Hierarchical and DBSCAN Clustering

Agenda

- Hierarchical Clustering
  - Distance Matrix
  - Linkage Methods
  - Dendrogram
- Case Study
- Density Based Clustering
  - DBSCAN

In the previous session...

- In the last session, we studied the K-means algorithm to cluster the dataset
- K-means is a simple algorithm to understand and implement. However, this algorithm has several flaws
- We need to provide the number of clusters (K) to the algorithm, which is not always easy to determine
- The initial centroid assignment affects the formation of final clusters
- The algorithm is sensitive to the presence of outliers

In this session...

- In this session, we study two more clustering techniques: Hierarchical clustering and DBSCAN
- For these two algorithms, there is no need to pre-specify the number of clusters
- Hierarchical clustering is useful when there is a hierarchical structure in the original data
- DBSCAN identifies the noise/ outliers in the data
- The DBSCAN algorithm can form clusters of any arbitrary shape

Hierarchical Clustering

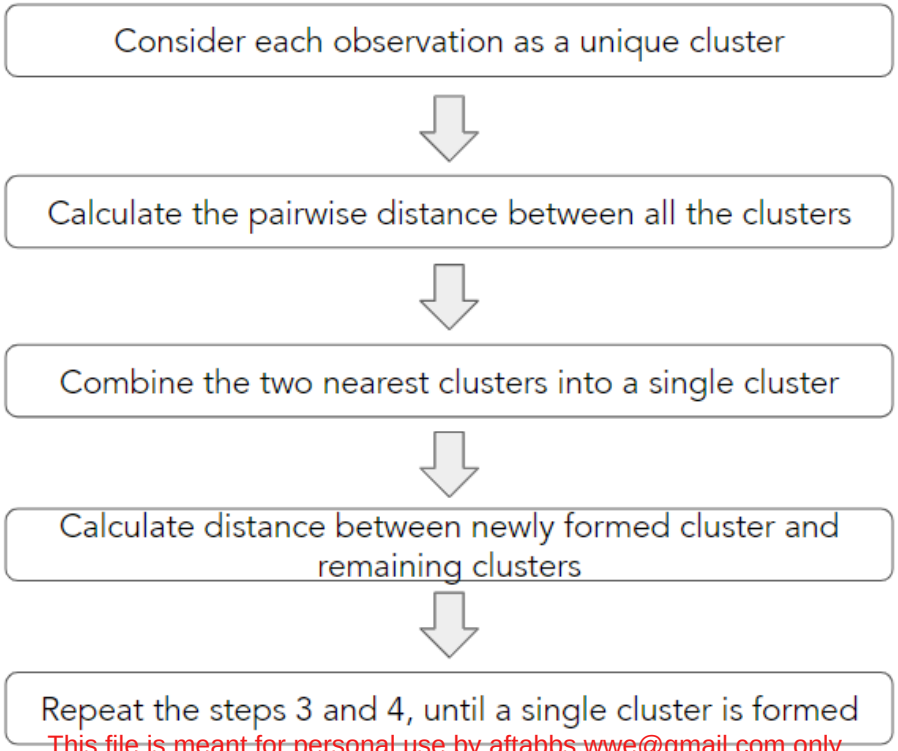
Hierarchical clustering

- Hierarchy based clustering method
- Two main types: Agglomerative (bottom to top approach) and Divisive (top to bottom approach)
- No need to pre-define the number of clusters

Agglomerative clustering

- Most popular hierarchical clustering method
- It considers the bottom to top approach
- The similar observations are clustered together to form a bigger cluster, considering each observation as a unique cluster in the initial step
- The process continues till all the observations are fused in a single cluster
- A dendrogram is used to visualize such cluster formation

Agglomerative clustering - procedure



Great Learning
POWER AHEAD

Python function

In python, the AgglomerativeClustering() performs Agglomerative clustering on the data

```
# import the function
from sklearn.cluster import AgglomerativeClustering

# pass the number of required clusters
model = AgglomerativeClustering(n_clusters = 2)

# fit and predict the cluster labels
model.fit_predict(data)
```

Distance Matrix

Distance matrix

- In the initial step of clustering, each observation is considered as a cluster
- The distance matrix returns the pairwise distance between all these observations
- The pairwise distance can be calculated using various distance measures like Manhattan, Euclidean, Minkowski and so on
- This matrix is used to find the two closest clusters

Great Learning
POWER AHEAD

Question:

Find the distance matrix for the given data using Euclidean distance.

	X	Y
A	0.1	0.4
B	0.25	0.32
C	0.29	0.19

Great Learning
POWER AHEAD

Answer:

To obtain a distance matrix, calculate the Euclidean distance between all the points.

$$Distance[A, B] = \sqrt[2]{(x-a)^2 + (y-b)^2}$$
$$= \sqrt[2]{(0.1-0.25)^2 + (0.4-0.32)^2}$$
$$= \sqrt[2]{0.0225 + 0.0064}$$
$$= \sqrt[2]{0.0289}$$
$$= 0.17$$

	X	Y
A	0.1	0.4
B	0.25	0.32
C	0.29	0.19

The distance between data points A and B is 0.17. Similarly we can calculate the distance between all the points.

Great Learning
POWER AHEAD

Answer:

By calculating all the distances, we obtain the distance matrix. This matrix returns all the pairwise distances.

