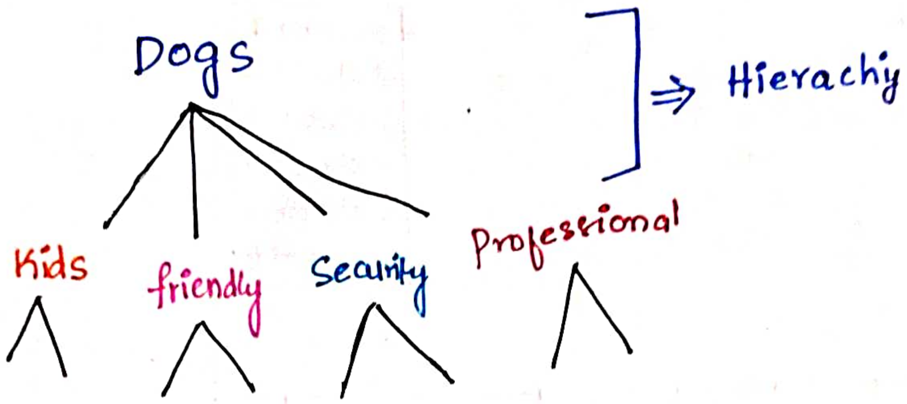
* Hierarchical Clustering [HC]

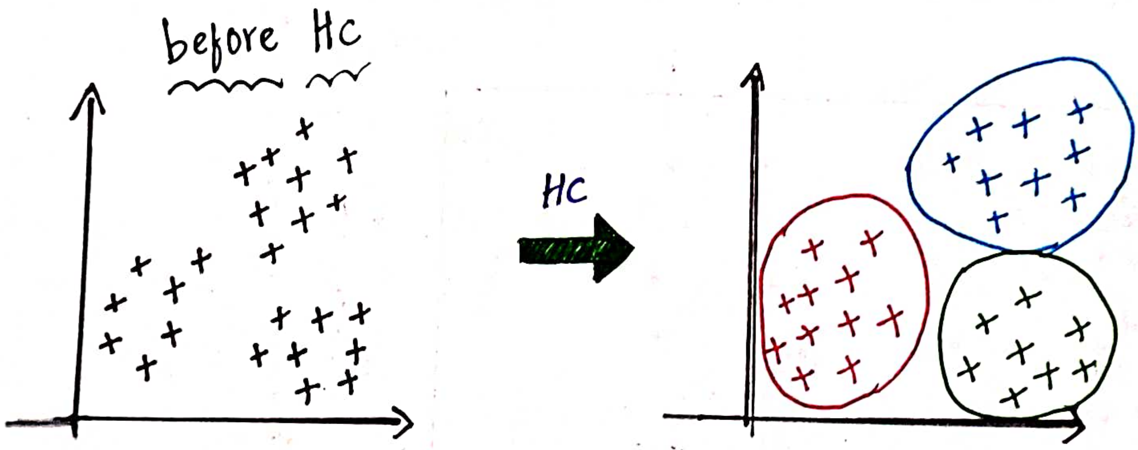
Combine data points which are very close by distance

Ex:-



* Hierarchical clustering algorithms build a hierarchy of clusters where "Each node is a cluster" consists of clusters of its daughter nodes.

