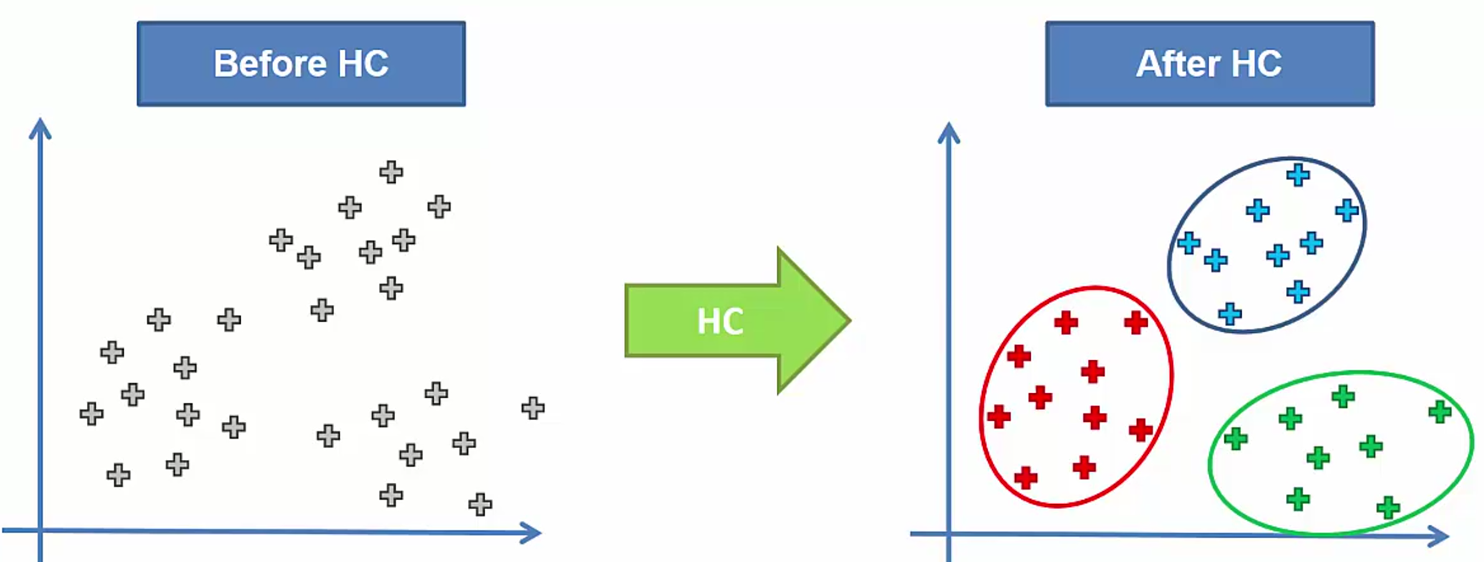
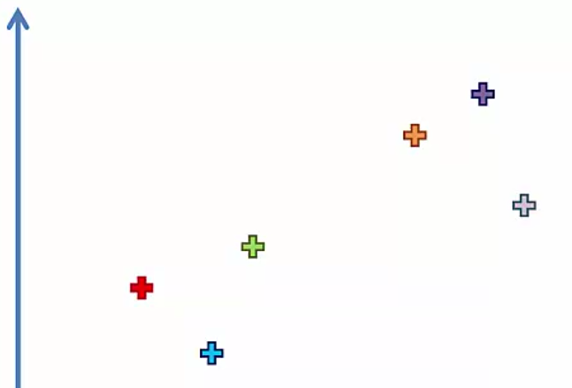
**Hierarchical Clustering**

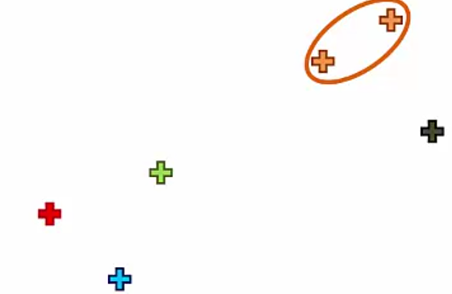
* Finds clusters in DP’s just like K-means, sometimes even the same result, but w/ a different method



