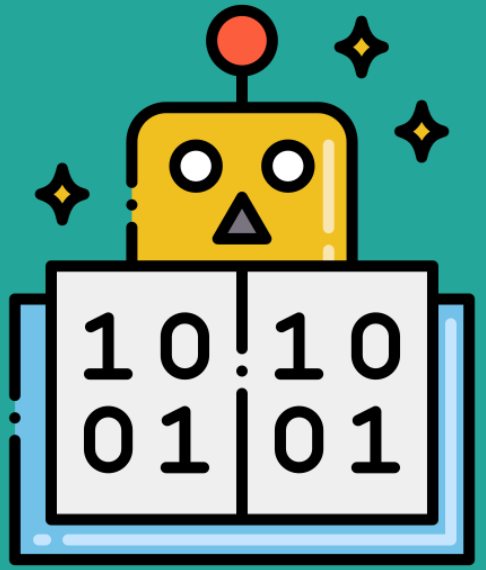
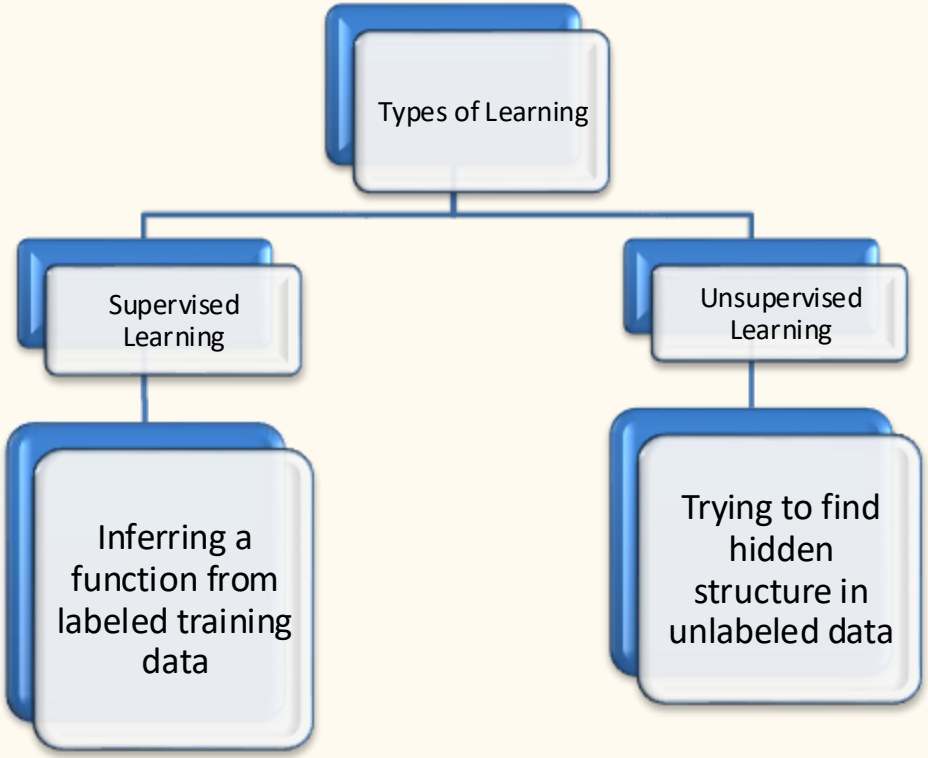
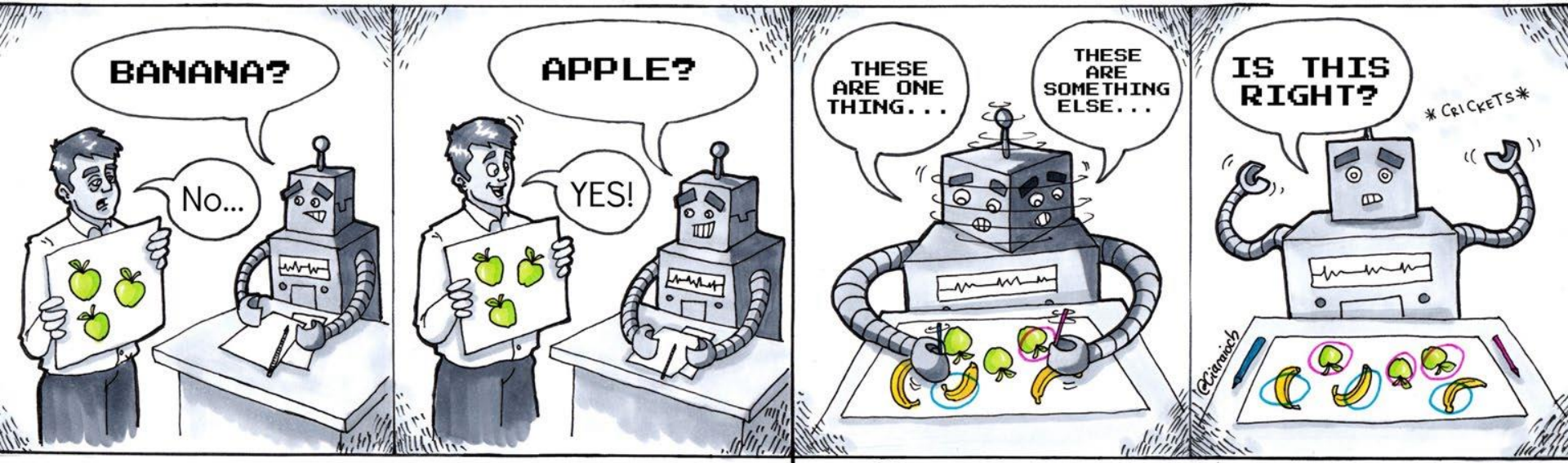
Introduction



Supervised vs. Unsupervised Learning





Supervised Learning

Unsupervised Learning

Example Clustering

Task 1 : Group These Set of Document into 3 Groups based on meaning

Doc1 : Health , Medicine, Doctor

Doc 2 : Machine Learning, Computer

Doc 3 : Environment, Planet

Doc 4 : Pollution, Climate Crisis

Doc 5 : Covid, Health , Doctor

Task 1 : Group These Set of Document into 3 Groups.

Doc1 : Health , Medicine, Doctor

Doc 2 : Machine Learning, Computer

Doc 3 : Environment, Planet

Doc 4 : Pollution, Climate Crisis

Doc 5 : Covid, Health , Doctor

Task 1 : Group These Set of Document into 3 Groups.

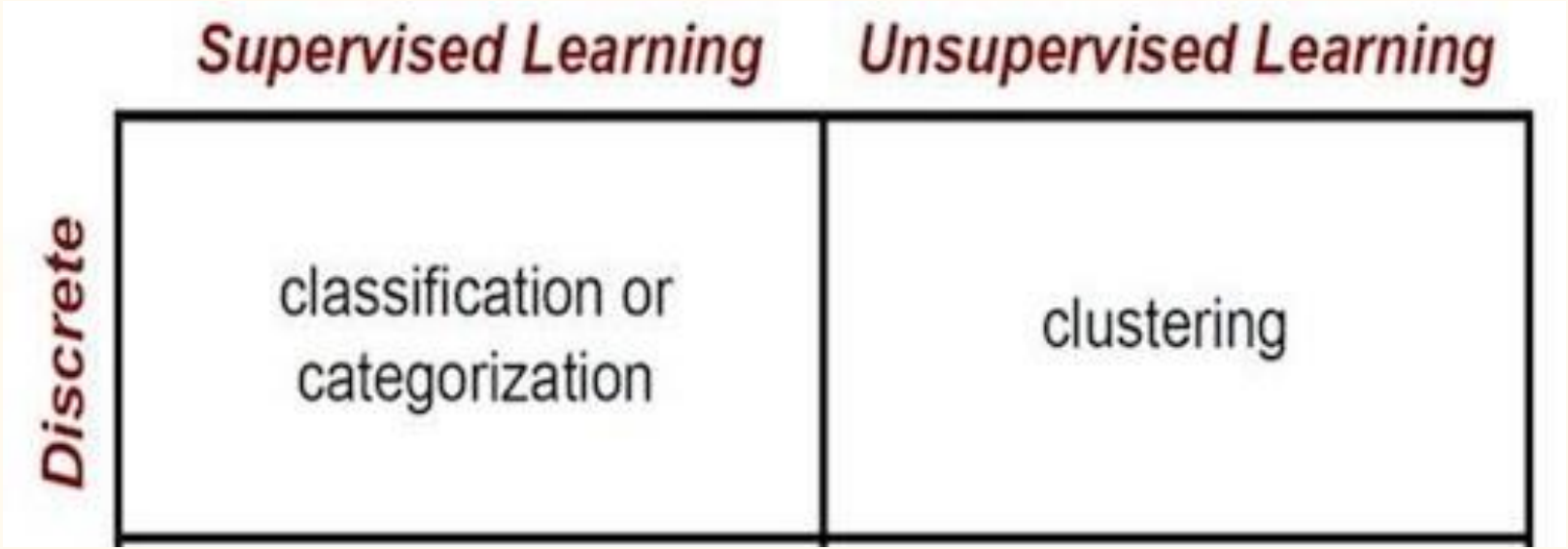
Doc 3 : Environment, Planet

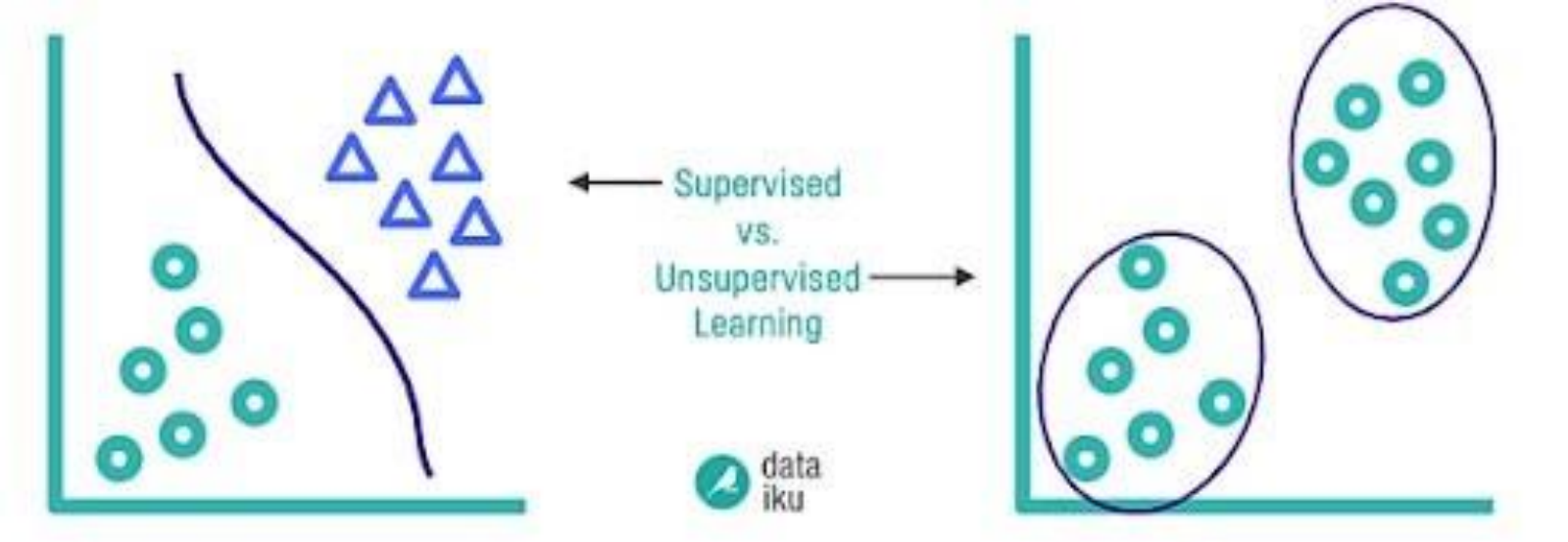
Doc1 : Health , Medicine, Doctor

Doc 4 : Pollution, Climate Crisis

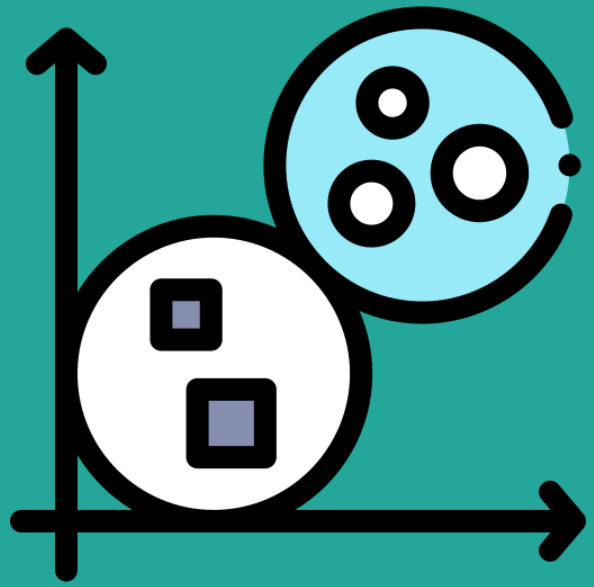
Doc 5 : Covid, Health , Doctor

Doc 2 : Machine Learning, Computer


Supervised Learning
Unsupervised Learning
Discrete
classification or categorization
clustering


Supervised
vs.
Unsupervised
Learning
data
iku

Types of Clustering



What is a Cluster?

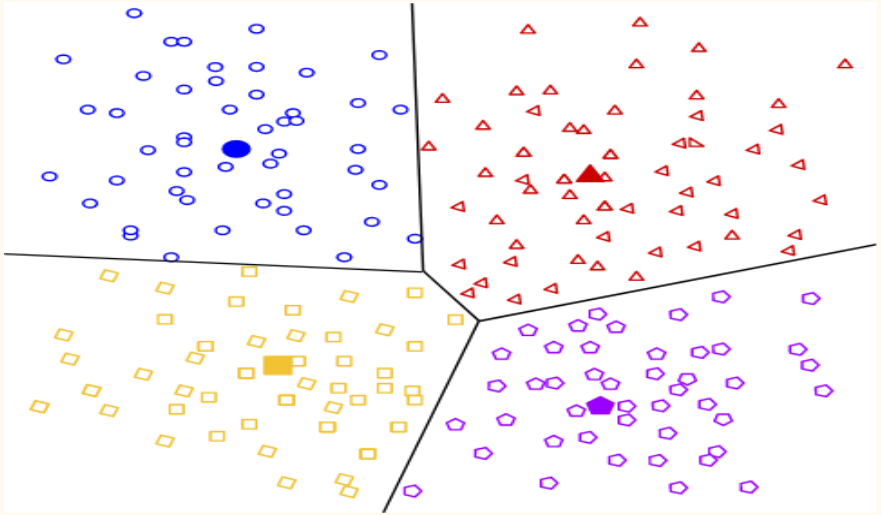
- Unsupervised Learning algorithms are classified into two categories.
- **Clustering:** Clustering is a technique of grouping objects into clusters.
- **Association:** helps in finding the relationships between variables in a large database.

Types of Clustering

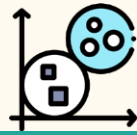
- Centroid-based Clustering
- Density-based Clustering
- Hierarchical Clustering
- Distribution-based Clustering

Centroid-based Clustering

- **Centroid-based clustering** organizes the data into non-hierarchical clusters, in contrast to hierarchical clustering defined below.
- k-means is the most widely-used centroid-based clustering algorithm. Centroid-based algorithms are efficient but sensitive to initial conditions and outliers.