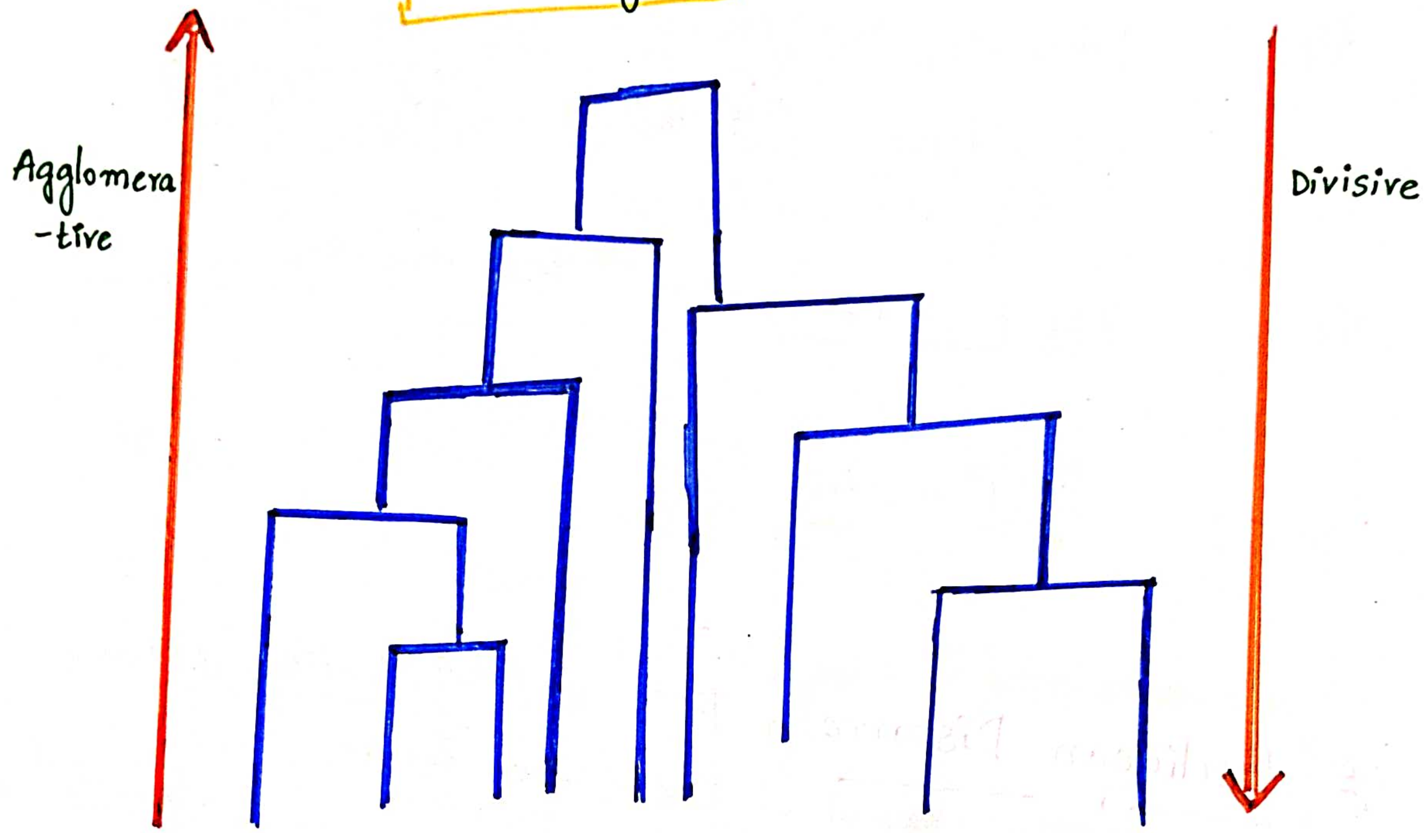
What "HC" does ?



Same as K-means but different process.

* Hierarchical clustering

Dendogram



* They are two types clussering.
    * Algomerative (From bottom to top)
    * Divisive (From Top to bottom)

Most of The Time, we use, This Techinque.

