Assignment 3

1. Conceptual Question

- Client A is a biotech company working on a large-scale genome sequencing project. They need to cluster genome sequences to identify patterns and anomalies.

  Using hierarchical clustering, it would allow for anomalies to be easily spotted with outliers to the major groups. The hierarchical nature of splitting a genome sequence makes this method of clustering ideal for the use case, also allowing us to model by genome.

- Client B is Amazon. They have a massive inventory catalogue with more than 1 million records. They want to cluster their products for better inventory management.

  K-means clustering could work well for this massive dataset as it scales well. Considering a client like Amazon, it would also be realistic so assume they know how many categories of products they have, something we can cluster the records by. By calculating distances between these products we can discover similar products easily. Prices and popularity can also be taken into consideration, popularity especially for updating inventories.

- Client C is an online encyclopaedia platform similar to Wikipedia. They want to cluster their vast number of topics to improve navigation for their users.

  DBScan would be a nice choice for this client. Allowing common topics to be clustered together, something that could help with placing articles into navigation groups and routes. Calculating similarities between articles would help this case further, categorising by similar contents.

3. DBSCAN Clustering

a. To perform DBSCAN on the data points we first need to calculate a distance matrix: *(the code for this can be found at the bottom of the notebook)*

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0.000000	14.764823	15.556349	9.000000	10.049876	5.385165	14.866069	14.764823	6.324555	9.000000	11.000000	8.246211	10.770330	6.708204	12.165525
1	14.764823	0.000000	4.472136	8.062258	12.369317	11.180340	5.000000	0.000000	8.602325	8.062258	13.601471	11.045361	4.242641	8.062258	5.099020
2	15.556349	4.472136	0.000000	11.180340	10.049876	10.816654	1.000000	4.472136	10.295630	11.180340	11.000000	9.486833	7.071068	9.433981	9.055385
3	9.000000	8.062258	11.180340	0.000000	12.806248	8.602325	11.045361	8.062258	3.605551	0.000000	14.212670	10.630146	4.123106	4.242641	3.605551
4	10.049876	12.369317	10.049876	12.806248	0.000000	5.099020	9.055385	12.369317	9.433981	12.806248	1.414214	2.236068	10.816654	8.602325	13.601471
5	5.385165	11.180340	10.816654	8.602325	5.099020	0.000000	10.000000	11.180340	5.000000	8.602325	6.324555	3.000000	8.062258	4.472136	10.440307
6	14.866069	5.000000	1.000000	11.045361	9.055385	10.000000	0.000000	5.000000	9.848858	11.045361	10.000000	8.544004	7.000000	8.944272	9.219544
7	14.764823	0.000000	4.472136	8.062258	12.369317	11.180340	5.000000	0.000000	8.602325	8.062258	13.601471	11.045361	4.242641	8.062258	5.099020
8	6.324555	8.602325	10.295630	3.605551	9.433981	5.000000	9.848858	8.602325	0.000000	3.605551	10.816654	7.211103	4.472136	1.000000	6.000000
9	9.000000	8.062258	11.180340	0.000000	12.806248	8.602325	11.045361	8.062258	3.605551	0.000000	14.212670	10.630146	4.123106	4.242641	3.605551
10	11.000000	13.601471	11.000000	14.212670	1.414214	6.324555	10.000000	13.601471	10.816654	14.212670	0.000000	3.605551	12.206556	10.000000	15.000000
11	8.246211	11.045361	9.486833	10.630146	2.236068	3.000000	8.544004	11.045361	7.211103	10.630146	3.605551	0.000000	8.944272	6.403124	11.661904
12	10.770330	4.242641	7.071068	4.123106	10.816654	8.062258	7.000000	4.242641	4.472136	4.123106	12.206556	8.944272	0.000000	4.123106	2.828427
13	6.708204	8.062258	9.433981	4.242641	8.602325	4.472136	8.944272	8.062258	1.000000	4.242641	10.000000	6.403124	4.123106	0.000000	6.082763
14	12.165525	5.099020	9.055385	3.605551	13.601471	10.440307	9.219544	5.099020	6.000000	3.605551	15.000000	11.661904	2.828427	6.082763	0.000000

