# Hierarchical Clustering

Name – SNEHA M - 21060641049
JOY ANGELIN - 21060641022

## SUPERVISED LEARNING

Deals with the “**labelled**” data

Training a machine using information that is neither classified nor labeled and allows the machine to act on that information without guidance.

## UNSUPERVISED LEARNING

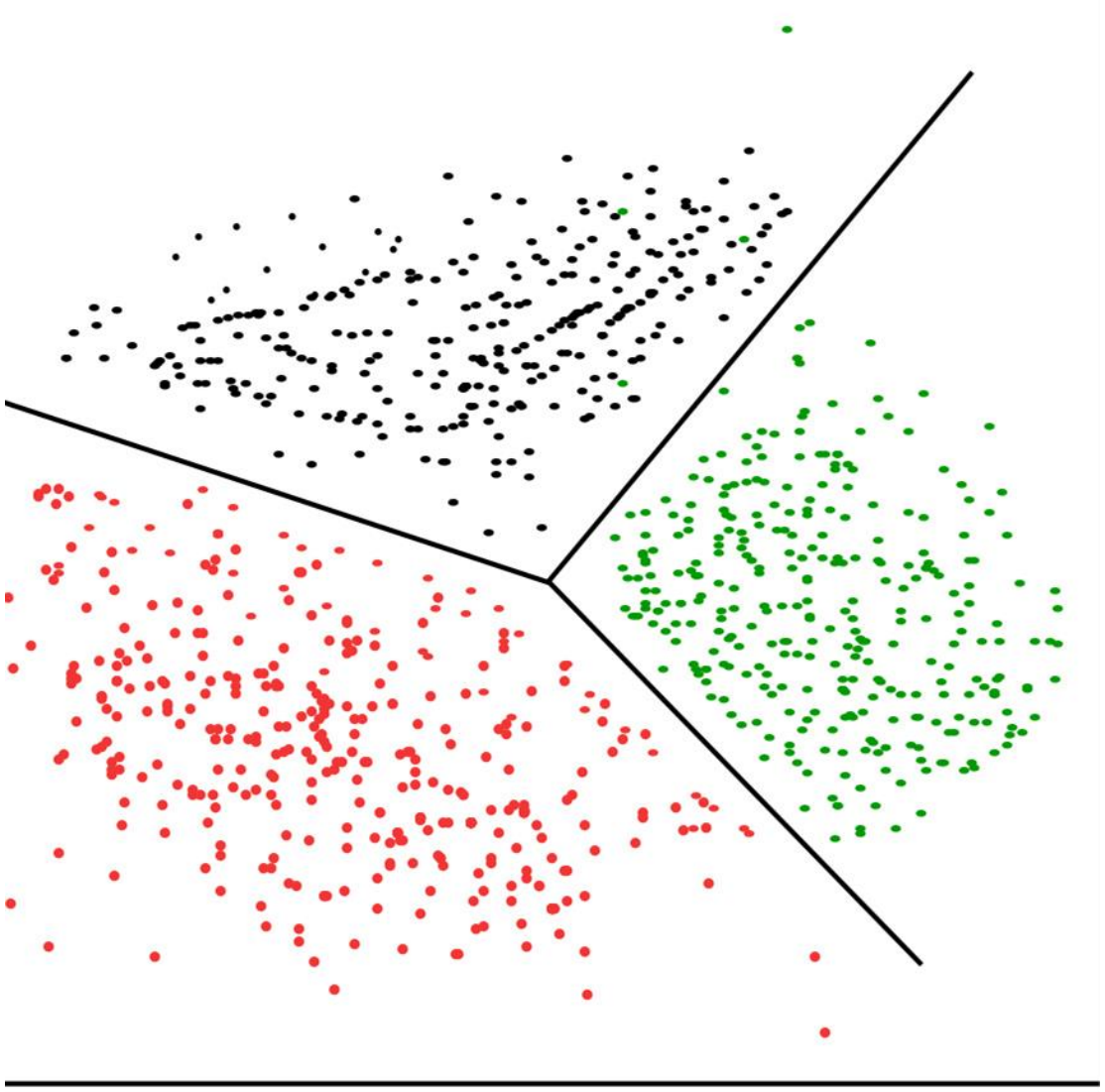
Deals with the “**unlabeled**” data.

Where you don’t have to supervise the model. Rather, you need to allow the model to work on its own to discover information

This is unsupervised learning, where you are not taught but you learn from the data

(in this case data about a dog.)

Had this been supervised learning, the family friend would have told the baby that it's a dog.



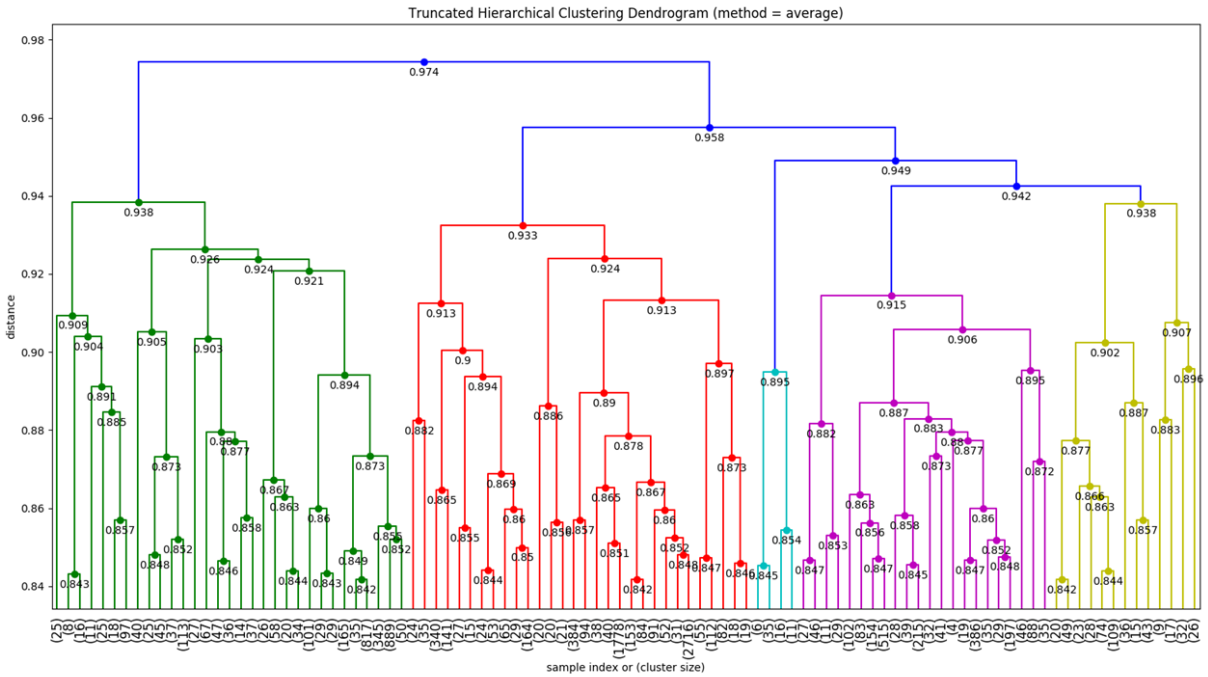
## WHAT IS CLUSTERING?

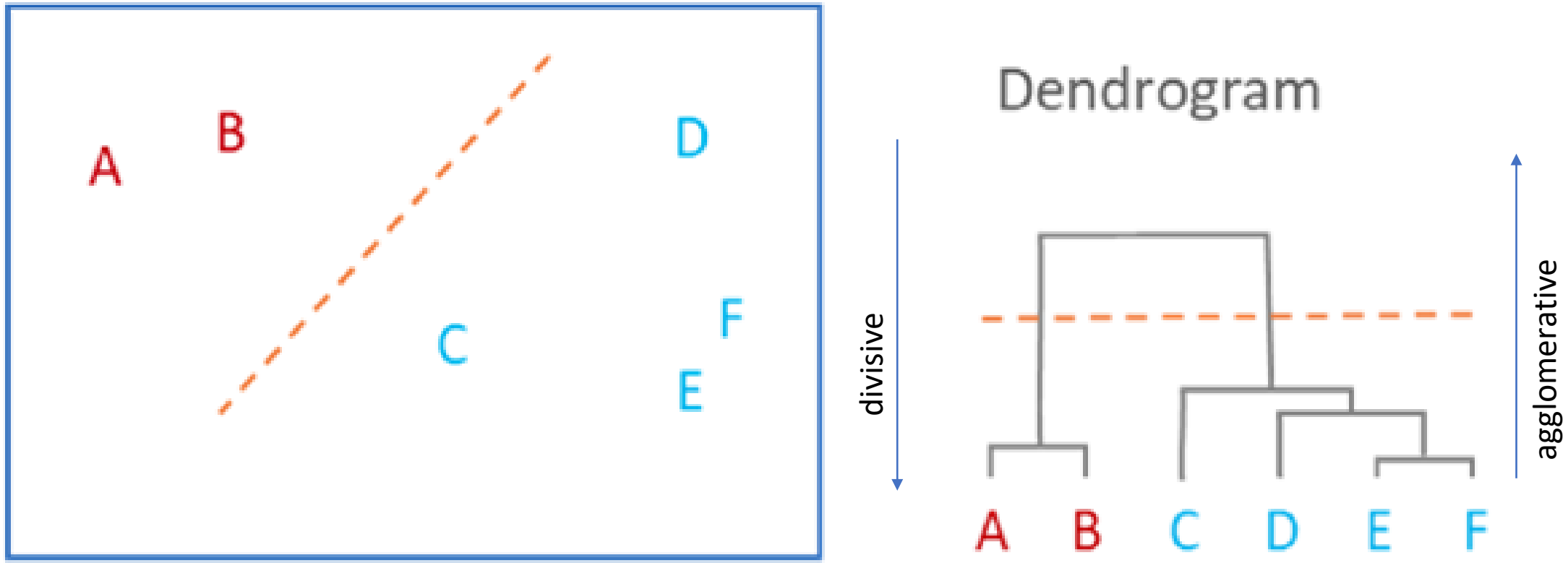
Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups.

# HIERARCHICAL CLUSTERING

It is the grouping of observations to form a **hierarchy of clusters**, where this hierarchy resembles a tree structure, called a dendrogram. Traditional hierarchical algorithms use a similarity or **distance matrix**.

The endpoint is a **set of clusters**, where each cluster is distinct from each other cluster, and the objects within each cluster are **broadly similar** to each other.

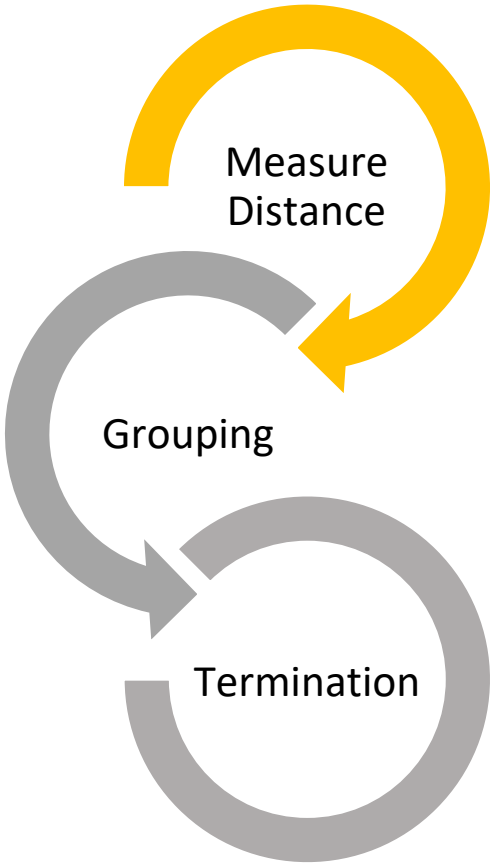




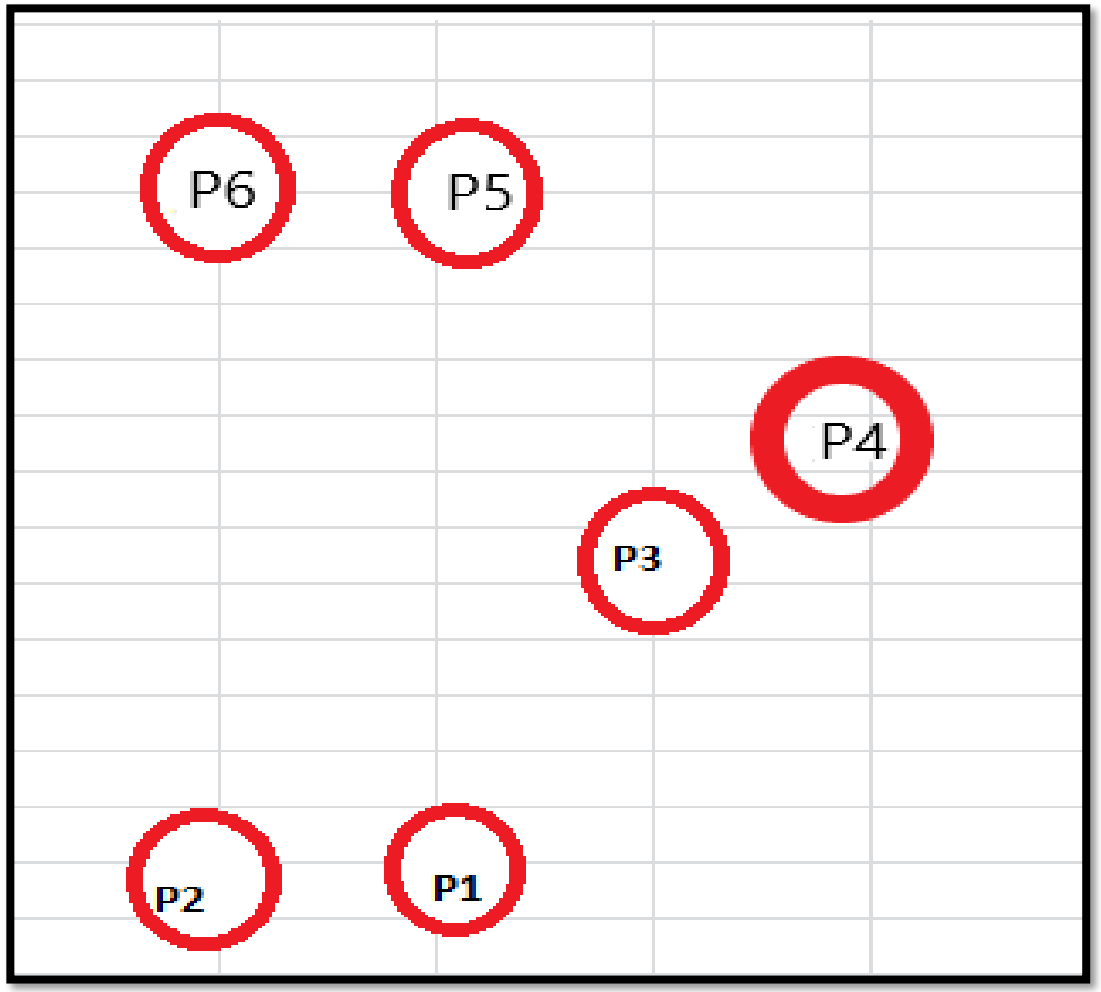

# What is a Dendrogram?

- A diagram that shows the hierarchical relationship between objects.
- Dendrogram is a summary of the distance matrix
- Size of the dendrogram can be manually declared