→ Steps of Hierachical clustering of (Algomerative)

$\boxed{\text{STep : 1}}$ Make Each data point a single point cluster (# That forms N clusters)

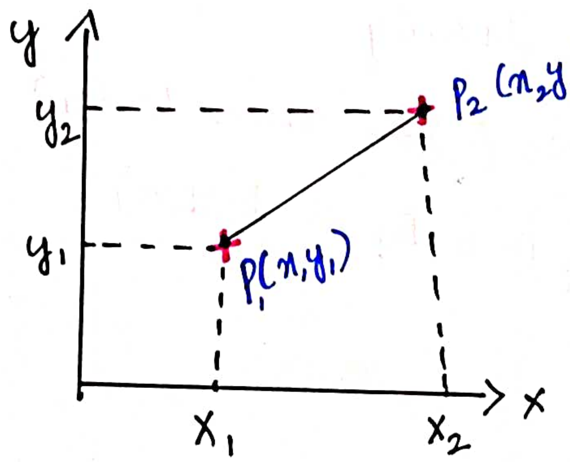
Step:2 Take the two closest data points and Make them One cluster (# that forms N-1 clusters)

Step:3 Take the two closest clusters and make them One cluster (# that forms N-2 clusters)

Step:4 Repeat Step 3, until there is only one cluster

Step 5: Final model.

Euclidean Distance :- it is used to identify distance between two points.

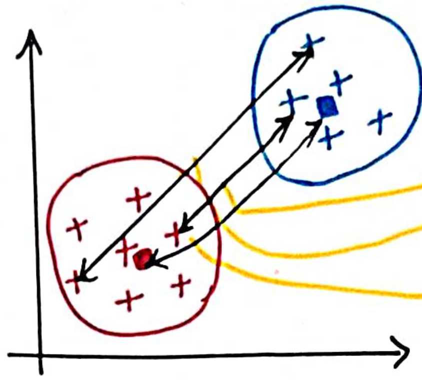


* Euclidean distance between P_1 and P_2

$$= \boxed{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

→ Distance between clusters.

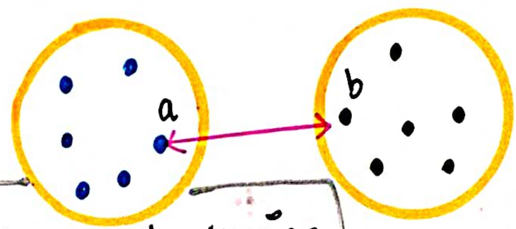
Distance b/w two clusters.



- option 1 : closest Points
- option 2 : furthest Points
- option 3 : Average Distance
- option 4 : Distance between Centroids

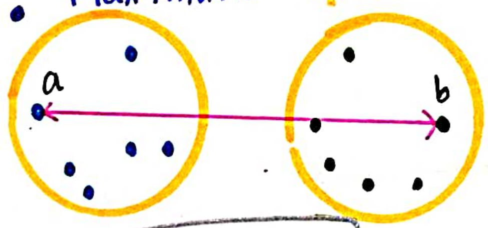
* Single-linkage clustering

- Minimum distance between clusters.