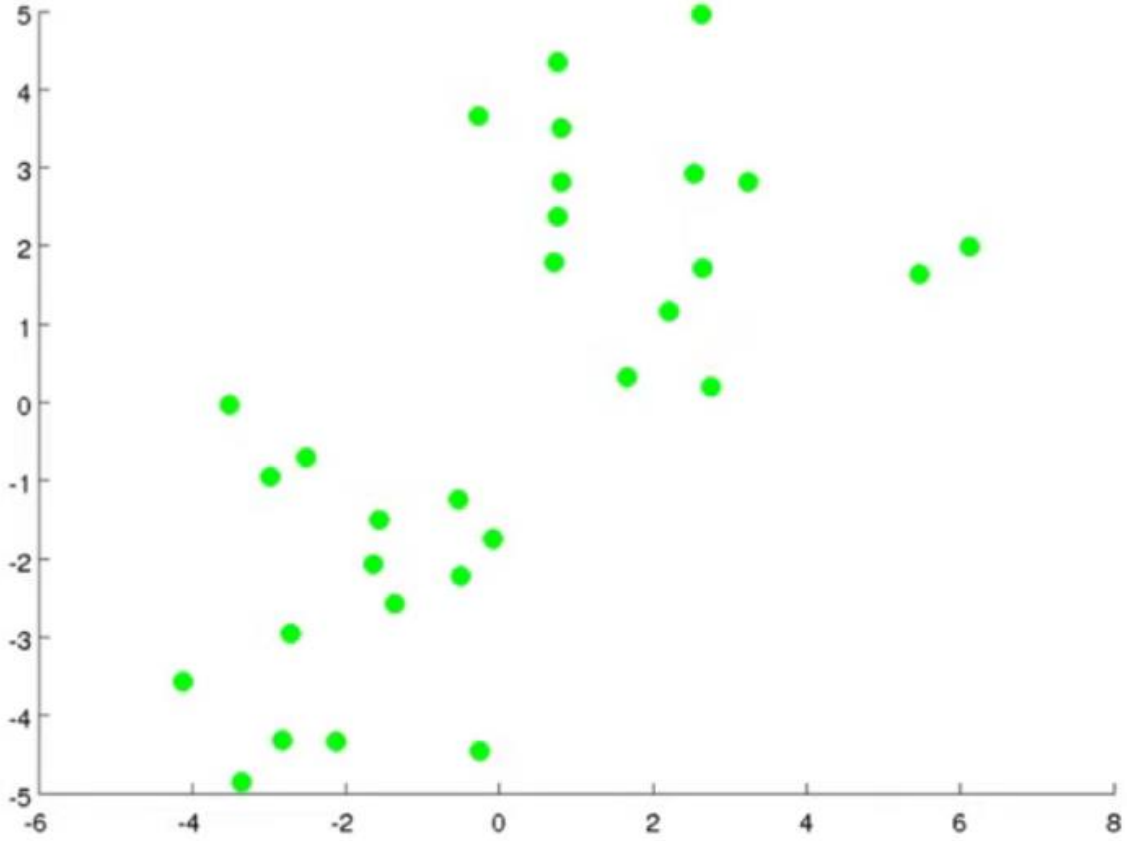
One such Centroid Based Clustering Algorithm Is K-Means

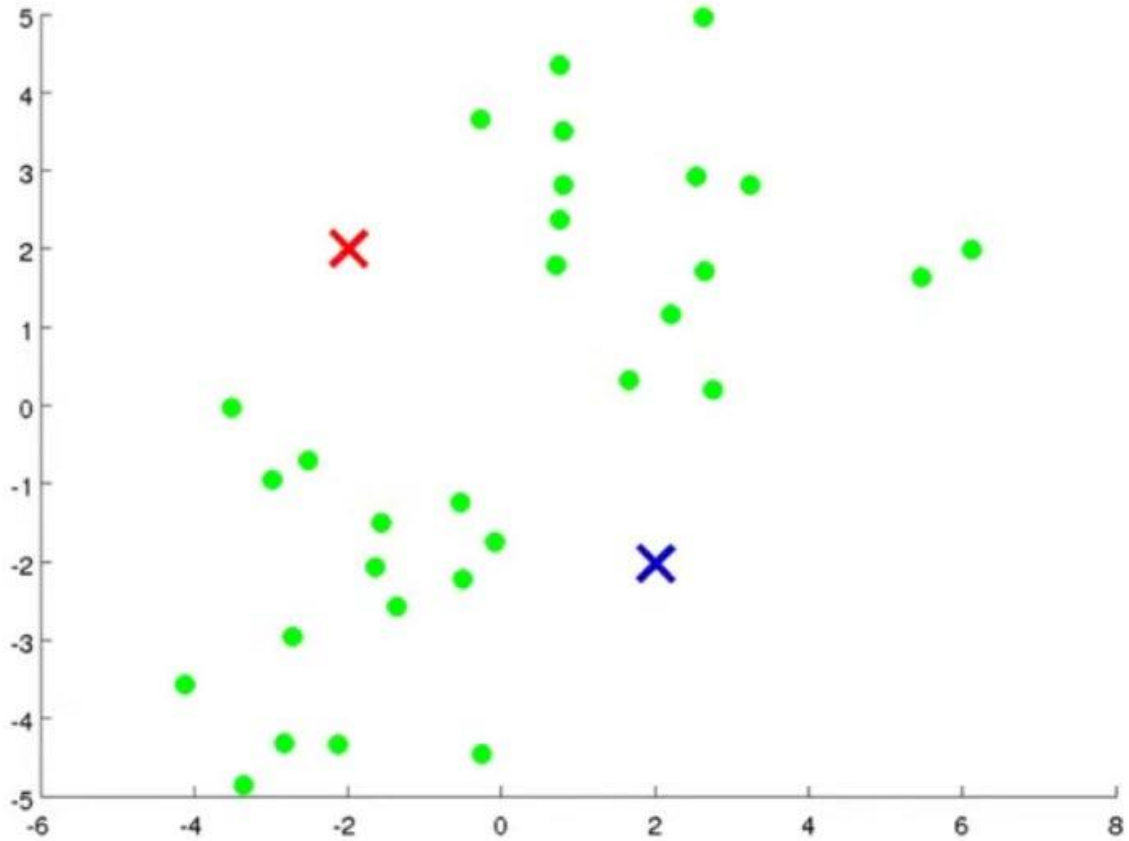


K Means Intuition

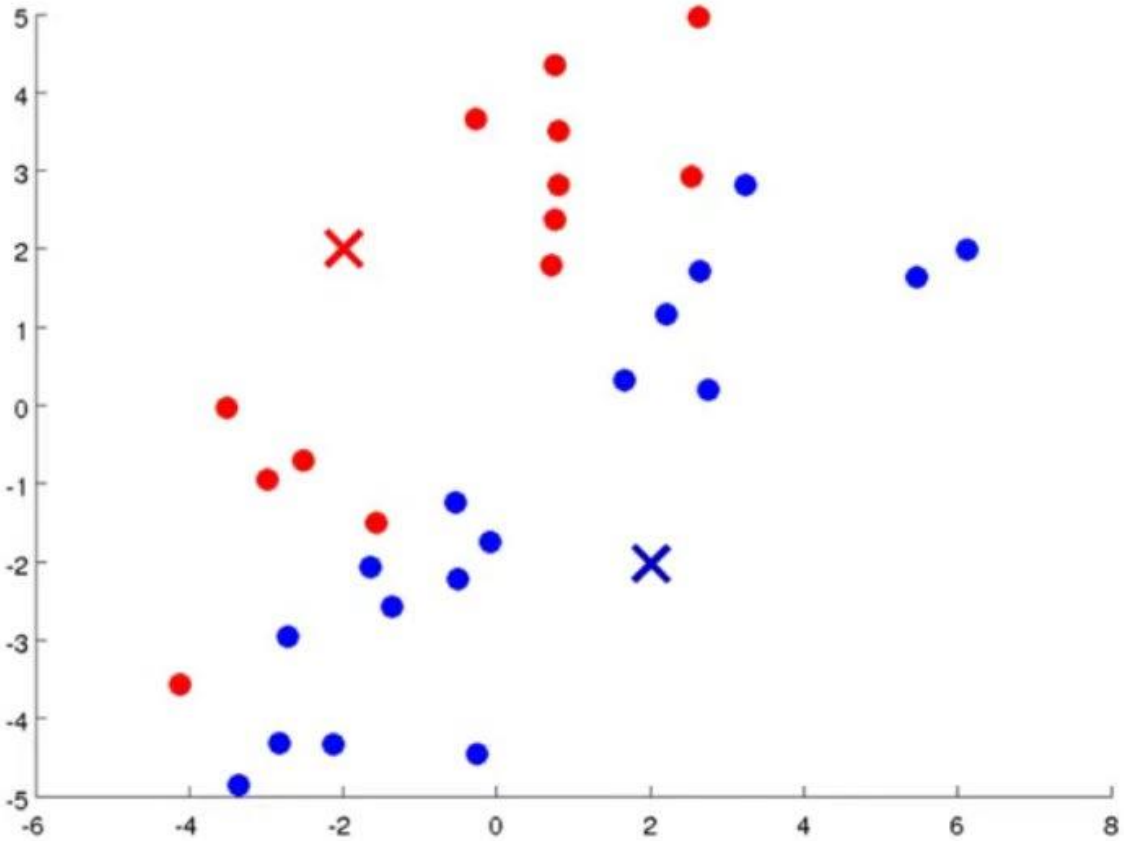
Basic Steps

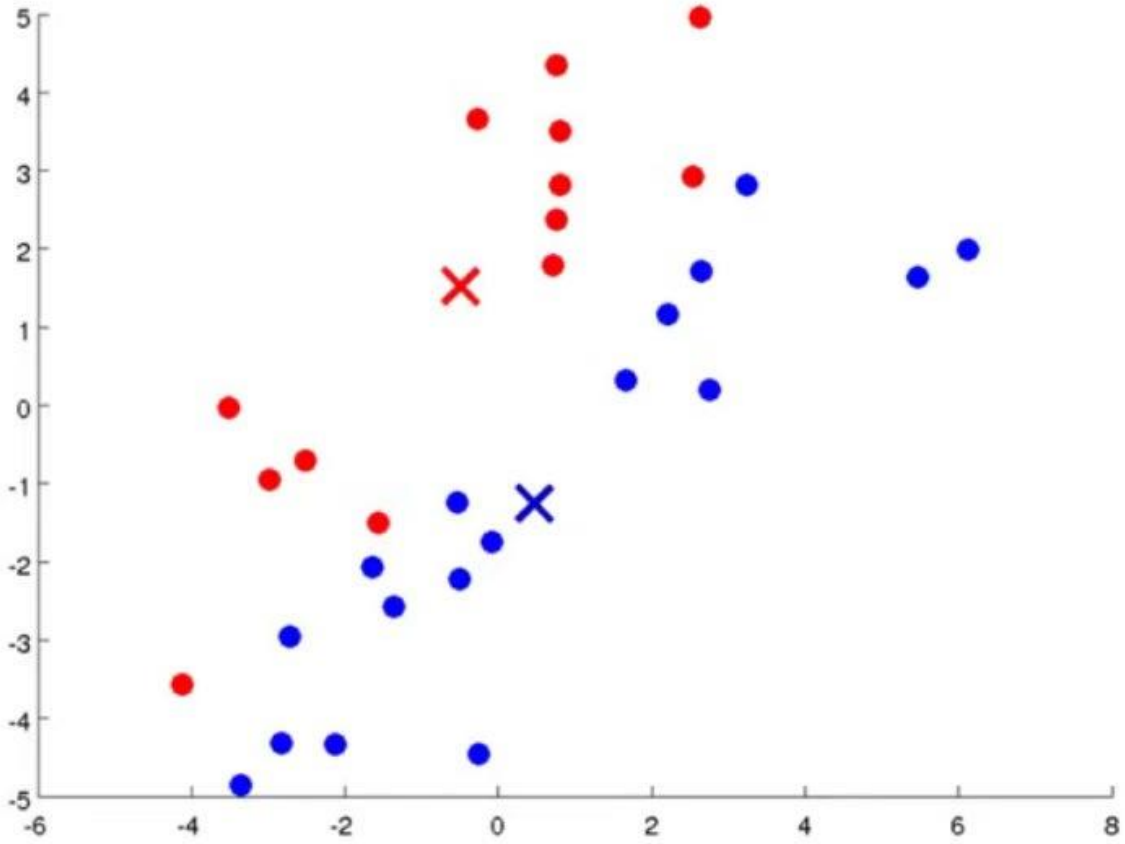
1) Specify number of clusters *K*.
2) Set cluster center rendomly
3)Keep iterating until there is no change to the centroids.
4)Compute the sum of the squared distance between data points and all centroids.
5)Assign each data point to the closest cluster (centroid).
6)Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

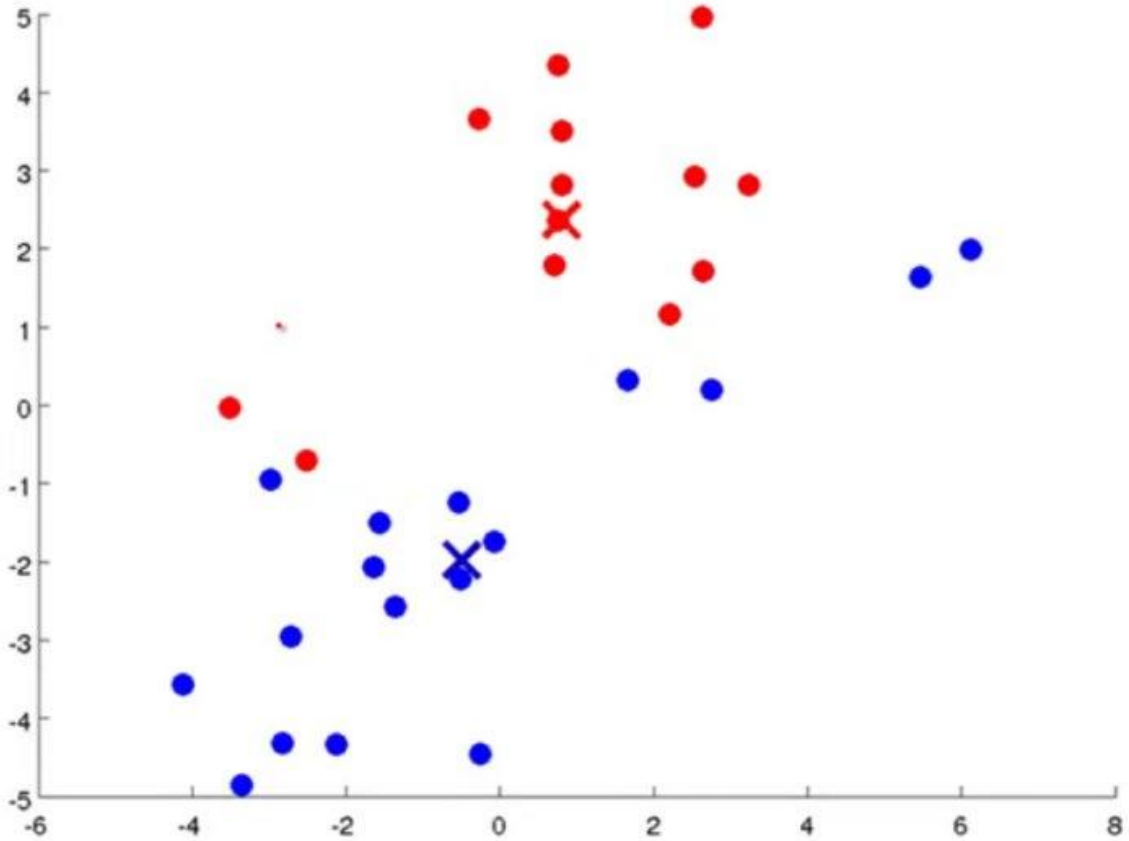

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8



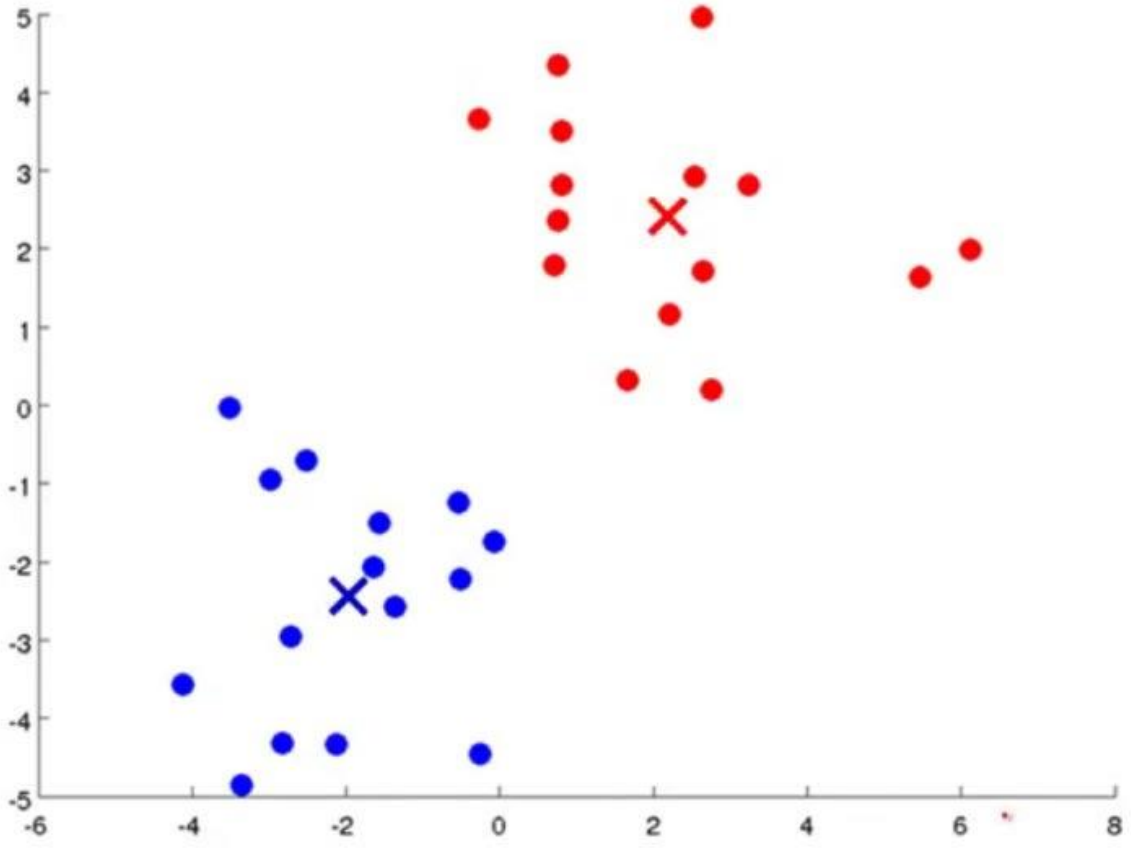
Slides from Andrew Ng | Machine Learning Coursera Lecture 13.2