	A	B	C
A	0	0.17	0.28
B	0.17	0	0.14
C	0.28	0.14	0

Distance matrix

- The distance of a point from itself will always be zero
- Thus, the diagonal of the distance matrix will be zero
- Here, we have 3x3 matrix for 3 clusters, as each point is considered as a cluster
- We have to group these clusters such that at the end we are left with a single cluster that consists of all the observations

	A	B	C
A	0	0.17	0.28
B	0.17	0	0.14
C	0.28	0.14	0

Distance matrix

- In each iteration, our goal is to find the closest pair
- Here we can see that the distance between the points B and C is the minimum
- Thus, we group B and C in one cluster (B, C)

	A	B	C
A	0	0.17	0.28
B	0.17	0	0.14
C	0.28	0.14	0

Distance matrix

- We will have to update the distances in the distance matrix
- The distance between the ungrouped clusters/elements will remain the same
- But, how we will calculate the distance between the ungrouped clusters and the newly created cluster (B, C)?
- Here is where the different linkage methods are used

Linkage Methods

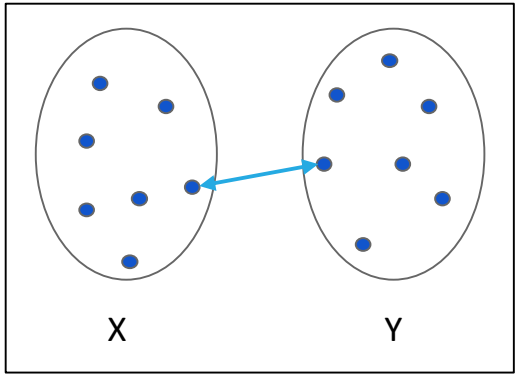
Linkage methods

- Similarity between the clusters (inter-cluster distance) can be measured using various types of linkages
- Some of the types are: Single, Complete, Average, Centroid linkage

Single linkage

It is defined as the minimum distance between the points of two clusters.

$$d(X,Y) = min\{d(x,y)|x\epsilon X, y\epsilon Y\}$$



- The method can create the non-elliptical clusters
- It can produce undesirable results in the presence of outliers
- It causes a chaining effect, where clusters have merged since at least one point in a cluster is closest to a point in another cluster. This forms a long and elongated cluster

Single linkage

In our example, the single linkage between the cluster (B, C) and A is

$$Min[dist(B, A), dist(C, A)]$$
$$= Min[0.17, 0.28]$$
$$= 0.17$$

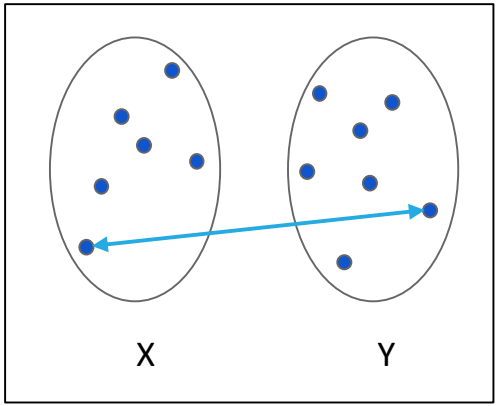
	A	(B, C)
A	0	**?**
(B, C)	**?**	0

	A	(B, C)
A	0	**0.17**
(B, C)	**0.17**	0

Complete linkage

It is defined as the maximum distance between the points of the two different clusters.

$$d(X,Y) = max\{d(x,y)|x\epsilon X, y\epsilon Y\}$$



- The method returns more stable clusters, with nearly equal diameter
- It avoids the chaining effect
- It is less sensitive to outliers
- It breaks the large clusters and it is biased towards globular clusters

Complete linkage

The complete linkage between the cluster (B, C) and A is

$$Max[dist(B, A), dist(C, A)]$$
$$= Max[0.17, 0.28]$$
$$= 0.28$$

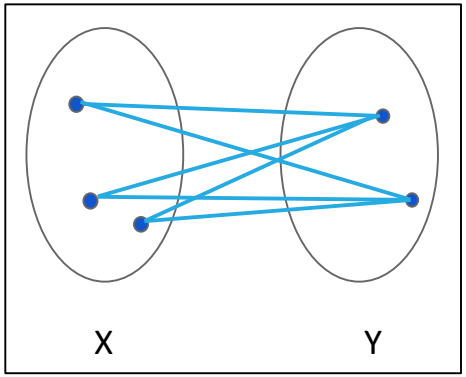
	A	(B, C)
A	0	**?**
(B, C)	**?**	0

	A	(B, C)
A	0	**0.28**
(B, C)	**0.28**	0

Average linkage

It is defined as the average of all the pairwise distances between the two clusters.

$$d(X,Y) = \frac{1}{n_X n_Y} \sum_{x \epsilon X, y \epsilon Y} d(x,y)$$



Where,

n_X : Number of elements in the cluster X

n_Y : Number of elements in the cluster Y

- This method balances between the single and complete linkage
- It forms compact clusters and the method is robust to outliers

Average linkage

The average linkage between the cluster (B, C) and A is

$$AVG[dist(B, A), dist(C, A)]$$
$$= \frac{1}{2}[dist(B, A) + dist(C, A)]$$
$$= \frac{1}{2}[0.17 + 0.28]$$
$$= 0.225$$

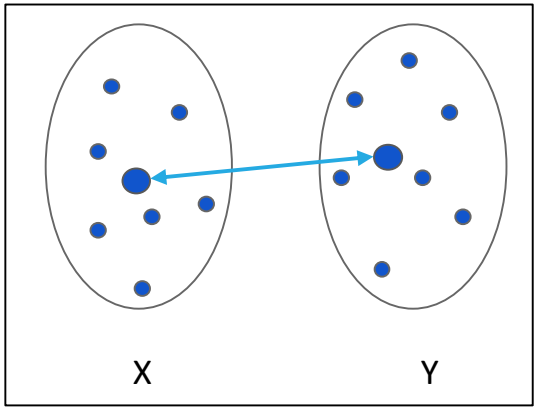
	A	(B, C)
A	0	**?**
(B, C)	**?**	0

	A	(B, C)
A	0	**0.225**
(B, C)	**0.225**	0

Centroid linkage

It is defined as the distance between the centroids (means) of the two clusters.

$$d(X, Y) = d(X_c, Y_c)$$



Where,

X_c : Centroid of the cluster X

Y_c : Centroid of the cluster Y

- It creates similar clusters as average linkage
- This method suffers a major drawback of inversion. i.e. a smaller cluster can be more similar to the newly merged larger cluster rather than the individual clusters

Centroid linkage

The centroid linkage between the cluster (B, C) and A is

$$dist(centroid(A), centroid(B, C))$$
$$= dist((0.1, 0.4), (0.27, 0.255))$$
$$= \sqrt{(0.1 - 0.27)^2 + (0.4 - 0.255)^2}$$
$$= 0.223$$