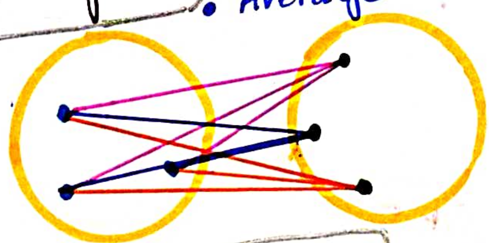
* Complete-linkage clustering

- Maximum distance between clusters



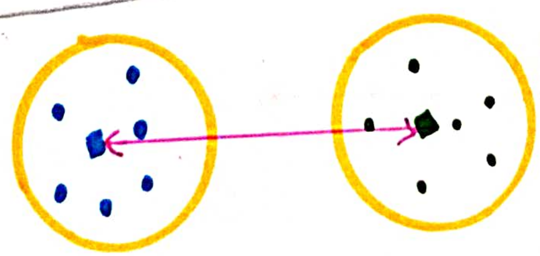
* Average Linkage clustering

- Average distance between clusters



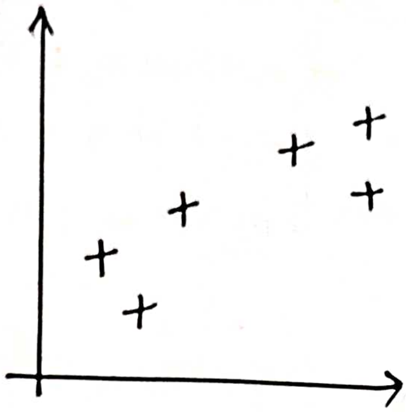
* Centroid Linkage clustering

- distance between cluster Centroids

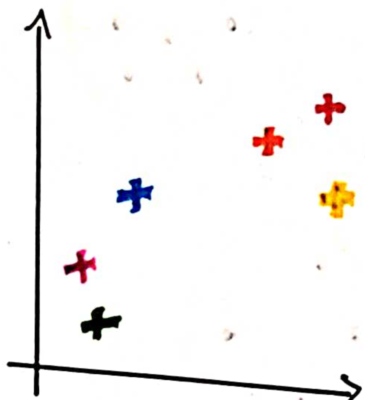


Agglomerative HC

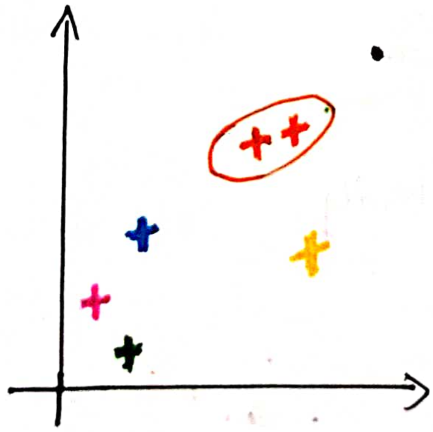
Consider The following dataset of N=6 data points



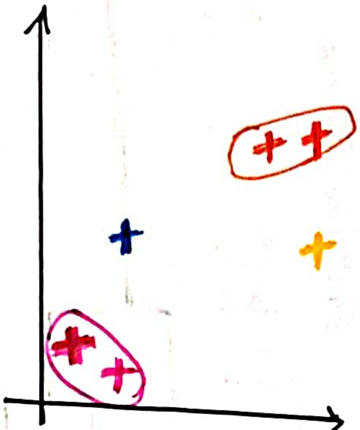
STep : 1 (Make Each data point a single point cluster) that forms 6 clusters)



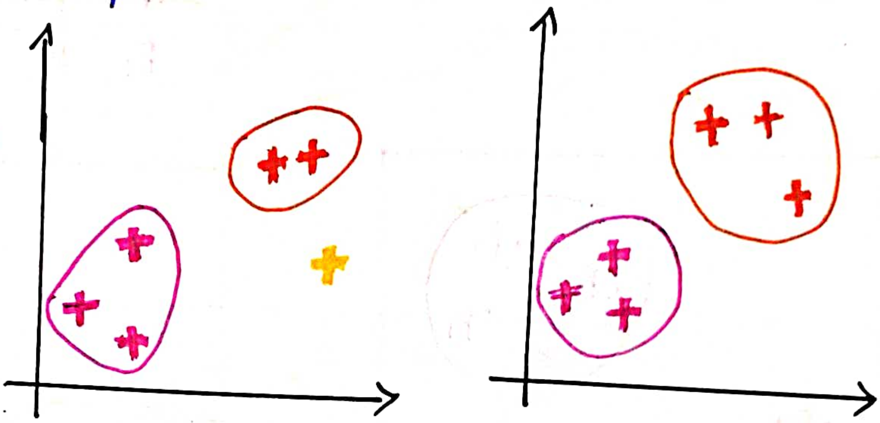
STep : 2 (Take the two closest data points and make them one cluster → that forms 5 clusters)
(n-1)



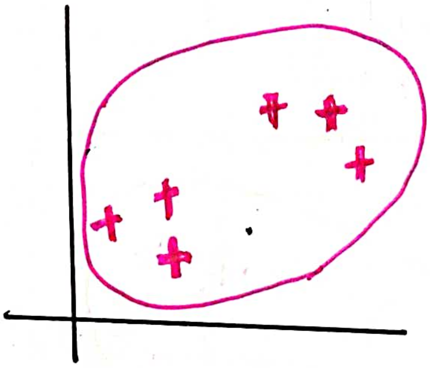
$\boxed{\text{STep 3}}$: Take the two closest clusters and make them one cluster → That forms 4 clusters (n-2)



