Assignment – 3

Apurva Mandalika **UIN - 334003575**

Question - 2. DBSCAN Clustering

Given Dataset →

a - (5,8) g - (7,4)

b - (6,7) h - (9,4)

c - (6,5) i - (3,3)

d - (2,4) j - (8,2)

e - (3,4) k - (7,5)

f - (5,4)

① Using $\epsilon = 2$, minpts $= 5$

$$d_\infty (x,y) = \max_{i=1}^{d} \{ |x_i - y_i| \}.$$

Distance matrix for datapoints.

	a	b	c	d	e	f	g	h	i	j	k	
a	0	1	3	4	4	4	4	4	5	6	3	a: b
b	1	0	2	4	3	3	3	3	4	5	2	b: a, c, k
c	3	2	0	4	3	1	1	3	3	3	1	c: b, f, g, k
d	4	4	4	0	1	3	5	7	1	6	5	d: e, i.
e	4	3	3	1	0	2	4	6	1	5	4	e: d, f, i
f	4	3	1	3	2	0	2	4	2	3	2	f: c, e, g, i, k
g	4	3	1	5	4	2	0	2	4	2	1	g: c, f, h, j, k
h	4	3	3	7	6	4	2	0	6	2	2	h: g, j, k
i	5	4	3	1	1	2	4	6	0	5	4	i: d, e, f
j	6	5	3	6	5	3	2	2	5	0	3	j: g, h
k	3	2	1	5	4	2	1	2	4	3	0	k: b, c, f, g, h.

Point	Status
a	Noise
b	Noise → Border
$\boxed{c}$	Core
d	Noise
e	Noise → Border
$\boxed{f}$	Core
$\boxed{g}$	Core
h	Noise → Border
i	Noise → Border
j	Noise → Border
$\boxed{k}$	Core

After calculation we get 4 core points, 5 Border points and 2 noise points.

⇒ Core points → c, f, g, k

c : b, f, g, k

f : c, e, g, i, k

g : c, f, h, j, k

k : b, c, f, g, h.

⇒ Border points → b, e, h, i, j

⇒ Noise points → a, d.

② Using $\epsilon = 1$, minpts = 6 and L_{min}

$$L_{min}(x, y) = \min_{i=1}^{d} \{|x_i - y_i|\}$$

Distance matrix for datapoints

	a	b	c	d	e	f	g	h	i	j	k	
a	0	1	1	3	2	0	2	4	2	3	2	a: b, c, f
b	1	0	0	3	3	1	1	3	3	2	1	b: a, c, f, g k
c	1	0	0	1	1	1	1	1	2	2	0	c: a, b, d, e, f, g, h, k
d	3	3	1	0	0	0	0	0	1	2	1	d: c, e, f, g, h, i, k.
e	2	3	1	0	0	0	0	0	0	2	1	e: c, d, f, g, h, i, k.
f	0	1	1	0	0	0	0	0	1	2	1	f: a, b, c, d, e, g, h, i, k.
g	2	1	1	0	0	0	0	0	1	1	0	g: b, c, d, e, f, h, i, j, k.
h	4	3	1	0	0	0	0	0	1	1	1	h: c, d, e, f, g, i, j, k.
i	2	3	2	1	0	1	1	1	0	1	2	i: d, e, f, g, h, j.
j	3	2	2	2	2	2	1	1	1	0	1	j: g, h, i, k
k	2	1	0	1	1	1	0	1	2	1	0	k: b, c, d, e, f, g, h, j.

Point	Status
a	Noise → Border
b	Core
c	Core
d	Core
e	Core
f	Core
g	Core
h	Core
i	Core
j	Noise → Border
k	Core

After calculation we get 9 core points and 2 Border points.

⇒ Core points = b, c, d, e, f, g, h, i, k.

⇒ Border points = a, j.

⇒ Noise points = 0 (Null)

Q.1(b)

	a	b	c	d	e	f	g	h	i	j	k
a	0	2	4	7	6	4	6	8	7	9	5
b		0	2	7	6	4	4	6	7	7	3
c			0	5	4	2	2	4	5	5	1
d				0	1	3	5	7	2	8	6
e					0	2	4	6	1	7	5
f						0	2	4	3	5	3
g							0	2	5	3	1
h								0	7	3	3
i									0	6	6
j										0	4
k											0

a = (5,8)
b = (6,7)
c = (6,5)
d = (2,4)
e = (3,4)
f = (5,4)
g = (7,4)
h = (9,4)
i = (3,3)
j = (8,2)
k = (7,5)

	e,i	a	b	c	d	f	g	h	j	k
e,i	0	7	7	5	2	3	5	7	6	6
a	7	0	2	4	7	4	6	8	9	5
b			0	2	7	4	4	6	7	3
c				0	5	2	2	4	5	1
d					0	3	5	7	8	6
f						0	2	4	5	3
g							0	2	3	1
h								0	3	3
j									0	4
k										0