	A	(B, C)
A	0	**?**
(B, C)	**?**	0

	A	(B, C)
A	0	**0.225**
(B, C)	**0.225**	0

Great Learning
POWER AHEAD

Ward Linkage (ward minimum variance method)

- By default, the scikit-learn library of python considers the ‘ward’ linkage
- The clusters are merged; if the new cluster minimizes the variance
- It is a computationally intensive method
- It is given by the formula:

$$d(X,Y) = \sqrt{\frac{2.n_X.n_Y}{n_X+n_Y}}.d(X_c,Y_c)$$

X_c : Centroid of the cluster X

Y_c : Centroid of the cluster Y

and

n_X : Number of elements in the cluster X

n_Y : Number of elements in the cluster Y

Summary

Single Linkage

- Can create non-elliptical clusters
- Sensitive to outliers
- Prone to chaining effect

Complete Linkage

- Creates more compact clusters
- Biased towards globular clusters
- Breaks large clusters

Average Linkage

- Balances between single and complete linkage
- Robust to outliers
- Biased towards globular clusters

Centroid Linkage

- Cluster formation is similar to average linkage
- Can cause inversion

Ward Linkage

- Most effective in presence of outliers
- Biased towards globular clusters

Great Learning
POWER AHEAD

Python function

In python, the linkage() returns the linkage matrix. It provides the distance between the cluster given the linkage method.

```
# import the function
from scipy.cluster.hierarchy import linkage

# instantiate linkage object with data & linkage method
# 'link' returns the linkage matrix
link = linkage(df_data, method = 'single')

# print the linkage matrix
print(link)
```

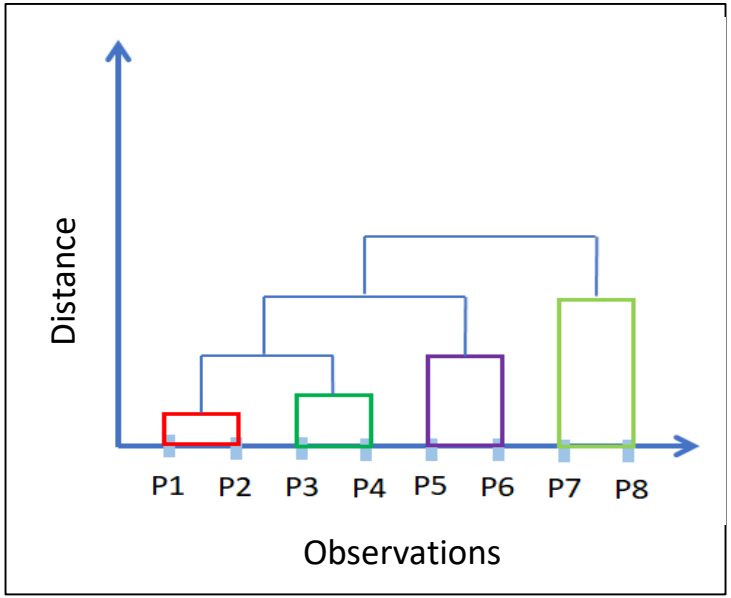
Dendrogram

Dendrogram

- It is a very useful technique to visualize the clusters
- It is a tree-based hierarchical structure that can be used to decide the required number of clusters
- Different linkage methods result in the formation of different dendrograms
- Observations linked at a low height represents more similar observations
- Dissimilar observations fuse at a higher level in the dendrogram

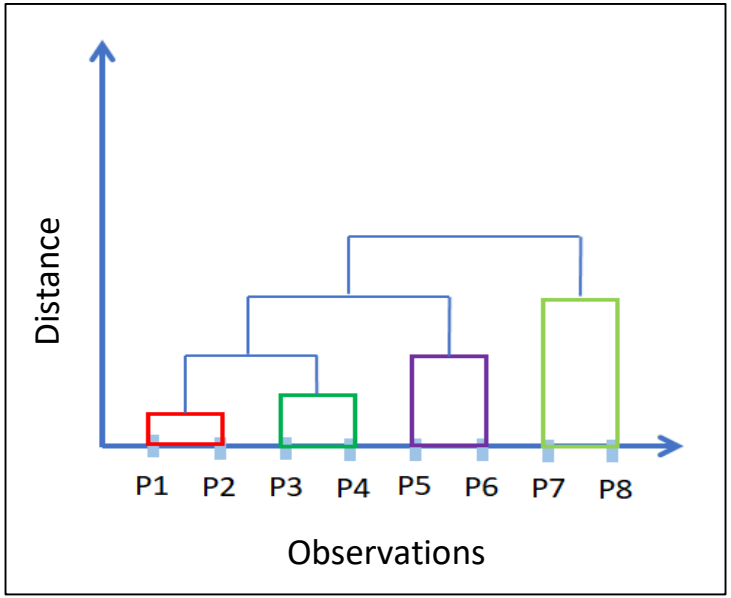
Dendrogram

- X-axis of the dendrogram represents the data point, each considered as a single cluster and the distance is given on the Y-axis
- Each single cluster is known as ‘leaf’
- The horizontal line is known as ‘clade’ which represents the merging of clusters



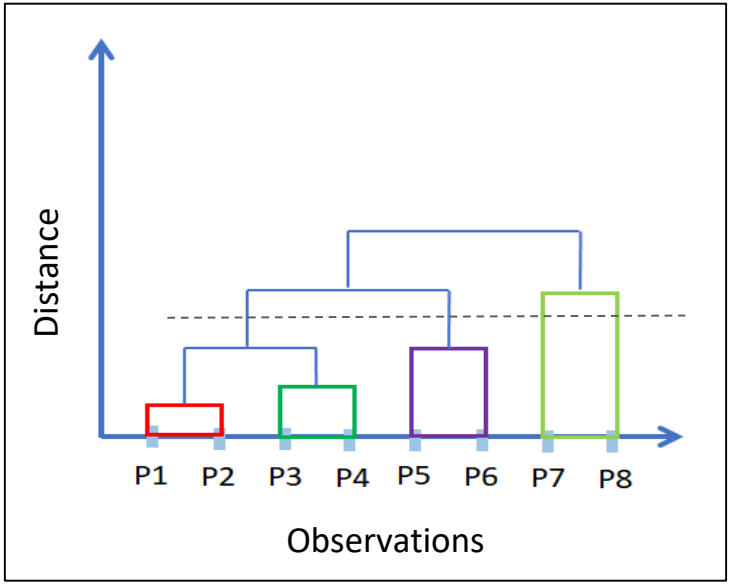
Dendrogram

- P1 and P2 are clustered at the lowest height, which implies more similarity between the observations
- The clusters (P1, P2) and (P3, P4) are clustered to form a bigger cluster. Then this cluster is fused with (P5, P6) to form a even bigger cluster
- Finally, this cluster is fused with (P7, P8) to form a single cluster