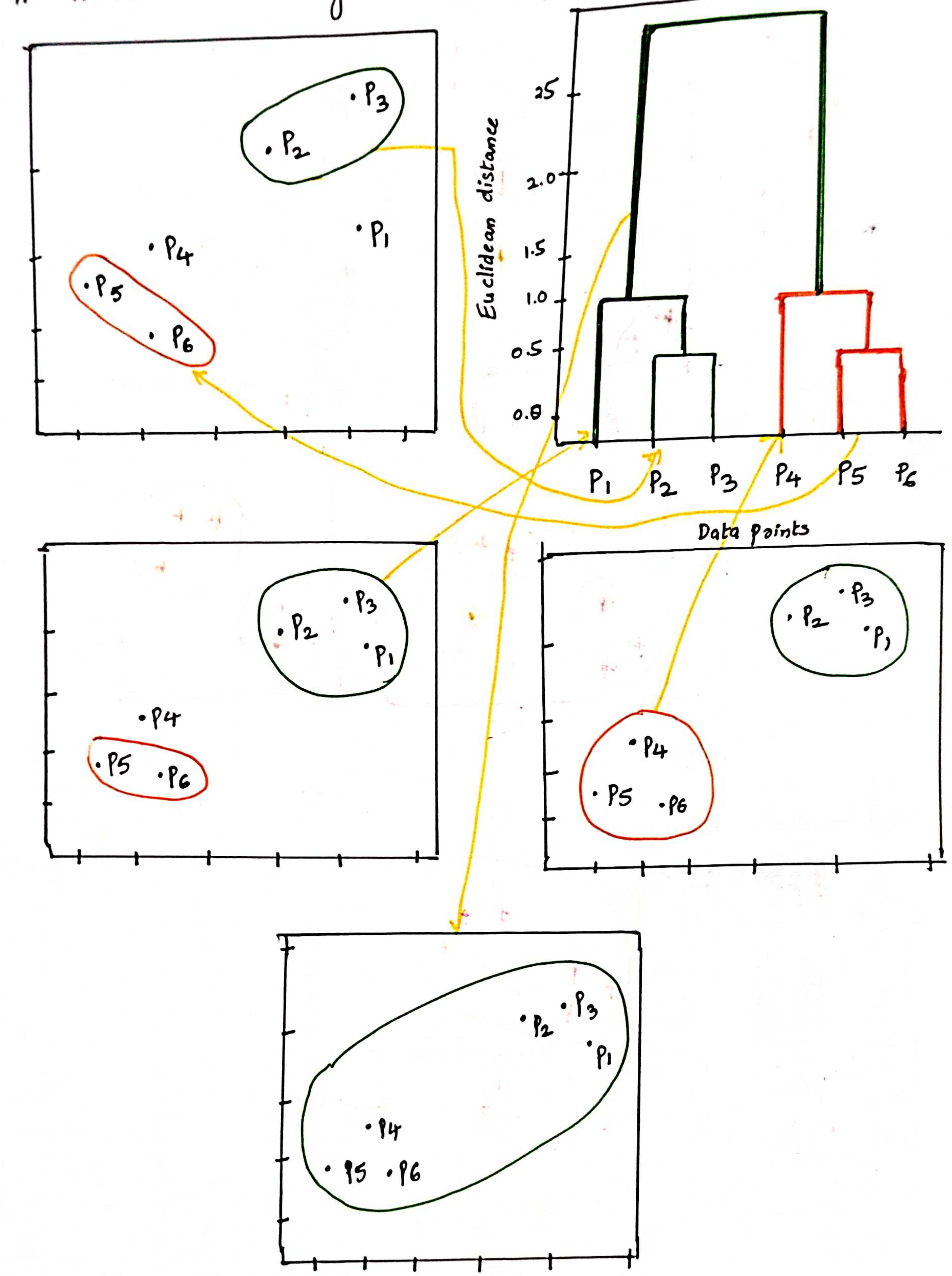
$\boxed{\text{STep 4}}$: Repeat step 3. until There is only one cluster.