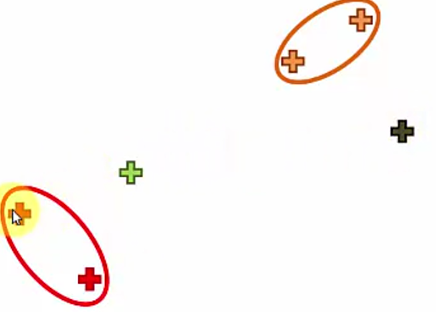
* 2 types:
* **Divisive** (top-down)
* **Agglomerative** (bottom-up)
* 1) Make each DP a cluster (N clusters)

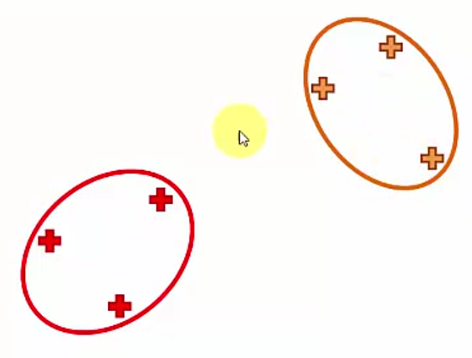
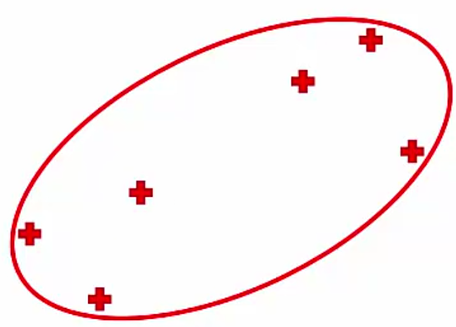


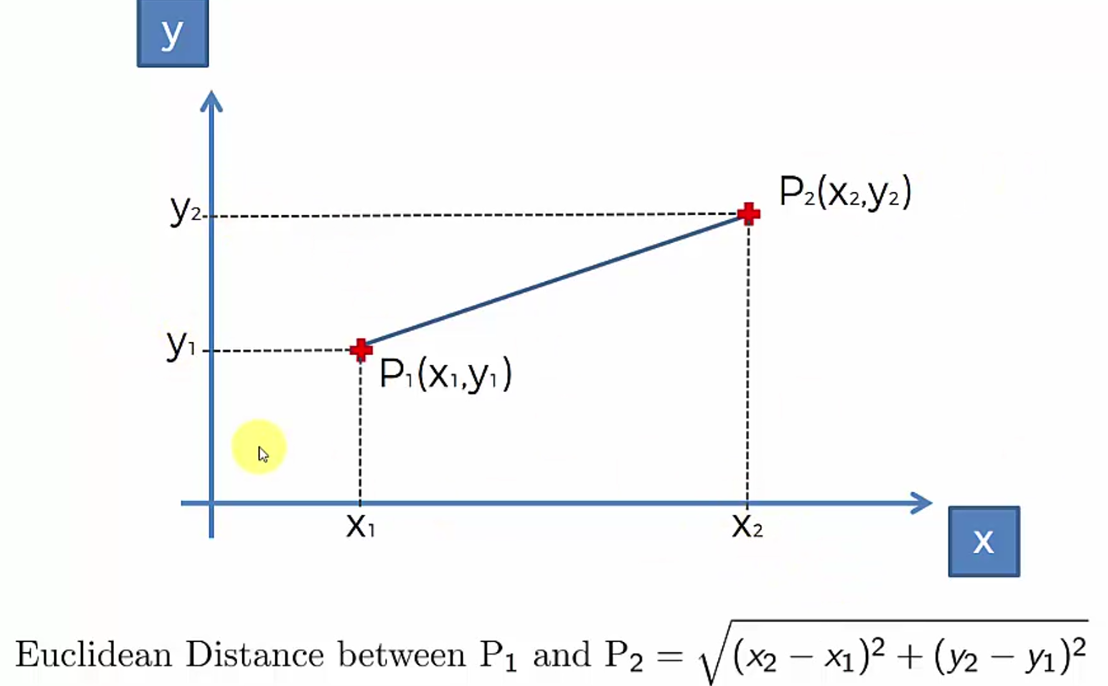