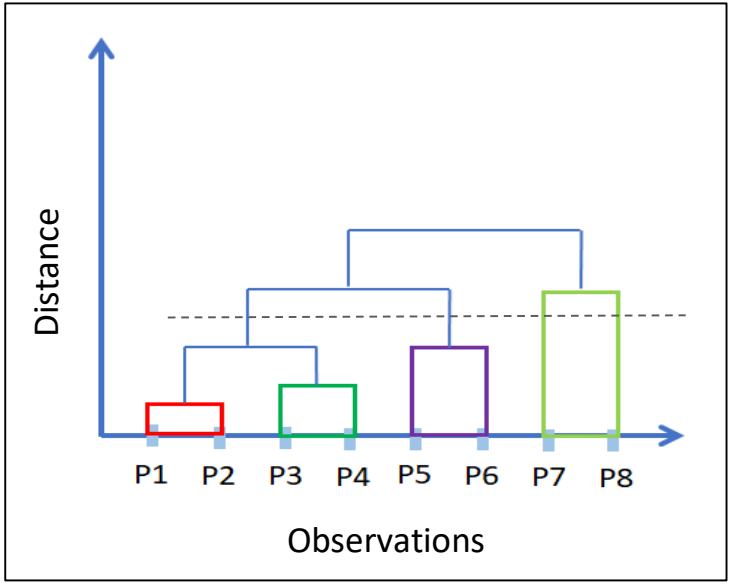
Cutting the dendrogram

- The number of clusters depends on the height at which the dendrogram is cut
- Cutting the dendrogram at different heights results in the formation of distinct clusters
- The optimal number of clusters is the number that remains constant for the larger distance on the y-axis



Dendrogram

- The black line shows the height at which the dendrogram is being cut
- This line intersects the dendrogram at 4 distinct points, which gives 4 clusters namely (P1, P2, P3, P4), (P5, P6), P7, and P8
- The clusters P7 and P8 are clustered at higher distance than the remaining observations, which suggests more dissimilarity between these points



Cophenetic correlation coefficient

- Quantifies how the dendrogram has represented the dissimilarities between the observations
- It is defined as the correlation coefficient between cophenetic distances and the actual distance between the observations
- The cophenetic distance between the points P_i and P_j is the height represented on the Y-axis of dendrogram at which P_i and P_j are first linked together
- The actual distance is the pairwise distance between the observations represented in the distance matrix

Cophenetic correlation coefficient

- The value close to 1 represents the best linkage quality
- It is mostly used in biostatistics to evaluate the cluster models
- In python, the linkage matrix provides the cophenetic distance

```
# import the function
from scipy.cluster.hierarchy import cophenet

# pass the linkage matrix and actual distance
# 1st output of the cophenet() is the correlation coefficient
coeff, cophnet_dist = cophenet(linkage_matrix, actual_dist)

# print the cophnetic correlation coefficient
print(coeff)
```

Great Learning
POWER AHEAD

Python function

In python, the dendrogram() plots the dendrogram for the given linkage matrix.

```
# import the function
from scipy.cluster.hierarchy import dendrogram

# plot the dendrogram
# pass the linkage matrix
dendrogram(linkage_matrix)

# display the plot
plt.show()
```

Summary

- Merits:
  - Does not require a pre-specified number of clusters
  - Hierarchical relation between the clusters can be identified
  - Dendrogram provides a clear representation of clusters
- Demerits:
  - Different dendrograms are produced for different linkage methods
  - Selecting an optimal number of clusters using dendrogram is sometimes difficult
  - Time complexity is high

Great Learning
POWER AHEAD

Silhouette score

- One of the ways to decide the number of clusters is the silhouette score
- We calculate the silhouette score for different values of K (similar to K-means)
- The value of K with the highest silhouette score can be considered as an optimal number of clusters

Case Study

Case study: group the data

Consider the data of flower's petal length and petal width in millimeters for different flowers.

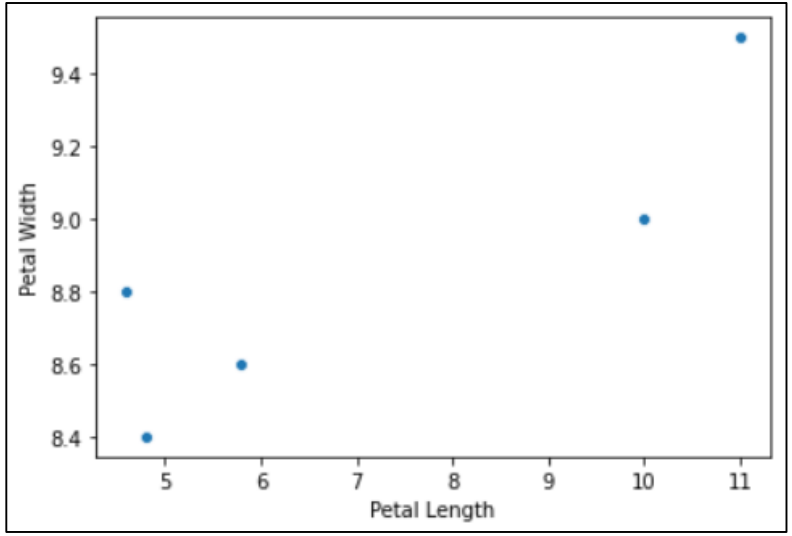
Petal Length	4.8	11	5.8	10	4.6
Petal Width	8.4	9.5	8.6	9	8.8

Can we group the data that belongs to the same kind of flower?