Final :

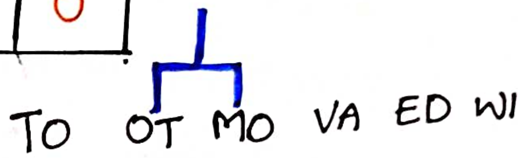


How Do Dendograms Work ?



Example

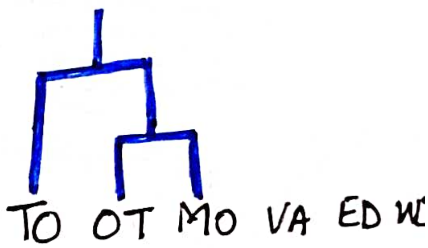
	TO	OT	VA	MO	WI	ED
TO	0	351	3363	505	1510	2699
OT		0	3543	167	1676	2840
VA			0	3690	1867	819
MO				0	1824	2976
WI					0	1195
ED						0



OT and Mo (combined)

Distances are recalculated again.

	TO	OT/MO	VA	WI	ED
TO		351	3363	1510	2699
OT/MO			3543	1676	2840
VA				1867	819
WI					1195
ED					



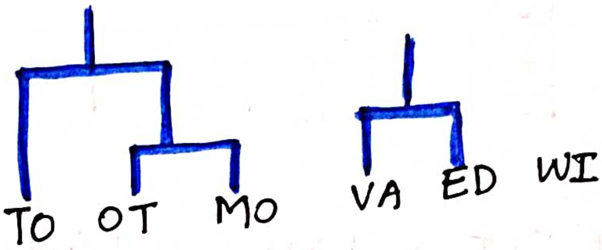
OT/MO and To (combined)

Distances are recalculated again, to make as one cluster.

	To/OT/MO	VA	WI	ED
To/OT/MO		3543	1676	2840
VA			1867	819
WI				1195
ED				

ED and VA (combined)

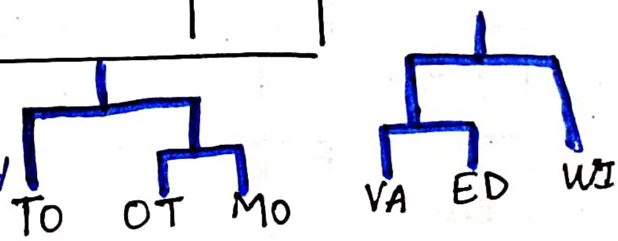
Distances are recalculated again



	To/OT/MO	VA/ED	WI
To/OT/MO		2840	1676
VA/ED			1667 # value wrong.

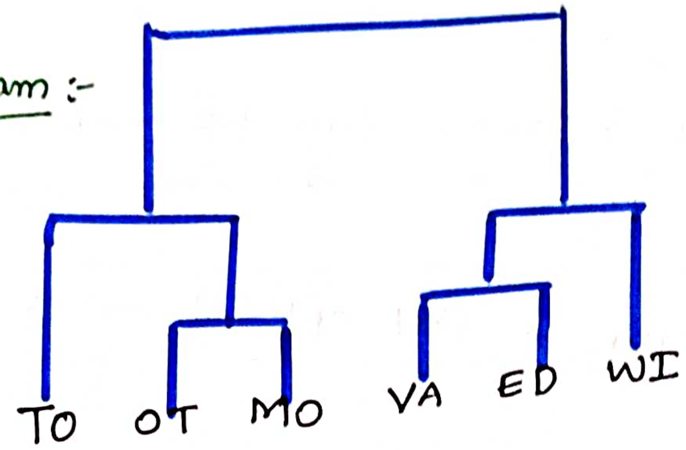
WI and VA/ED (combined)