* 2) Take 2 closest DP’s + combine into single cluster (N – 1 clusters)



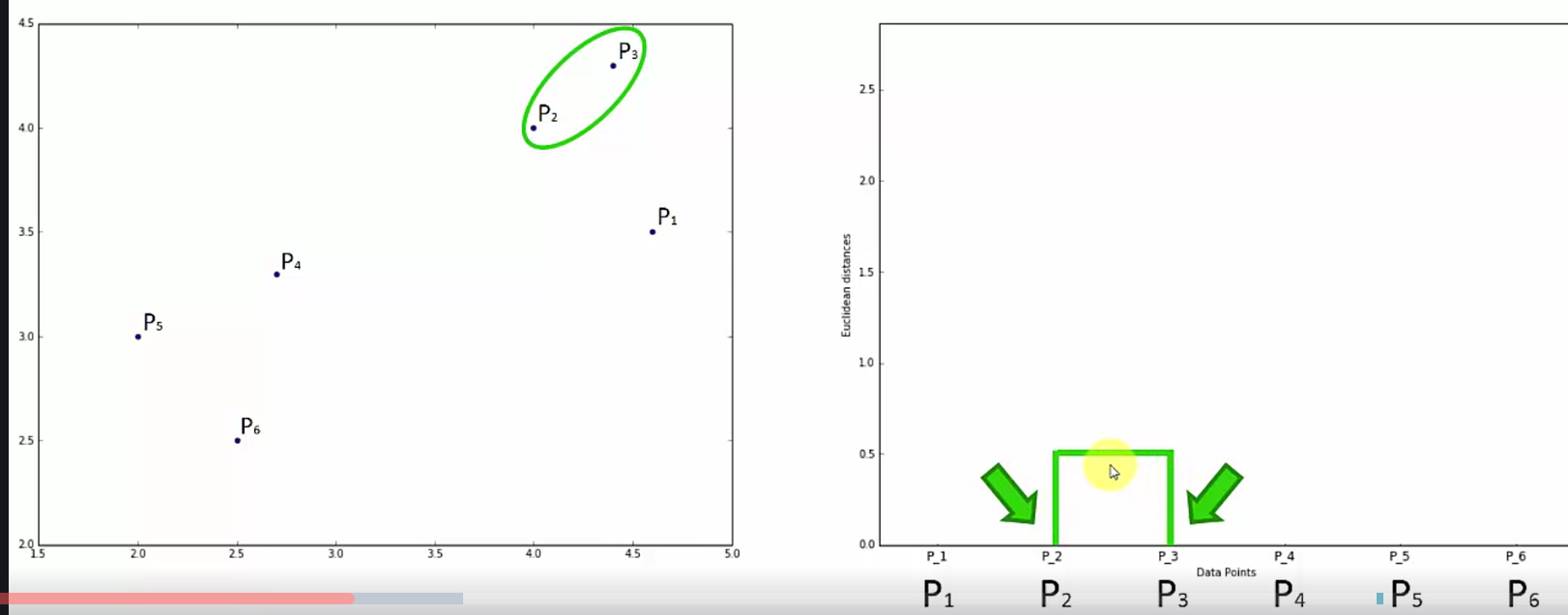
* 3) Take 2 closest *clusters* + combine into single cluster (N – 1 clusters)

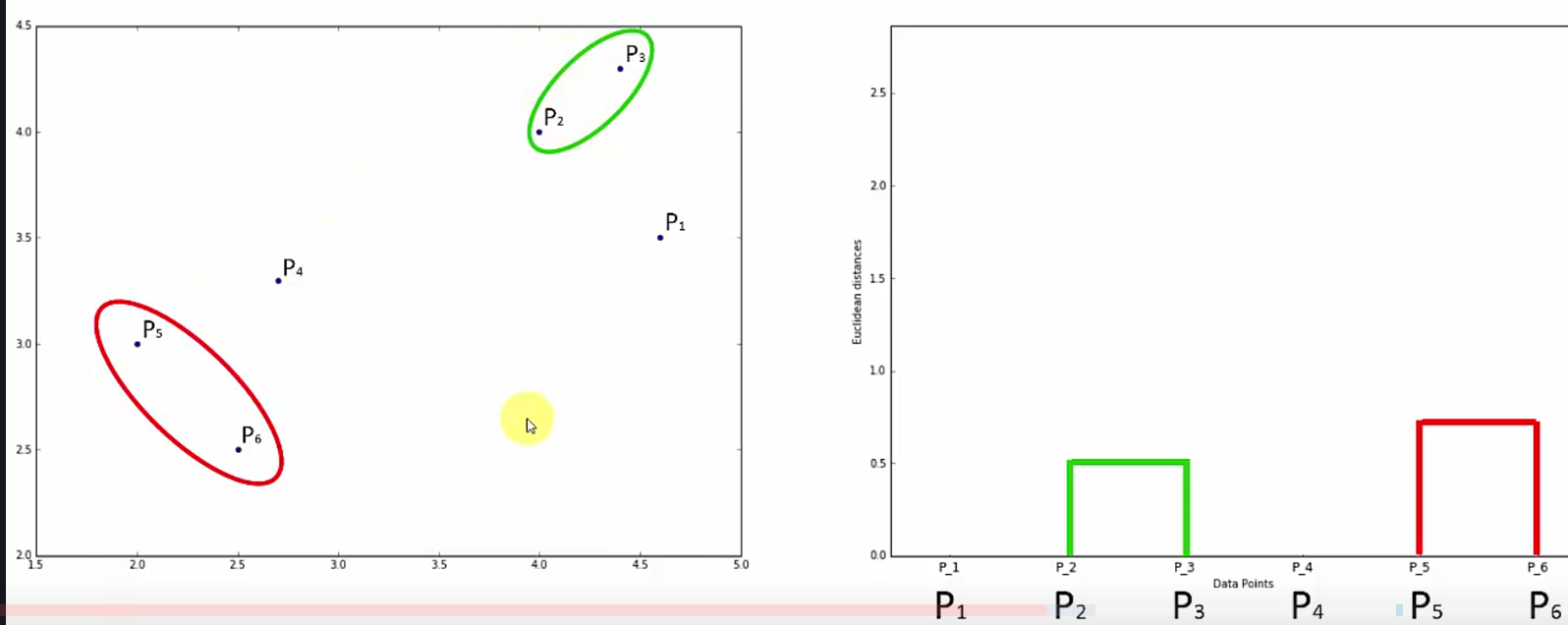