Case study

Use the Euclidean distance as a proximity measure to calculate the distance between the data points.

Use the ‘single’ linkage method to calculate the distance between the two clusters.



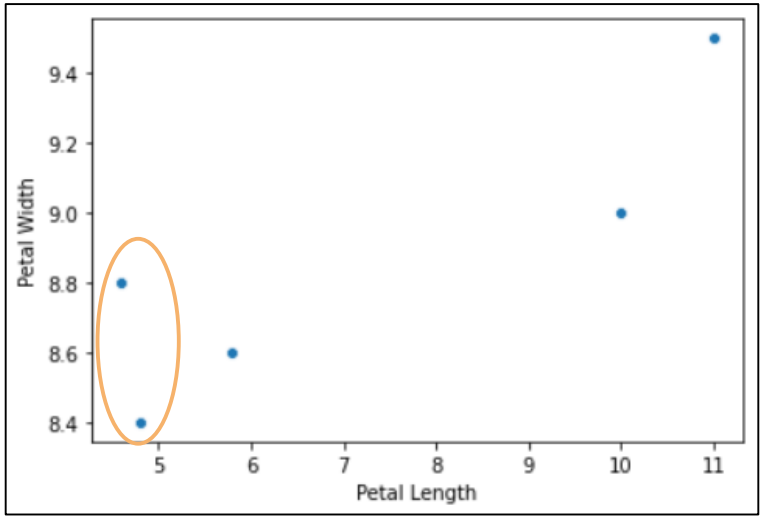
Case study

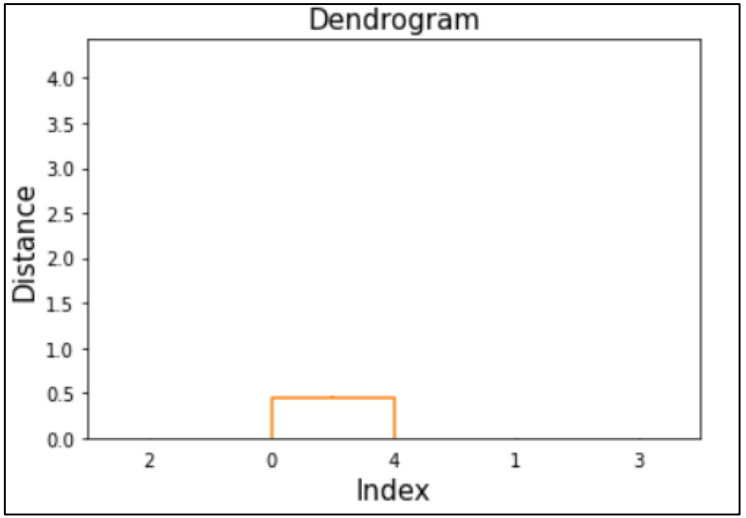
Calculate the Euclidean distance between the data points.

The distance matrix shows that the first and last points are closest.

	0	1	2	3	4
0	0	6.296824597	1.019803903	5.234500931	0.447213595
1	6.296824597	0	5.277309921	1.118033989	6.438167441
2	1.019803903	5.277309921	0	4.219004622	1.216552506
3	5.234500931	1.118033989	4.219004622	0	5.403702434
4	0.447213595	6.438167441	1.216552506	5.403702434	0

Case study





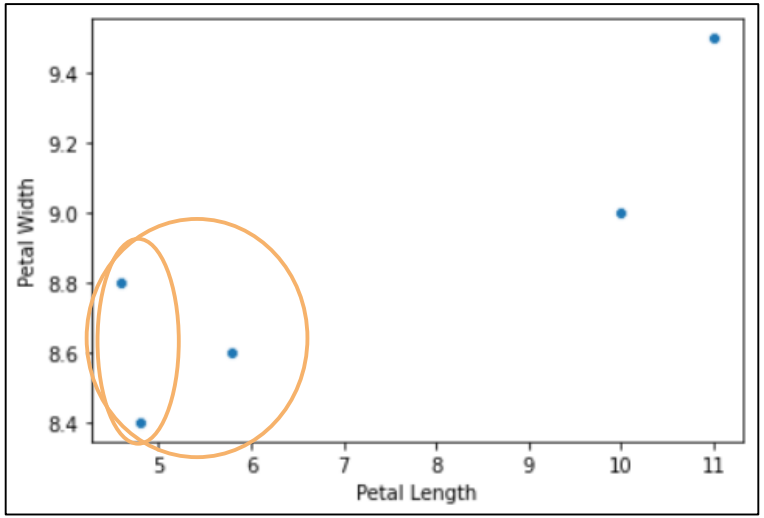
Case study

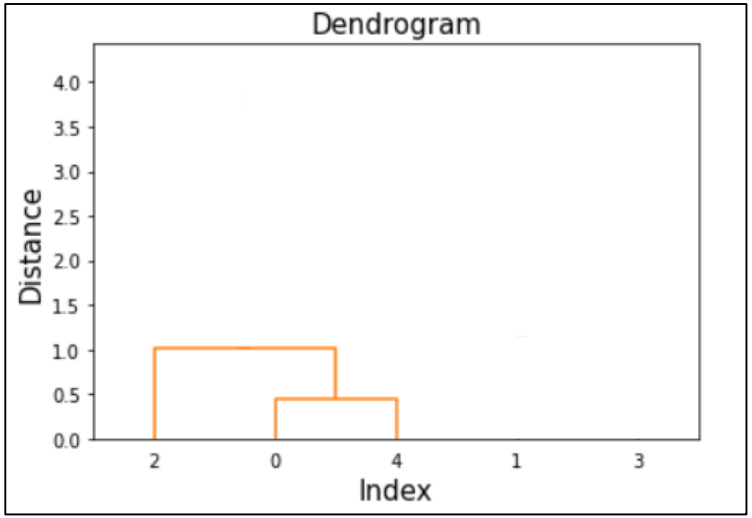
Consider (0,4) as a new cluster and calculate the distance between the cluster and data points using a single linkage. Other distances will remain the same.