5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8

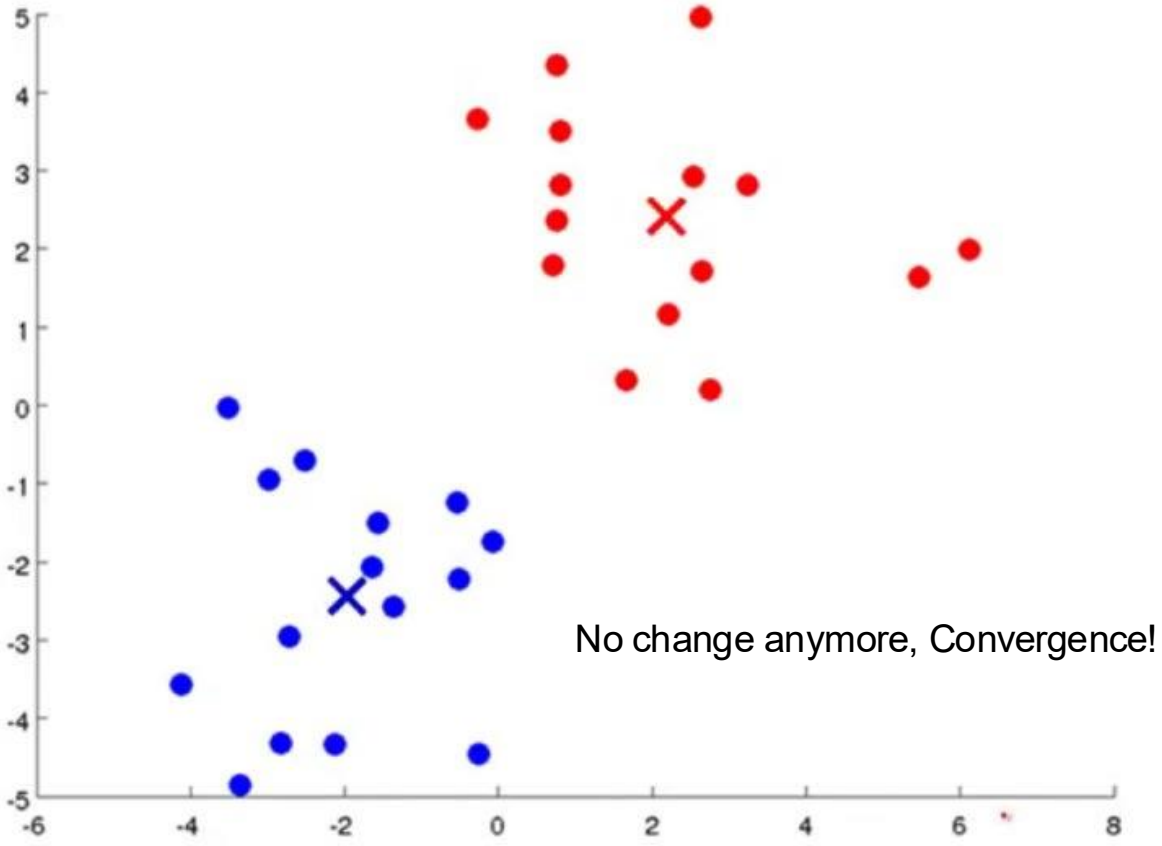

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8



Slides from Andrew Ng | Machine Learning Coursera Lecture 13.2



Slides from Andrew Ng | Machine Learning Coursera Lecture 13.2

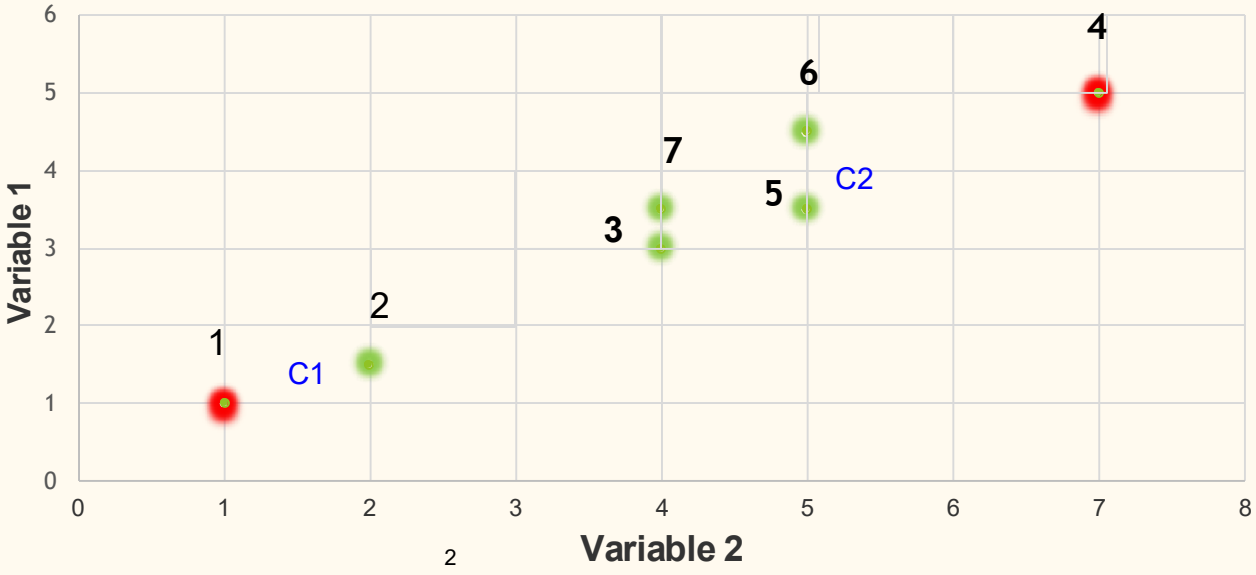

5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-4
-2
0
2
4
6
8
No change anymore, Convergence!

K Means

A Simple example k-means (using K=2)

Individual	Variable 1	Variable 2
1	1	1
2	1.5	2
3	3	4
4	5	7
5	3.5	5
6	4.5	5
7	3.5	4.5

K= 2



Step 1

Initialization: Randomly we choose following two centroids (k=2) for two clusters. In this case the 2 centroid are: m1=(1.0,1.0) and m2=(5.0,7.0).

	Individual	Mean Vector
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

Step 2

	Centroid 1	Centroid 2
1	$\sqrt{(1-1)^2 + (1-1)^2} = 0$	$\sqrt{(5-1)^2 + (7-1)^2} = 7.21$
2	$\sqrt{(1-1.5)^2 + (1-2)^2} = 1.12$	$\sqrt{(5-1.5)^2 + (7-2)^2} = 6.10$
3	$\sqrt{(1-3)^2 + (1-4)^2} = 3.61$	$\sqrt{(5-3)^2 + (7-4)^2} = 3.61$
4	$\sqrt{(1-5)^2 + (1-7)^2} = 7.21$	$\sqrt{(5-5)^2 + (7-7)^2} = 0$
5	$\sqrt{(1-3.5)^2 + (1-5)^2} = 4.72$	$\sqrt{(5-3.5)^2 + (7-5)^2} = 2.5$
6	$\sqrt{(1-4.5)^2 + (1-5)^2} = 5.31$	$\sqrt{(5-4.5)^2 + (7-5)^2} = 2.06$
7	$\sqrt{(1-3.5)^2 + (1-4.5)^2} = 4.30$	$\sqrt{(5-3.5)^2 + (7-4.5)^2} = 2.92$

- Thus, we obtain two clusters containing: {1,2,3} and {4,5,6,7}.
- Their new centroids are:

Group 1 $=\left(\frac{1+1.5+3}{3}\right)$, $\left(\frac{1+2+4}{3}\right) = (1.83 , 2.33)$

Group 2 $=\left(\frac{5+3.5+4.5+3.5}{4}\right)$, $\left(\frac{7+5+5+4.5}{4}\right) = (4.12 , 5.38)$

Step 3

	Centroid 1	Centroid 2
1	$\sqrt{(1.83-1)^2+(2.33-1)^2}=\mathbf{1.57}$	$\sqrt{(4.12-1)^2+(5.38-1)^2}=\mathbf{5.38}$
2	$\sqrt{(1.83-1.5)^2+(2.33-2)^2}=\mathbf{0.47}$	$\sqrt{(4.12-1.5)^2+(5.38-2)^2}=\mathbf{4.29}$
3	$\sqrt{(1.83-3)^2+(2.33-4)^2}=\mathbf{2.04}$	$\sqrt{(4.12-3)^2+(5.38-4)^2}=\mathbf{1.78}$
4	$\sqrt{(1.83-5)^2+(2.33-7)^2}=\mathbf{5.64}$	$\sqrt{(4.12-5)^2+(5.38-7)^2}=\mathbf{1.84}$
5	$\sqrt{(1.83-3.5)^2+(2.33-5)^2}=\mathbf{3.15}$	$\sqrt{(4.12-3.5)^2+(5.38-5)^2}=\mathbf{0.73}$
6	$\sqrt{(1.83-4.5)^2+(2.33-5)^2}=\mathbf{3.78}$	$\sqrt{(4.12-4.5)^2+(5.38-5)^2}=\mathbf{0.54}$
7	$\sqrt{(1.83-3.5)^2+(2.33-4.5)^2}=\mathbf{2.74}$	$\sqrt{(4.12-3.5)^2+(5.38-4.5)^2}=\mathbf{1.08}$

Therefore, the new clusters are:

{1,2} and {3,4,5,6,7}

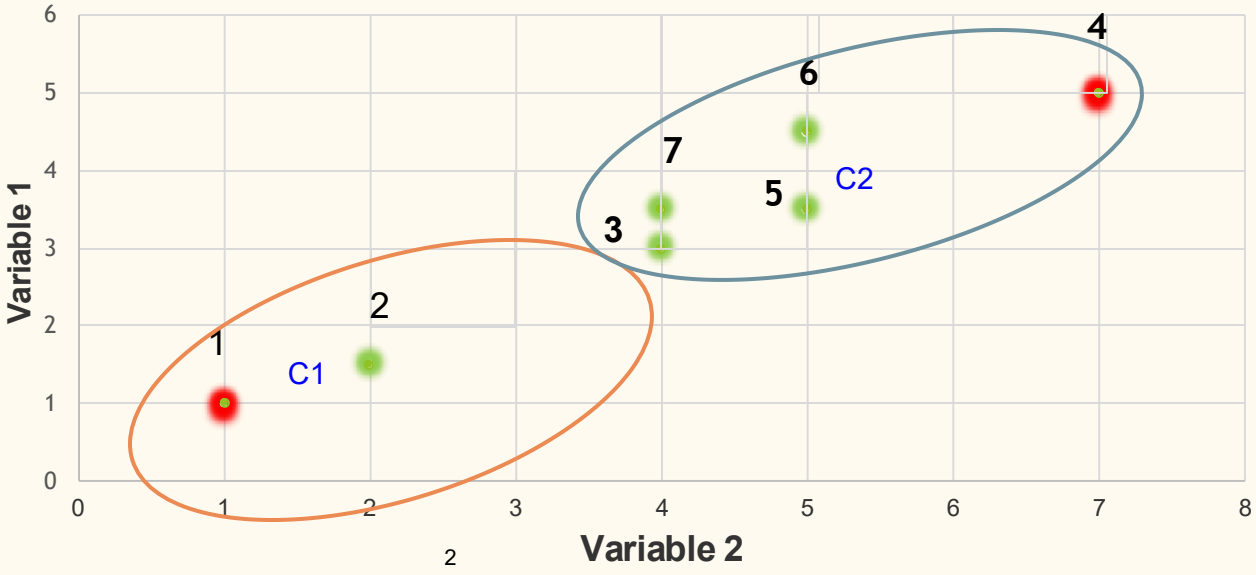
Group 1 $= \left(\frac{1+1.5}{2}\right), \left(\frac{1+2}{2}\right)$ **=(1.25 , 1.5)**

Group 2 $= \left(\frac{3+5+3.5+4.5+3.5}{5}\right), \left(\frac{4+7+5+5+4.5}{5}\right)$ **=(3.9 , 5.1)**

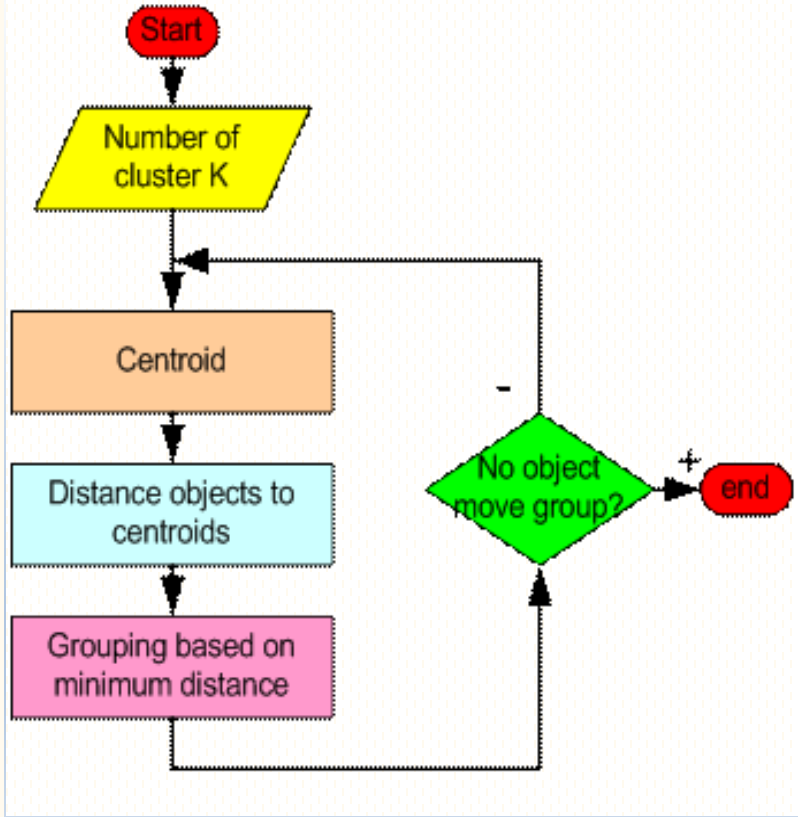
Step 4

	Centroid 1	Centroid 2
1	$\sqrt{(1.25-1)^2+(1.5-1)^2} = \mathbf{0.58}$	$\sqrt{(3.9-1)^2+(5.1-1)^2} = \mathbf{5.02}$
2	$\sqrt{(1.25-1.5)^2+(1.5-2)^2} = \mathbf{0.56}$	$\sqrt{(3.9-1.5)^2+(5.1-2)^2} = \mathbf{3.92}$
3	$\sqrt{(1.25-3)^2+(1.5-4)^2} = \mathbf{3.05}$	$\sqrt{(3.9-3)^2+(5.1-4)^2} = \mathbf{1.42}$
4	$\sqrt{(1.25-5)^2+(1.5-7)^2} = \mathbf{6.66}$	$\sqrt{(3.9-5)^2+(5.1-7)^2} = \mathbf{2.20}$
5	$\sqrt{(1.25-3.5)^2+(1.5-5)^2} = \mathbf{4.16}$	$\sqrt{(3.9-3.5)^2+(5.1-5)^2} = \mathbf{0.41}$
6	$\sqrt{(1.25-4.5)^2+(1.5-5)^2} = \mathbf{4.78}$	$\sqrt{(3.9-4.5)^2+(5.1-5)^2} = \mathbf{0.61}$
7	$\sqrt{(1.25-3.5)^2+(1.5-4.5)^2} = \mathbf{3.75}$	$\sqrt{(3.9-3.5)^2+(5.1-4.5)^2} = \mathbf{0.72}$

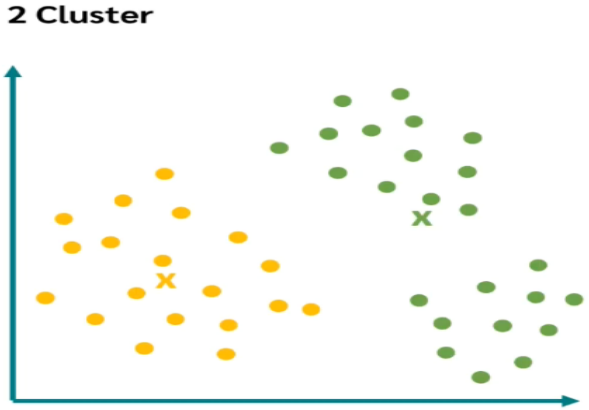
- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.