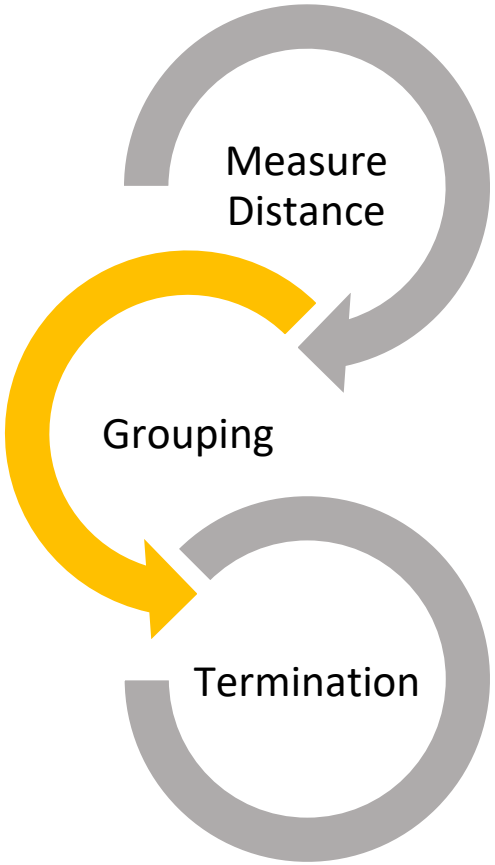
# HOW TO MAKE A DENDROGRAM?

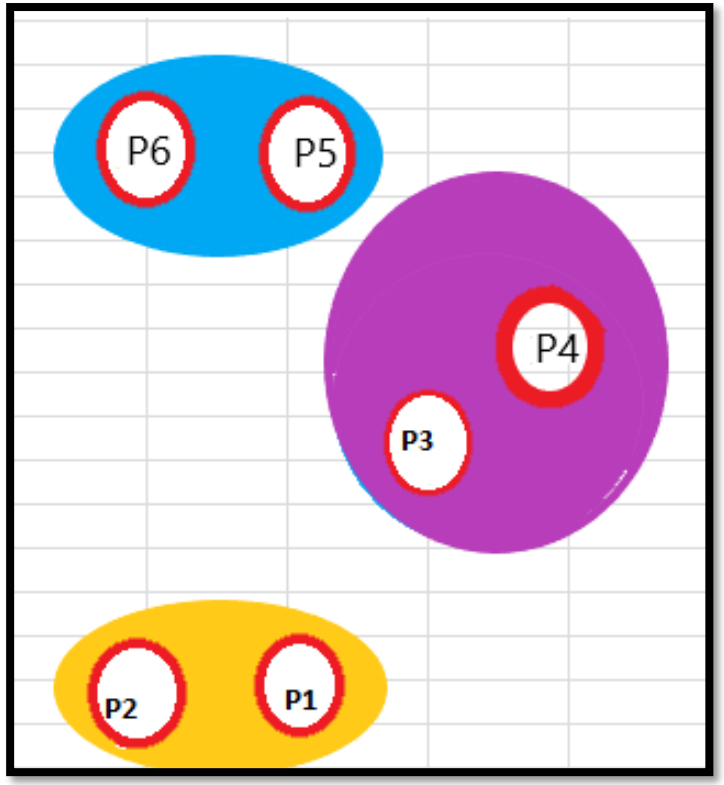


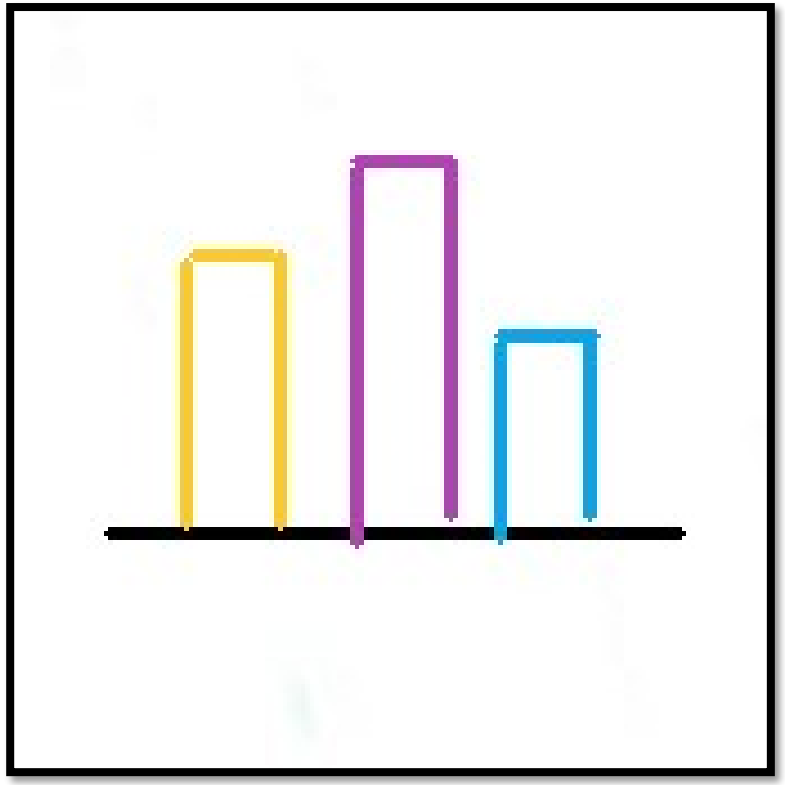

- Each Data Point is a cluster of its own
- We try to find the least distance between two clusters.



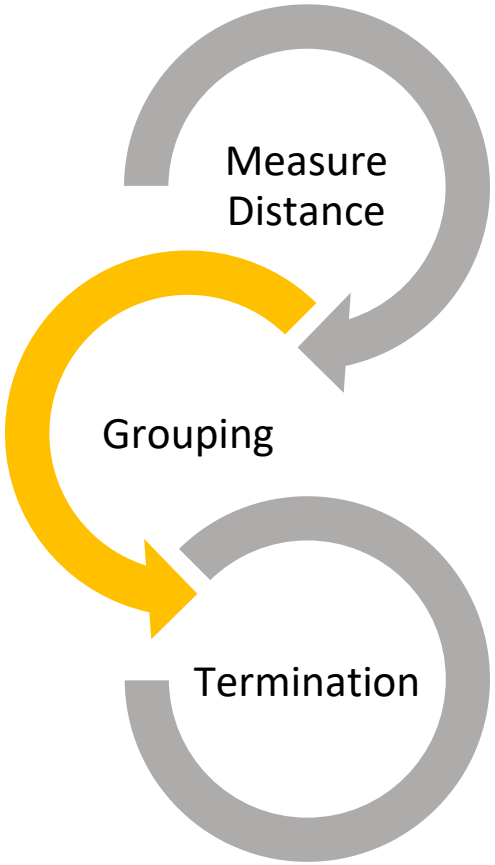
The two nearest clusters are merged together.

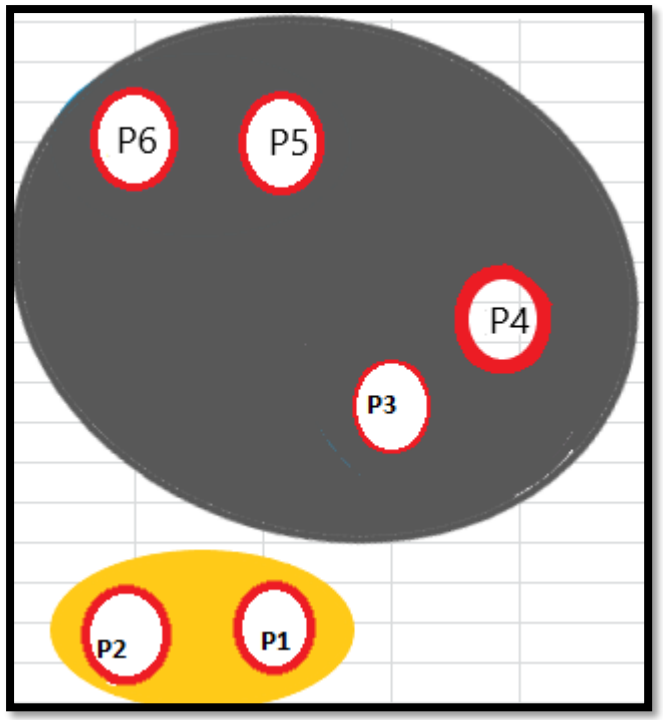


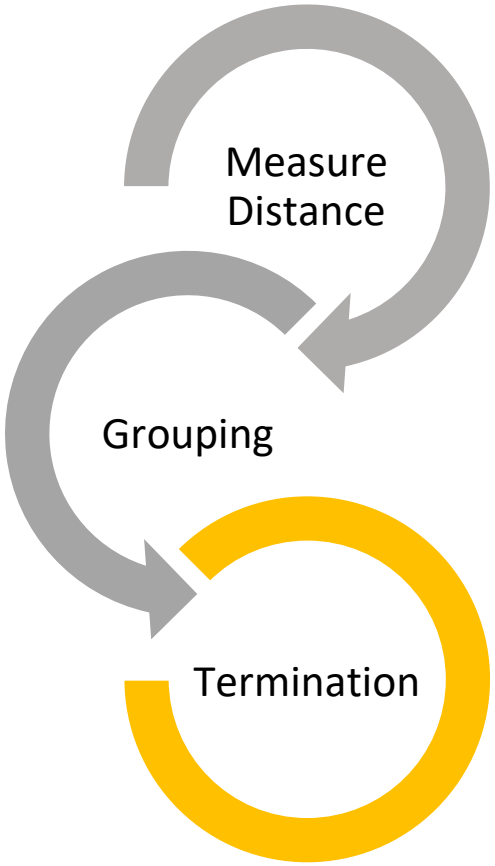



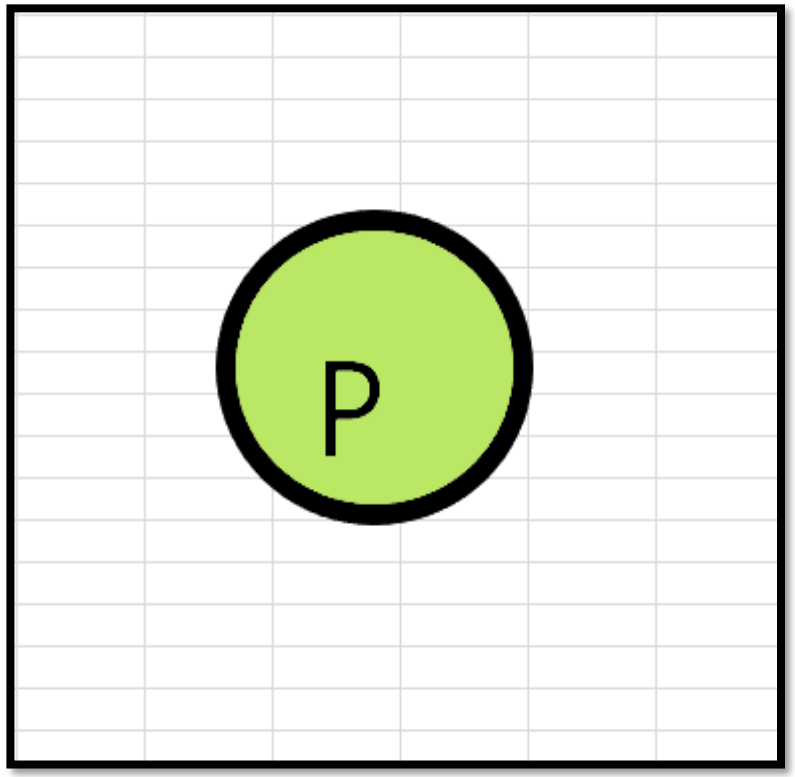
The two nearest clusters are merged together.

