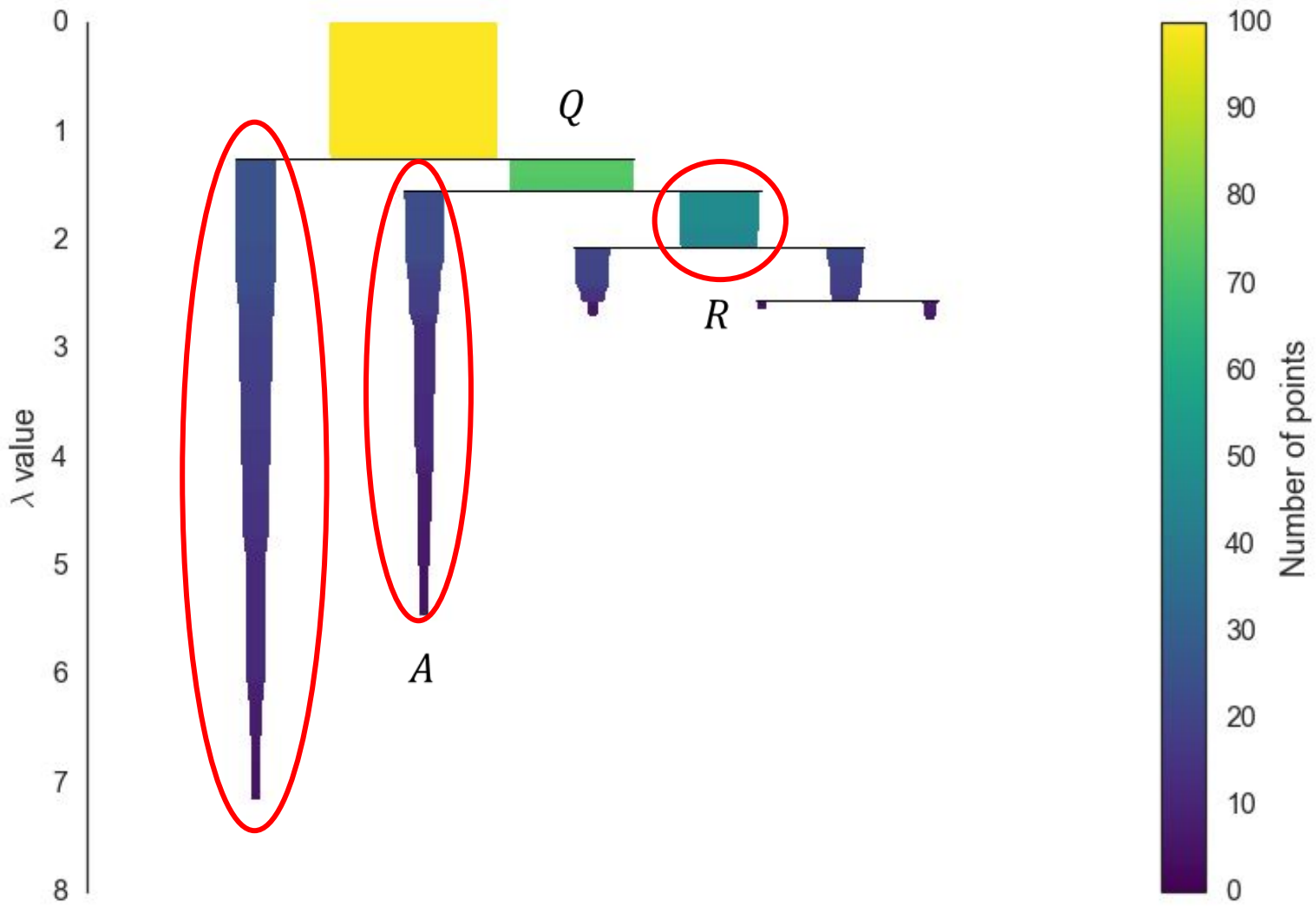
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