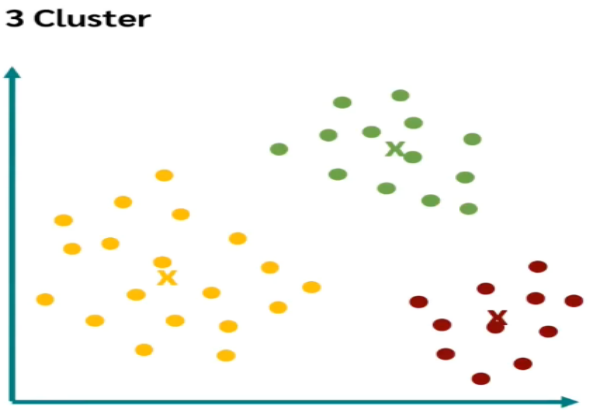

Variable 1
Variable 2
0
1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
C1
C2

How the K-Mean Clustering algorithm works?



Optimal cluster number

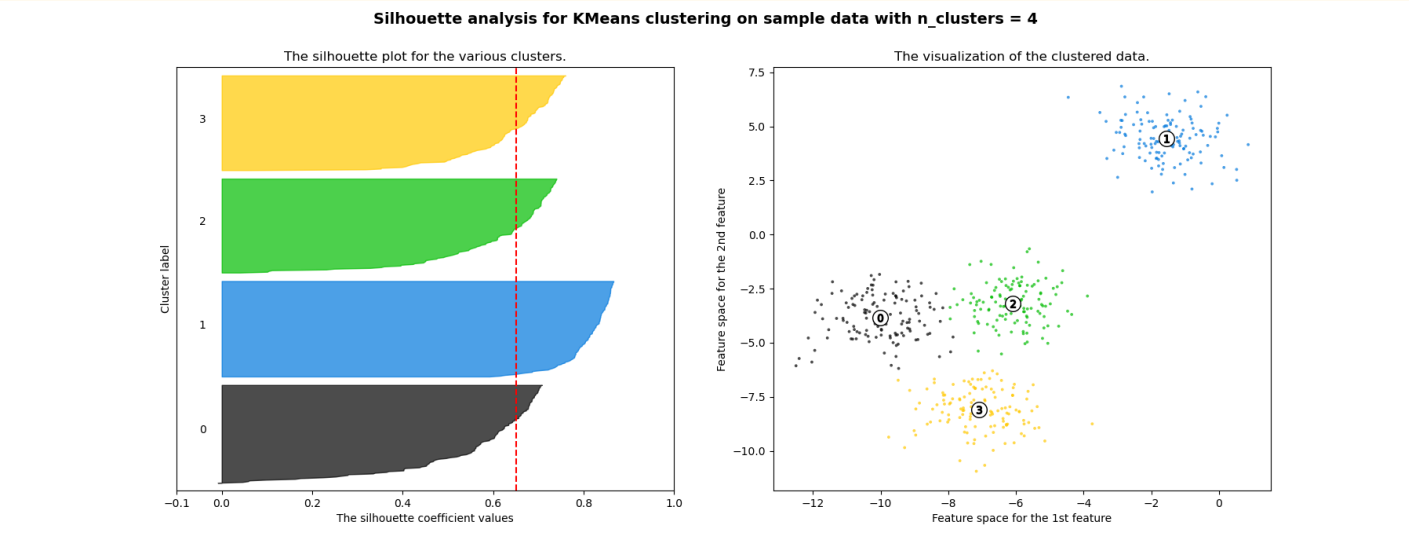




- With each new cluster the summed distance in the clusters becomes smaller and smaller.
- If there are as many clusters as points, zero comes out
- How many clusters should be used?

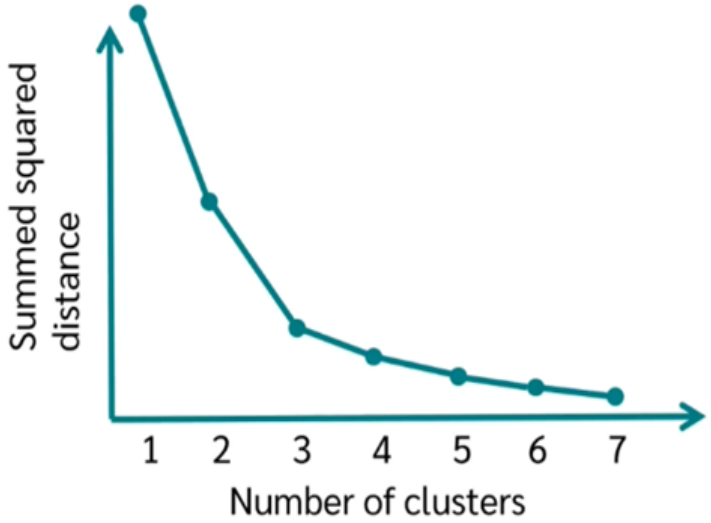
How to identify the best “K” ?

- Elbow method: It is to calculate the sum the distances from data points to centroids and aims at minimising the sum to an optimal value.
- Silhouette analysis: it measures how similar a point is to its own cluster compared to other clusters.



Elbow Method

Elbow method gives us an idea on what a good *k* number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters' centroids.



- With each additional cluster the summed distance between the points and the cluster center becomes smaller and smaller
- However, there is a cluster number from which each additional cluster reduces the summed distance only slightly.
- This point is used as number for the clusters.

Application Case:

Image Segmentation and Compression

What is Segmentation?

Segmentation is to partition an image into regions each of which has a reasonably homogeneous visual appearance or which corresponds to objects or parts of objects

Original image

$K = 10$

Original image

$K = 3$

Original image

$K = 2$

Pros and Cons

Pros and cons

Advantages of k-means

1. Relatively simple to implement.
2. Scales to large data sets.
3. Easily adapts to new examples.

Disadvantages of k-means

1. Choosing \(k\) manually.
2. Being dependent on initial values.
3. Scaling with number of dimensions.

How to Use?

sklearn.cluster.KMeans

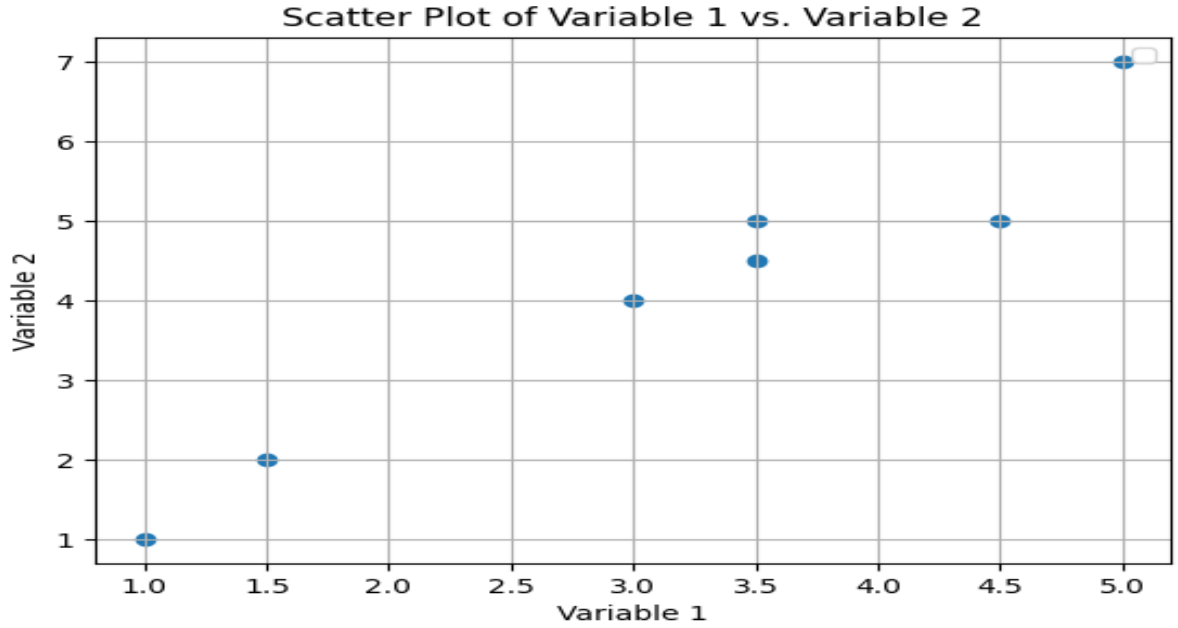
```
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
%matplotlib inline
```

```
df = np.array([
    [1, 1],
    [1.5, 2],
    [3, 4],
    [5, 7],
    [3.5, 5],
    [4.5, 5],
    [3.5, 4.5]
])
```

```
df_table = pd.DataFrame(df, columns=["Variable 1", "Variable 2"])
```

```
plt.scatter(df_table["Variable 1"], df_table["Variable 2"])
plt.xlabel("Variable 1")
plt.ylabel("Variable 2")
plt.title("Scatter Plot of Variable 1 vs. Variable 2")
plt.legend()
plt.grid(True)
plt.show()
```

```
<ipython-input-21-e628bb50a8c5>:5: UserWarning: No artists with labels found to put in legend.
  plt.legend()
```



```
km = KMeans(n_clusters=2)
y_predicted = km.fit_predict(df_table)
y_predicted
```

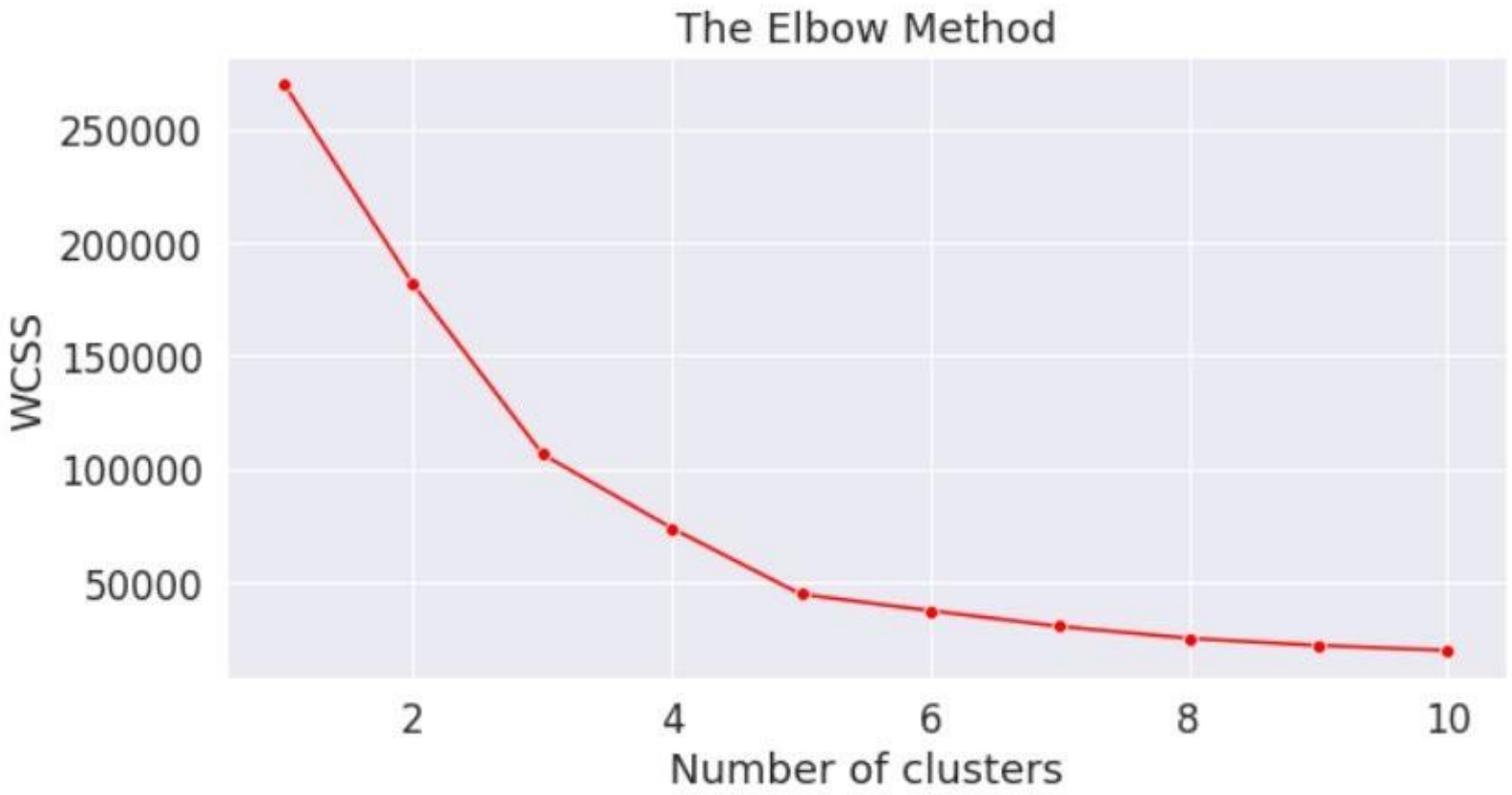
```
array([1, 1, 0, 0, 0, 0, 0], dtype=int32)
```

```
df_table['cluster']=y_predicted
df_table.head()
```

	Variable 1	Variable 2	cluster
0	1.0	1.0	1
1	1.5	2.0	1
2	3.0	4.0	0
3	5.0	7.0	0
4	3.5	5.0	0

```python
# Using the elbow method to find the optimal number of clusters
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(X)
    # inertia method returns wcss for that model
    wcss.append(kmeans.inertia_)
```

```python
plt.figure(figsize=(10,5))
sns.lineplot(range(1, 11), wcss,marker='o',color='red')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

Step by Step KMeans Explained in Detail

```
# Fitting K-Means to the dataset
kmeans = KMeans(n_clusters = 5, init = 'k-means++', random_state = 42)
y_kmeans = kmeans.fit_predict(X)
```

Step by Step KMeans Explained in Detail

```
# Visualising the clusters
plt.figure(figsize=(15,7))
sns.scatterplot(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], color = 'yellow', label = 'Cluster 1',
s=50)
sns.scatterplot(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], color = 'blue', label = 'Cluster 2',s=
50)
sns.scatterplot(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], color = 'green', label = 'Cluster 3',s
=50)
sns.scatterplot(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], color = 'grey', label = 'Cluster 4',s=
50)
sns.scatterplot(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], color = 'orange', label = 'Cluster 5',
s=50)
sns.scatterplot(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], color = 'red',
                label = 'Centroids',s=300,marker=',')
plt.grid(False)
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```


Step by Step KMeans Explained in Detail

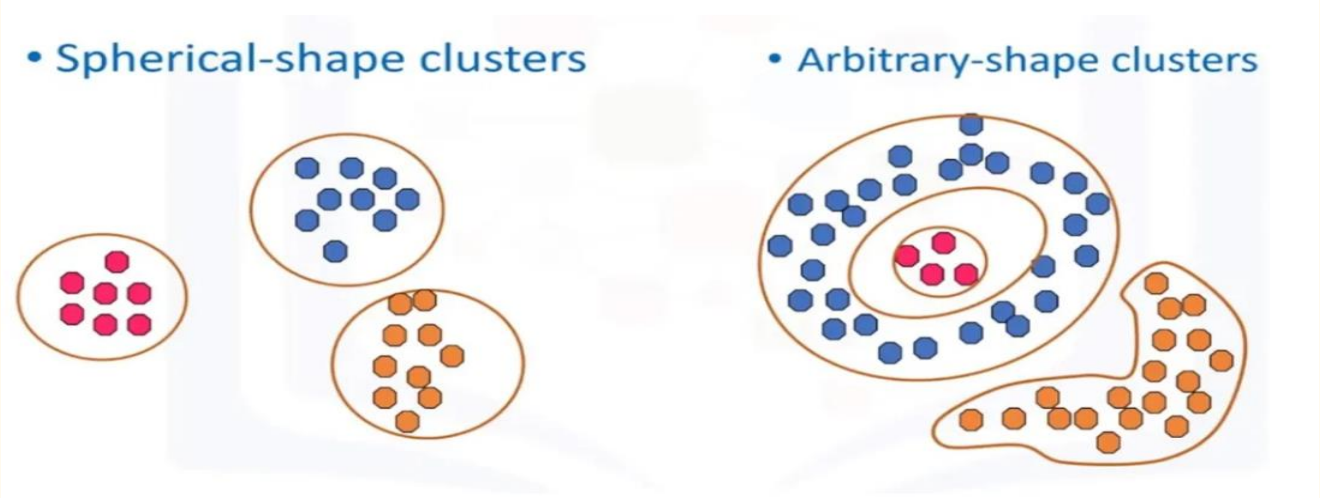
2. Density-Based Clustering Method

- These regard clusters as dense regions of objects in the data space that are separated by regions of low density (representing noise).
- Density-based clustering methods have been developed To discover clusters with arbitrary shape.

Algorithms:
DBSCAN

Density-based clustering

- These algorithms have difficulty with data of varying densities and high dimensions. Further, by design, these algorithms do not assign outliers to clusters.



Density-Based Spatial Clustering of Applications with Noise

- (DBSCAN) is a base algorithm for density-based clustering. It can discover clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers.

- The DBSCAN algorithm uses two parameters:
- **minPts:** The minimum number of points (a threshold) clustered together for a region to be considered dense.
- **eps (ε):** A distance measure that will be used to locate the points in the neighborhood of any point.