Distances are recalculated again

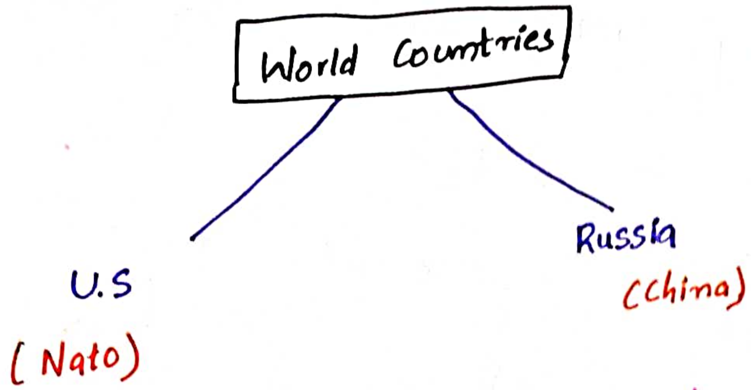


	To/OT/MO	VA/ED/WI
To/OT/MO		1676
VA/ED/WI		

Dendogram :-



Ex:-



if U.S attack russia. it is going to identify which country is going to support russia.

which Country is directly paaticipate in war.

Which are very close to Each other. it is going to identify.

this is Hêerarchical clustering.

Code :- Same data & Same steps till modelling

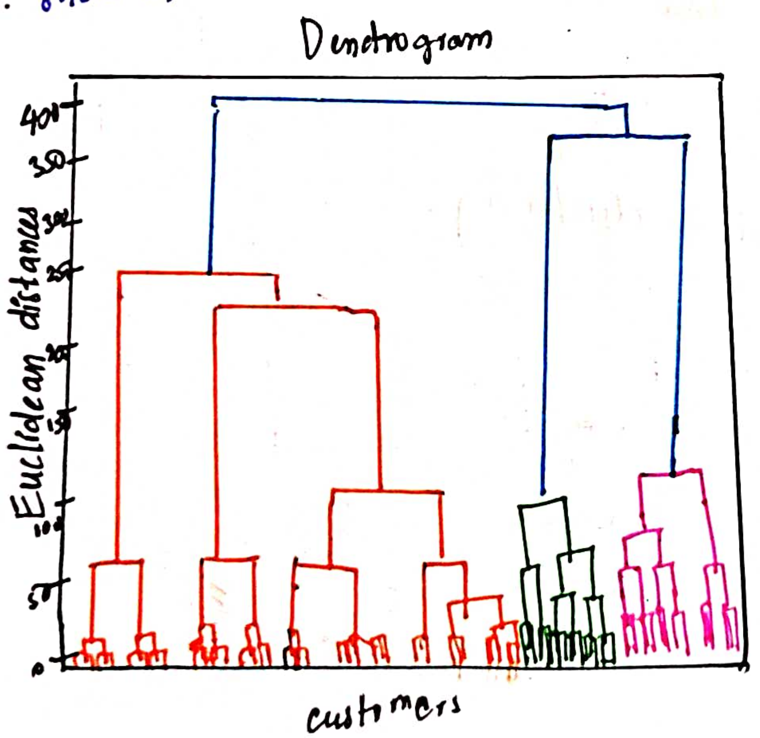
Using the dendrogram to find optimal no. of clusters.

```
import scipy.cluster.hierarchy as sch

# dendrogram = sch.dendrogram (sch.linkage(x,
                                method = "ward"))
```

└→ Based on Centroids

```
# plt.title ("Dendrogram")
# plt.xlabel ( "Customers")
# plt.ylabel ( "Euclidean distances")
# plt.show()
```



Q:- How to identify optimal clusters during dendrogram?

A:- biggest disadvantage with "HC" is we can't identify "optimal no. of clusters". only in K-means we identify.