The distance matrix shows that the 3rd point is closest to the cluster (0,4).

	(0,4)	1	2	3
(0,4)	0	6.296824597	1.019803903	5.234500931
1	6.296824597	0	5.277309921	1.118033989
2	1.019803903	5.277309921	0	4.219004622
3	5.234500931	1.118033989	4.219004622	0

Case study





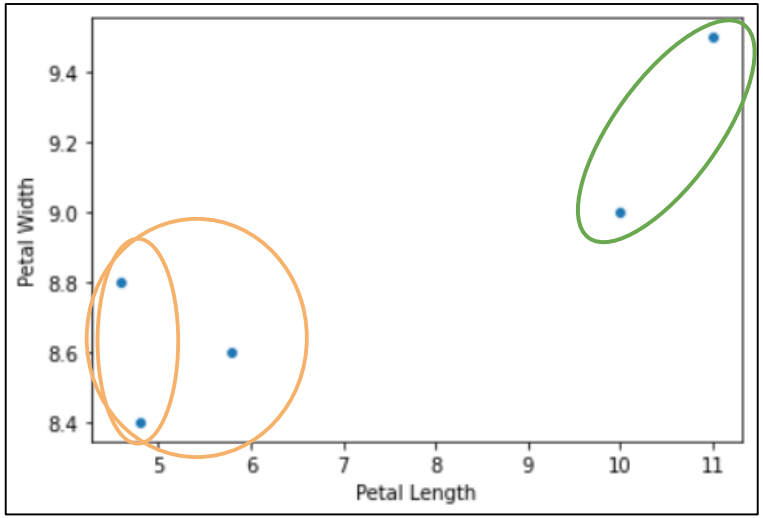
Case study

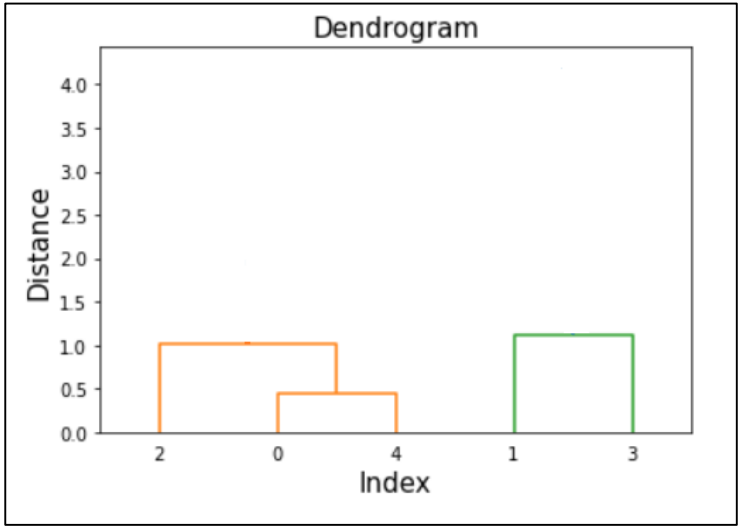
Consider ((0,4),2) as a new cluster and calculate the distance between the cluster and data points using a single linkage. Other distances will remain the same.

The distance matrix shows that the 2nd and 4th point are closest to each other.

	((0,4),2)	1	3
((0,4),2)	0	5.277309921	4.219004622
1	5.277309921	0	1.118033989
3	4.219004622	1.118033989	0

Case study





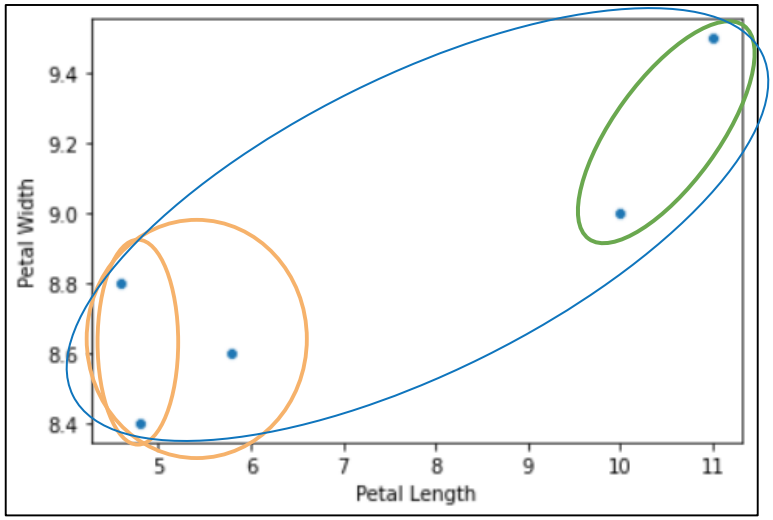
Case study

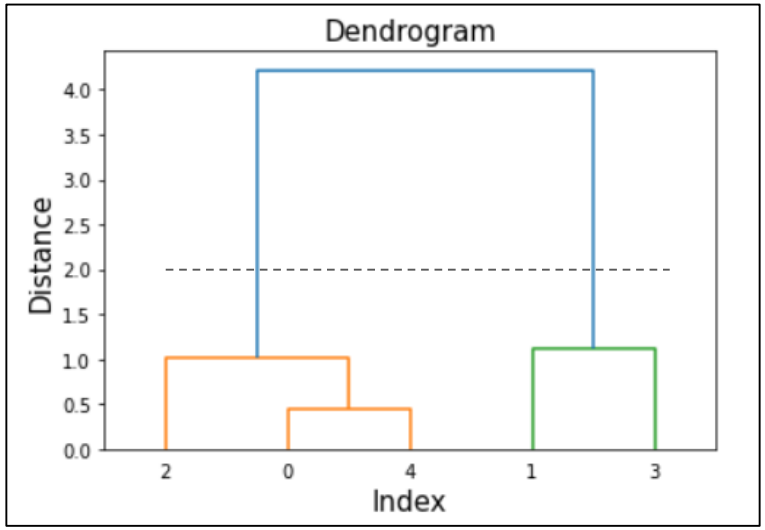
Consider (1,3) as a new cluster and calculate the distance between the clusters (1,3) and ((0,4),2) using a single linkage.