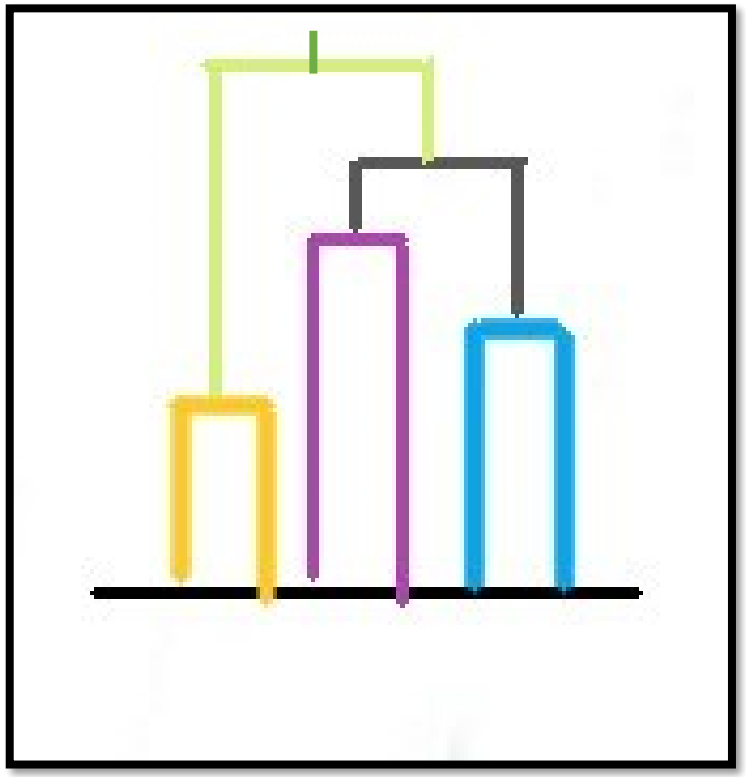



We terminate to when we are left with only one cluster.



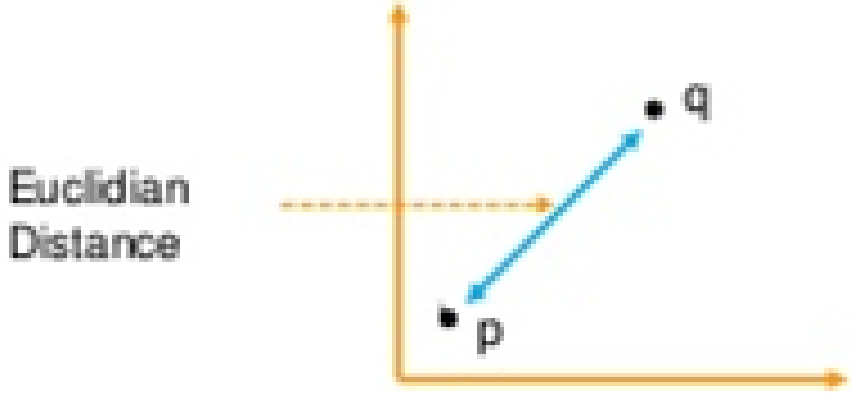



# DISTANCE MEASURE

- **Euclidean Distance Measure**
- **Squared Euclidean Distance Measure**
- **Manhattan Distance Measure**
- **Cosine Distance Measure**

The Euclidean distance  is the ordinary straight line
It is the distance between two points in Euclidean space

$$d=\sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}$$



**Euclidean Distance Measure**

**Squared Euclidean Distance Measure**

**Manhattan Distance Measure**

**Cosine Distance Measure**

The Euclidean squared distance metric uses the same equation as the Euclidean distance metric but does not take square root.

$$d = \sum_{i=1}^{n} (q_i - p_i)^2$$

**Euclidean Distance Measure**

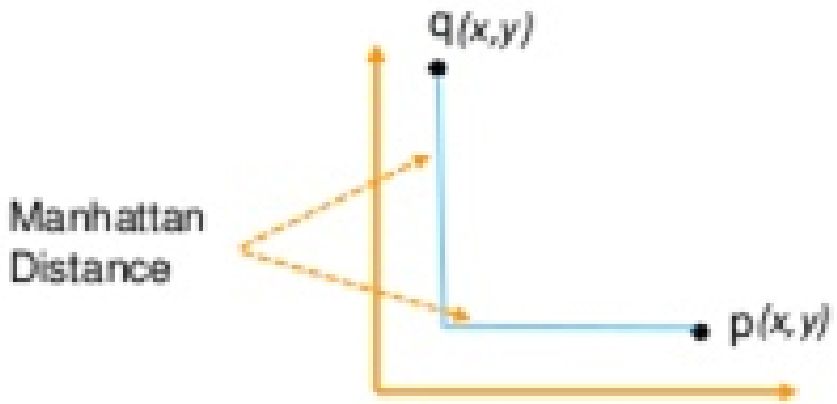
**Squared Euclidean Distance Measure**

**Manhattan Distance Measure**

**Cosine Distance Measure**

The Manhattan distance is a simple **sum of the horizontal and vertical components** or the distance between two points measured along axis at right angles

$$d = \sum_{i=1}^{n} |q_x - p_x| + |q_y - p_y|$$



**Euclidean Distance Measure**

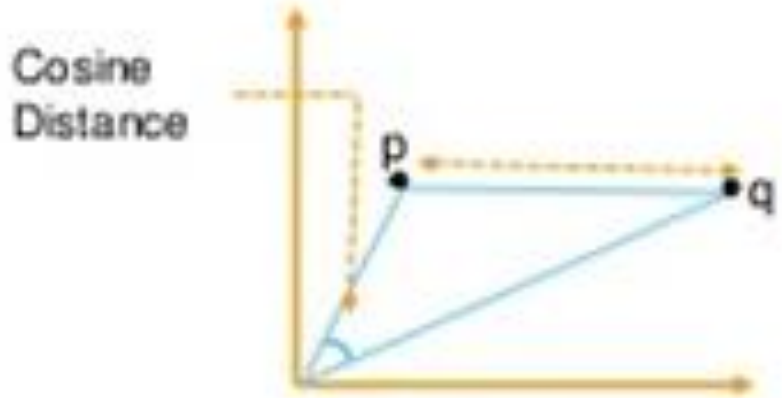
**Squared Euclidean Distance Measure**

**Manhattan Distance Measure**

**Cosine Distance Measure**

The cosine distance similarity measures the angle between two vectors

$$d = \frac{\sum_{i=0}^{n-1} q_i - p_x}{\sum_{i=0}^{n-1} (q_i)^2 \times \sum_{i=0}^{n-1} (p_i)^2}$$

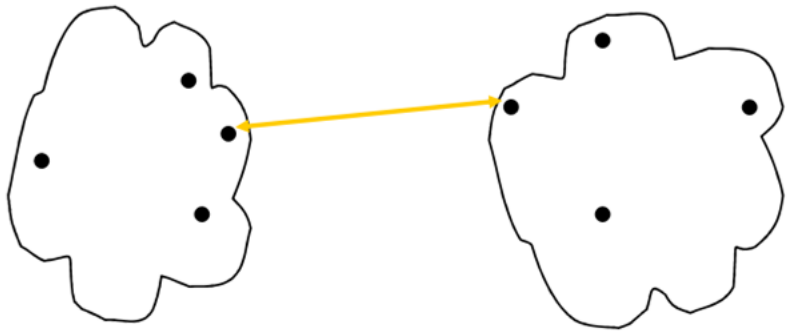


# MEASURE FOR THE DISTANCE BETWEEN TWO CLUSTERS

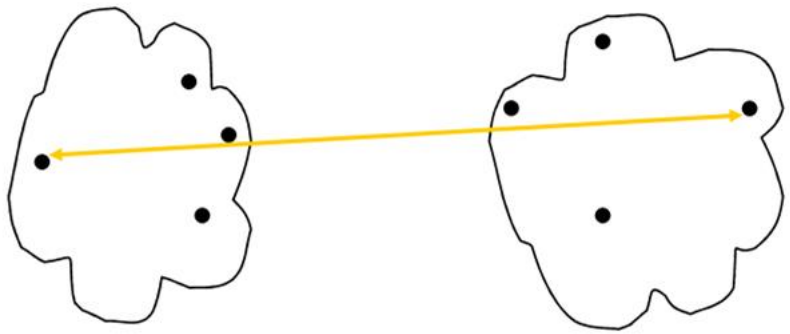
## SINGLE LINKAGE

Can handle non elliptical shapes

Sensitive to noise and outliers



## COMPLETE LINKAGE



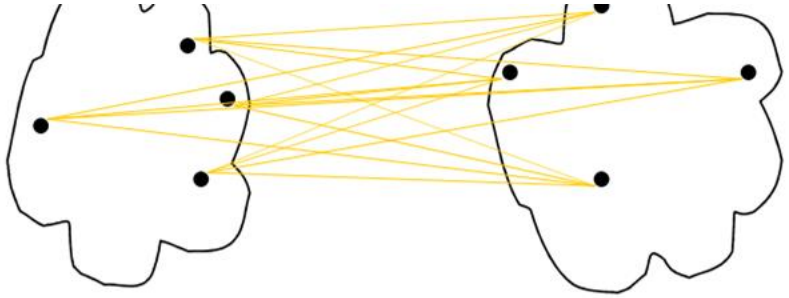
Less susceptible to noise and outliers

Tends to break large clusters and biased towards globular clusters

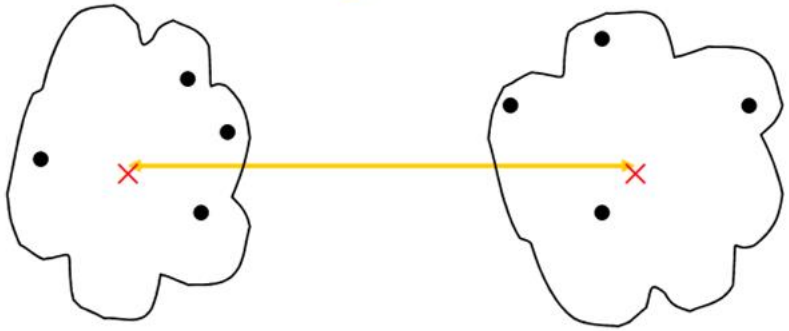
## AVERAGE LINKAGE

Less susceptible to noise and outliers

Biased toward globular clusters

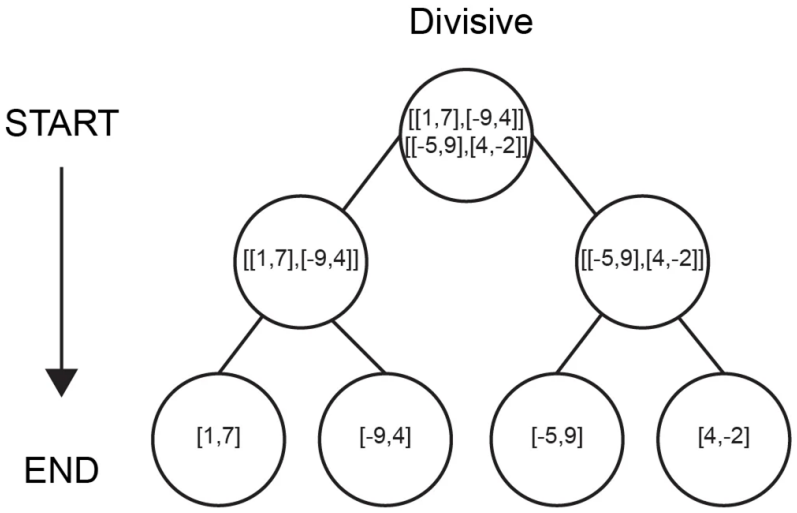


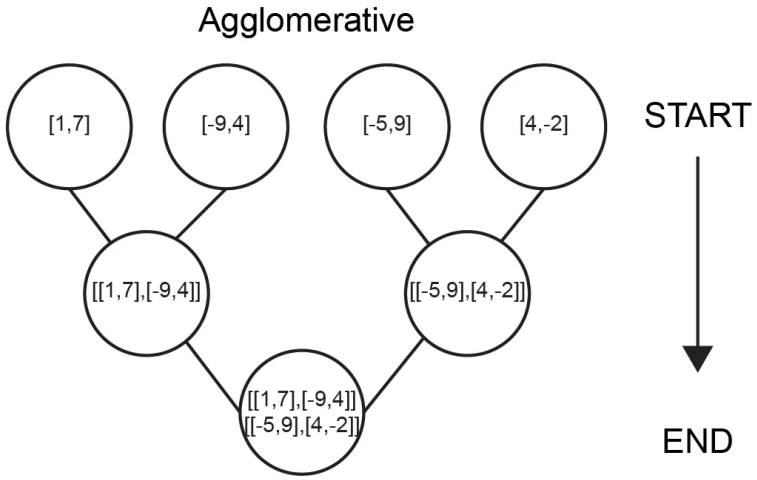
## CENTROID LINKAGE



Does well in separating clusters if there is any noise between the clusters.

Biased towards globular clusters




# Hierarchical clustering

## DIVISIVE CLUSTERING

- This is a "top-down" approach: all observations start in one cluster.
- **Splits** are performed recursively as one moves down the hierarchy.