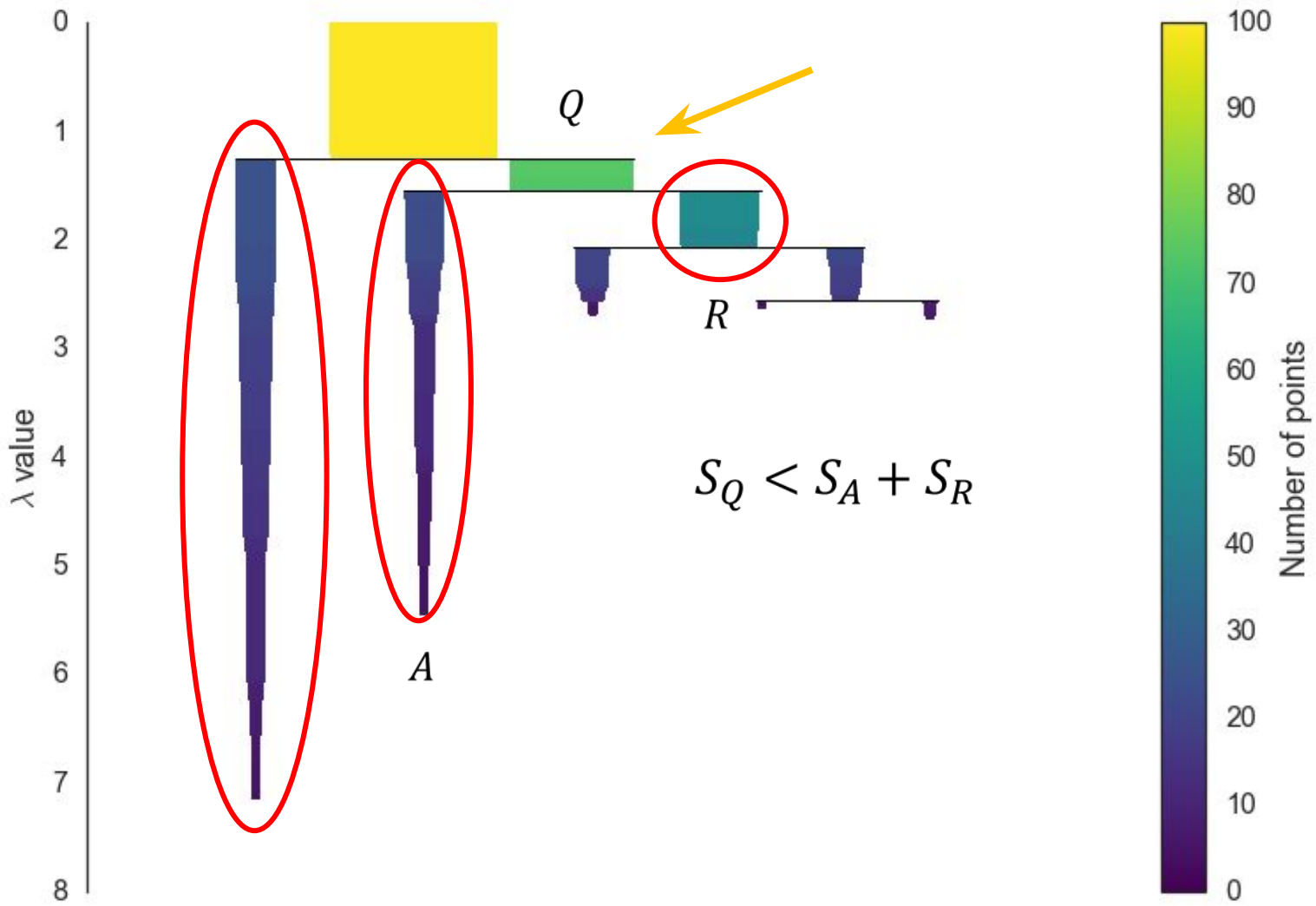
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