(e,i) = (3,3)

	e,i	c,k	a	b	d	f	g	h	j
e,i	0	5	7	7	2	3	5	7	6
c,k	5	0	4	2	5	2	2	4	5
a			0	2	7	4	6	8	9
b				0	7	4	4	6	7
d					0	3	5	7	8
f						0	2	4	5
g							0	2	3
h								0	3
j									0

$(c,k) = (6,5)$

↓

	e,i,d	c,k	a	b	f	g	h	j
e,i,d	0	6	8	8	4	6	8	7
c,k	6	0	4	2	2	2	4	5
a			0	2	4	6	8	9
b				0	4	4	6	7
f					0	2	4	5
g						0	2	3
h							0	3
j								0

$(e,i,d) = (2,3)$

↓

	e,i,d	c,k,g	a	b	f	h	j
e,i,d	0	5	8	8	4	8	7
c,k,g	5	0	5	3	1	3	4
a			0	2	4	8	9
b				0	4	6	7
f					0	4	5
h						0	3
j							0

(c,k,g) = (6,4).

	e,i,d	c,k,g,f	a	b	h	j
e,i,d	0	4	8	8	8	7
c,k,g,f	4	0	4	2	4	5
a			0	2	8	9
b				0	6	7
h					0	3
j						0

c,k,g,f = (5,4)

$\downarrow$

	e,i,d	c,k,g f,b	a	h	j
e,i,d	0	4	4	4	5
c,k,g, f,b	4	0	4	4	5
a			0	8	9
h				0	3
j					0

c,k,g,f,b = (5,4).

{e,i,b}, {c,k,g,f,b}, {a}, {h,j}.

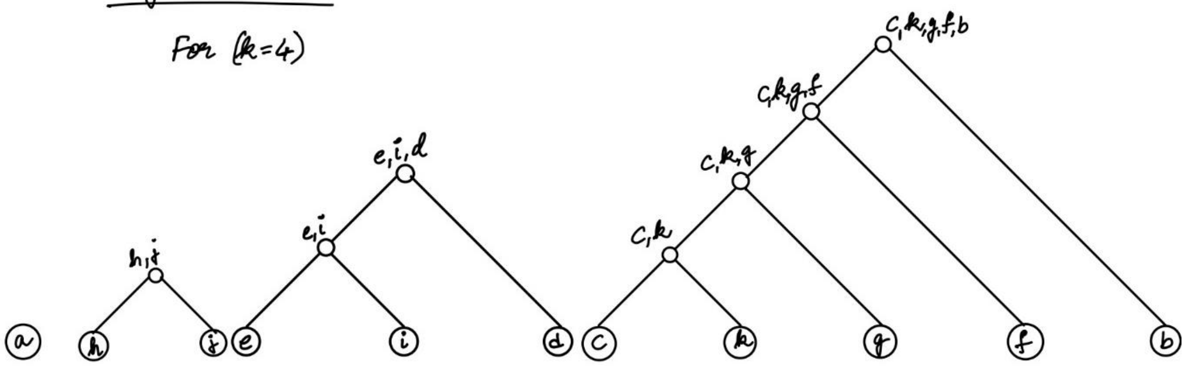
Merge Order Table

[Taking minimum feature values of data points].

Clustering	Clusters
C_1	{a} {b} {c} {d} {e} {f} {g} {h} {i} {j} {k}
C_2	{e, i} {a} {b} {c} {d} {f} {g} {h} {j} {k}
C_3	{e, i} {c, k} {a} {b} {d} {f} {g} {h} {j}
C_4	{e, i, d} {c, k} {a} {b} {f} {g} {h} {j}.
C_5	{e, i, d} {c, k, g} {a} {b} {f} {h} {j}
C_6	{e, i, d} {c, k, g, f} {a} {b} {h} {j}
C_7	{e, i, d} {c, k, g, f, b} {a} {h} {j}
C_8	{e, i, d} {c, k, g, f, b} {h, j} {a}.