DBSCAN

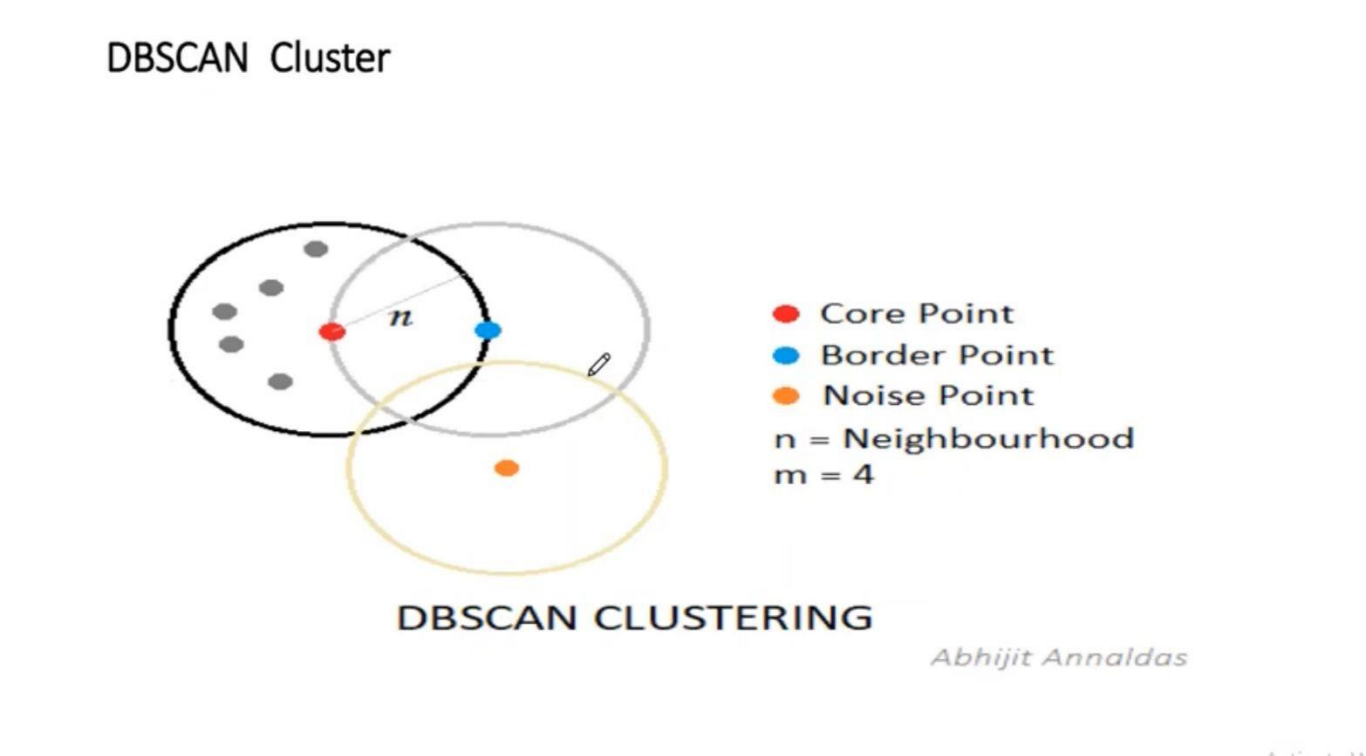
DBSCAN revolves around three key concepts:

1.Core Points: These are points that have at least a minimum number of other points (MinPts) within a specified distance (ε or epsilon).

2.Border Points: These are points that are within the ε distance of a core point but don't have MinPts neighbors themselves.

3.Noise Points: These are points that are neither core points nor border points. They're not close enough to any cluster to be included.

DBSCAN


DBSCAN Cluster
n
Core Point
Border Point
Noise Point
n = Neighbourhood
m = 4
DBSCAN CLUSTERING
Abhijit Annaldas

DBSCAN Example

Q. Given the points A(3, 7), B(4, 6), C(5, 5), D(6, 4), E(7, 3), F(6, 2), G(7, 2) and H(8, 4), Find the core points and outliers using DBSCAN. Take Eps = 2.5 and MinPts = 3.

Example

Solution:
Given, Epsilon(*Eps*) = 2.5
Minimum Points(*MinPts*) = 3
Let's represent the given data points in tabular form:

Data Points	X	Y
A	3	7
B	4	6
C	5	5
D	6	4
E	7	3
F	6	2
G	7	2
H	8	4

Data points

Solution

Calculating distance between data point A and other data points as:

$d(A, B) = \sqrt{(4-3)^2 + (6-7)^2} = \sqrt{(1)^2 + (-1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$

$d(A, C) = \sqrt{(5-3)^2 + (5-7)^2} = \sqrt{(2)^2 + (-2)^2} = \sqrt{4+4} = \sqrt{8} = 2.8284$

$d(A, D) = \sqrt{(6-3)^2 + (4-7)^2} = \sqrt{(3)^2 + (-3)^2} = \sqrt{9+9} = \sqrt{18} = 4.2426$

$d(A, E) = \sqrt{(7-3)^2 + (3-7)^2} = \sqrt{(4)^2 + (-4)^2} = \sqrt{16+16} = \sqrt{32} = 5.6569$

$d(A, F) = \sqrt{(6-3)^2 + (2-7)^2} = \sqrt{(3)^2 + (-5)^2} = \sqrt{9+25} = \sqrt{34} = 5.8310$

$d(A, G) = \sqrt{(7-3)^2 + (2-7)^2} = \sqrt{(4)^2 + (-5)^2} = \sqrt{16+25} = \sqrt{41} = 6.4031$

$d(A, H) = \sqrt{(8-3)^2 + (4-7)^2} = \sqrt{(5)^2 + (-3)^2} = \sqrt{25+9} = \sqrt{34} = 5.8310$

Calculating distance between data point B and other data points as:

$d(B, C) = \sqrt{(5-4)^2 + (5-6)^2} = \sqrt{(1)^2 + (-1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$

$d(B, D) = \sqrt{(6-4)^2 + (4-6)^2} = \sqrt{(2)^2 + (-2)^2} = \sqrt{4+4} = \sqrt{8} = 2.8284$

$d(B, E) = \sqrt{(7-4)^2 + (3-6)^2} = \sqrt{(3)^2 + (-3)^2} = \sqrt{9+9} = \sqrt{18} = 4.2426$

$d(B, F) = \sqrt{(6-4)^2 + (2-6)^2} = \sqrt{(2)^2 + (-4)^2} = \sqrt{4+16} = \sqrt{20} = 4.4721$

$d(B, G) = \sqrt{(7-4)^2 + (2-6)^2} = \sqrt{(3)^2 + (-4)^2} = \sqrt{9+16} = \sqrt{25} = 5$

$d(B, H) = \sqrt{(8-4)^2 + (4-6)^2} = \sqrt{(4)^2 + (-2)^2} = \sqrt{16+4} = \sqrt{20} = 4.4721$

Solution

Calculating distance between data point C and other data points as:

$d(C, D) = \sqrt{(6-5)^2 + (4-5)^2} = \sqrt{(1)^2 + (-1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$

$d(C, E) = \sqrt{(7-5)^2 + (3-5)^2} = \sqrt{(2)^2 + (-2)^2} = \sqrt{4+4} = \sqrt{8} = 2.8284$

$d(C, F) = \sqrt{(6-5)^2 + (2-5)^2} = \sqrt{(1)^2 + (-3)^2} = \sqrt{1+9} = \sqrt{10} = 3.1623$

$d(C, G) = \sqrt{(7-5)^2 + (2-5)^2} = \sqrt{(2)^2 + (-3)^2} = \sqrt{4+9} = \sqrt{13} = 3.6056$

$d(C, H) = \sqrt{(8-5)^2 + (4-5)^2} = \sqrt{(3)^2 + (-1)^2} = \sqrt{9+1} = \sqrt{10} = 3.1623$

Calculating distance between data point D and other data points as:

$d(D, E) = \sqrt{(7-6)^2 + (3-4)^2} = \sqrt{(1)^2 + (-1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$

$d(D, F) = \sqrt{(6-6)^2 + (2-4)^2} = \sqrt{(0)^2 + (-2)^2} = \sqrt{0+4} = \sqrt{4} = 2$

$d(D, G) = \sqrt{(7-6)^2 + (2-4)^2} = \sqrt{(1)^2 + (-2)^2} = \sqrt{1+4} = \sqrt{5} = 2.2361$

$d(D, H) = \sqrt{(8-6)^2 + (4-4)^2} = \sqrt{(2)^2 + (0)^2} = \sqrt{4+0} = \sqrt{4} = 2$

Solution

Calculating distance between data point E and other data points as:

$$d(E, F) = \sqrt{(6-7)^2 + (2-3)^2} = \sqrt{(-1)^2 + (-1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$$

$$d(E, G) = \sqrt{(7-7)^2 + (2-3)^2} = \sqrt{(0)^2 + (-1)^2} = \sqrt{0+1} = \sqrt{1} = 1$$

$$d(E, H) = \sqrt{(8-7)^2 + (4-3)^2} = \sqrt{(1)^2 + (1)^2} = \sqrt{1+1} = \sqrt{2} = 1.4142$$

Calculating distance between data point F and other data points as:

$$d(F, G) = \sqrt{(7-6)^2 + (2-2)^2} = \sqrt{(1)^2 + (0)^2} = \sqrt{1+0} = \sqrt{1} = 1$$

$$d(F, H) = \sqrt{(8-6)^2 + (4-2)^2} = \sqrt{(2)^2 + (2)^2} = \sqrt{4+4} = \sqrt{8} = 2.8284$$

Calculating distance between data point G and other data points as:

$$d(G, H) = \sqrt{(8-7)^2 + (4-2)^2} = \sqrt{(1)^2 + (2)^2} = \sqrt{1+4} = \sqrt{5} = 2.2361$$

The final distance matrix becomes

Data Points	A	B	C	D	E	F	G	H
A	0	1.4142	2.8284	4.2426	5.6569	5.831	6.4031	5.831
B		0	1.4142	2.8284	4.2426	4.4721	5	4.4721
C			0	1.4142	2.8284	3.1623	3.6056	3.1623
D				0	1.4142	2	2.2361	2
E					0	1.4142	1	1.4142
F						0	1	2.8284
G							0	2.2361
H								0

Solution

N(A) = {B};
N(B) = {A, C};
N(C) = {B, D};
N(D) = {C, E, F, G, H};
N(E) = {D, F, G, H};
N(F) = {D, E, G};
N(G) = {D, E, F, H};
N(H) = {D, E, G};

Result

data points A, B and C have neighbors <= **MinPts (i.e. 3)** so can't be considered as core points. Since they belong to the neighborhood of other data points, hence there exist *no outliers* in the given set of data points.
Data points D, E, F, G and H have neighbors >= **MinPts (i.e. 3)** and hence are the *core data points*.

Pros and cons

Pros of DBSCAN:

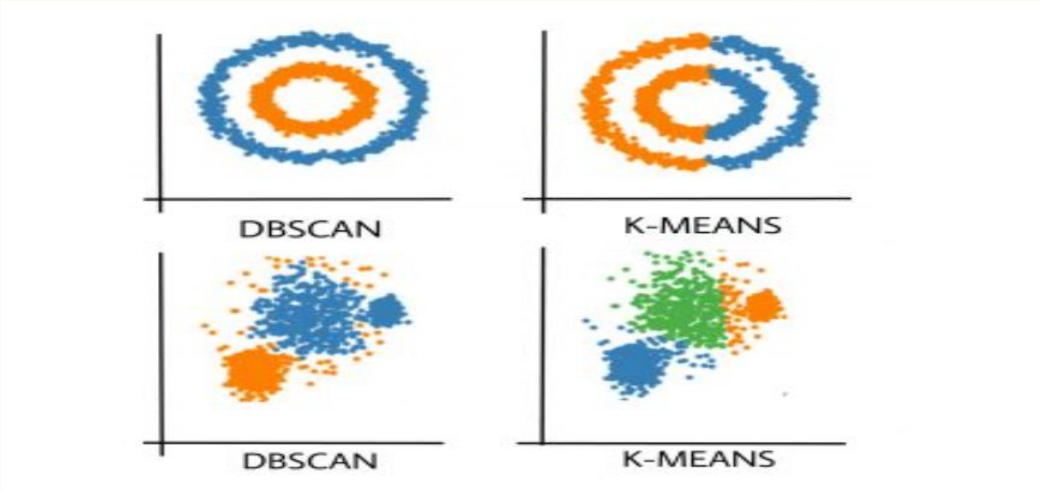
-DBSCAN can discover clusters of arbitrary shape, unlike k-means.
-It is robust to noise, as it can identify points that do not belong to any cluster as outliers.
-It does not require the number of clusters to be specified in advance.

Cons OF DBSCAN:

-It is sensitive to the choice of the Eps and MinPts parameters.
-It does not work well with clusters of varying densities.
-It has a high computational cost when the number of data points is large.
-It is not guaranteed to find all clusters in the data.

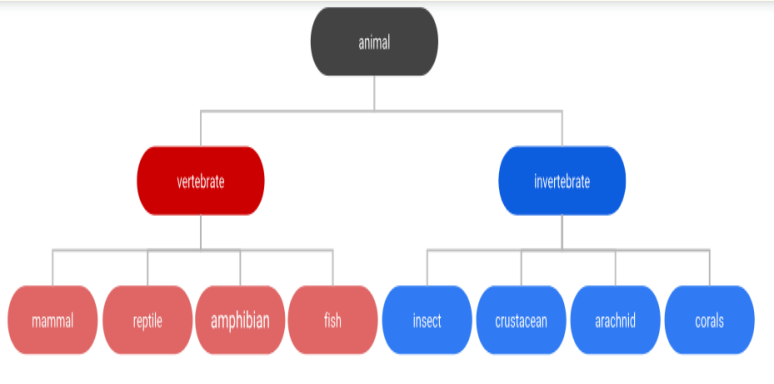
K-means vs DBSCAN

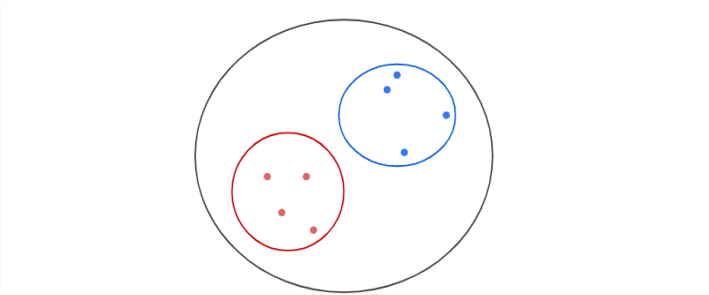
K-means assigns all points to a cluster even if they do not belong in any .
Density-based clustering locates regions of high density and separates outliers .



3-Hierarchical clustering

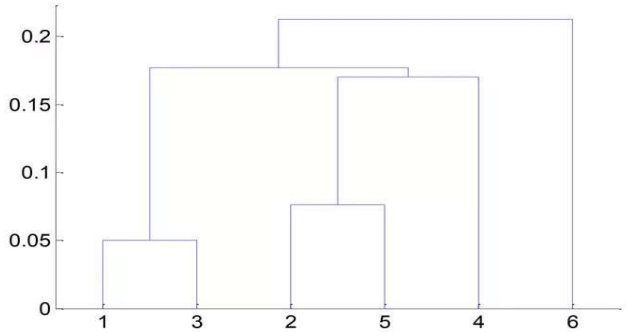
- **Hierarchical clustering** creates a tree of clusters.
- Hierarchical clustering, not surprisingly, is well suited to hierarchical data, such as taxonomies.
- In addition, another advantage is that any number of clusters can be chosen by cutting the tree at the right level.