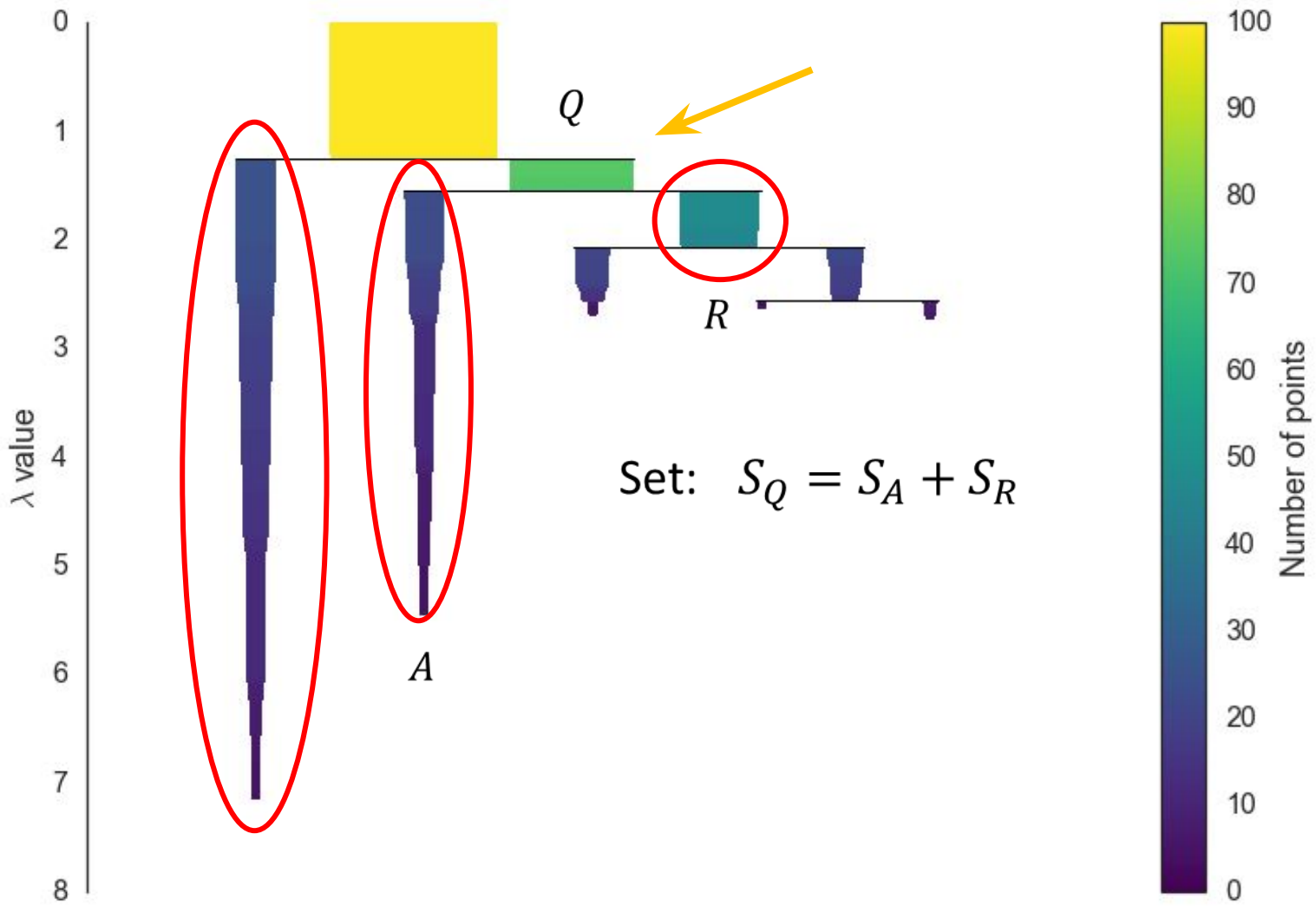
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