Hierarchical clustering model :

from sklearn. cluster import AgglomerativeClustering.

small (cluster) · capital (Agglomerative, Clustering) · (no gap)

hc = AgglomerativeClustering (n_clusters = 5, affinity = "Euclidean", linkage = "ward")

Predict

y_hc = hc. fit_predict (x)

y_hc

Out: array ([4, 3, 4, 3, 4, 3, ...
. . . .
. 1,1,1,1, 1, 1 . - -
-- --- - - - - - - -
0, 2]

Visualising clusters

cluster 1

```
# plt.scatter(x[y_hc = =0,0], x[y_hc = =0,1], S=100,
                    C= "red", Label= "cluster 1")
```

cluster 2

```
# plt.scatter(x[y_hc== 1,0], x[y_hc = = 1,1], S=100,
                    C = "blue", label = "cluster 2")
```

cluster 3

```
# plt.scatter(x[y_hc ==2,0], x[y_hc = = 2,1], S=100,
                    C= "green", label = "cluster 3")
```

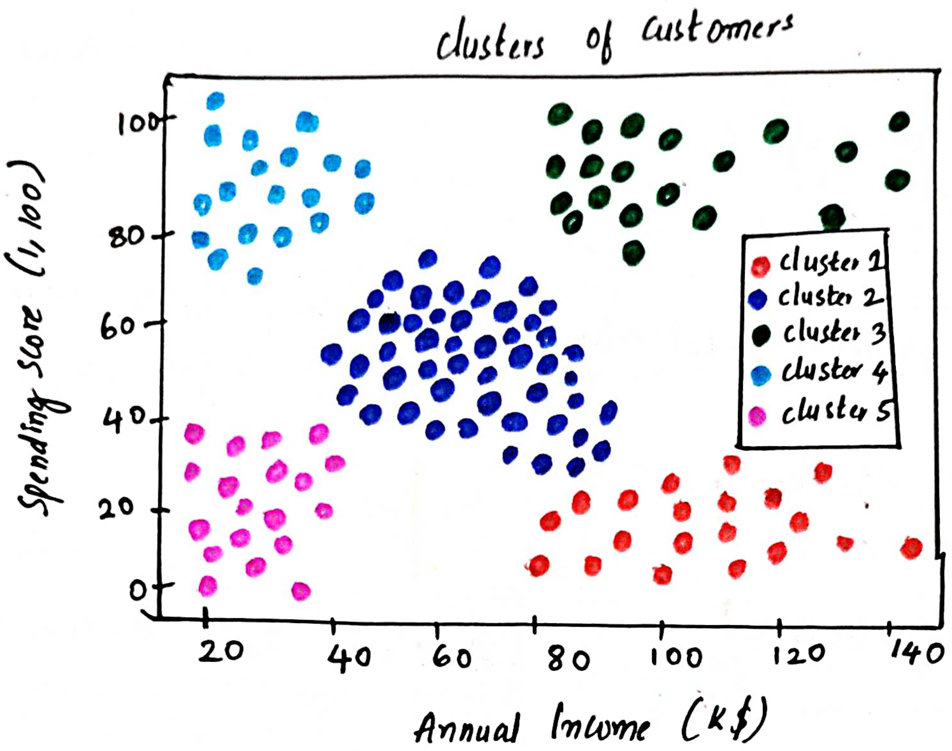
cluster 4

```
# plt.scatter(x[y_hc = = 3,0], x[y_hc == 3,1], S=100,
                    C= "cyan", Label = "cluster 4")
```

cluster 5

```
# plt.scatter(x[y_hc = = 4,0], x[y_hc = = 4,1], S=100,
                    C= "magenta", label= "cluster 5")

# plt.title("clusters of customers")
# plt.xlabel(" Annual Income (K$)")
# plt.ylabel(" spending Score (1-100)")
# plt. show()
# plt. legend()
```



04/05/22
5:00 Am.