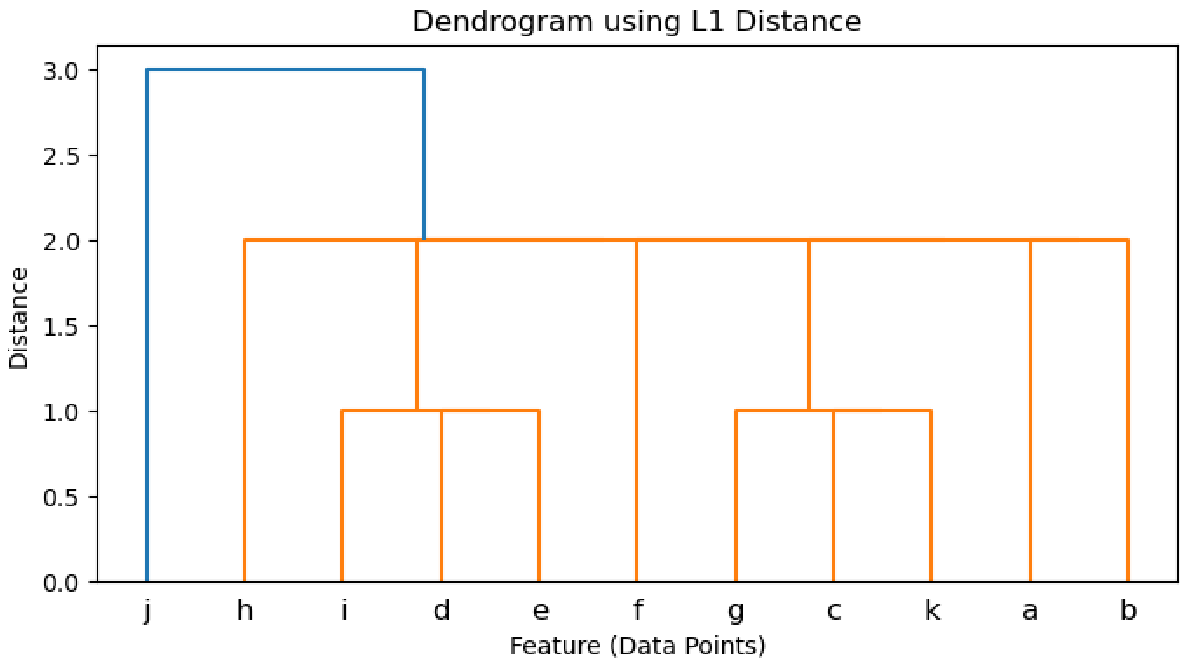
Merge Order Tree

For (k=4)