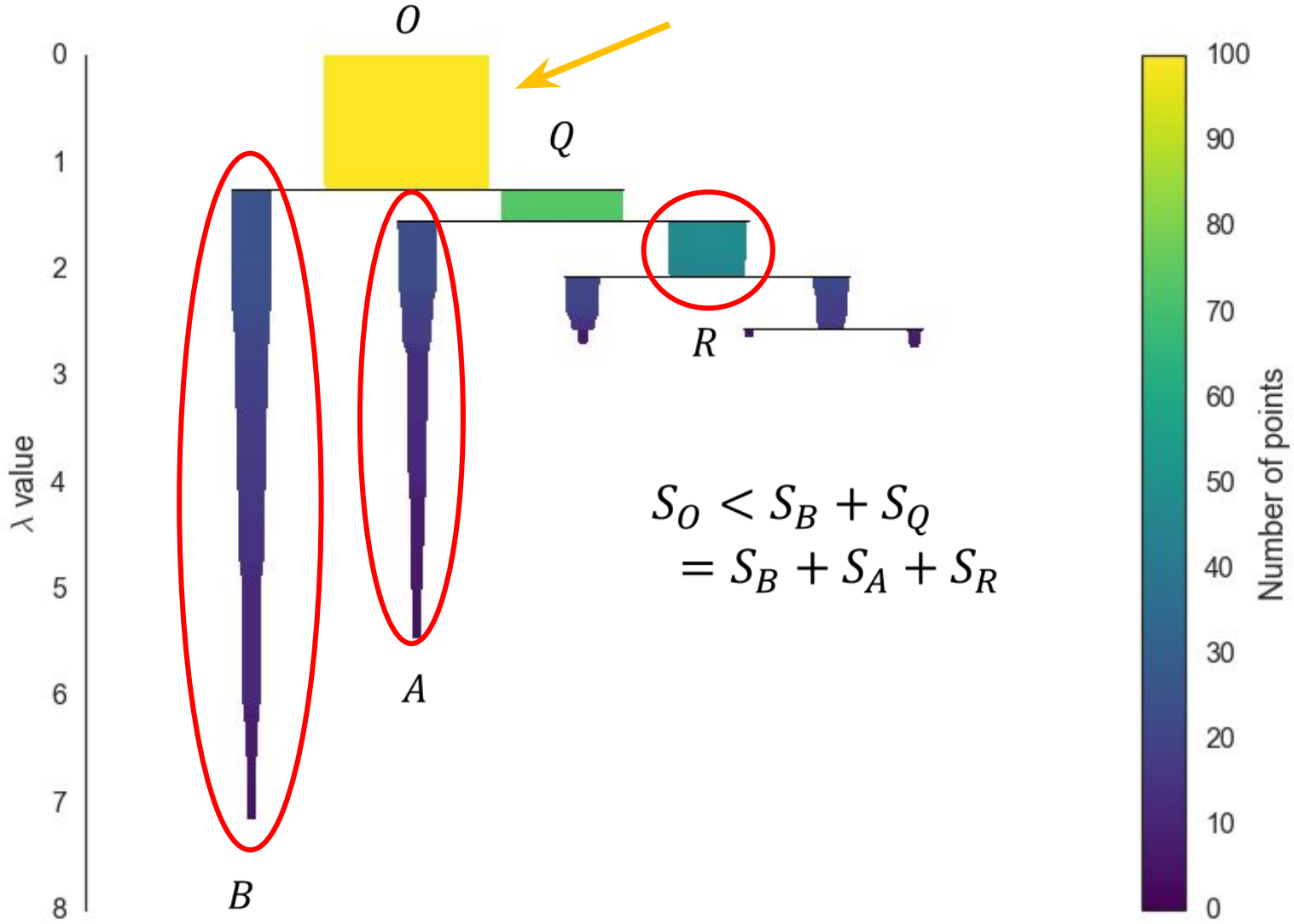
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