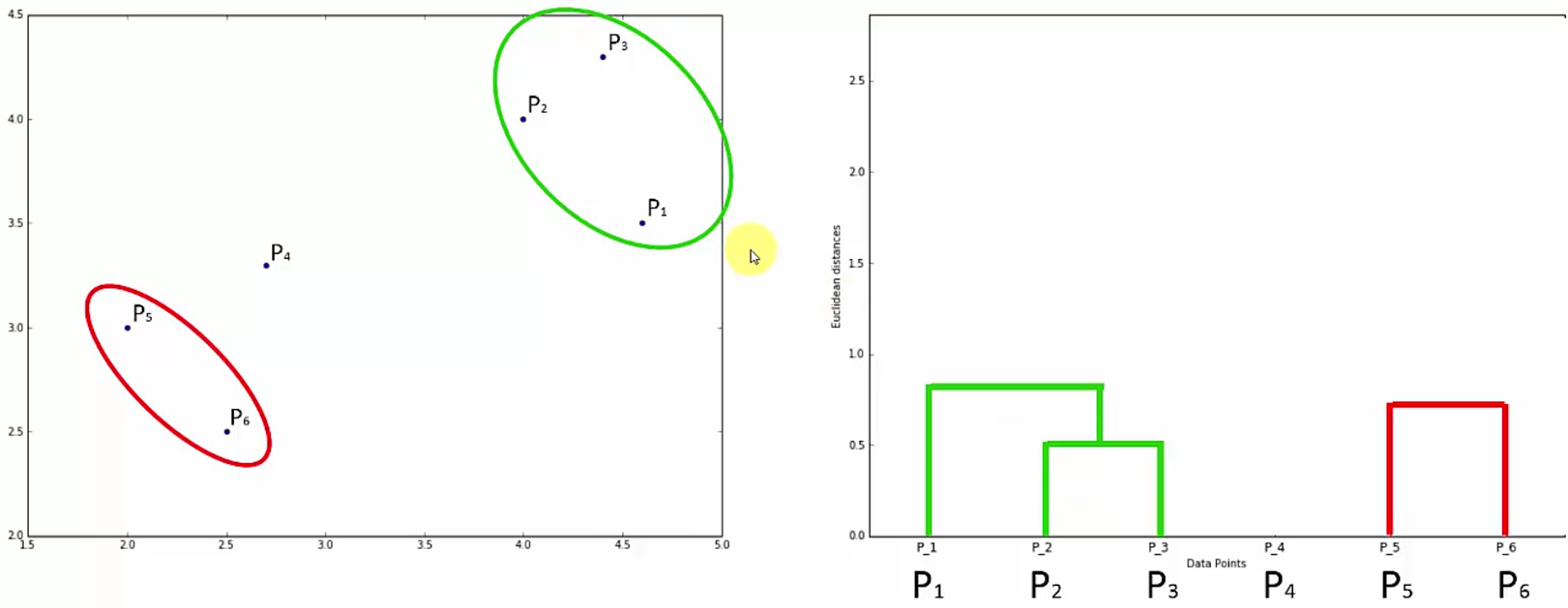
* 4) Repeat until 1 cluster remains (algorithm converges)
*  🡺 
* Remember Euclidean distance

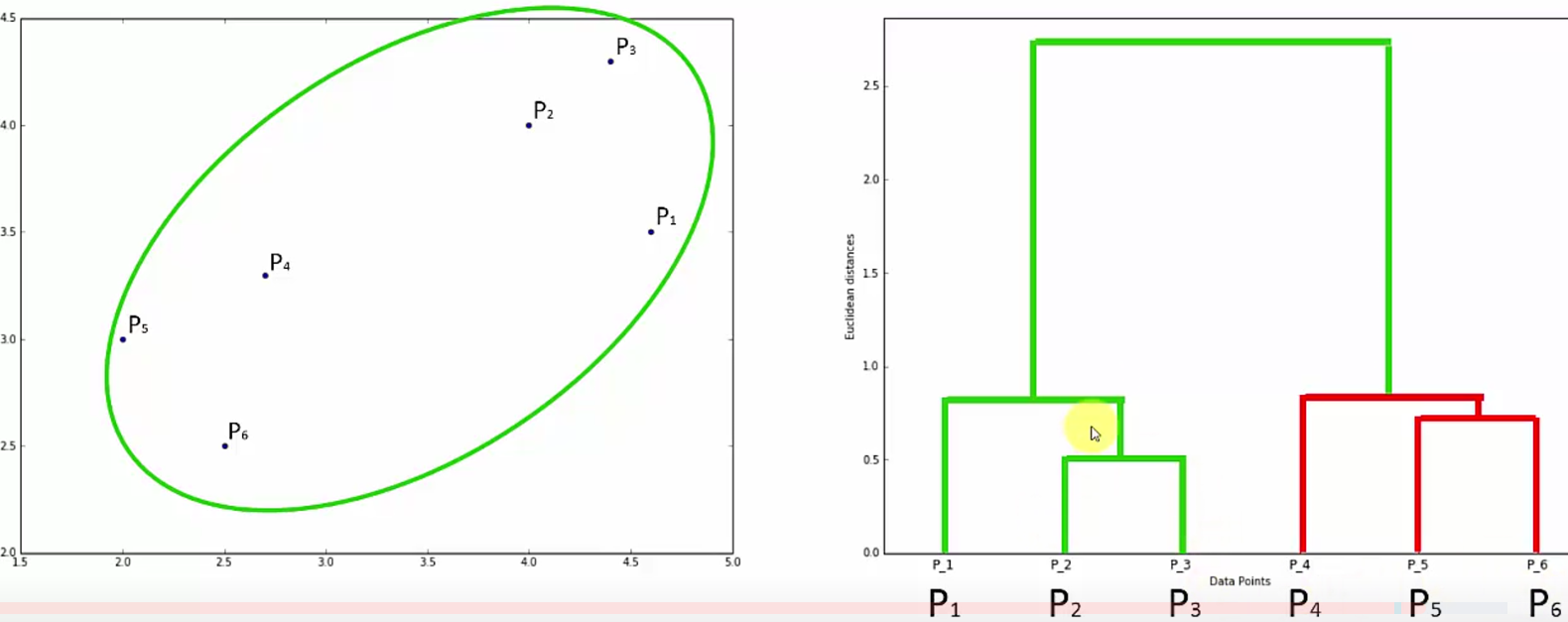


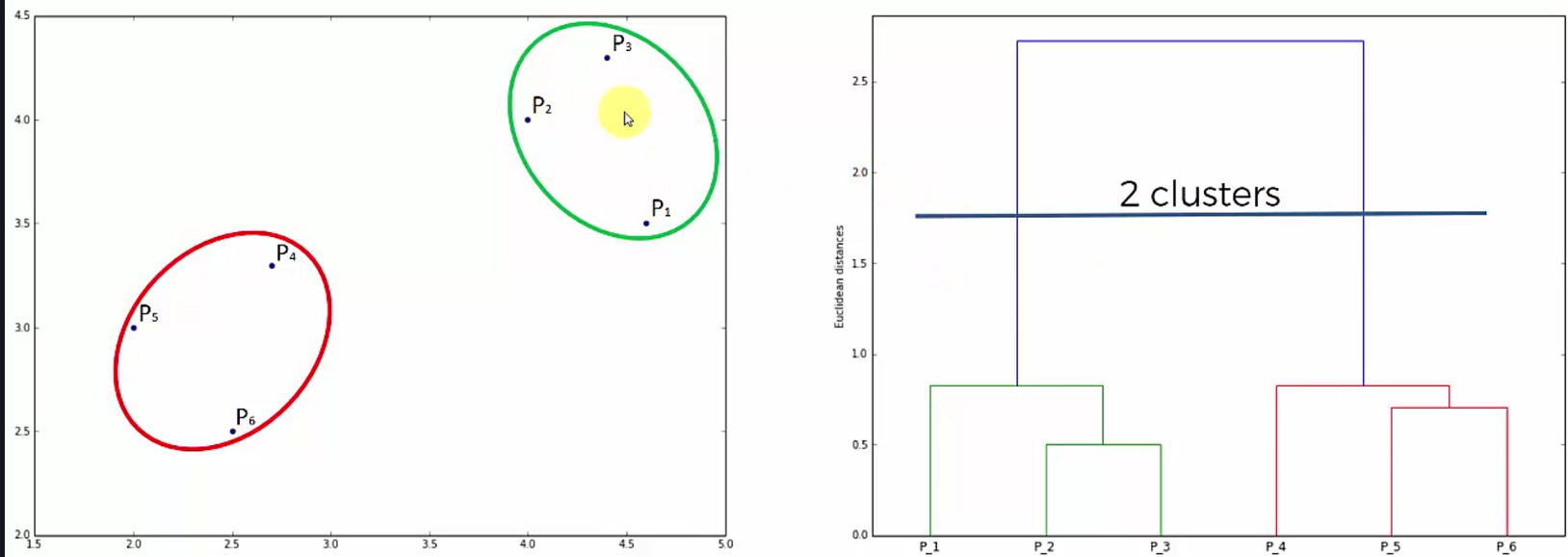
* Multiple ways to measure distance between clusters