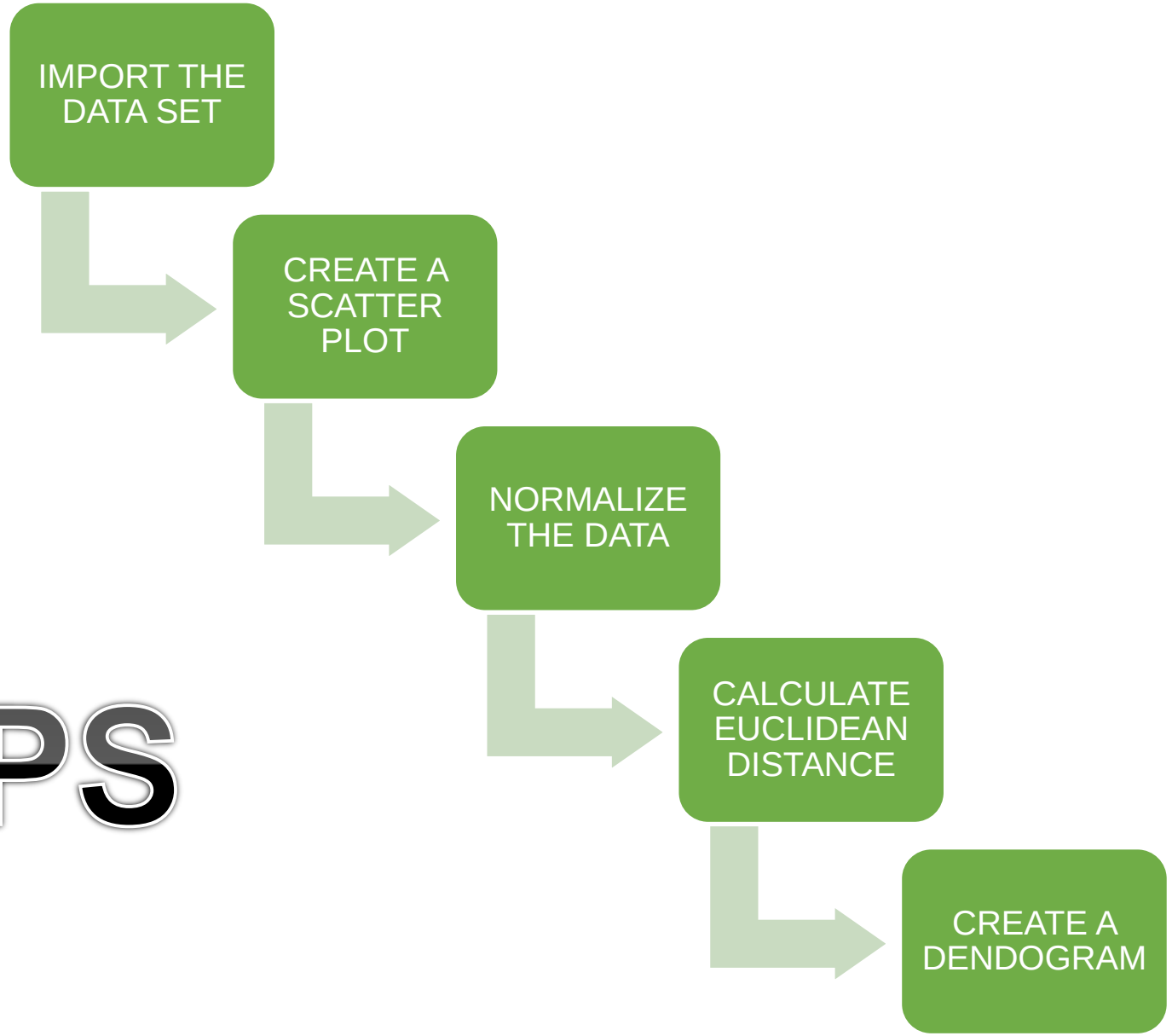
## AGGLOMERATIVE CLUSTERING

- This is a "bottom-up" approach: each observation starts in its own cluster.
- **Pairs** of clusters are merged as one moves up the hierarchy.

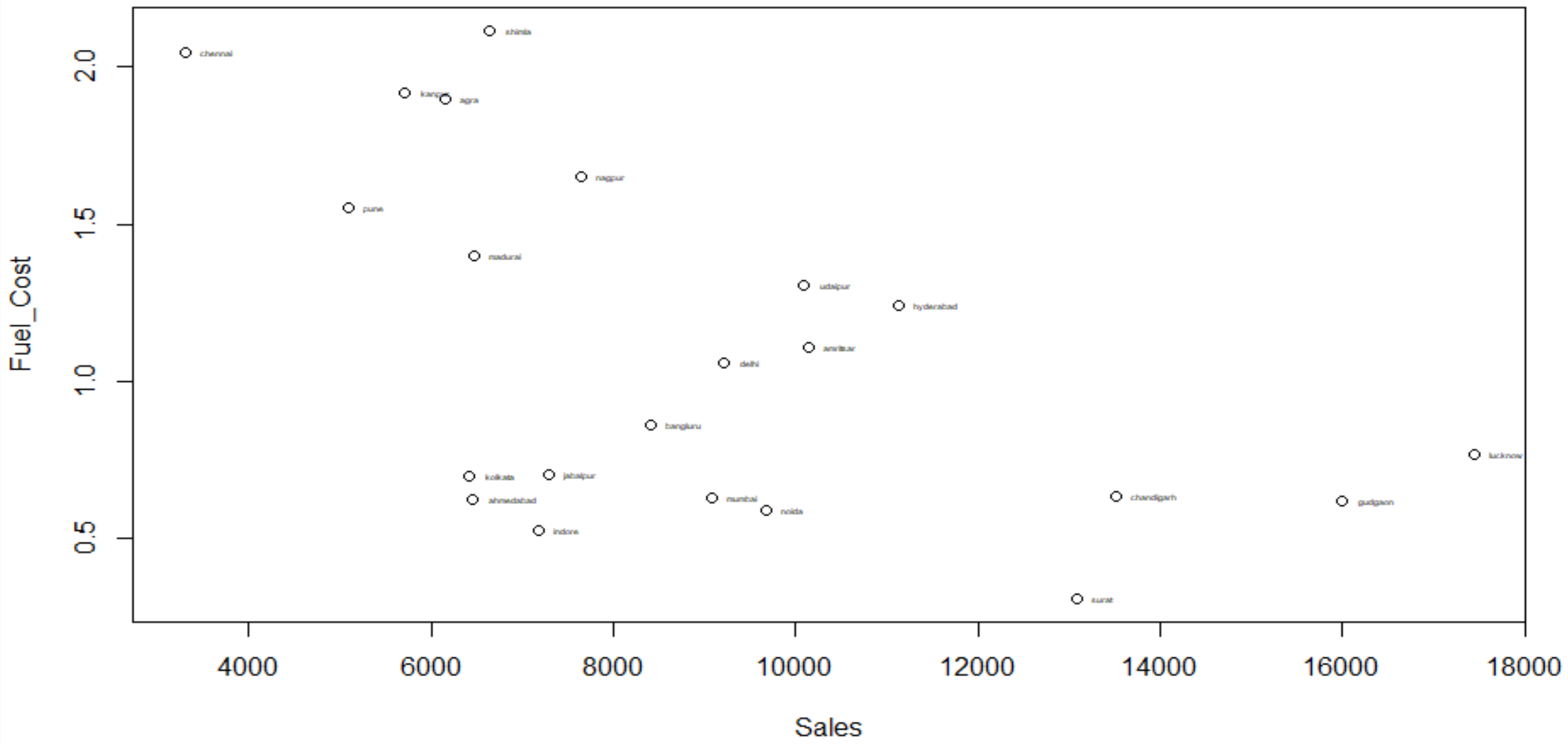
# How to perform

# HIERARCHICAL CLUSTERING?

# STEPS :-


IMPORT THE DATA SET
CREATE A SCATTER PLOT
NORMALIZE THE DATA
CALCULATE EUCLIDEAN DISTANCE
CREATE A DENDOGRAM

| CITY | Fixed_charge | RoR | Cost | Load | D Demand | Sales | Nuclear | Fuel_Cost |
|---|---|---|---|---|---|---|---|---|
| mumbai | 1.06 | 9.2 | 151 | 54.4 | 1.6 | 9077 | 0 | 0.628 |
| pune | 0.89 | 10.3 | 202 | 57.9 | 2.2 | 5088 | 25.3 | 1.555 |
| delhi | 1.43 | 15.4 | 113 | 53 | 3.4 | 9212 | 0 | 1.058 |
| kolkata | 1.02 | 11.2 | 168 | 56 | 0.3 | 6423 | 34.3 | 0.7 |
| chennai | 1.49 | 8.8 | 192 | 51.2 | 1 | 3300 | 15.6 | 2.044 |
| hyderabad | 1.32 | 13.5 | 111 | 60 | -2.2 | 11127 | 22.5 | 1.241 |
| nagpur | 1.22 | 12.2 | 175 | 67.6 | 2.2 | 7642 | 0 | 1.652 |
| surat | 1.1 | 9.2 | 245 | 57 | 3.3 | 13082 | 0 | 0.309 |
| bangluru | 1.34 | 13 | 168 | 60.4 | 7.2 | 8406 | 0 | 0.862 |
| ahmedabad | 1.12 | 12.4 | 197 | 53 | 2.7 | 6455 | 39.2 | 0.623 |
| lucknow | 0.75 | 7.5 | 173 | 51.5 | 6.5 | 17441 | 0 | 0.768 |
| agra | 1.13 | 10.9 | 178 | 62 | 3.7 | 6154 | 0 | 1.897 |
| indore | 1.15 | 12.7 | 199 | 53.7 | 6.4 | 7179 | 50.2 | 0.527 |
| noida | 1.09 | 12 | 96 | 49.8 | 1.4 | 9673 | 0 | 0.588 |
| madurai | 0.96 | 7.6 | 164 | 62.2 | -0.1 | 6468 | 0.9 | 1.4 |
| gudgaon | 1.16 | 9.9 | 252 | 56 | 9.2 | 15991 | 0 | 0.62 |
| kanpur | 0.76 | 6.4 | 136 | 61.9 | 9 | 5714 | 8.3 | 1.92 |
| amritsar | 1.05 | 12.6 | 150 | 56.7 | 2.7 | 10140 | 0 | 1.108 |
| chandigarh | 1.16 | 11.7 | 104 | 54 | -2.1 | 13507 | 0 | 0.636 |
| jabalpur | 1.2 | 11.8 | 148 | 59.9 | 3.5 | 7287 | 41.1 | 0.702 |
| shimla | 1.04 | 8.6 | 204 | 61 | 3.5 | 6650 | 0 | 2.116 |
| udaipur | 1.07 | 9.3 | 174 | 54.3 | 5.9 | 10093 | 26.6 | 1.306 |

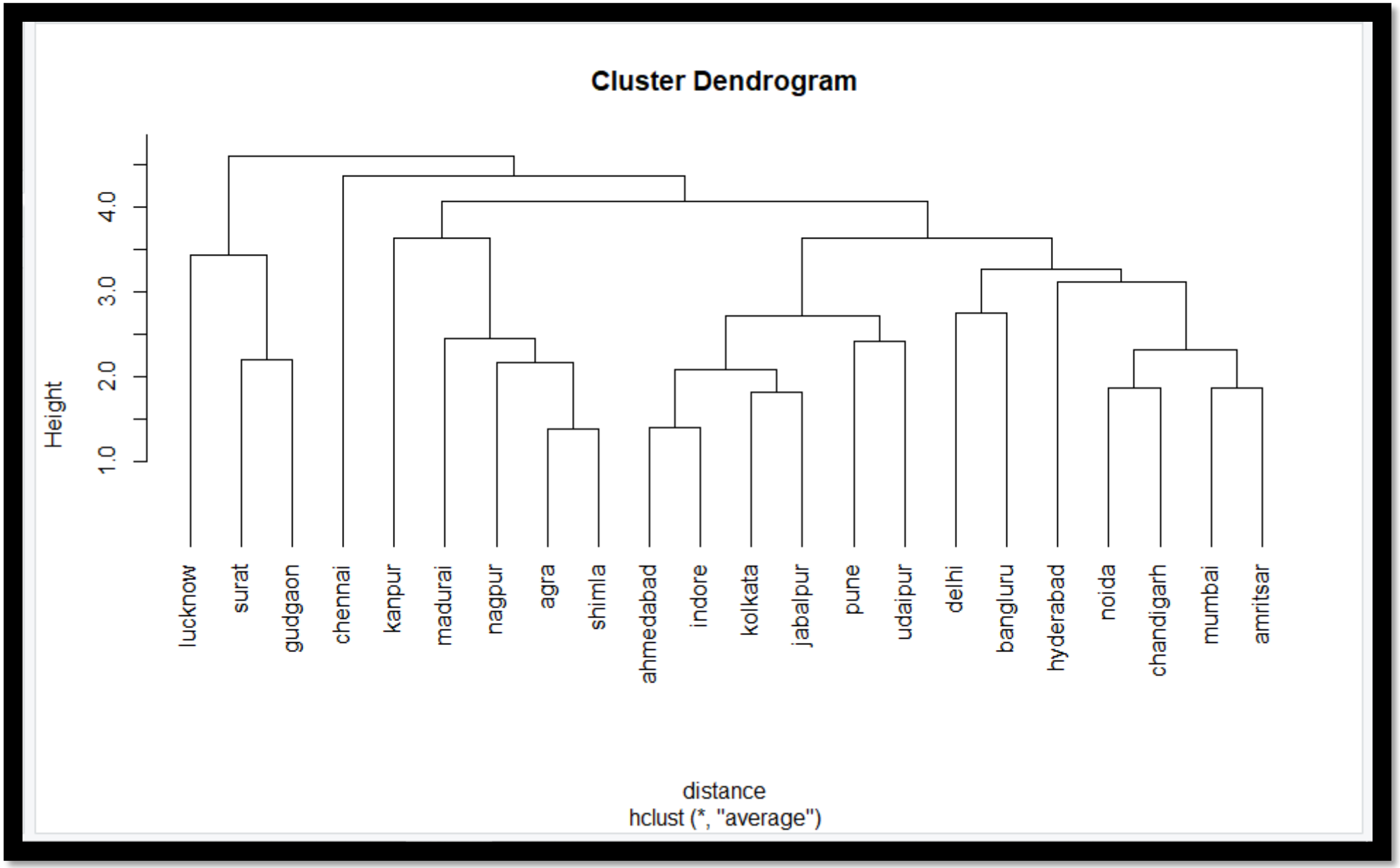

Fuel_Cost
0.5
1.0
1.5
2.0
Sales
4000
6000
8000
10000
12000
14000
16000
18000
chennai
shimla
kanpur
agra
nagpur
pune
madurai
udaipur
hyderabad
amritsar
delhi
bangluru
lucknow
kolkata
jabalpur
ahmedabad
mumbai
noida
chandigarh
gudgaon
indore
surat