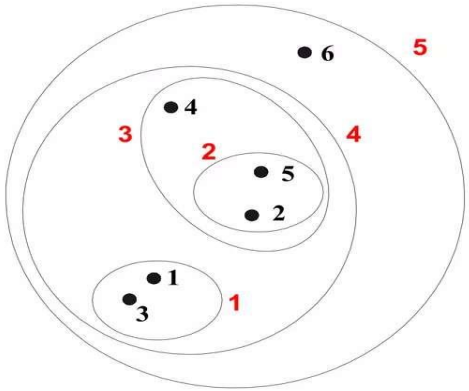




Hierarchical Clustering

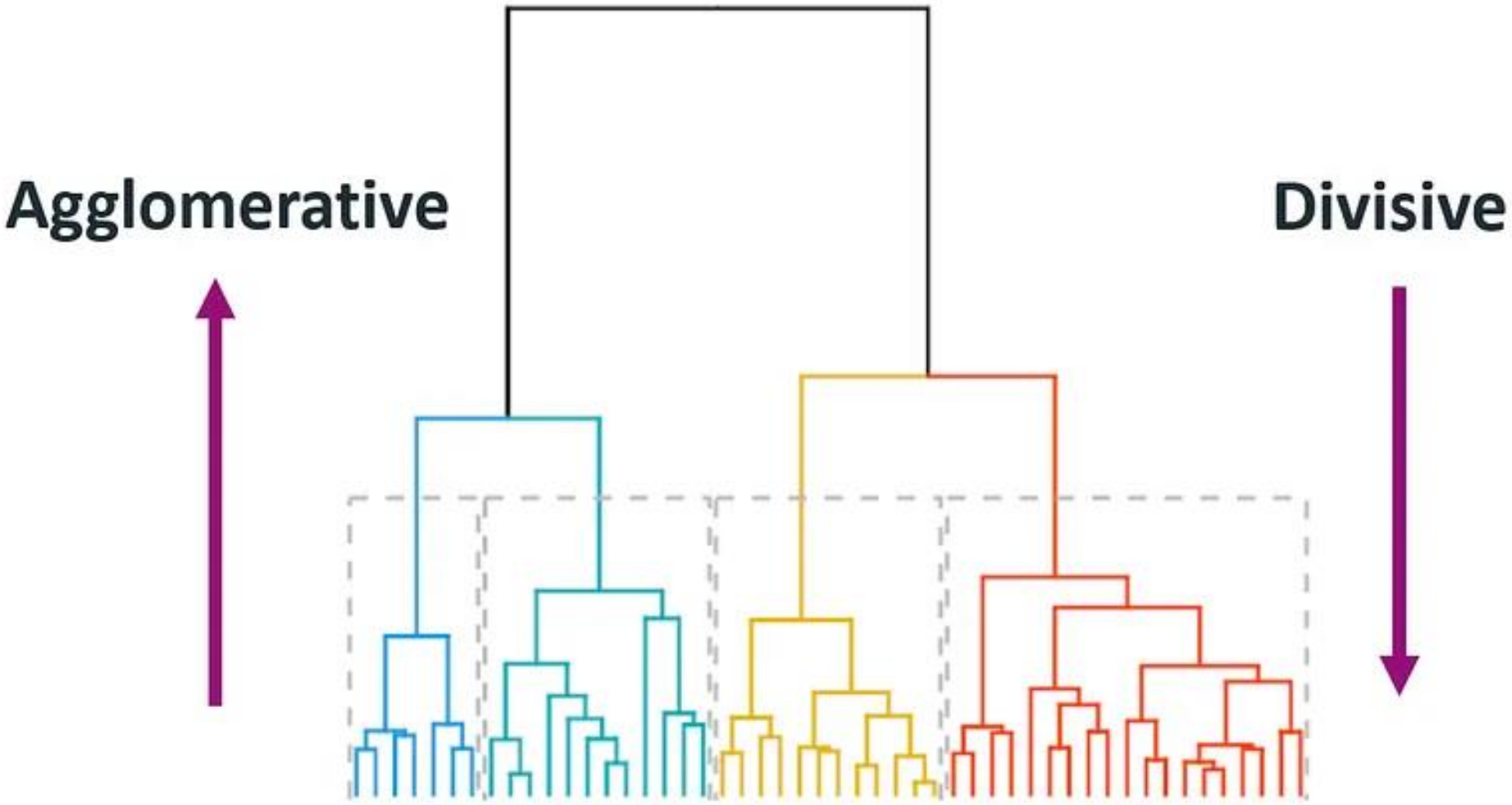
- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits





Hierarchical Clustering

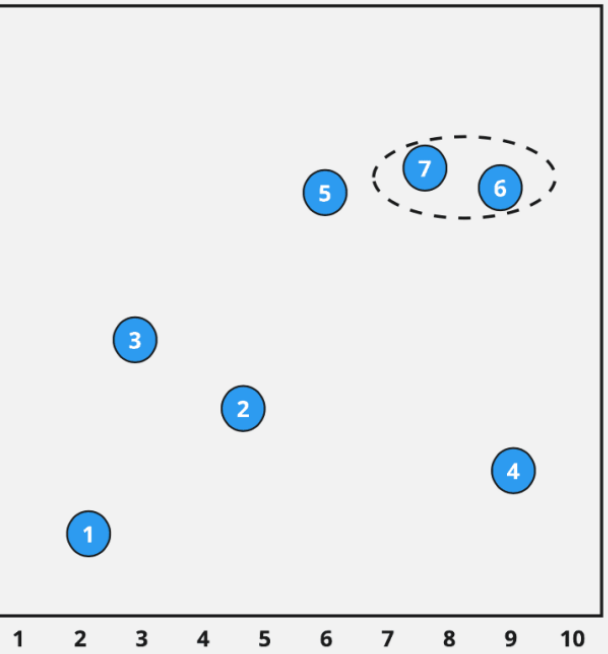
- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains an individual point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time


Agglomerative
Divisive

Agglomerative Clustering Algorithm

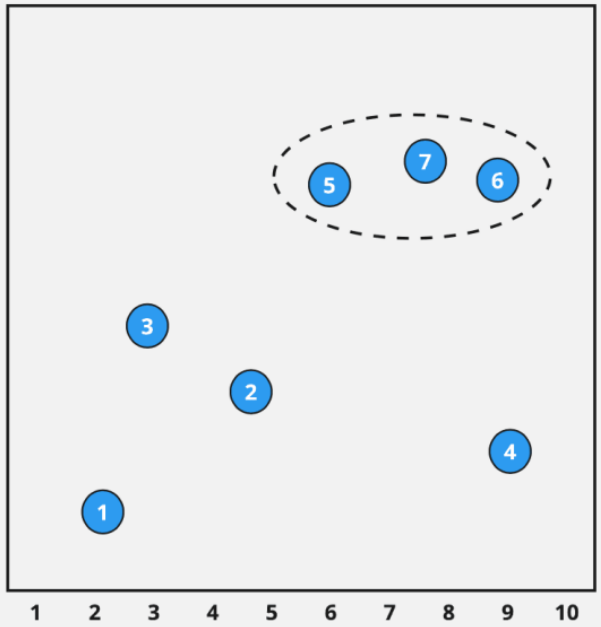
- **Key Idea: Successively merge closest clusters**
- Basic algorithm
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

Step 02



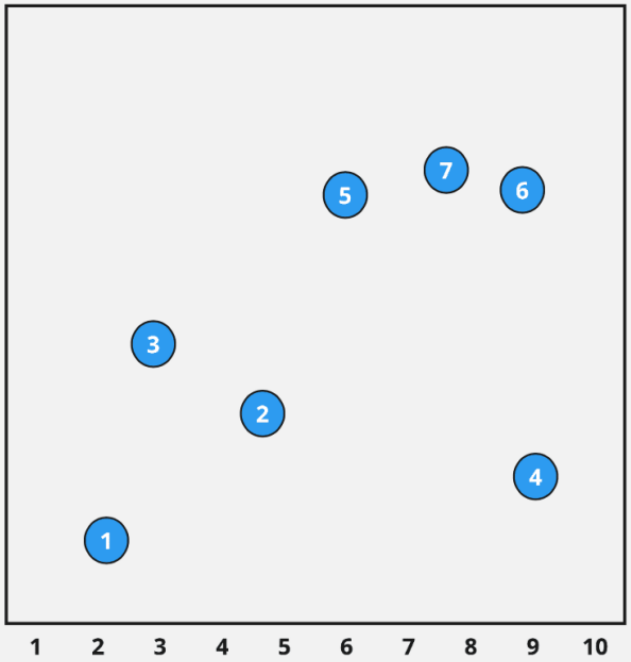
dataaspirant.com

Step 03



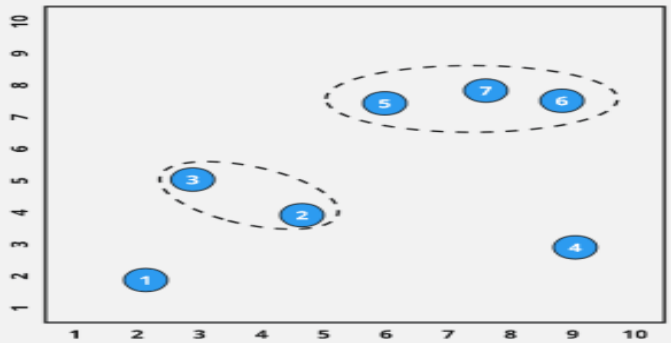
dataaspirant.com

Step 01

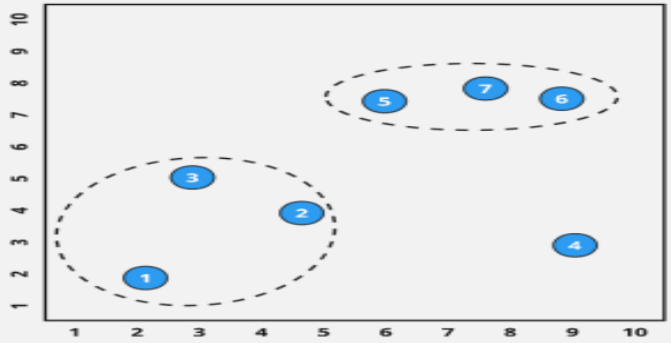


dataaspirant.com

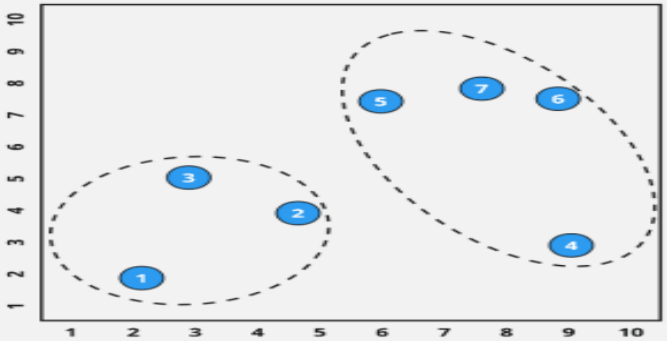
Step 04



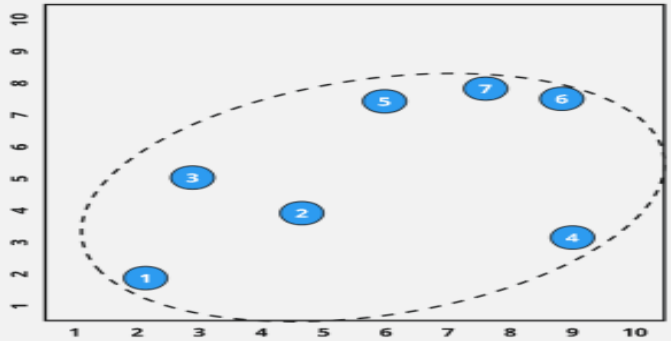
Step 05



Step 06

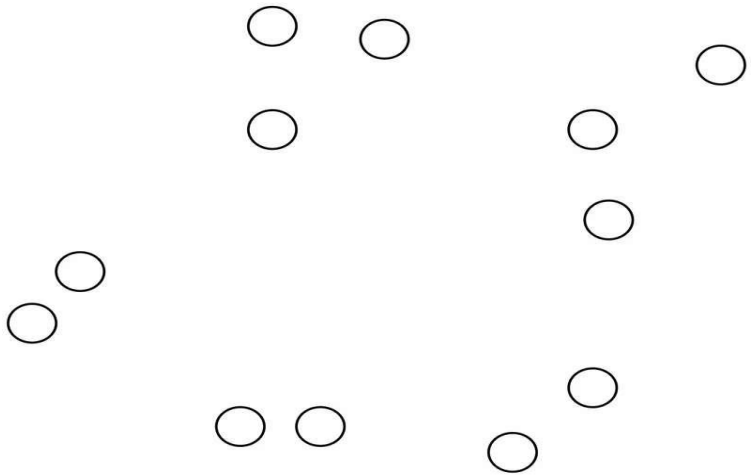


Step 07



Steps 1 and 2

- Start with clusters of individual points and a proximity matrix

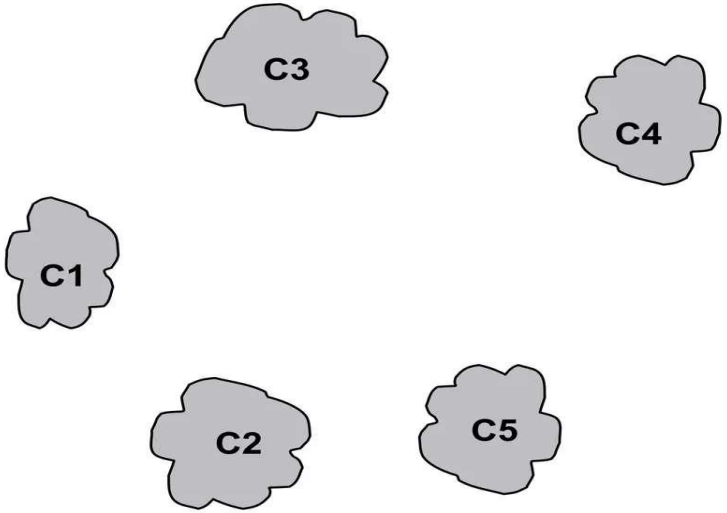


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

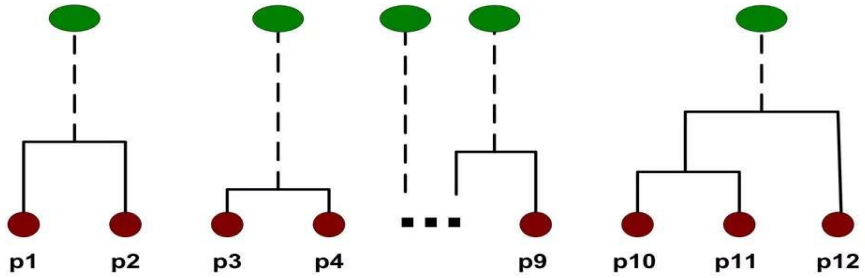
Intermediate Situation

- After some merging steps, we have some clusters



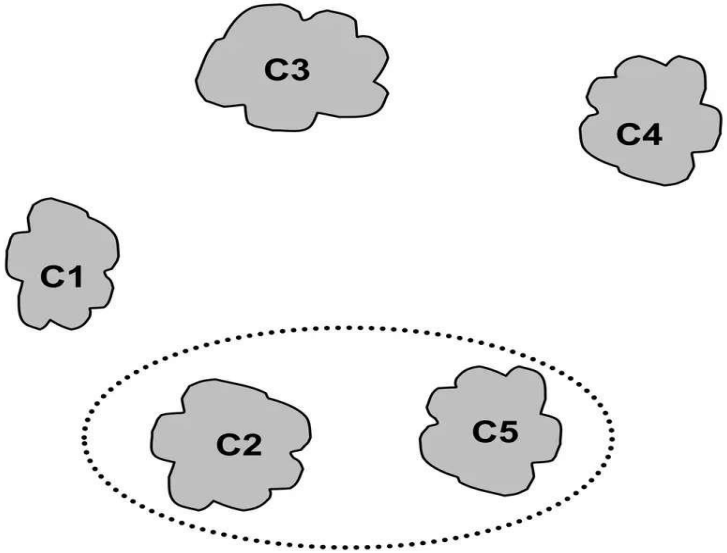
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



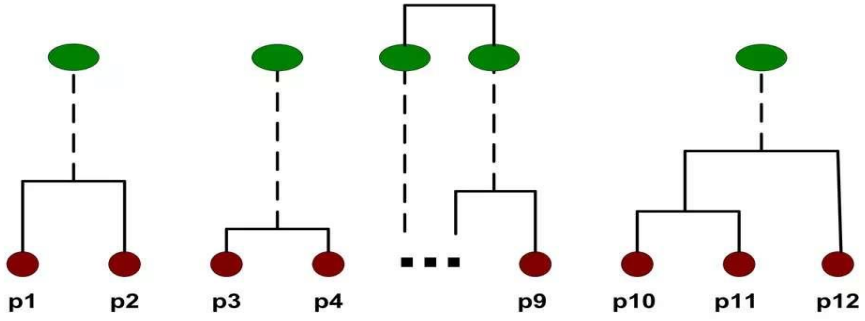
Step 4

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.



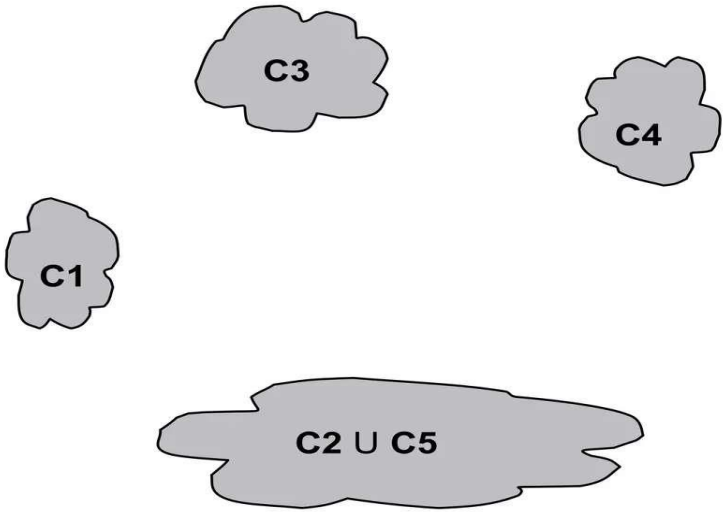
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



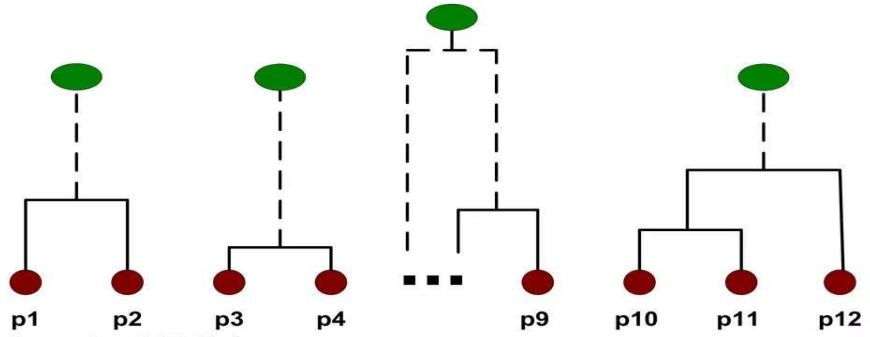
Step 5

- The question is “How do we update the proximity matrix?”