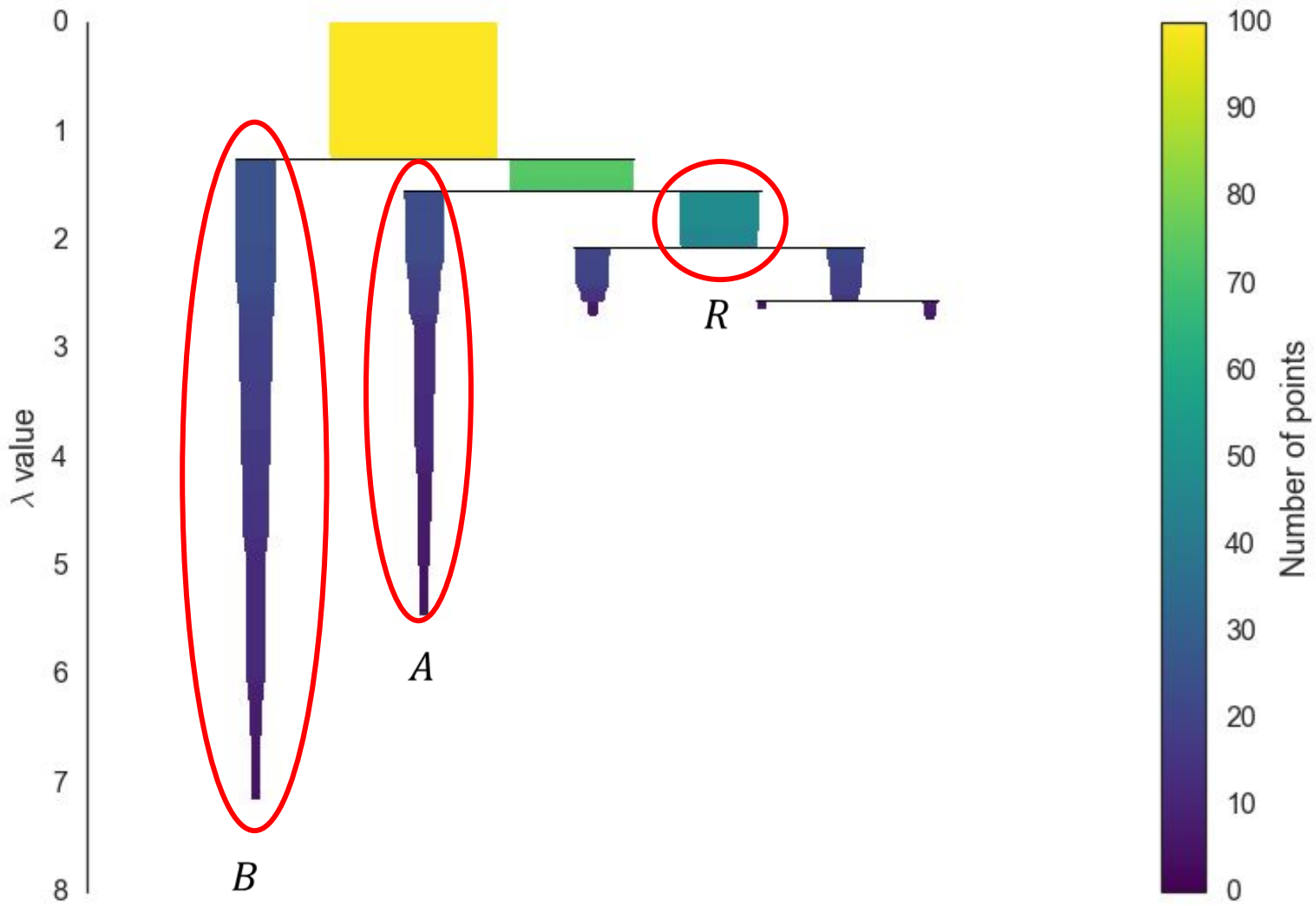
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