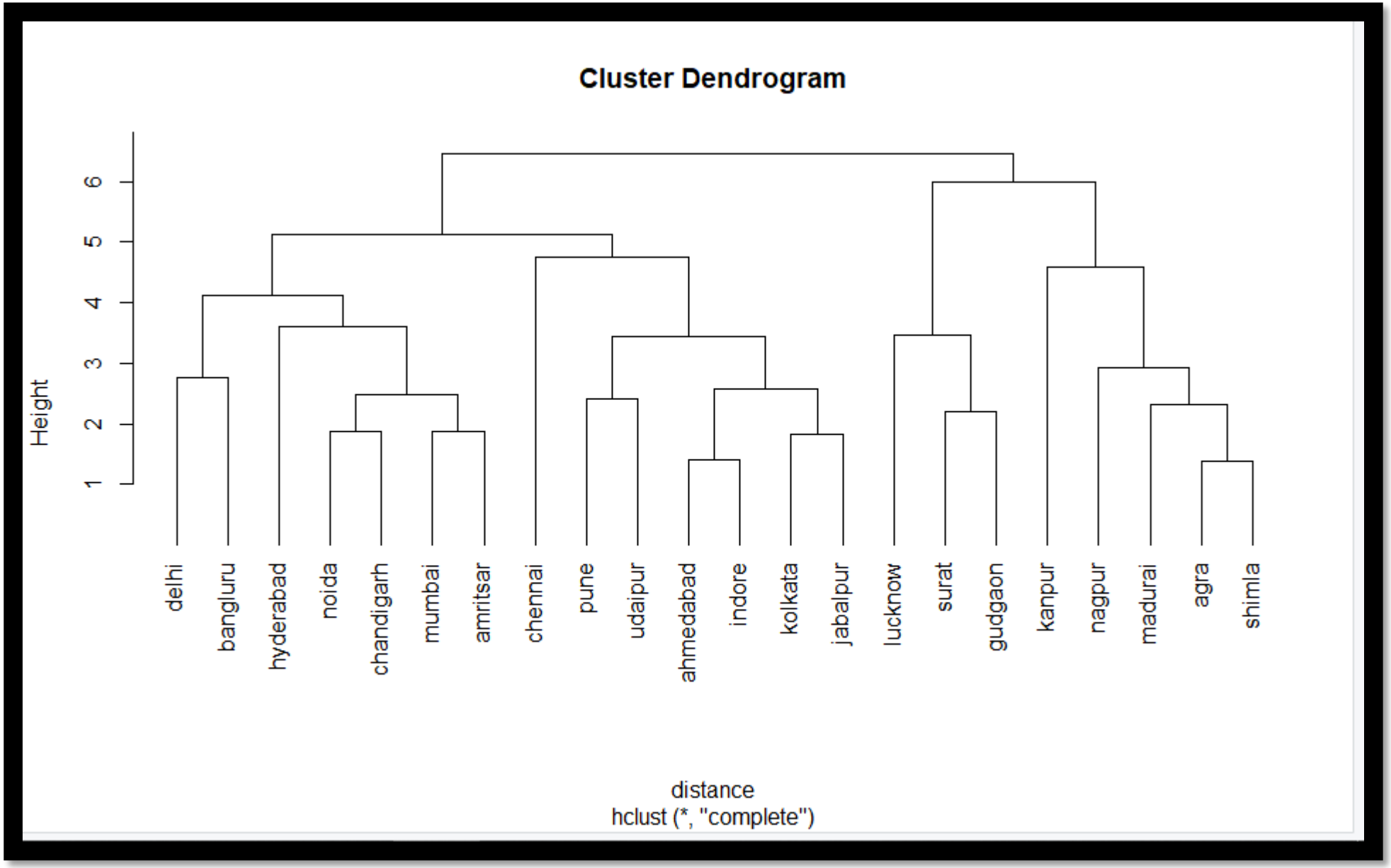
```r
plot(RoR~Sales, mydata)
with(mydata,text(RoR~Sales, labels = City, pos = 4, cex = .3))



#normalization

z <- mydata[,-c(1)]
m <- apply(z,2,mean)
s <- apply(z,2,sd)
z <- scale(z,m,s)


m
```

```r



#calculate the Euclidean Distance

distance <- dist(z)
distance
print(distance, digits=2)


```

```
22 2.739910 3.512207 3.352644 3.457129 3.628061 2.548060 3.967618 2.618050 3.012264
> print(distance, digits=2)
     1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21
2  3.1
3  3.7 4.9
4  2.5 2.2 4.1
5  4.1 3.9 4.5 4.1
6  3.6 4.2 3.0 3.2 4.6
7  3.9 3.4 4.2 4.0 4.6 3.4
8  2.7 3.9 5.0 3.7 5.2 4.9 4.4
9  3.3 4.0 2.8 3.8 4.5 3.7 2.8 3.6
10 3.1 2.7 3.9 1.5 4.0 3.8 4.5 3.7 3.6
11 3.5 4.8 5.9 4.9 6.5 6.0 6.0 3.5 5.2 5.1
12 3.2 2.4 4.0 3.5 3.6 3.7 1.7 4.1 2.7 3.9 5.2
13 4.0 3.4 4.4 2.6 4.8 4.6 5.0 4.1 3.7 1.4 5.3 4.5
14 2.1 4.3 2.7 3.2 4.8 3.5 4.9 4.3 3.8 3.6 4.3 4.3 4.4
15 2.6 2.5 5.2 3.2 4.3 4.1 2.9 3.8 4.1 4.3 4.7 2.3 5.1 4.2
16 4.0 4.8 5.3 5.0 5.8 5.8 5.0 2.2 3.6 4.5 3.4 4.6 4.4 5.2 5.2
17 4.4 3.6 6.4 4.9 5.6 6.1 4.6 5.4 4.9 5.5 4.8 3.5 5.6 5.6 3.4 5.6
18 1.9 2.9 2.7 2.7 4.3 2.9 2.9 3.2 2.4 3.1 3.9 2.5 3.8 2.3 3.0 4.0 4.4
19 2.4 4.6 3.2 3.5 5.1 2.6 4.5 4.1 4.1 4.1 4.5 4.4 5.0 1.9 4.0 5.2 6.1 2.5
20 3.2 3.0 3.7 1.8 4.4 2.9 3.5 4.1 2.9 2.1 5.4 3.4 2.2 3.7 3.8 4.8 4.9 2.9 3.9
21 3.5 2.3 5.1 3.9 3.6 4.6 2.7 4.0 3.7 4.4 4.9 1.4 4.9 4.9 2.1 4.6 3.1 3.2 5.0 4.1
22 2.5 2.4 4.1 2.6 3.8 4.0 4.0 3.2 3.2 2.6 3.4 3.0 2.7 3.5 3.4 3.5 3.6 2.5 4.0 2.6 3.0
>
```

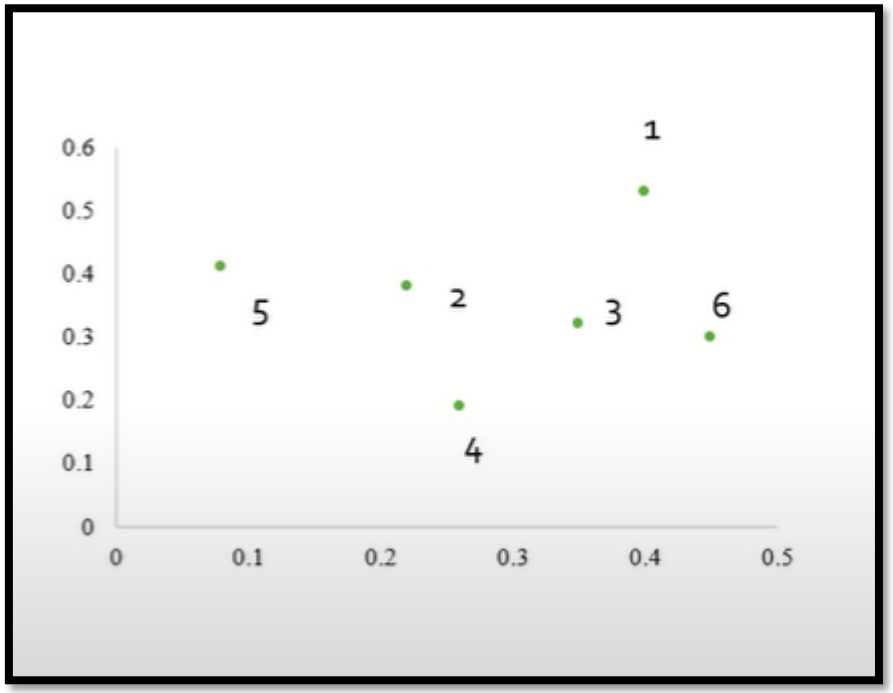
```r

#clustering dendogram

hc.l <- hclust(distance)
plot(hc.l)
plot(hc.l, labels= mydata$City, hang =-1)

#clustering dendogram average

hc.l <- hclust(distance, method = "average")
plot(hc.l, labels= mydata$City, hang =-1)
```


Cluster Dendrogram
Height
1.0
2.0
3.0
4.0
lucknow
surat
gudgaon
chennai
kanpur
madurai
nagpur
agra
shimla
ahmedabad
indore
kolkata
jabalpur
pune
udaipur
delhi
bangluru
hyderabad
noida
chandigarh
mumbai
amritsar
distance
hclust (*, "average")


Cluster Dendrogram
Height
1
2
3
4
5
6
delhi
bangluru
hyderabad
noida
chandigarh
mumbai
amritsar
chennai
pune
udaipur
ahmedabad
indore
kolkata
jabalpur
lucknow
surat
gudgaon
kanpur
nagpur
madurai
agra
shimla
distance
hclust (*, "complete")

- Find the clusters using single link technique.
- Use Euclidean distance

| | X | Y |
|---|---|---|
| P1 | 0.4 | 0.53 |
| P2 | 0.22 | 0.38 |
| P3 | 0.35 | 0.32 |
| P4 | 0.26 | 0.19 |
| P5 | 0.08 | 0.41 |
| P6 | 0.45 | 0.30 |



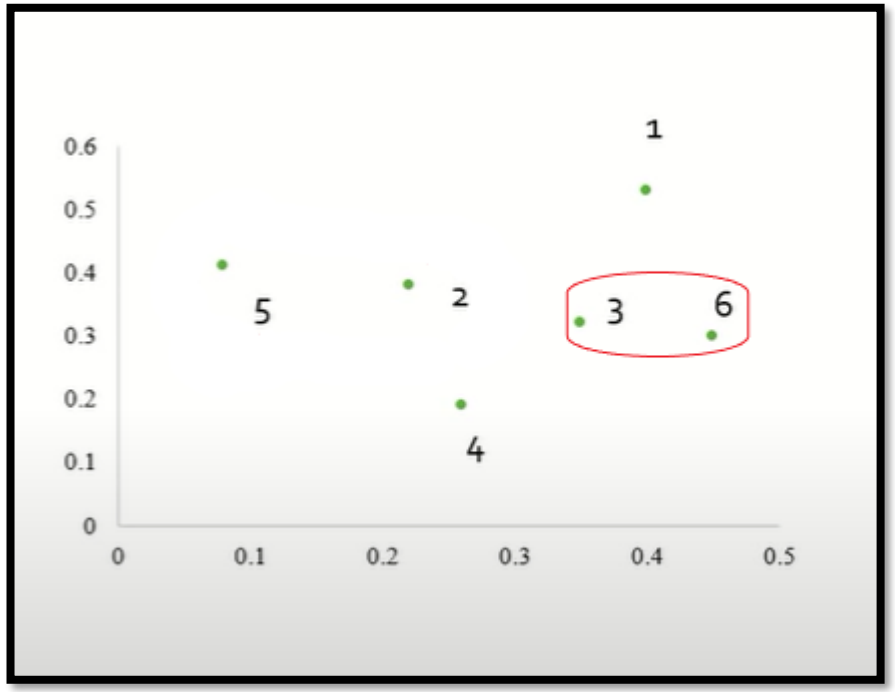
Euclidean Distance:-

Distance [ P1 , P2 ] = [ (x , y), (a , b) ] = $\sqrt{(x-a)^2 + (y-b)^2}$

| | P1 | P2 | P3 | P4 | P5 | P6 |
|---|---|---|---|---|---|---|
| P1 | 0 | | | | | |
| P2 | 0.23 | 0 | | | | |
| P3 | 0.22 | 0.15 | 0 | | | |
| P4 | 0.37 | 0.20 | 0.15 | 0 | | |
| P5 | 0.34 | 0.14 | 0.28 | 0.29 | 0 | |
| P6 | 0.23 | 0.25 | 0.11 | 0.22 | 0.39 | 0 |

- Using distances we get a ‘DISTANCE MATRIX’
- It is a Hollow Matrix.

| | P1 | P2 | P3 | P4 | P5 | P6 |
|---|---|---|---|---|---|---|
| P1 | 0 | | | | | |
| P2 | 0.23 | 0 | | | | |
| P3 | 0.22 | 0.15 | 0 | | | |
| P4 | 0.37 | 0.20 | 0.15 | 0 | | |
| P5 | 0.34 | 0.14 | 0.28 | 0.29 | 0 | |
| P6 | 0.23 | 0.25 | 0.11 | 0.22 | 0.39 | 0 |