Now we have obtained the two clusters. These can be merged into a single cluster.

	((0,4),2)	(1,3)
((0,4),2)	0	4.219004622
(1,3)	4.219004622	0

Case study





If we cut the dendrogram at a height of 2.0, then we get two distinct clusters (0,2,4) and (1,3).

Case study

The cophenetic coefficient is 0.9679, which is close to 1. Thus we can say that clustering is quite good.

Cophenetic Distance	Actual Distance
4.21900462	6.2968246
1.0198039	1.0198039
4.21900462	5.23450093
0.4472136	0.4472136
4.21900462	5.27730992
1.11803399	1.11803399
4.21900462	6.43816744
4.21900462	4.21900462
1.0198039	1.21655251
4.21900462	5.40370243

Density Based Clustering

Density based clustering

- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is mostly used density-based clustering algorithm
- This technique can form clusters of non-globular shapes
- It considers a cluster as a continuous region of high density
- Regions of low density are identified as noise/ outliers

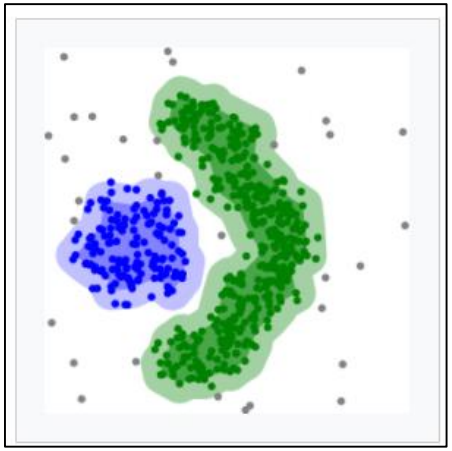


Image ref.

Parameters in DBSCAN

- There are two parameters in the DBSCAN algorithm: epsilon (ε) and the minimum number of samples (min_samples or minPts)
- epsilon (ε) is the radius of the neighbourhood for a data point
- Minimum number of samples is the lower bound for the count of data points in the neighbourhood of a core point

Terminologies

- A core point is a data point which has at least a minimum number of samples in its ε-neighbourhood (including itself) or otherwise; it is a non-core point
- A data point is ‘directly density reachable’ from a core point if it is in the ε-neighbourhood of a core point
- A border point is a point which is not a core point but, it is directly density reachable from a core point

Terminologies

- A point *s* is ‘density reachable’ from a core point (c) if there is a path p_1=c, p_2,..., p_n=*s*, such that p_{i+1} is directly density reachable from p_i
- The definition of density reachable indicates that the points on the path must be the core points (the last point can be an exception!)
- Two points are ‘density connected’ if there is a third point from which both the points are density reachable
- A point which is not density reachable from any point is known as ‘noise’ point or outlier

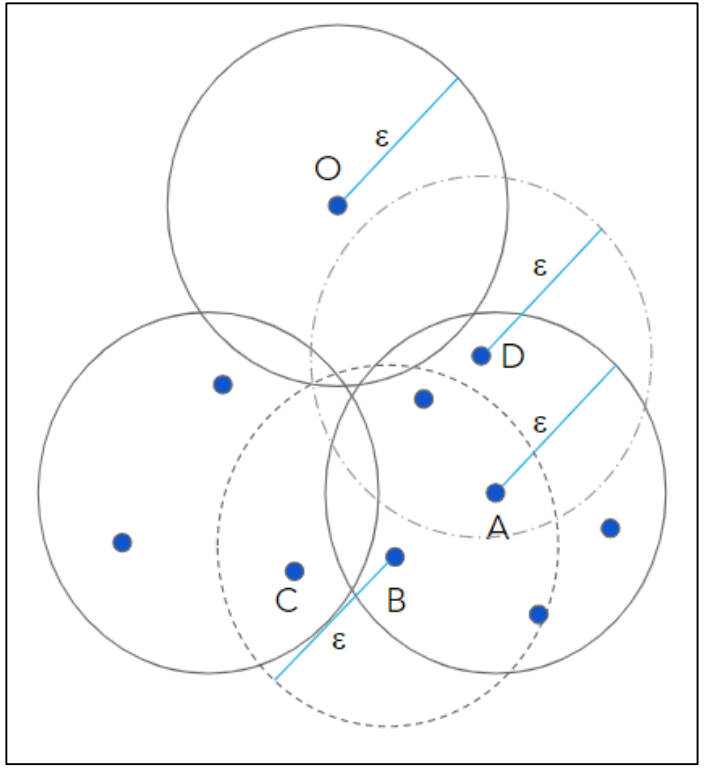
Great Learning
POWER AHEAD

All the non-core points are directly density reachable from core-points and not vice-versa.