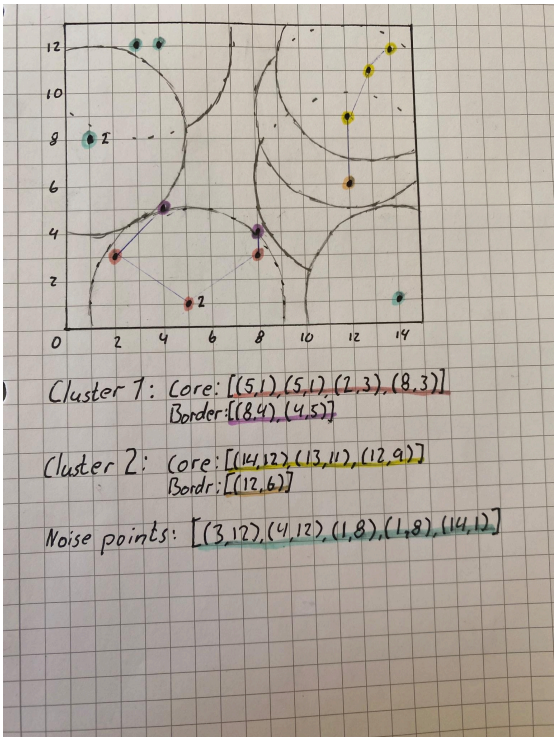
If we then select every row with at least MinPts number of values less than Eps we can see our clusters' core points:

If we look at p_3 we see that its distance to points p_9, p_14 and p_8 are less than Eps = 4.

Furthermore, we can choose our border points if they have a distance less than Eps but are not in the cluster.

Take point p_12, its distance to p_14 is lesser than Eps, but only to one point within the cluster, therefore it becomes a border point as it does not have enough neighbours in the cluster to become a core point, but its distance is not far enough to be considered a noise point.

Going through this method for every point we get:



Cluster 1 core points

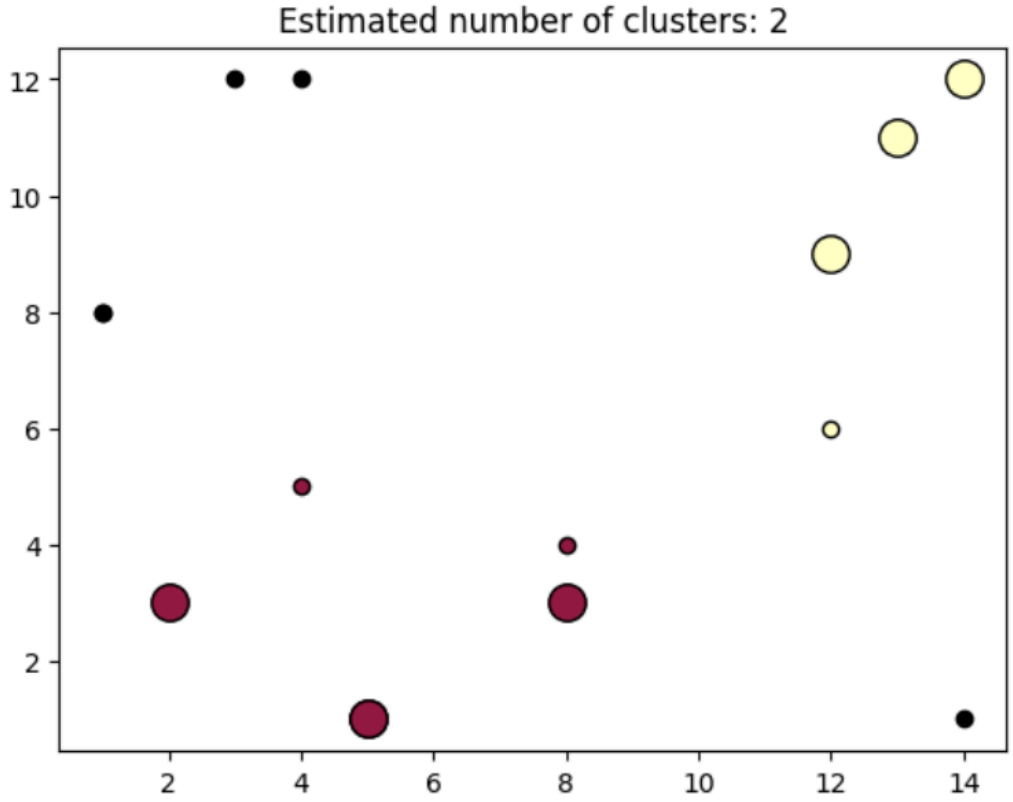
Cluster 1 border points

Cluster 2 core points

Cluster 2 border points

Noise points

b. As we can see this aligns with the results in the notebook:



4. Hierarchical Agglomerative Clustering (HAC)

a. HAC works by iteratively merging the two closest clusters together. It starts off seeing every point as its own cluster before merging the two closest until only one remains. Looking at two of the different methods we can use with HAC, MAX- and MIN-link; MAX-link will look at the two furthest away points between clusters, the lowest of these will determine which clusters are merged. On the contrary to this MIN-link looks at the two closest points between clusters, and merges based on that metric.

   HAC can give you useful hierarchies of the data, allowing you to 'cut' into whichever level you deem fit.

b. Firstly we have to create a distance matrix for the data points:

	A	**B**	**C**	**D**	**E**
A	0	4.123	4.123	1.000	7.211
B	4.123	0	7.071	3.162	7.000
C	4.123	7.071	0	4.472	5.385
D	1.000	3.162	4.472	0	6.708
E	7.211	7.000	5.385	6.708	0

   To begin with every point is considered its own cluster.

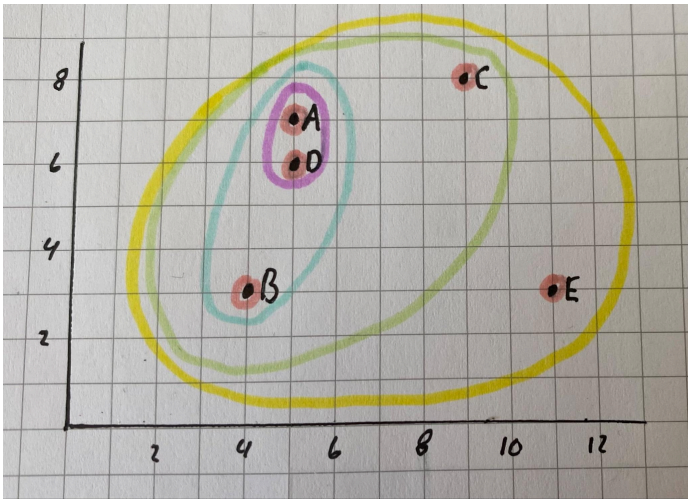
   Using the distance matrix we can start by merging the two closest points: This causes A to be merged with D to form the first cluster.

As we are using MIN-link for the first example, we look for the two closest points. The closest points are B and D, then the clusters now are [A, D, B], with the rest alone in their own clusters. We keep doing this until every point is covered:

2: [A, D, B, C]

3: [A, D, B, C, E]

This can be visualised as such:



Cluster 1

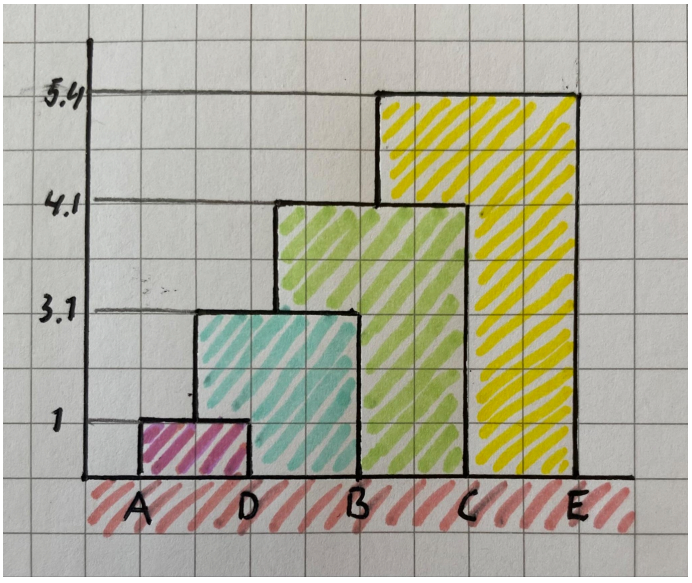
Cluster 2

Cluster 3

Cluster 4

Cluster 5

The related dendrogram can be visualised as such:



It seems to me that both MIN-link and MAX-link will give the same result on this dataset (?)