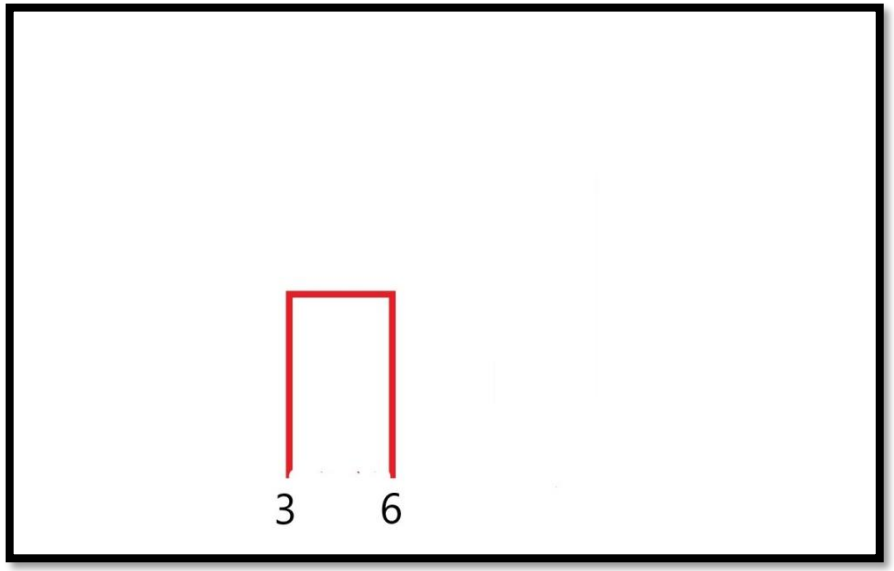



To update the distance matrix

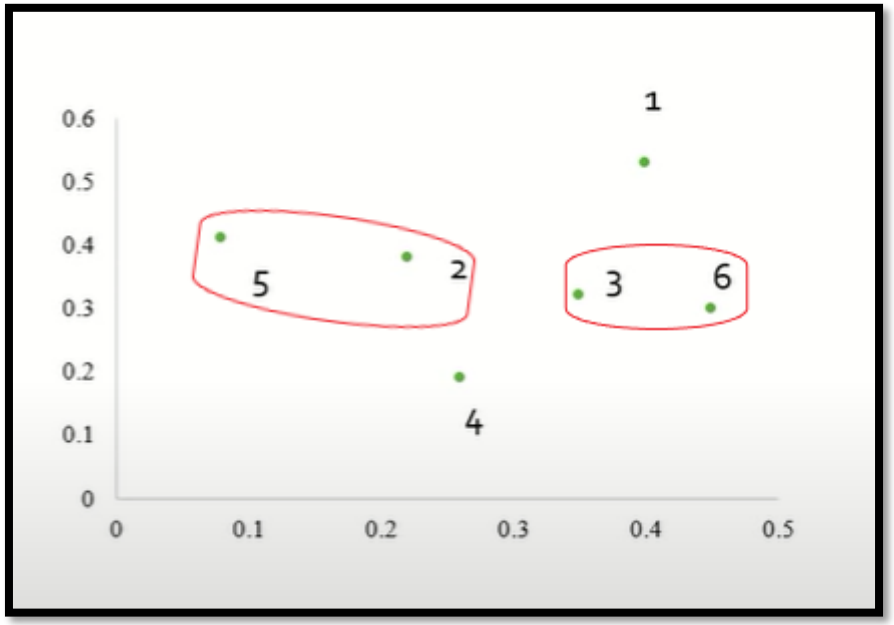
MIN [ dist (P3 , P6 ) , point ]

= MIN [ dist (P3,point) , (P6,point) ]

- This is new Distance

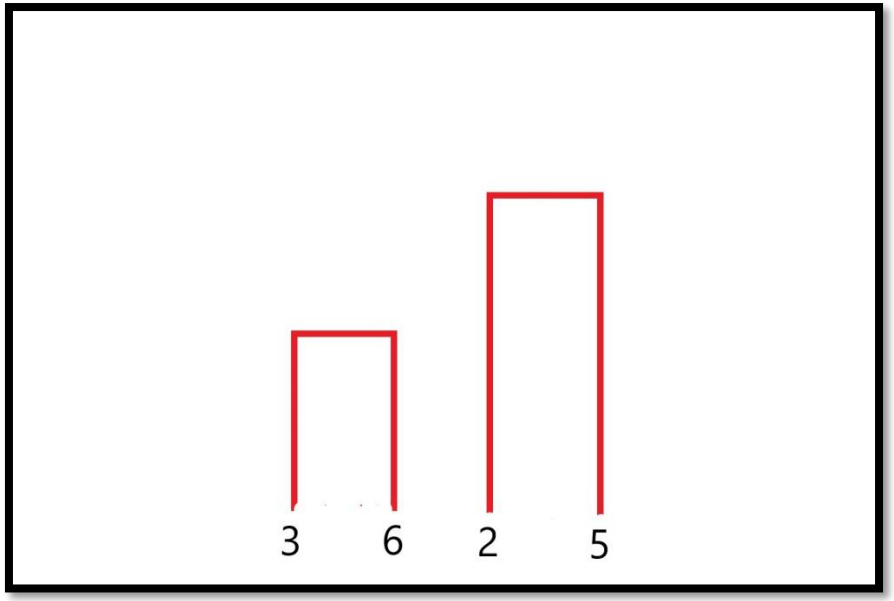


| | P1 | P2 | P3,P6 | P4 | P5 |
|---|---|---|---|---|---|
| P1 | 0 | | | | |
| P2 | 0.23 | 0 | | | |
| P3,P6 | 0.22 | 0.15 | 0 | | |
| P4 | 0.37 | 0.20 | 0.15 | 0 | |
| P5 | 0.34 | 0.14 | 0.28 | 0.29 | 0 |




- New Distance Matrix

| | P1 | P2 | P3,P6 | P4 | P5 |
|---|---|---|---|---|---|
| P1 | 0 | | | | |
| P2 | 0.23 | 0 | | | |
| P3,P6 | 0.22 | 0.15 | 0 | | |
| P4 | 0.37 | 0.20 | 0.15 | 0 | |
| P5 | 0.34 | 0.14 | 0.28 | 0.29 | 0 |



<u>To update the distance matrix</u>

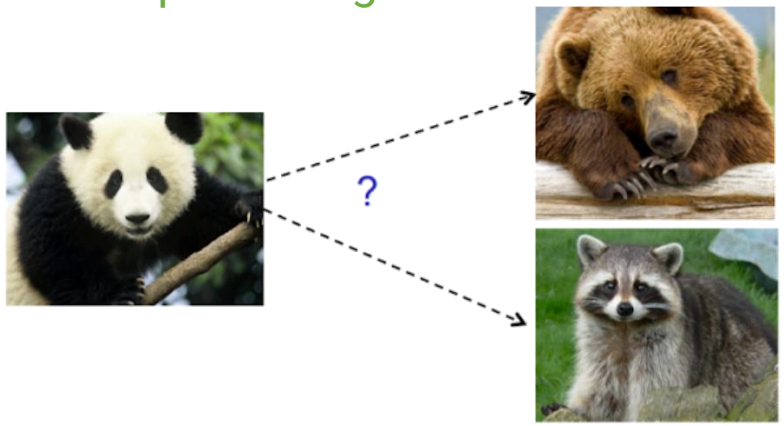
MIN [ dist (P2 , P5 ) , point ]

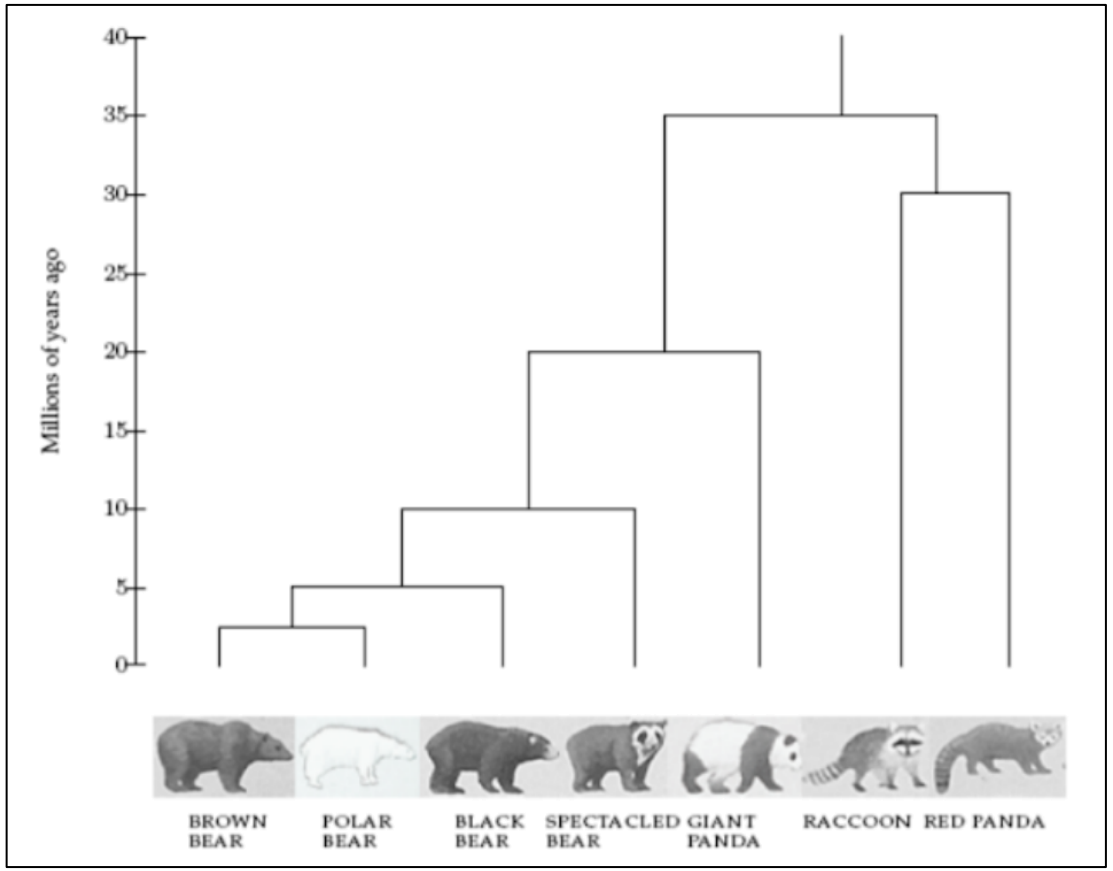
= MIN [ dist (P2,point) , (P5,point) ]

# APPLICATIONS : Charting Evolution through Phylogenetic Trees

How can we relate different species together?

*Are giant pandas closer to bears or racoons?*




- Nowadays, we can use DNA sequencing and hierarchical clustering to find the phylogenetic tree of animal evolution:

- Generate the DNA sequences
  Calculate the edit distance between all sequences.
  Calculate the DNA similarities based on the edit distances.
  Construct the phylogenetic tree.

- As a result of this experiment, the researchers were able to place the giant pandas closer to bears.

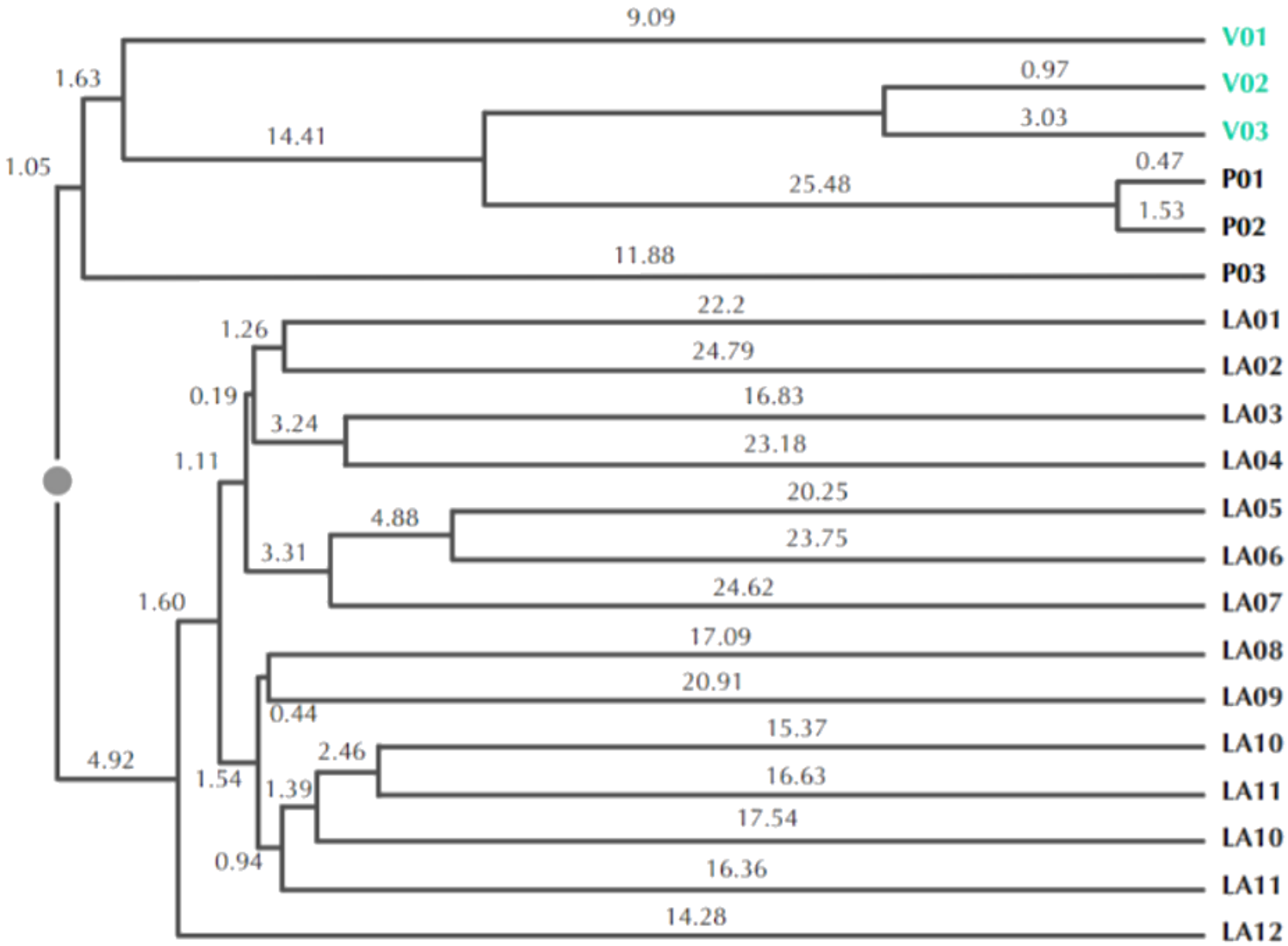
# APPLICATIONS : Tracking Viruses through Phylogenetic Tree

Can we find where a viral outbreak originated?