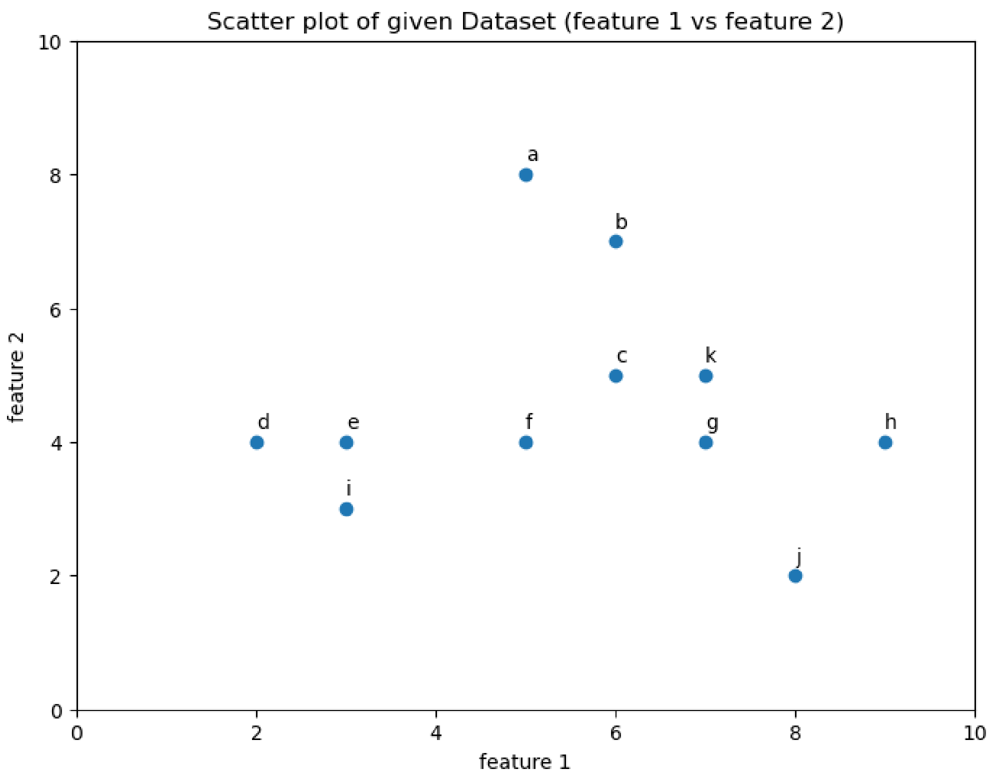
1. **Hierarchical Clustering**

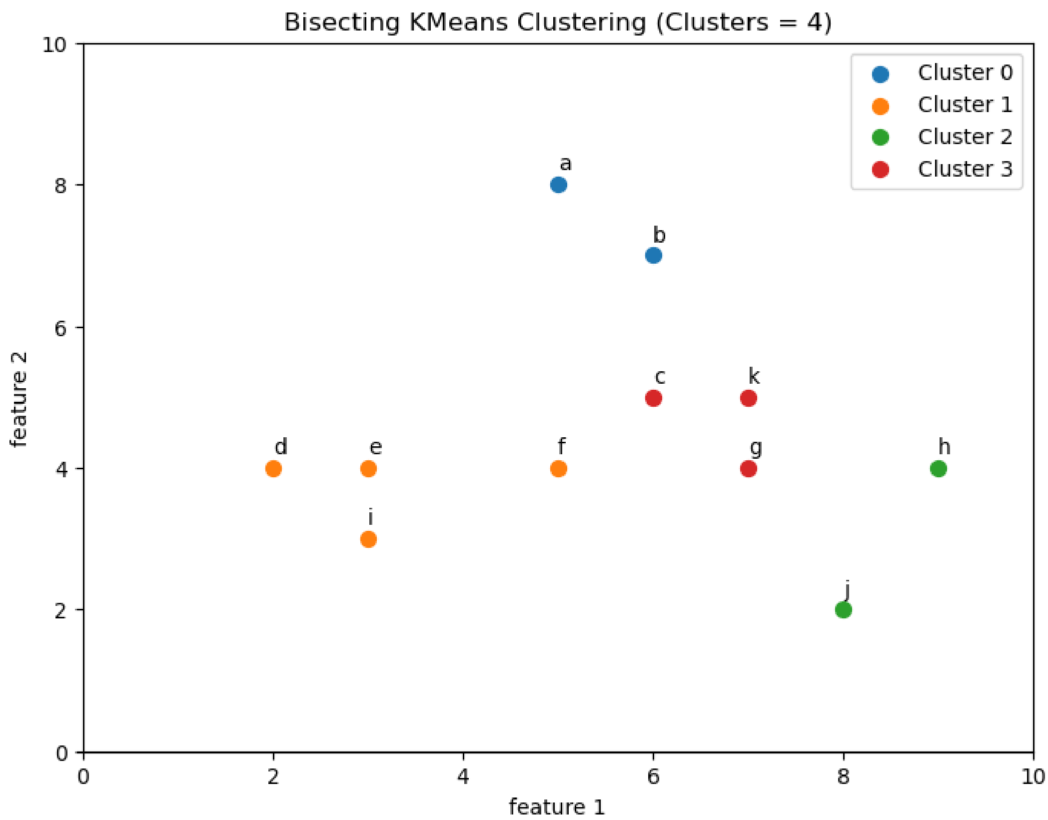
   a.



