BT3041 – Analysis and Interpretation of Biological Data
Assignment -1

Name: Renuka Kolusu
Roll No. BS19B018

Question 1:

The solution to this problem is coded using Python3 and the file has been attached in the folder.

Parameters:

The parameters used to obtain 4 clusters are:
Epsilon radius (ε) : shows the maximum radius of the neighbor around a data point
Min_points : represents the minimum number of points in distance of epsilon neighborhood of a selected data point.

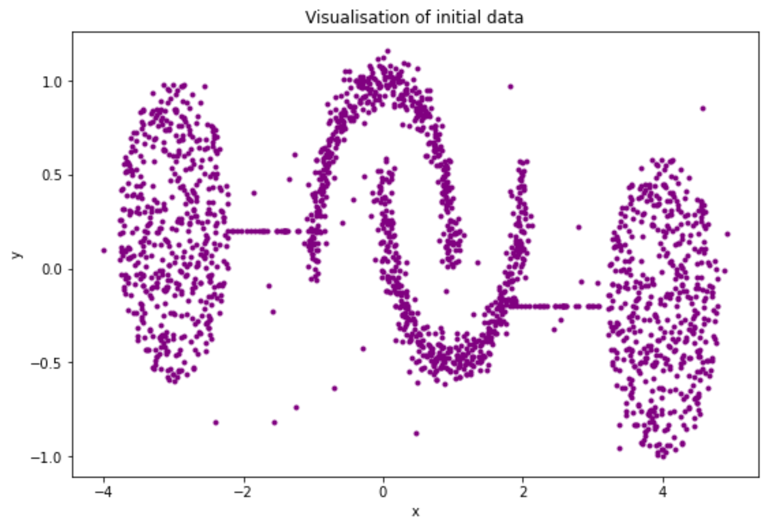
Procedure:

- The distance between all pairs of points are calculated
- The number of nieghbours around a point is calculated by considering all distances that are less than eps.
- Points that have more than min_points number of neighbor are labelled as ***core points***
- Points that have less than min_points number of neighbors within eps but is in neighborhood of a core point are labelled as ***border points***
- Points that have less than min_points number of neighbors and are not neighbors of core points are considered as outliers and labelled as ***noise points***
- Cluster allocation is done in way where all accessible core points and their neighbors are assigned the same cluster number
- Once all the accessible core points and their neighbors are given a cluster label, the current cluster label is incremented and the cluster allocation is done using the next set of core points.

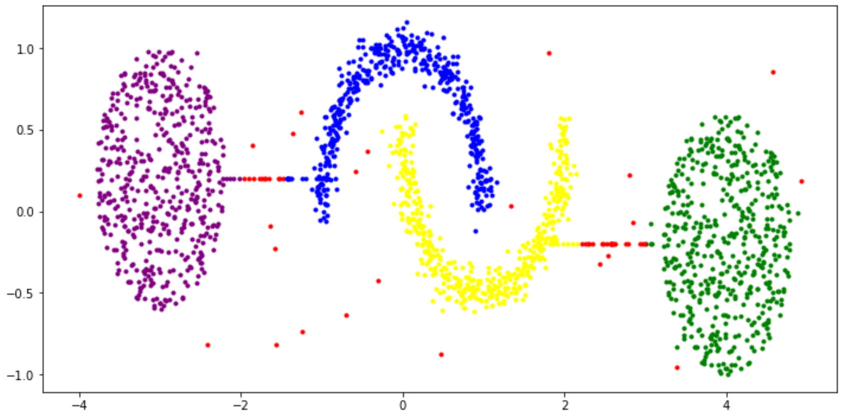
Code details:

Function ***def neighhbor_p(data,p,max_dist)*** to find all neighbors of a given point within a specified radius
Function ***dbscan(data, eps, minpts)*** will do is
Initially classifies and labels points as core and noise depending on the minimum poins within epsilon radius from the point
Now classifying the noise points that lie within epsilon radisu of core points as border points
And then forming clusters by adding the core and border points in the neighborhood of a core point
And if a cluster has points less than min_points it is classified as noise
And return the complete resultant data as a dictionary and also the points label list of number of points in particular cluster and noise points as well
Function ***plot(data,pt_label)*** gives the data visualization of that data clustering

The Data visualisation of dataset given before clustering:

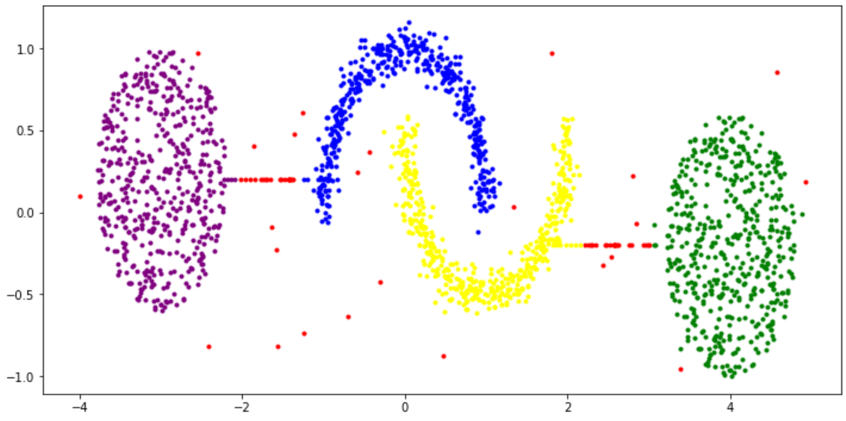


The Data visualisation of dataset given after clustering: at epsilon 0.23 and min_pts=25



Cluster 1 : Purple
Cluster 2: Blue
Cluster 3: Green
Cluster 4: yellow
Noise points: Red

The Data visualisation of dataset given after clustering: at epsilon 0.21 and min_pts=25

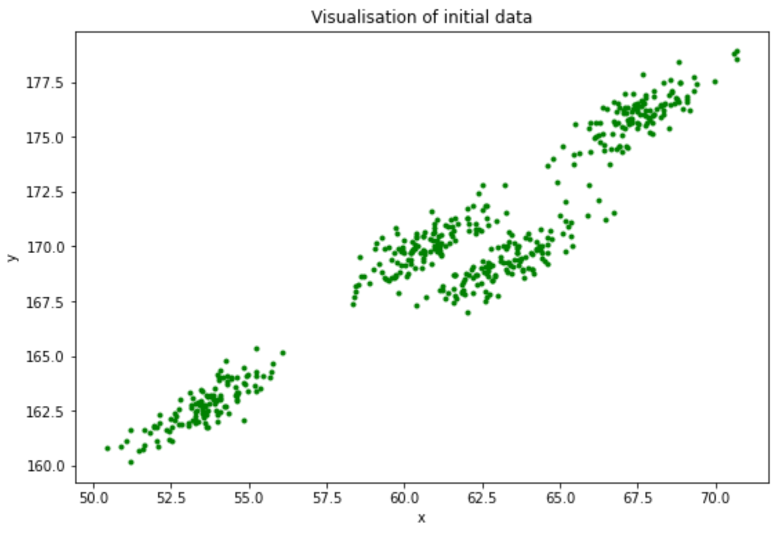


And resultant clustered data:

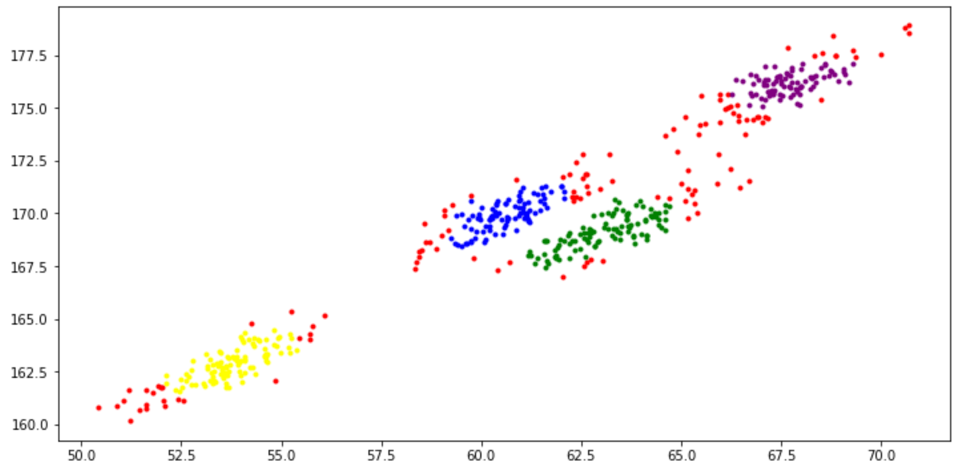
Total number of points 2000

Number of clusters	4
Number of points in cluster 1	509
Number of points in cluster 2	465
Number of points in cluster 3	510
Number of points in cluster 4	462
Number of noise points	54

The Data visualisation of dataset given before clustering:



The Data visualisation of dataset given after clustering: at epsilon 0.5 and min_pts=10



Cluster 1 : Purple
Cluster 2: Blue
Cluster 3: Green
Cluster 4: yellow
Noise points: Red

And resultant clustered data:
Total number of points: 500

Number of clusters	4
Number of points in cluster 1	85
Number of points in cluster 2	91
Number of points in cluster 3	102
Number of points in cluster 4	100
Number of noise points	122

Conclusion:

This illustrates how effective DBSCAN is at clustering data points according to how they are disturbed throughout the dataset. Although the only dependent parameters that affect the clustering pattern are eps and min_pts, it is not totally deterministic because the density varies over the range and makes it difficult to define the borders. It has the advantage that clusters of different shapes can be clearly identified using this approach.