* 2 closest points (1 in each cluster)
* 2 furthest points
* Average of all distances between all DPs
* Centroids distance
* **Dendrogram** = kind of like “memory” of HC algorithm



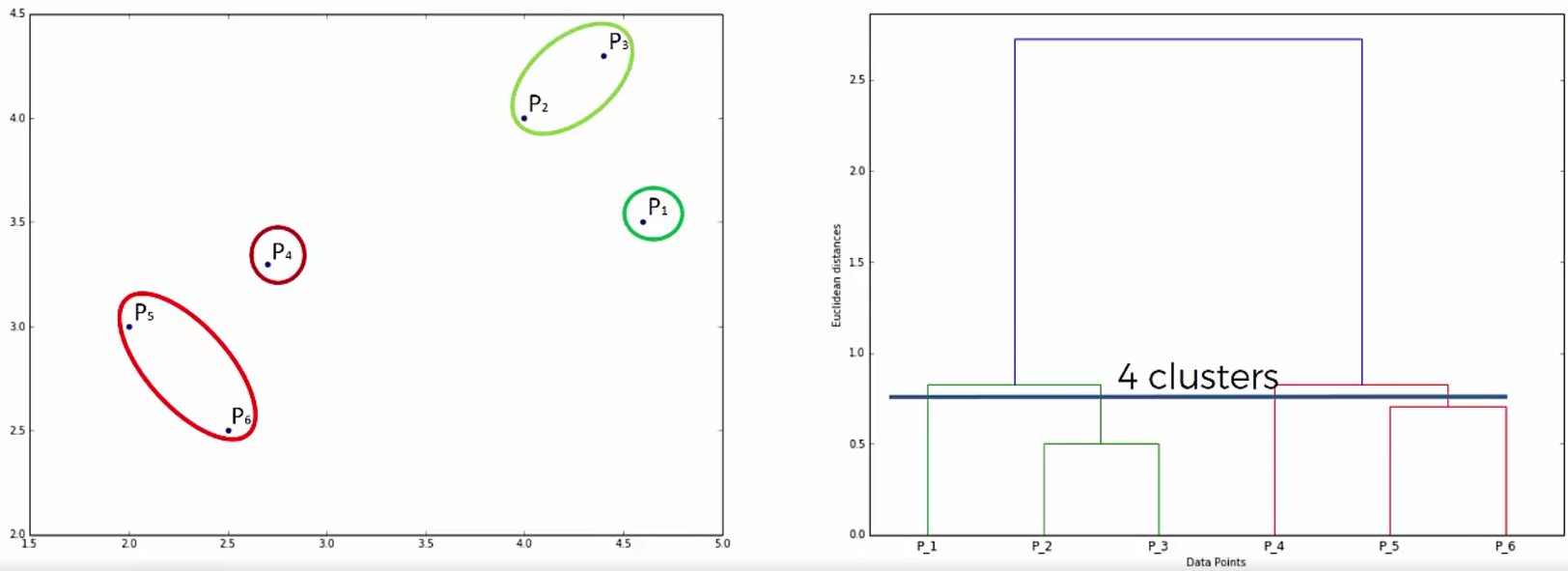
* Height of vertical bars = Euclidean distance = represents the “computed similarity” between the 2 points/clusters
* 



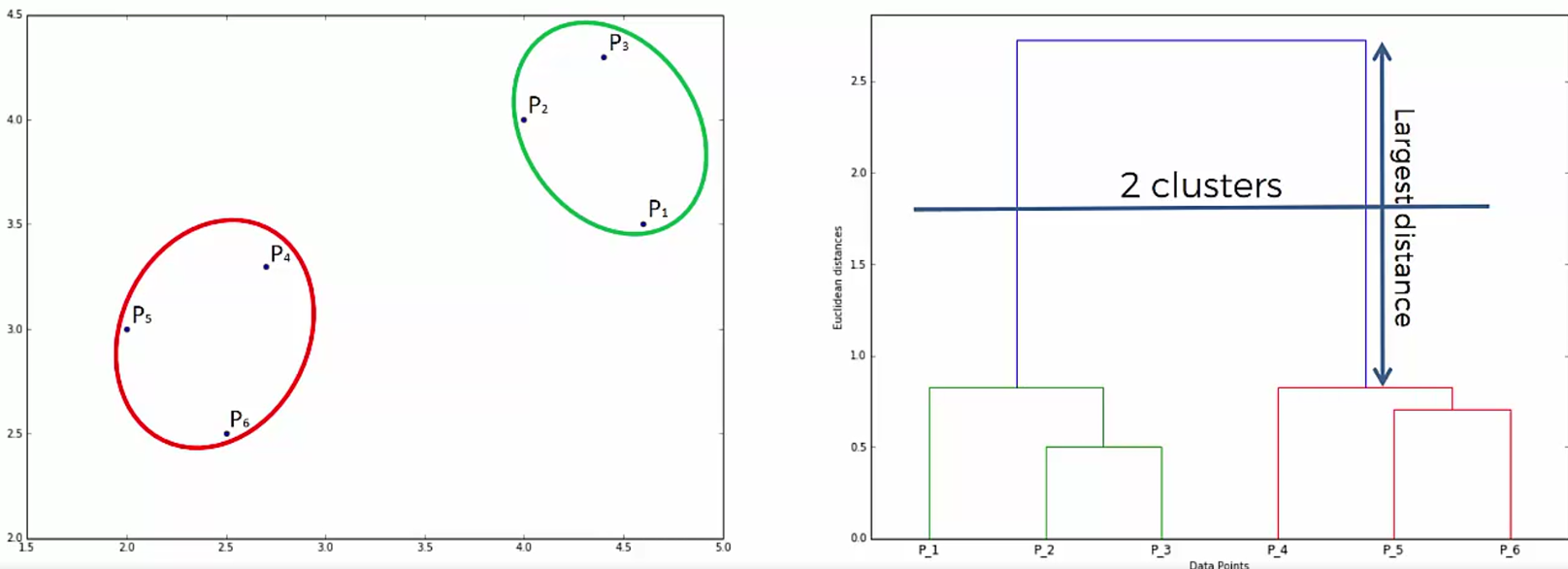
* Distance up from P1 = the “dissimilarity” between P2 + P3 cluster and P1
* 
* Then, look at the horizontal levels + set a distance/dissimilarity threshold (i.e. don’t want it to be > 1.75)



* This eliminates any clusters w/ dissimilarity > 1.75
* # of vertical lines the dendrogram crosses = final # of clusters



* How can we figure out an optimal # of clusters?
* One way = look for highest vertical distance on dendrogram that does not cross a hypothetical horizontal line (can’t use P1 or P4 b/c the cross the tops of P2 + P3 and P5 + P6)



* The guideline here is to take a threshold that will cross this largest distance + that gives us an “optimal” # of clusters