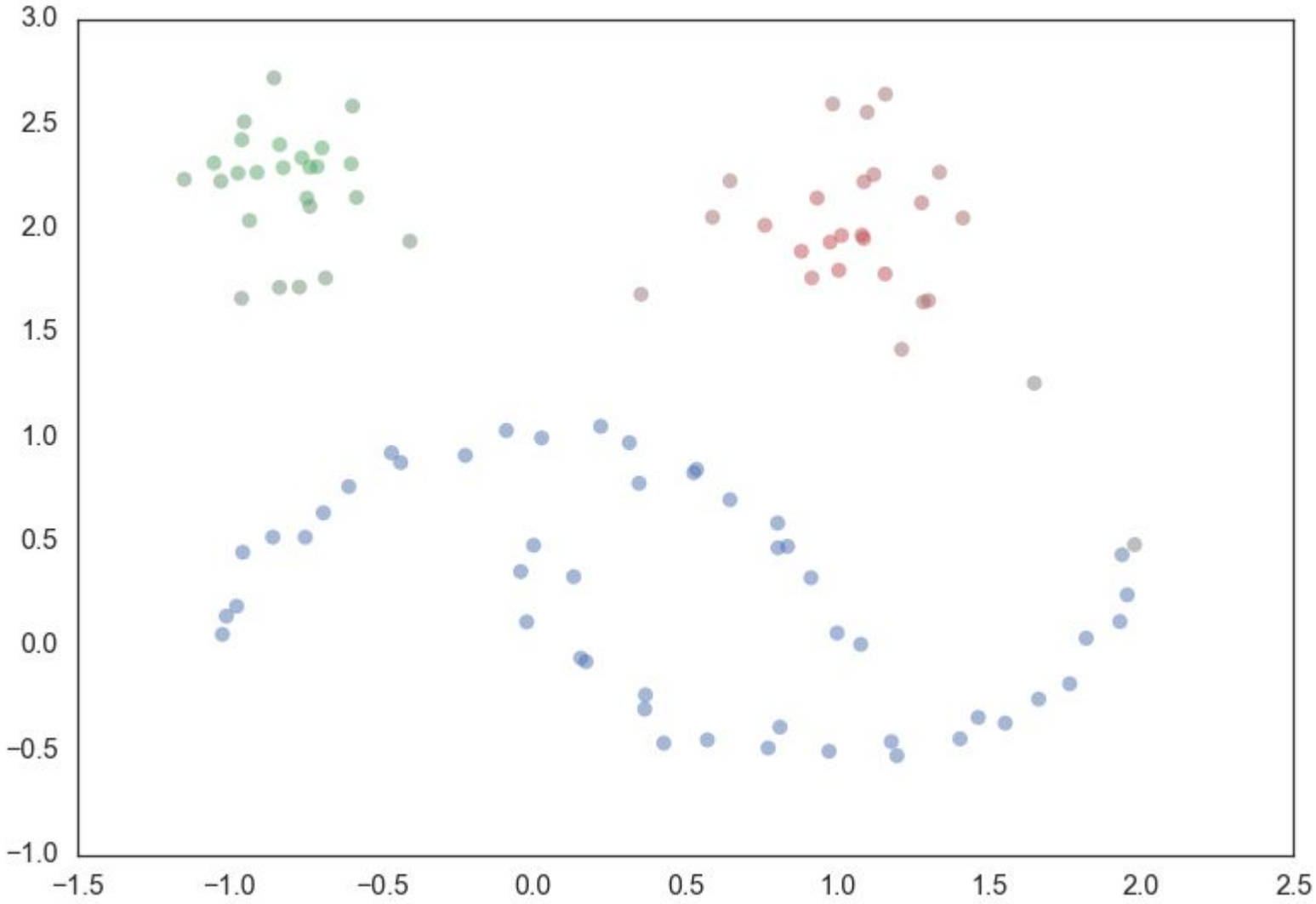
Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Extract the clusters



Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

Clustering result



Example taken from: http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

HDBSCAN - Summary

- Similar to OPTICS – provides a way to inspect several values of ϵ at the same time.
- Unlike OPTICS, provides a way to actually extract the clusters based on the local density.
- Default implementation has only **one** intuitive parameter – minClusterSize. (*But having less parameters also mean less control*..)
- High performance python implementation available, actively developed.
- Main caveat: for larger datasets, tends to cluster large number of points as noise.

Wrap-up

- Density based clustering methods:
  - Discover clusters of different sizes and shapes.
  - Automatically detect the number of clusters.
  - Can handle noise.
- Reviewed 3 Methods for density based clustering (but there are many more).
- Parameters / Complexity trade-off.

Thank you!

Resources

- Python notebook: https://github.com/nadavbar/density-based-clustering/blob/master/clustering_example.ipynb
- SKLearn clustering algorithms: http://scikit-learn.org/stable/modules/clustering.html
- "How HDBSCAN works": http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html
- HDBSCAN implementation: https://github.com/scikit-learn-contrib/hdbscan