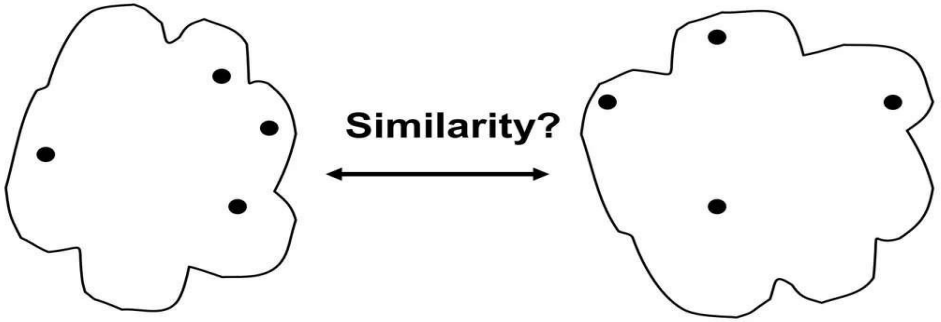


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

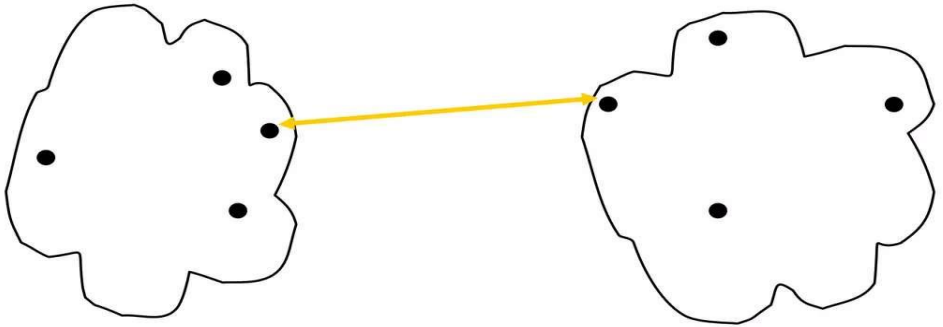


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

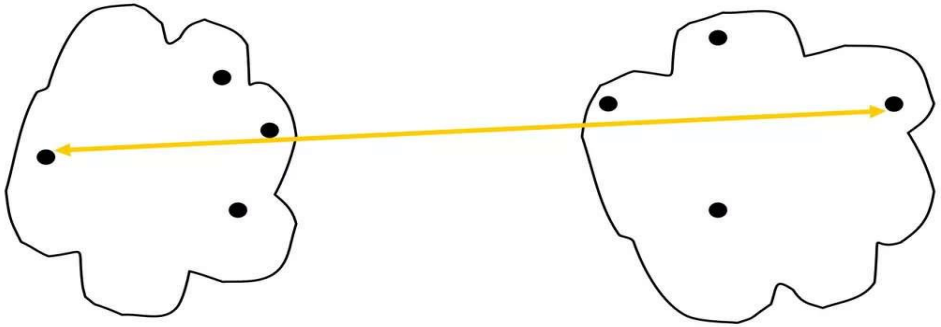


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity

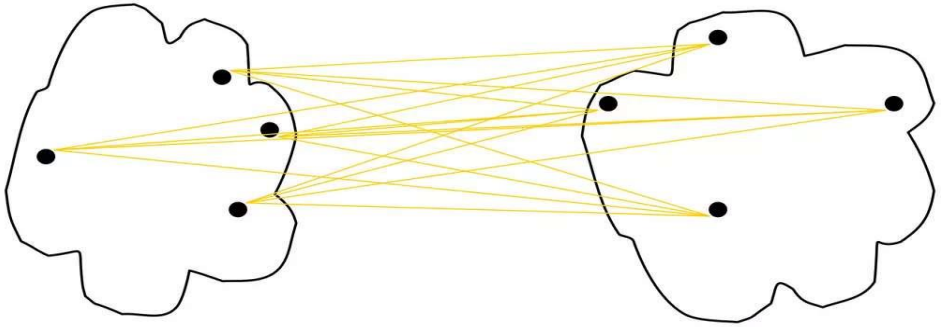


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity

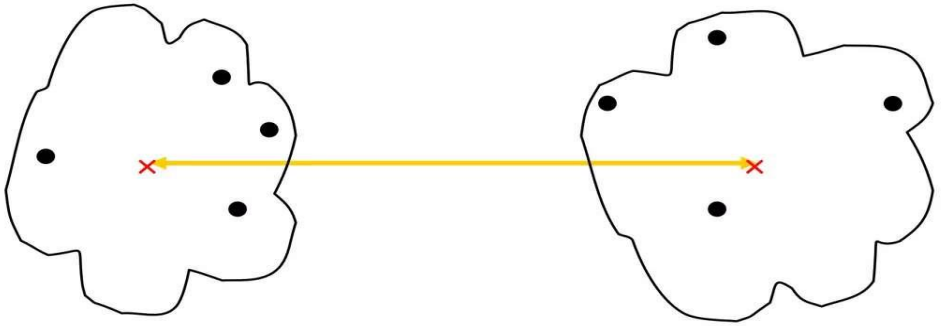


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity



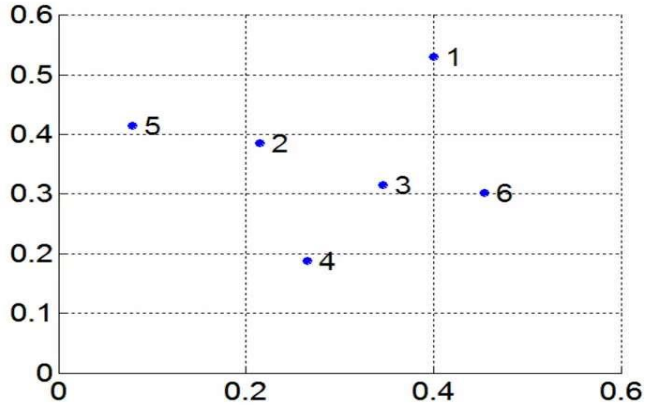
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

MIN or Single Link

- Proximity of two clusters is based on the two closest points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph
- Example:



Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.24	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),P1)]

MIN(dist(P3,P1), (P6,P1))

= min[(0.22,0.23)]

= 0.22

To update the distance matrix MIN[dist(P3,P6),P2)]

MIN(dist(P3,P2), (P6,P2))

= min[(0.15,0.25)]

= 0.15

To update the distance matrix MIN[dist(P3,P6),P4)]

MIN(dist(P3,P4), (P6,P4))

= min[(0.15,0.22)]

= 0.15

To update the distance matrix MIN[dist(P3,P6),P5)]

MIN(dist(P3,P5), (P6,P5))

= min[(0.28,0.39)]

= 0.28

The updated distance matrix for cluster P3, P6

	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

To update the distance matrix MIN[dist(P2,P5),P1)]

MIN[dist(P2,P1), (P5,P1)]

= min[(0.23,0.34)]

= 0.23

To update the distance matrix MIN[dist(P2,P5),(P3,P6)]

MIN[dist(P2,(P3,P6)), (P5,(P3,P6))]

= min[(0.15,0.28)]

= 0.15

To update the distance matrix MIN[dist(P2,P5),P4)]

MIN[dist(P2,P4), (P5,P4)]

= min[(0.20,0.29)]

= 0.20

The updated distance matrix for cluster P2,P5

	P1	P2,P5	P3,P6	P4
P1	0			
P2,P5	0.23	0		
P3,P6	0.22	0.15	0	
P4	0.37	0.20	0.15	0

To update the distance matrix MIN[dist((P2,P5),(P3,P6)),P1]

MIN[dist((P2,P5),P1), ((P3,P6),P1)]

= min[(0.23,0.22)]

= 0.22

To update the distance matrix MIN[dist((P2,P5),(P3,P6)),P4]

MIN[dist((P2,P5),P4), ((P3,P6),P4)]

= min[(0.20,0.15)]

= 0.15

The updated distance matrix for cluster P2,P5,P3,P6

	P1	P2,P5,P3,P6	P4
P1	0		
P2,P5,P3,P6	0.22	0	
P4	0.37	0.15	0

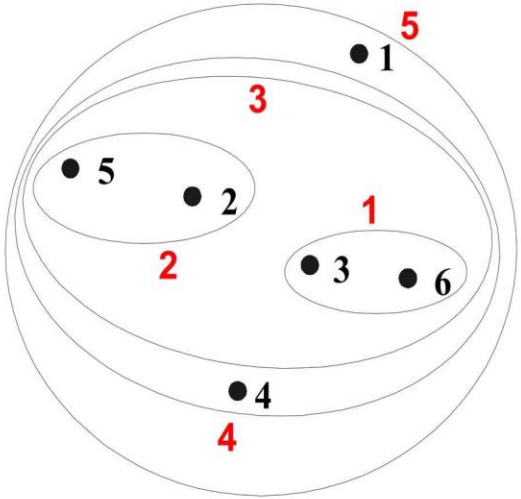
To update the distance matrix MIN[dist(P2,P5,P3,P6),P4]

MIN[dist((P2,P5,P3,P6),P1), (P4,P1)]

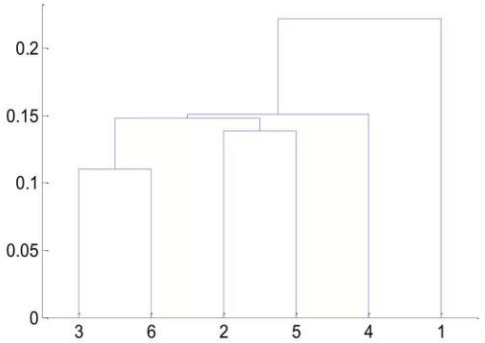
= min[(0.22,0.37)]

= 0.22

Hierarchical Clustering: MIN

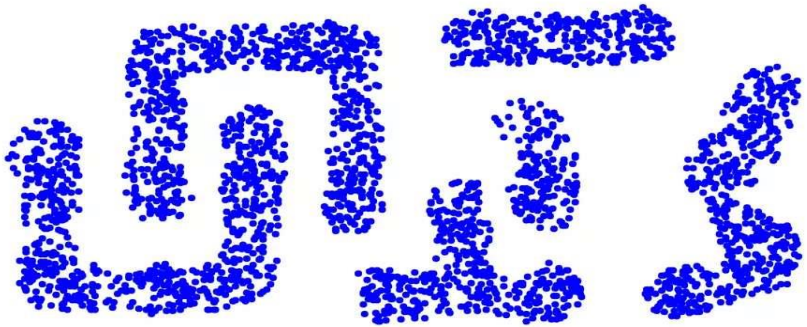


Nested Clusters

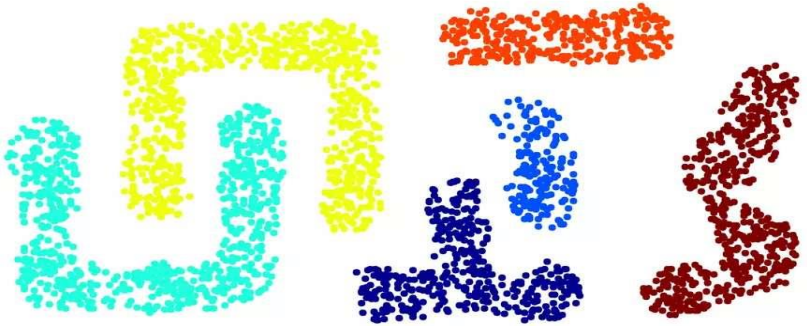


Dendrogram

Strength of MIN



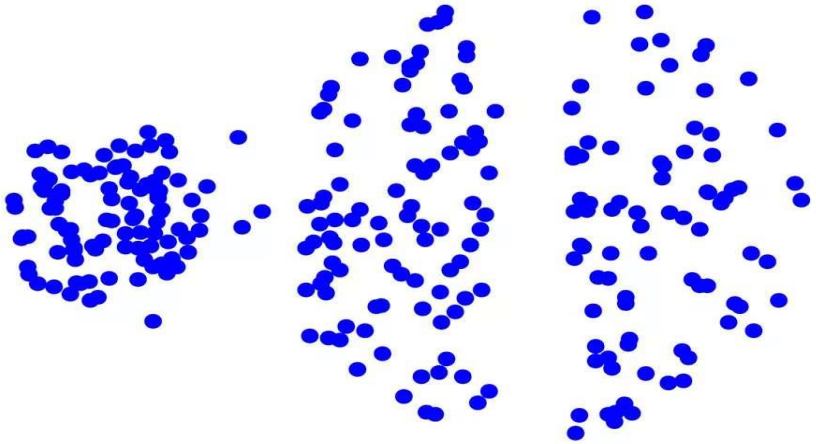
Original Points



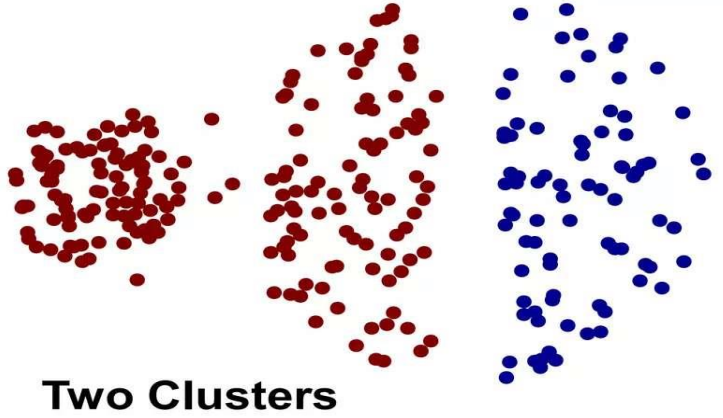
Six Clusters

- Can handle non-elliptical shapes

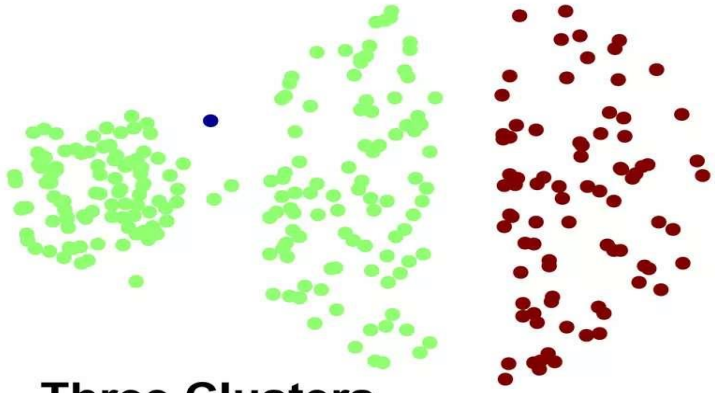
Limitations of MIN



Original Points



Two Clusters

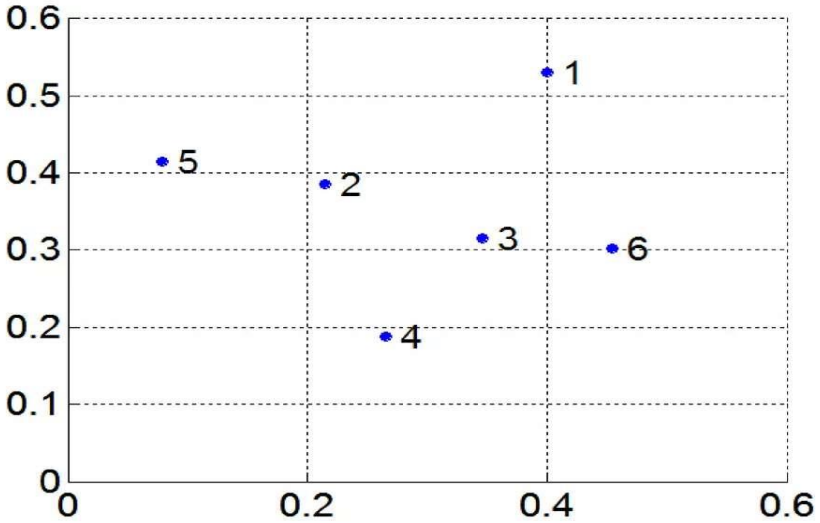


Three Clusters

- Sensitive to noise

MAX or Complete Linkage

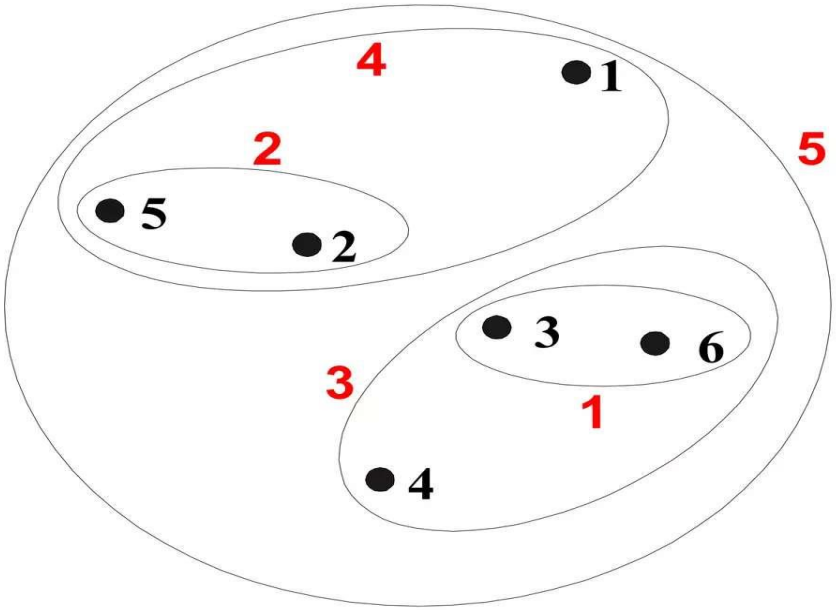
- Proximity of two clusters is based on the two most distant points in the different clusters
  - Determined by all pairs of points in the two clusters