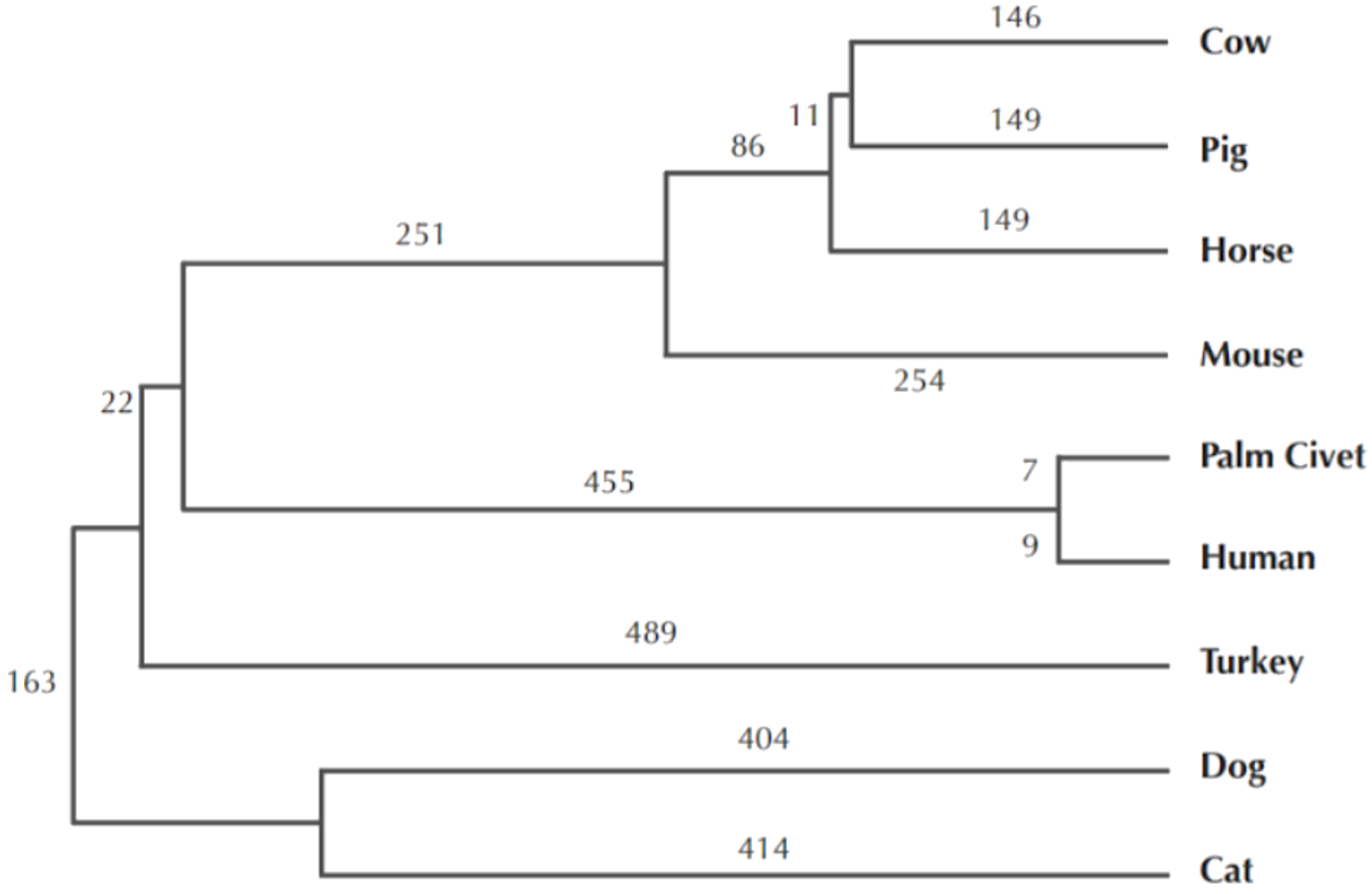
- Tracing these outbreaks to their source can give scientists additional data as to why and how the outbreak began, potentially saving lives.

- Viruses such as HIV have high mutation rates, which means the similarity of the DNA sequence of the same virus depends on the time since it was transmitted. This can be used to trace paths of transmission.

- **This method was used as evidence in a court case, wherein the victim's strand of HIV was found to be more similar to the accused patient's strand, compared to a control group.**



**A similar study was also done for finding the animal that gave the humans the SARS virus:**



## ADVANTAGES

**No need to pre-specify the number of clusters.**

**Easy to understand and implement.**

Produces an order of objects, which may be **informative for display and better visualization.**

## DISADVANTAGES

**Dendrogram is commonly misinterpreted**

**No objective function is directly minimized**

Once a decision is made to combine two clusters, it **cannot be undone**

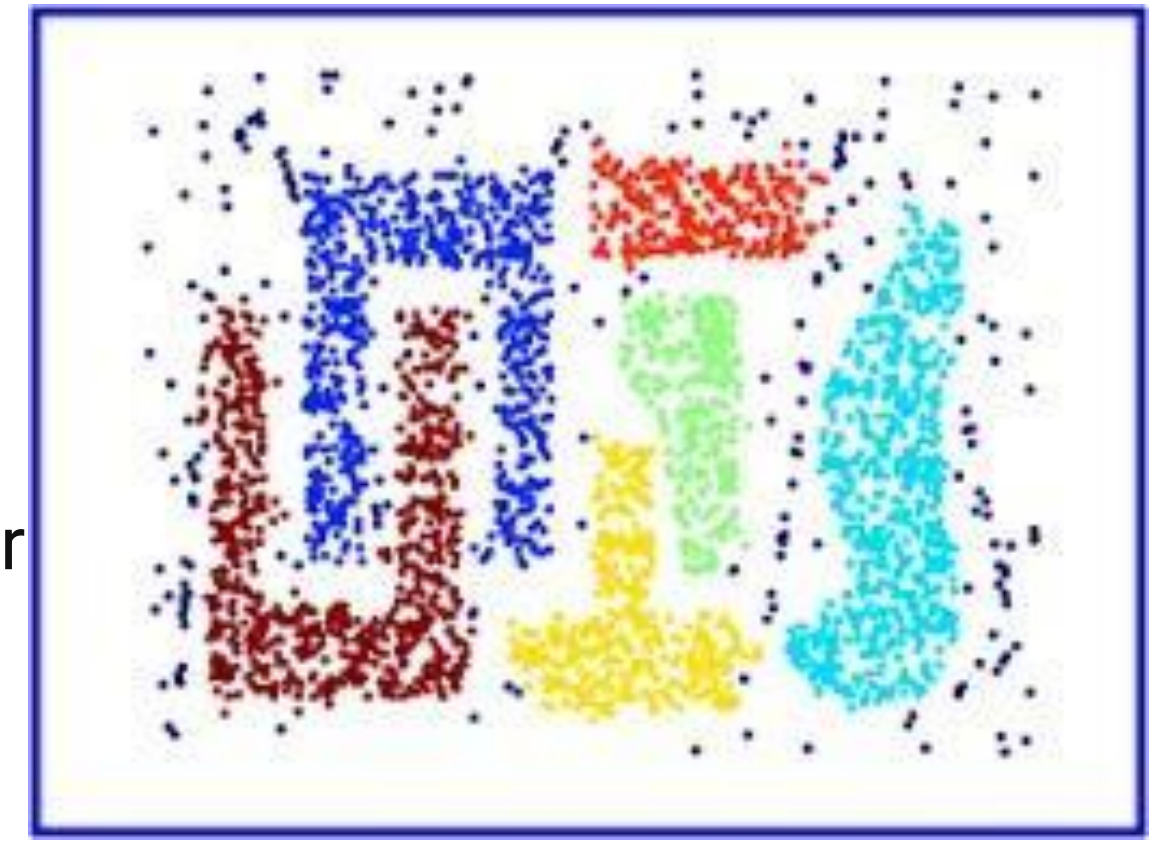
Does not work with **missing data**

Does not work well on very **large data sets**

**May not give best results** in all cases

# DENSITY-BASED CLUSTERING

Density-Based Clustering identifies distinctive groups/clusters in the data, based on the idea that a cluster in a data space is a contiguous region of high point density, separated from other such clusters by contiguous regions of low point density.



The data points in the separating regions of low point density are typically considered noise/outliers.

# The DBSCAN algorithm uses two parameters:

MINPTS

It is the minimum number of points (a threshold) clustered together, for a region to be considered dense.

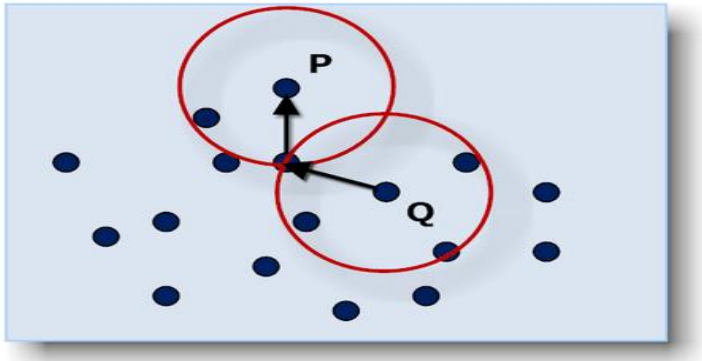
EPS (ε)

The distance that specifies the neighbourhoods. Two points are considered to be neighbours if the distance between them are less than or equal to eps.

These parameters can be understood if we explore two concepts called Density Reachability and Density Connectivity.

## Density Reachability -

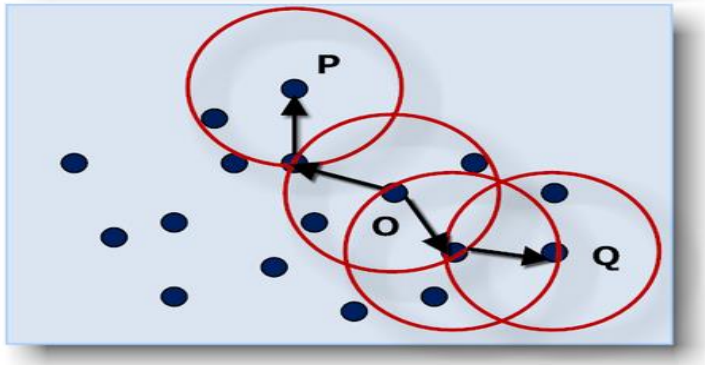
A point "p" is said to be density reachable from a point "q" if point "p" is within ε distance from point "q" and "q" has a sufficient number of points in its neighborhood which are within distance ε.




(a)

P is density-reachable from Q

Q is not density-reachable from P




(b)

P and Q are density connected to each other by O

## Density Connectivity -

A point "p" and "q" are said to be density connected if there exists a point "o" which has a sufficient number of points in its neighbors and both the points "p" and "q" are within the ε distance from "o".

**This is a chaining process.**
So, if "q" is neighbour of "o", "o" is neighbour of "s", "s" is a neighbour of "t" which in turn is neighbour of "p" implies that "q" is neighbour of "p".

**There are three types of points after the DBSCAN clustering is complete:**

## CORE POINT

A data point is considered to be a core point if it has a minimum number of neighbouring data points (min_pts) at an epsilon distance from it. (These min_pts include the original data points also.)

## BORDER POINT

A data point that has less than the minimum number of neighbouring data points needed but has at least one core point in the neighbourhood.

## NOISE POINT

A data point that is not a core point or a border point is considered noise or an outlier.