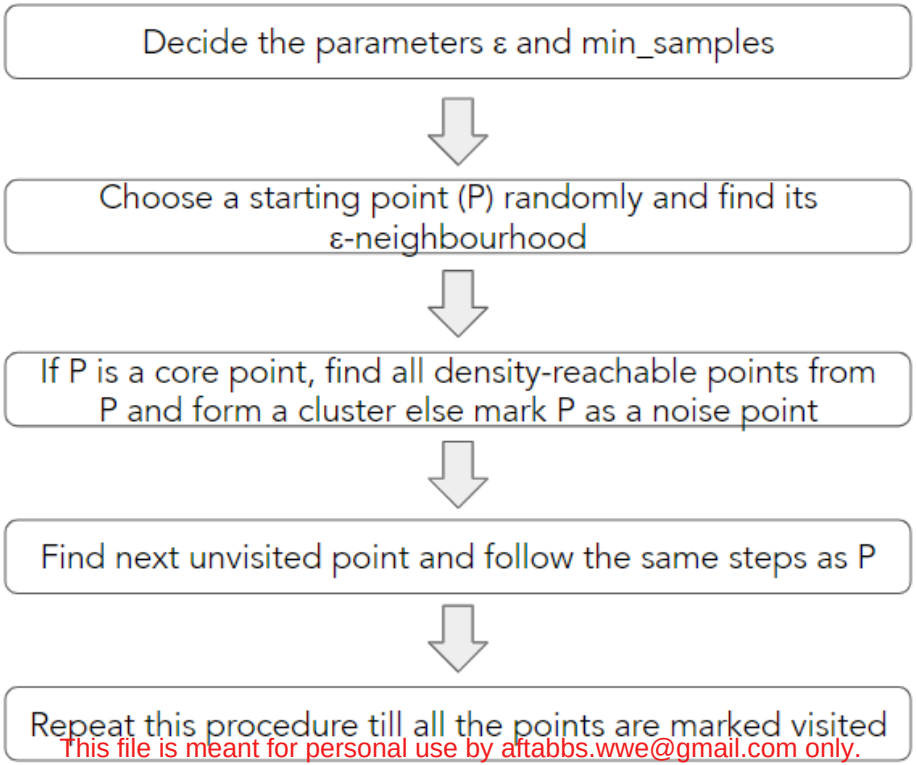
Terminologies

Consider min_samples = 5

A: Core point
B: Directly density reachable point from A
A and C are density connected and density reachable
D: Border point
O: Noise point



DBSCAN - procedure



Great Learning
POWER AHEAD

Python function

In python, the DBSCAN() performs DBSCAN clustering on the data.

```
# import the function
from sklearn.cluster import DBSCAN

# pass the epsilon radius for neighbourhood
# pass the number of minimum points
model = DBSCAN(eps = eps_radius, min_samples = m)

# fit and predict the cluster labels
model.fit_predict(data)
```

Summary

- Merits:
  - Does not require a pre-specified number of clusters
  - Useful to form clusters of any size and shape
  - Can be used to find outliers in the data
- Demerits:
  - Can not efficiently work with the clusters of varying densities
  - Does not work well with high dimensional data

WANT TO KNOW MORE?

Great Learning
POWER AHEAD

Read about:

Hierarchical DBSCAN

- A Hierarchical version of DBSCAN developed by Ricardo J.G.B. Campello, Davoud Moulavi and Joerg Sander
- Ref: Link

Summary - clustering techniques

K-means

- Partition-based most widely used algorithm
- Need to specify the number of clusters (K)
- Sensitive to noise/ outliers in the data
- Creates more circular clusters
- Cluster formation depends on the initial centroid assignment

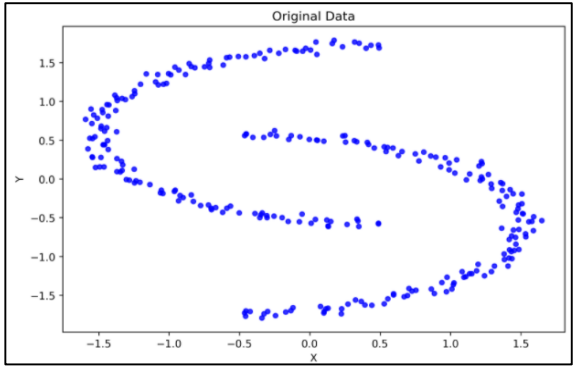
Hierarchical

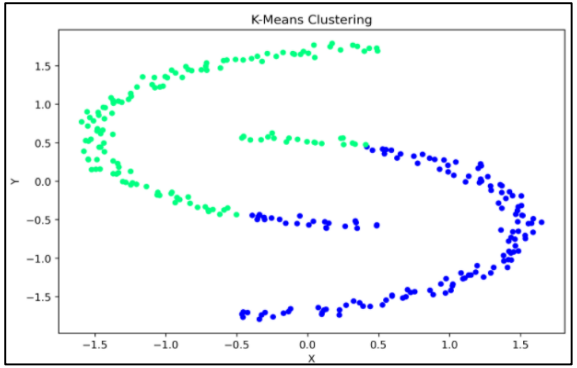
- Hierarchy-based algorithm
- No need to specify the number of clusters
- Sensitive to noise/ outliers in the data
- Useful in recovering the underlying hierarchical structure in the data
- Cluster shape depends on the distance metric

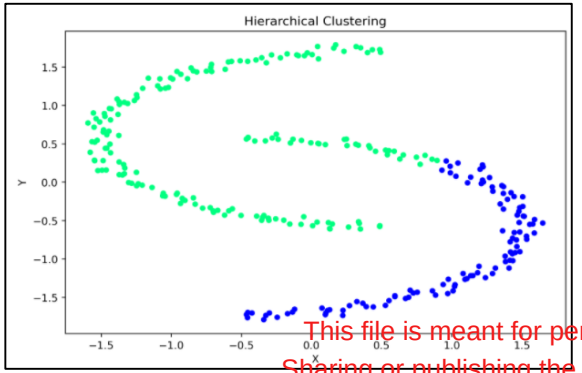
DBSCAN

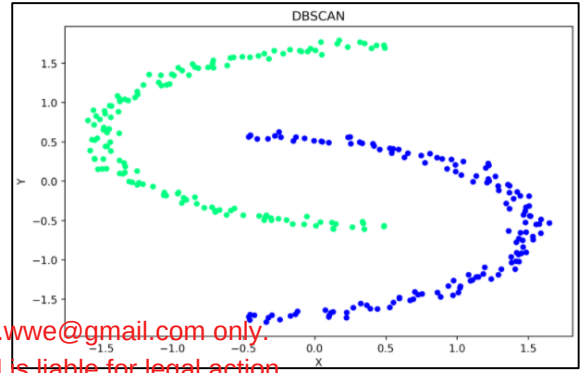
- Density-based algorithm. 2nd most widely used algorithm
- No need to specify the number of clusters
- need to specify epsilon (ε) and the minimum number of samples (min_samples)
- Less sensitive to noise. It identifies the outliers/ noise in the data
- Can create clusters of arbitrary shapes

Summary - clustering techniques









Great Learning
POWER AHEAD

Thank You