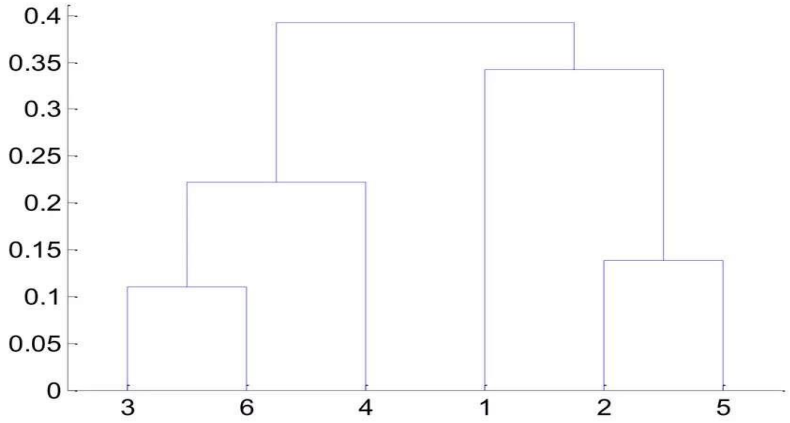
Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: MAX

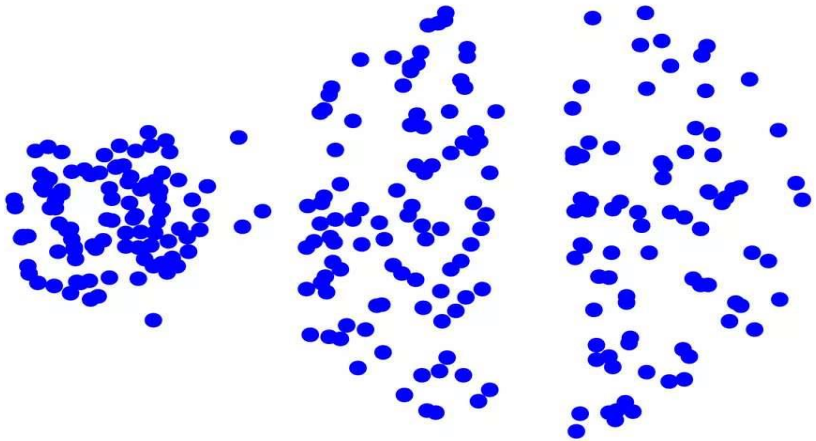


Nested Clusters

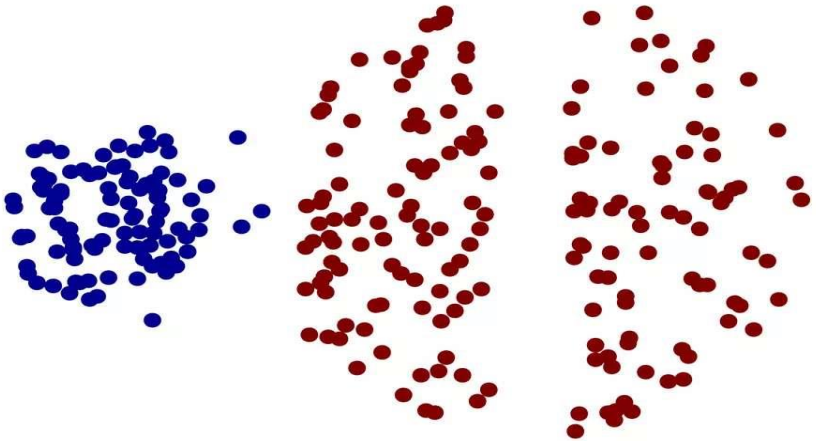


Dendrogram

Strength of MAX



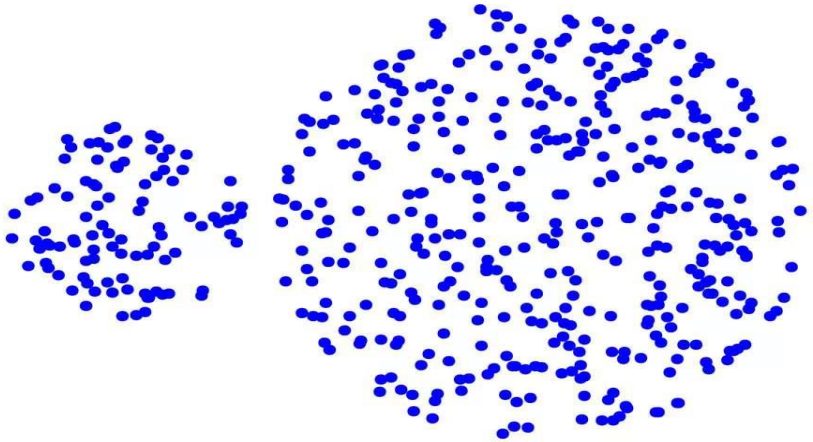
Original Points



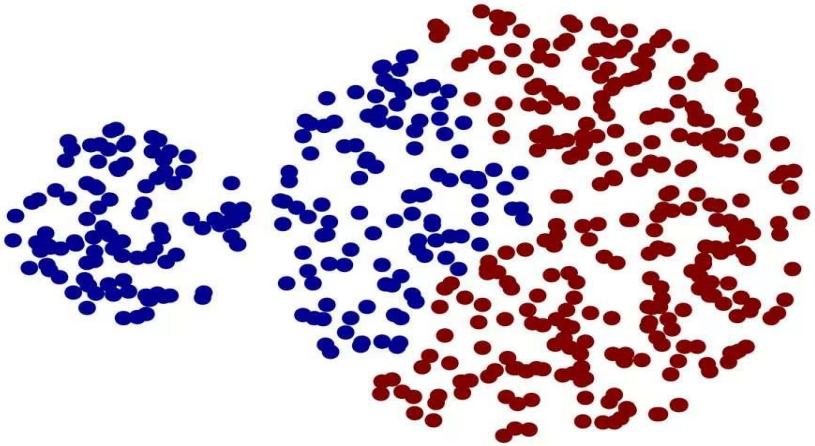
Two Clusters

- Less susceptible to noise

Limitations of MAX



Original Points



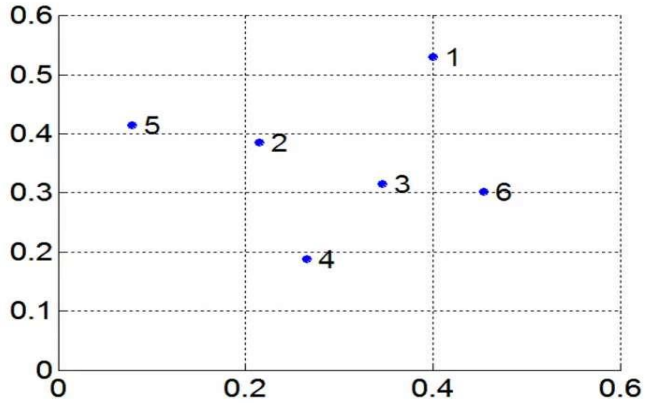
Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| \times |\text{Cluster}_j|}$$



Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

The distance matrix is, AVG[dist(P3,P6),P1]

$$\text{dist}((P3,P6),P1) = \tfrac{1}{2}\,(\text{dist}(P3,P1)+\text{dist}(P6,P1)$$
$$= \tfrac{1}{2}\,(0.22+0.23)$$
$$= \tfrac{1}{2}\,(0.45)$$
$$= 0.23$$

The distance matrix is, AVG[(dist(P3,P6),P2]

$$\text{dist}((P3,P6),P2) = \tfrac{1}{2}\,(\text{dist}(P3,P2)+\text{dist}(P6,P2)$$
$$= \tfrac{1}{2}\,(0.15+0.25)$$
$$= \tfrac{1}{2}\,(0.4)$$
$$= 0.2$$

The distance matrix is, AVG[dist(P3,P6),P4]

$$dist((P3,P6),P4) = \frac{1}{2}\,(dist(P3,P4)+dist(P6,P4)$$

$$= \frac{1}{2}\,(0.15+0.22)$$

$$= \frac{1}{2}\,(0.37)$$

$$= 0.19$$

The distance matrix is, AVG[dist(P3,P6),P5]

$$dist((P3,P6),P5) = \frac{1}{2}\,(dist(P3,P5)+dist(P6,P5)$$

$$= \frac{1}{2}\,(0.28+0.39)$$

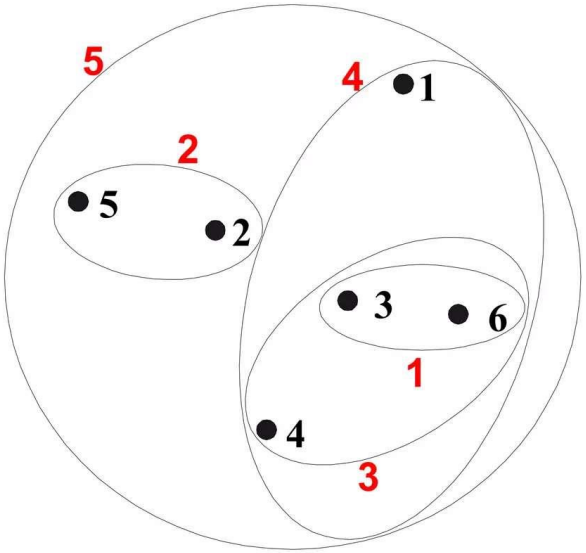
$$= \frac{1}{2}\,(0.67)$$

$$= 0.34$$

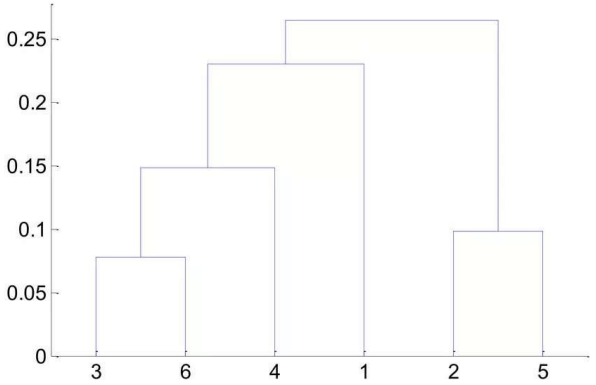
The updated distance matrix for cluster (P3,P6)

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.23	0.2	0		
P4	0.37	0.20	0.19	0	
P5	0.34	0.14	0.34	0.29	0

Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

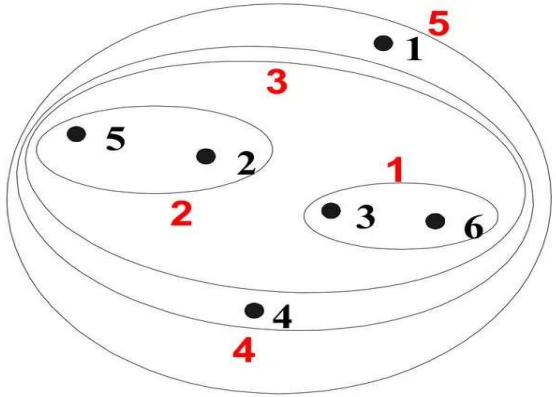
Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise
- Limitations
  - Biased towards globular clusters

Cluster Similarity: Ward's Method

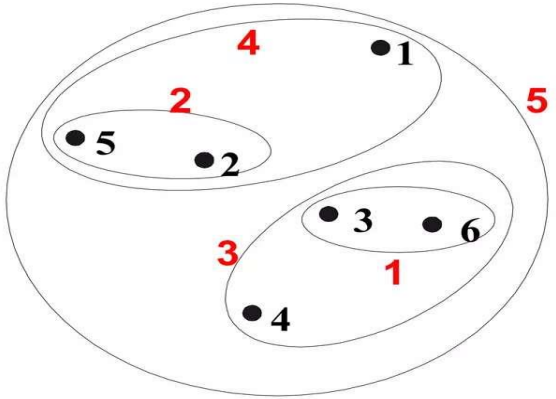
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

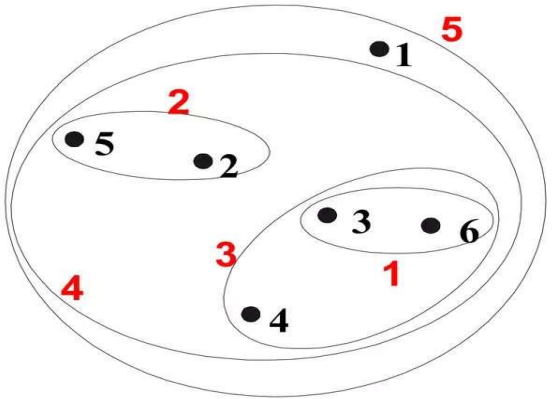
Hierarchical Clustering: Comparison



MIN

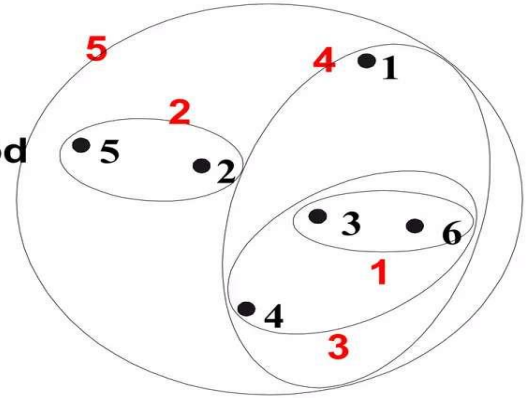
MAX





Group Average

Ward's Method



Difference ways to **measure** the **distance** between two clusters

- **Single Linkage**

  $D(c_1,c_2) = \min D(x_1,x_2)$

  Minimum distance or distance between closest elements in clusters

- **Complete Linkage**

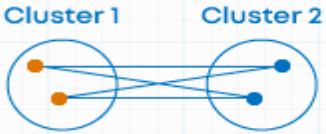
  $D(c_1,c_2) = \max D(x_1,x_2)$

  Maximum distance between elements in clusters

- **Average Linkage**

  $D(c_1,c_2) = \frac{1}{|c_1|}\frac{1}{|c_2|}\Sigma\Sigma D(x_1,x_2)$

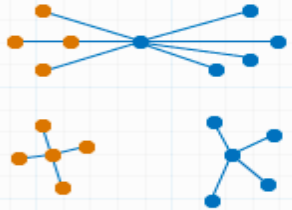
  Average of the distances of all pairs



- **Centroid Method**

  Combining clusters with minimum distance between the centroids of the two clusters

- **Ward's Method**
  - Combining clusters where increase in within cluster variance is to the smallest degree.
  - Objective is to minimize the total within cluster vairance



Advantages

- The agglomerative technique is easy to implement.
- It can produce an ordering of objects, which may be informative for the display.
- In agglomerative Clustering, there is no need to pre-specify the number of clusters.
- By the Agglomerative Clustering approach, smaller clusters will be created, which may discover similarities in data.

Disadvantages

- The agglomerative technique gives the best result in some cases only.
- The algorithm can never undo what was done previously, which means if the objects may have been incorrectly grouped at an earlier stage, and the same result should be close to ensure it.
- The usage of various distance metrics for measuring distances between the clusters may produce different results. So performing multiple experiments and then comparing the result is recommended to help the actual results' veracity.

Distribution-based Clustering > > GMM gaussian mixture model

- This clustering approach assumes data is composed of distributions, such as **Gaussian distributions**.
- As distance from the distribution's center increases, the probability that a point belongs to the distribution decreases. The bands show that decrease in probability.
- When you do not know the type of distribution in your data, you should use a different algorithm.