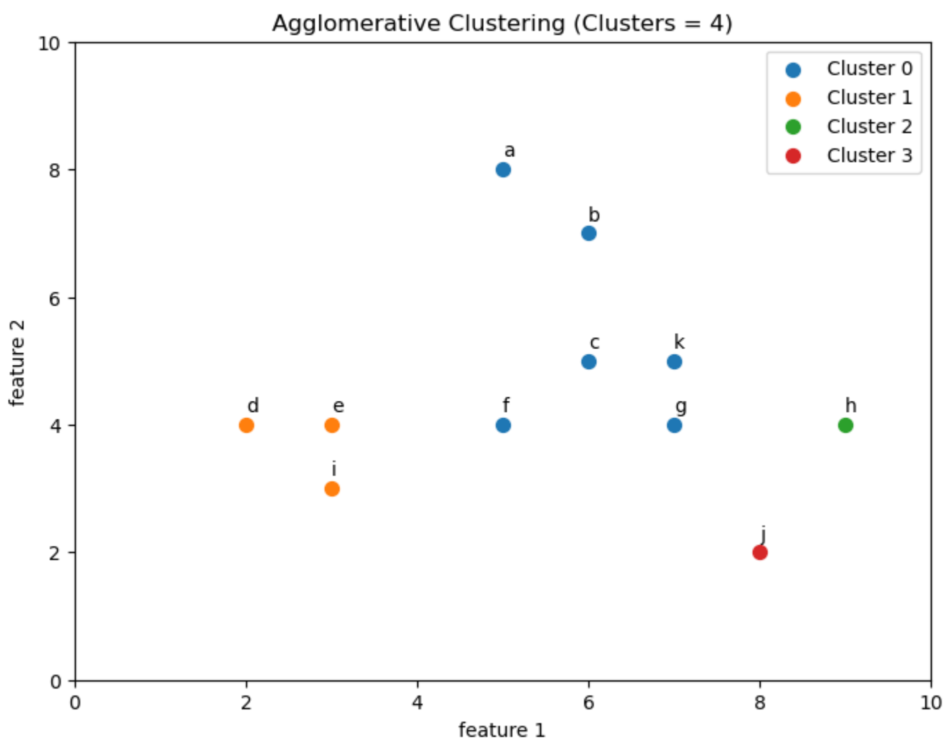
   b. **Showed above – solved by hand**

   c.





d.



Agglomerative Clusters: array([0, 0, 0, 1, 1, 0, 0, 2, 1, 3, 0], dtype=int64)

Bisecting KMeans Clusters: array([0, 0, 3, 1, 1, 1, 3, 2, 1, 2, 3])

```
Adjusted Rand Index: 0.3119266055045872
```

This score is positive. It means that there is some agreement between the two clustering methods, more than what would be expected by random chance. However, since the score is closer to 0 than to 1, it shows that the agreement between the two clustering results is moderate to low.

Discussion:

1. The two clustering approaches do not achieve the same results for this dataset. While they both identify 4 clusters, the assignment of data points to these clusters differs significantly.
2. Agglomerative clustering tends to create one large cluster (Cluster 0 with 6 elements) and two singleton clusters. This suggests that it might be identifying one main group and treating some points as outliers or highly distinct cases.
3. Bisecting K-Means, on the other hand, creates more balanced clusters, with sizes ranging from 2 to 4 elements. This suggests a more even partitioning of the data space, which might be preferable if you expect roughly equal-sized groups in your data.
4. The moderate ARI score (0.3119) confirms that while there is some agreement between the two methods, they produce notably different clusterings.
5. The agglomerative method seems to be more sensitive to potential outliers, as evidenced by the singleton clusters. Bisecting K-Means appears to be forcing these points into larger clusters, which might be beneficial if outliers are not a primary concern in your analysis.
6. The two methods agree on only a few point assignments, indicating that they are capturing different aspects of the data structure. This could be due to the inherent differences in how these algorithms work:

- Agglomerative clustering builds a hierarchy from the bottom-up
- Bisecting K-Means recursively splits clusters from the top-down

2. Shown above – solved by hand

3. a.

b.

c. `Silhouette Score for mean shift clustering: 0.77`

2. `Estimated Bandwidth: 0.73`

```
Silhouette Score for estimated bandwidth: 0.66
```

Estimated bandwith clustering leads to the optimal clustering of the dataset because for the given species - 3 the estimated band width clustered it in 3 clusters while the Mean shift clustering clustered it in 2 clusters.

4. **Gaussian Naive Bayes**

Training set Confusion Matrix
```
array([[47, 25],
       [28, 40]], dtype=int64)
```

Testing set Confusion Matrix
```
array([[19,  9],
       [14, 18]], dtype=int64)
```

Accuracy Score: 0.6166666666666667
Precision Score: 0.6666666666666666
Recall Score: 0.5625
F1 Score: 0.6101694915254238

K Nearest Neighbours

Training set Confusion Matrix
```
array([[67,  5],
       [ 2, 66]], dtype=int64)
```

Testing set Confusion Matrix
```
array([[23,  5],
       [ 3, 29]], dtype=int64)
```

Accuracy Score: 0.8666666666666667
Precision Score: 0.8529411764705882
Recall Score: 0.90625
F1 Score: 0.8787878787878787

Here for K nearest neighbours the F1 score, accuracy score, precision score and recall score, all are higher than that of the Gaussian bayes'. Hence KNN performs better than Gaussian Naive Bayes.

KNN

Advantages

1. KNN is intuitive and simple. It has no assumptions.
2. It is very easy to implement for multiclass problems and can be used for both classification and regression.

Disadvantages

1. KNN is a slow algorithm.
2. Imbalance data in KNN causes problems.

GNB

Advantages

1. Fast and flexible model gives highly reliable results.
2. Works well with large dataset.

Disadvantages

1. Large data record are required to achieve good results.
2. Sometimes shows lower performance than other classifiers.