2014CS50280
ASHLEY JAIN

HW 3

1 (a)

Point	x	y
1	0.4	0.53
2	0.22	0.38
3	0.35	0.32
4	0.26	0.19
5	0.08	0.41
6	0.45	0.3

⇒ Euclidean Distance

	1	2	3	4	5	6
1	0	-	-	-	-	-
2	0.234	0				
3	0.215	0.143	0			
4	0....	0.194	0.158	0		
5	....	0.143	...	...	0	
6	0.235	...	0.102	0.219	...	0

(symmetry matrix)

0.102 → shortest length

3, 6 → (3, 6) ⇒ one cluster

⇒

	1	2	3,6	4	5
1	0				
2	0.234	0			
3,6	...	0.143	0		
4	...	0.194	0.158	0	
5	...	0.143	...	...	0

5 (3,6) 2

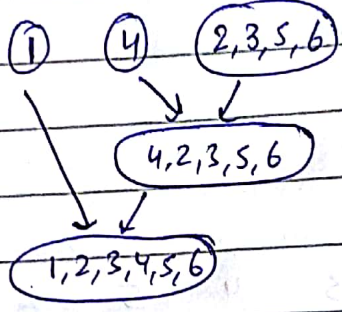
5 2 3,6

2 + 3,6 → 2,3,6 ⇒ new cluster

5 + 2,3,6 → 5,2,3,6 ⇒ new cluster

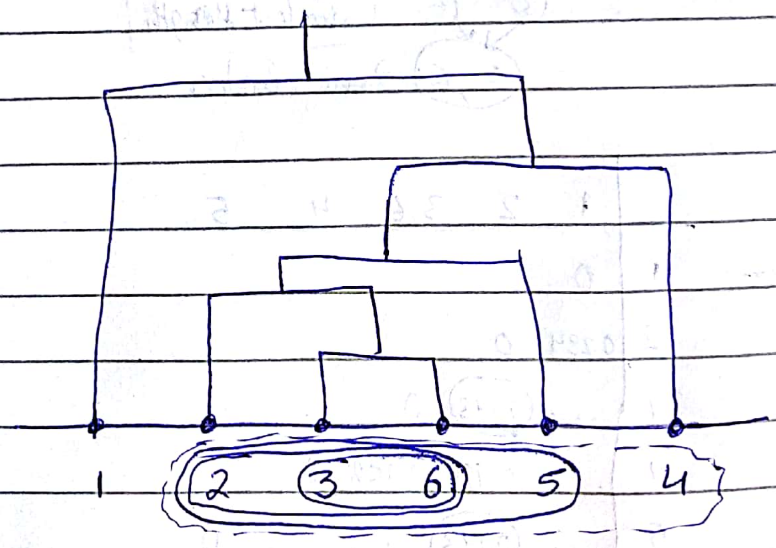
⇒

	1	2,3,5,6	4
1	0		
2,3,5,6	0.215	0	
4		0.158	0



Final answer.

Dendogram :-



(finding minimum value)

Date ___/___/___

Page ______

1 (b) ⇒ From previous part, we compute distance b/w all pair of (n) points ⇒ $O(n^2) + O(n) \approx O(n^2)$

⇒ Then again (after joining two point into single cluster), we compute distance b/w (n-1) objects. ⇒ $O[(n-1)^2]$

⋮

$O(1^2)$

Complexity ⇒ $O(n^2) + O[(n-1)^2] + O[(n-2)^2] + \dots$

$$\Rightarrow n^2 + (n-1)^2 + (n-2)^2 + \dots 1^2 = \sum_{i=1}^{n} i^2$$

$$= \frac{n(n+1)(2n+1)}{6}$$

$$= O(n^3).$$

∴ Computational Complexity ⇒ $O(n^3)$.

1 (c) ~~We ca~~ There could be better algorithm than naive algorithm $[O(n^3)]$ by using Minimum Spanning tree.

Consider 'E' $(=n^2)$ edges between every 'V' $(=n)$ points.

1. Sort all edges in non-decreasing order of their Euclidean distance as weight.
2. Pick smallest edge weight, Check if it forms a cycle with spanning tree formed so far (that means they are already in same cluster). If cycle is not formed, include this edge (merge two cluster into one), else discard it.

Date __/__/__

Page __

mycompanion

3. Repeat step (2) until there are $(n-1)$ edges in spanning tree. (Only one big cluster).

Complexity would be $O(E \log V)$ by definition.

And, $E = n^2$, $V = n$

then $O(n^2 \log n)$. Ans.

Scanned by CamScanner