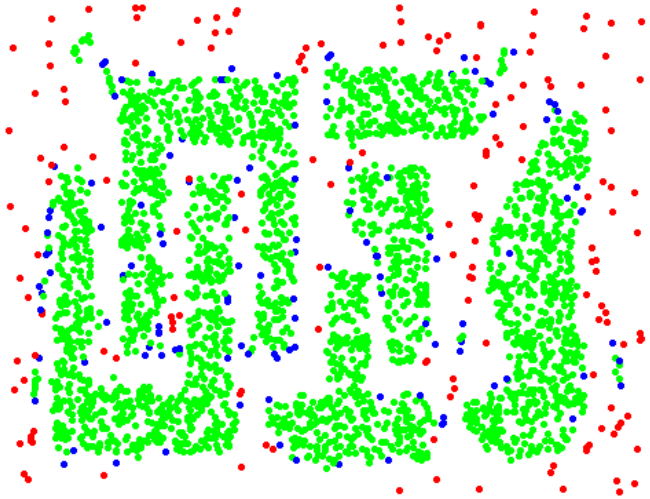
# ORIGINAL DATASET

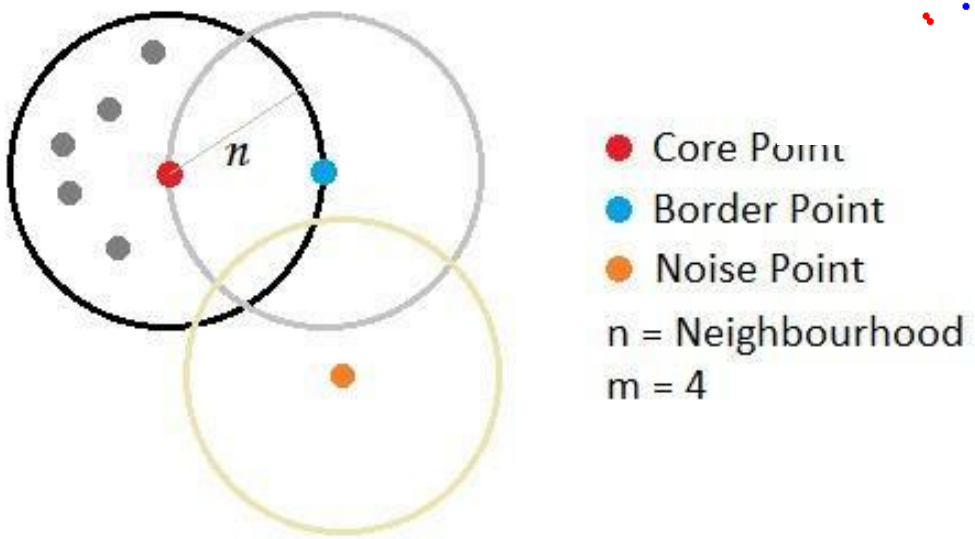
# DBSCAN DATASET

CORE

BORDER

NOISE






DBSCAN CLUSTERING

*Abhijit Annaldas*

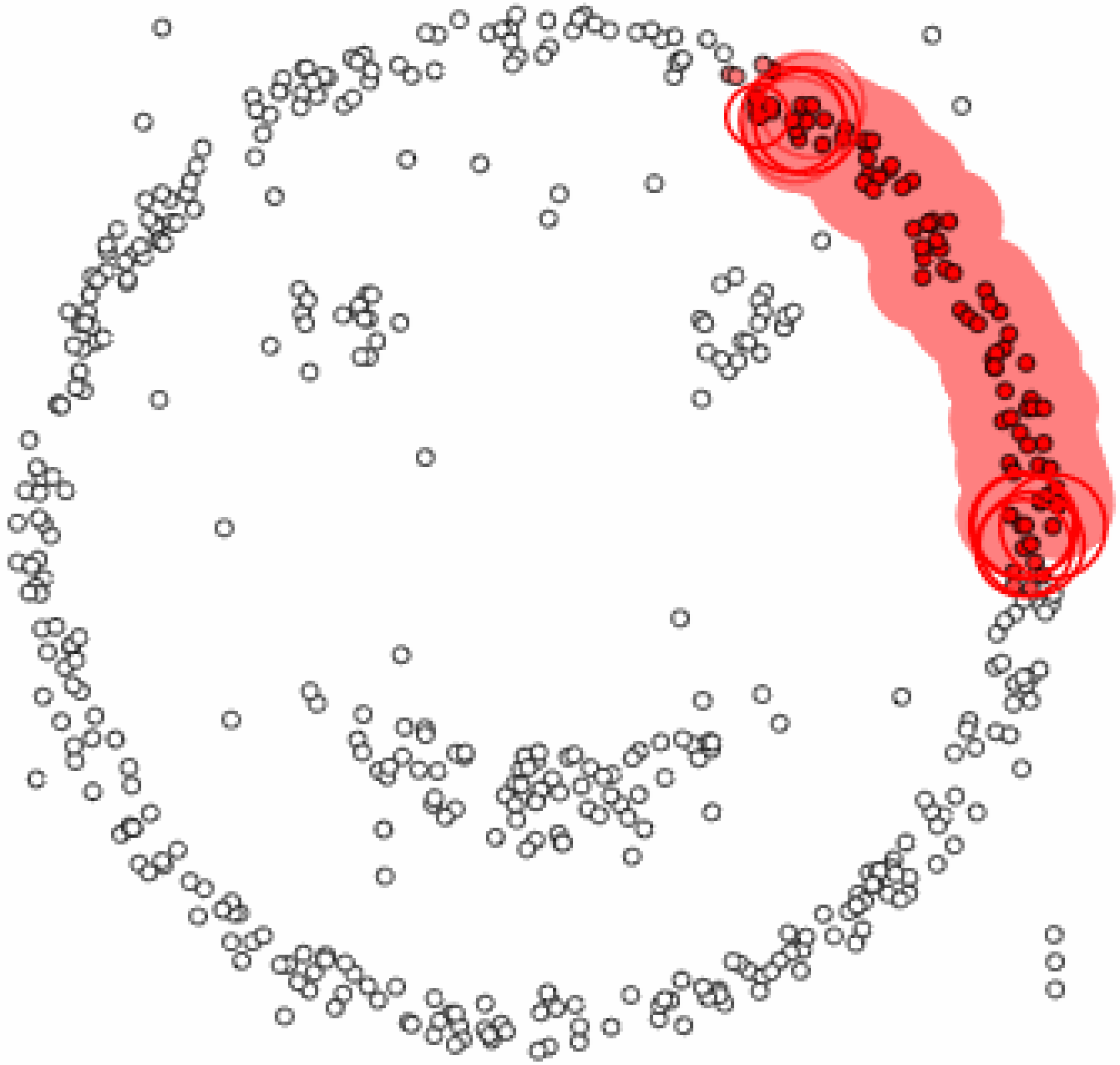
## THE DBSCAN ALGORITHM IS AS FOLLOWS:

- Randomly select a point p.
  Retrieve all the points that are density reachable from p with regard to the Maximum radius of the neighbourhood (EPS) and the minimum number of points within the eps neighbourhood (Min Pts).

- If the number of points in the neighbourhood is more than Min Pts then p is a core point. We assign a new cluster for the core point p.

- Find all its density connected points and assign them to the same cluster as the core point.  If p is not a core point, then mark it as a noise/outlier and move to the next point.

- Continue the process until all the points have been processed and visited.



epsilon = 1.00
minPoints = 4

# How to perform

# DENSITY BASED CLUSTERING IN R?

# APPLICATIONS OF DBSCAN CLUSTERING

- **Images of satellite**
- **Crystallography of x-ray**
- **Anomaly detection in temperature data**

## ADVANTAGES

**Does not require one to specify the number of clusters** in the data a priori, as opposed to k-means.

**Can find arbitrarily or non-linear shaped clusters.**

**Can even find a cluster completely surrounded** by (but not connected to) a different cluster.

**Requires just two parameters** and is mostly insensitive to the ordering of the points in the database.

**Has a notion of noise**, and is robust to outliers.

## DISADVANTAGES

**Fails in case of high varying density clusters. Cannot cluster data-sets with large differences** in densities well.

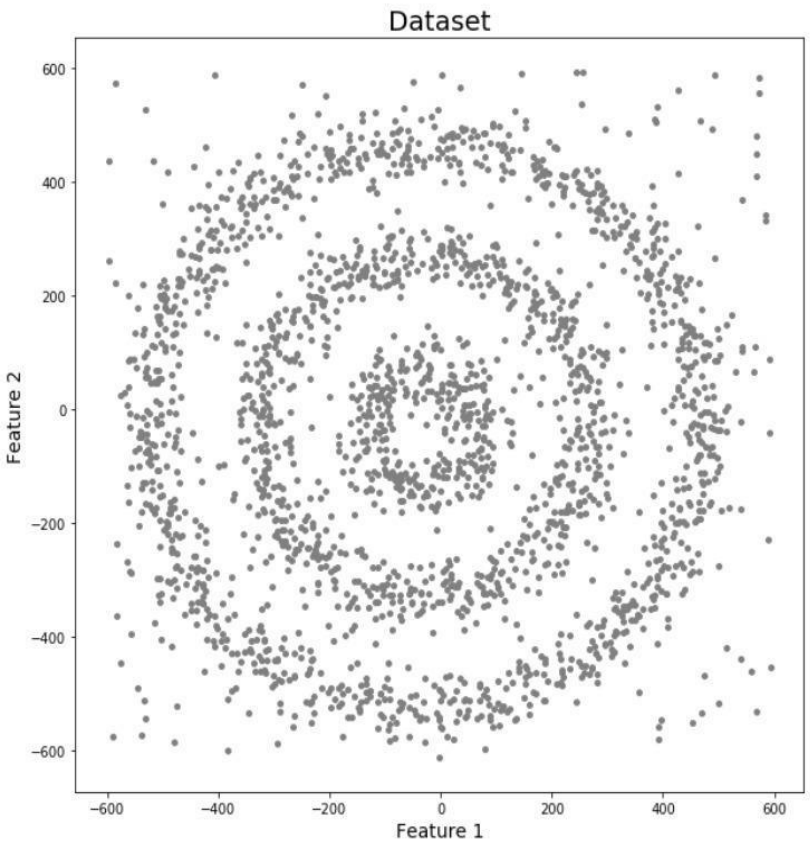
Hence the **minPts- eps combination cannot be chosen** appropriately for all clusters.

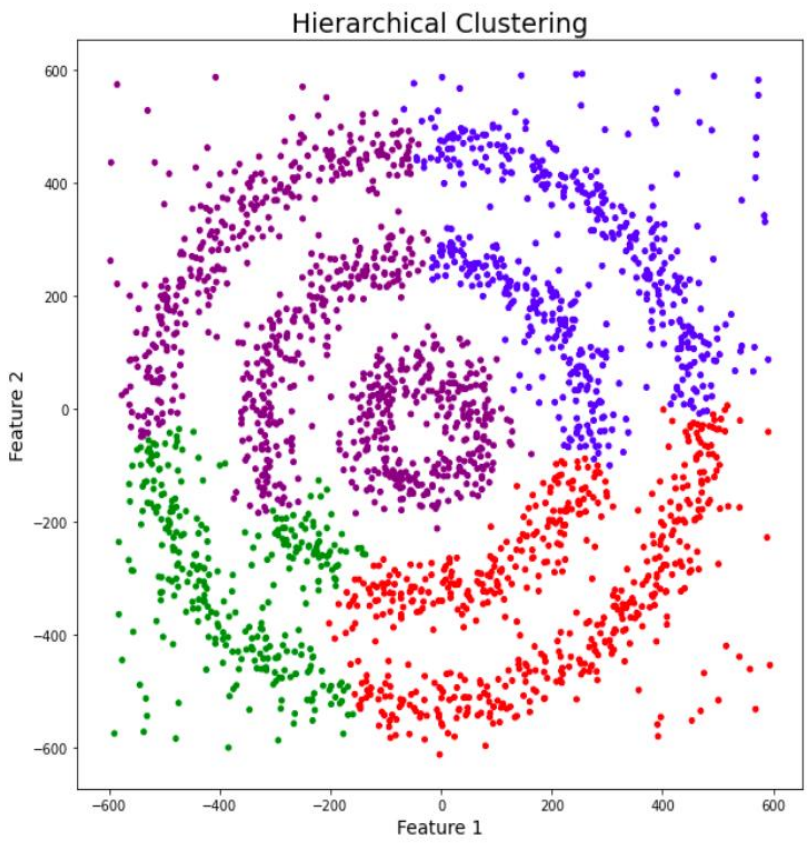
B**order points that are reachable from more than one cluster can be part of either cluster,** depending on the order in which the data are processed.

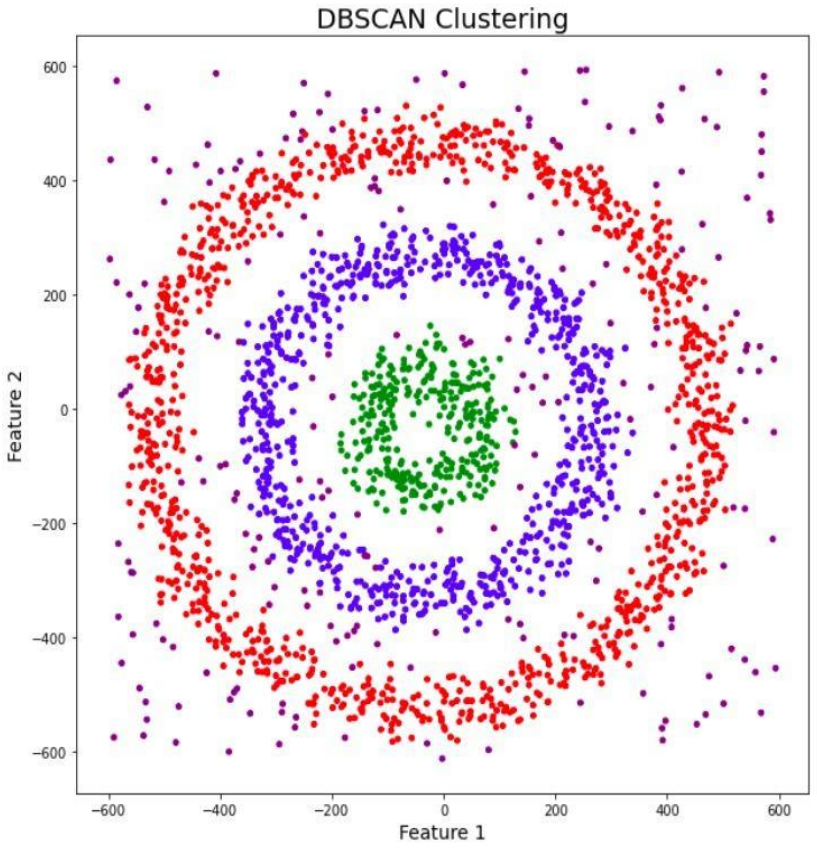
**Choosing a meaningful eps value can be difficult** if the data isn't well understood.

DBSCAN is **not entirely deterministic.**

# COMPARISON


Dataset
Feature 2
Feature 1
600
400
200
0
−200
−400
−600


Hierarchical Clustering
Feature 2
Feature 1
600
400
200
0
−200
−400
−600


DBSCAN Clustering
Feature 2
Feature 1
600
400
200
0
−200
−400
−600

# THANK YOU