# MACHINE LEARNING

# Distance Based Models

**Dr G.Kalyani**

**Department of Information Technology**

**Velagapudi Ramakrishna Siddhartha Engineering College**

# Topics

- Introduction
- Nearest Neighbor Classification
- Distance based Clustering
  - Partitioning Clustering
    - K-Means algorithm,
    - Clustering around medoids,
  - Hierarchical Clustering.

# Topics

- Introduction
- Nearest Neighbor Classification
- Distance based Clustering
  - Partitioning Clustering
    - K-Means algorithm,
    - Clustering around medoids,
  - Hierarchical Clustering.

# Definition of Distance Metric

**Definition 8.2 (Distance metric).** *Given an instance space $\mathscr{X}$, a* distance metric *is a function* $\text{Dis} : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ *such that for any* $x, y, z \in \mathscr{X}$:

1. *distances between a point and itself are zero:* $\text{Dis}(x, x) = 0$;
2. *all other distances are larger than zero: if* $x \neq y$ *then* $\text{Dis}(x, y) > 0$;
3. *distances are symmetric:* $\text{Dis}(y, x) = \text{Dis}(x, y)$;
4. *detours can not shorten the distance:* $\text{Dis}(x, z) \leq \text{Dis}(x, y) + \text{Dis}(y, z)$.

# Minkowski Distance

**Definition 8.1 (Minkowski distance).** *If $\mathscr{X} = \mathbb{R}^d$, the* Minkowski distance *of order $p > 0$ is defined as*

$$\boxed{\mathrm{Dis}_p(\mathbf{x},\mathbf{y}) = \left(\sum_{j=1}^{d} |x_j - y_j|^p\right)^{1/p} = ||\mathbf{x}-\mathbf{y}||_p}$$

*where $||\mathbf{z}||_p = \left(\sum_{j=1}^{d} |z_j|^p\right)^{1/p}$ is the* $p$-norm *(sometimes denoted $L_p$ norm) of the vector $\mathbf{z}$. We will often refer to* $\mathrm{Dis}_p$ *simply as the $p$-norm.*

# Euclidean Distance

- The ***2-norm refers to the familiar Euclidean distance:***

$$\boxed{\mathrm{Dis}_2(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{j=1}^{d}(x_j - y_j)^2}}$$

# Manhattan Distance

- ***1-norm denotes Manhattan distance or cityblock distance:***

$$\boxed{\mathrm{Dis}_1(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{d} |x_j - y_j|}$$

# Chebyshev Distance

- **Chebyshev Distance:**

$$\boxed{\mathrm{Dis}_{\infty}(\mathbf{x}, \mathbf{y}) = \max_j |x_j - y_j|.}$$

# Hamming Distance

- **Hamming Distance:** also called as 0-*norm (or L0 norm).* The corresponding distance counts the number of positions in which vectors **x and y differ.**

$$\boxed{\mathrm{Dis}_0(\mathbf{x},\mathbf{y}) = \sum_{j=1}^{d} (x_j - y_j)^0 = \sum_{j=1}^{d} I[x_j = y_j]}$$

Where $Z^0 = 0$ *for* $Z = 0$

$= 1$ *otherwise.*

*****_Hamming distance is used to calculate the distance between instances which are described with categorical attributes or dimensions._

# Examples for Distance Calculation

**Q1: Find the distance between X= (2, 3) and Y= (4, 1).**

- Euclidean Distance= $\sqrt{\sum_{j=1}^{d}(x_j - y_j)^2}$ $= \sqrt{(2-4)^2+(3-1)^2}$ = 2.83
- Manhattan Distance = $\sum_{j=1}^{d}|x_j - y_j|$ = |(2-4)|+|(3-1)| = 4
- Chebshev Dstance= $\max_j |x_j - y_j|.$ = max(|(2-4)|,|(3-1)|) = 2

**Q2: Find the distance between X= (yes, true) and Y= (no, true).**

- Hamming Distance= I(yes=no) + I(true=true) = 0 + 1 = 1

# Examples for Practice

- **Q3: Find the distance between the instances**

  X = (6, 2,4) and Y= (9, 1,-2).

- **Q4: Determine the distance between the instances**

  X = (yes, male, high, good) and Y=(No, male, high, Excellent).

# Topics

- Introduction
- Nearest Neighbor Classification
- Distance based Clustering
  - Partitioning Clustering
    - K-Means algorithm,
    - Clustering around medoids,
  - Hierarchical Clustering.

# Nearest Neighbor(NN) Classification

- Distance based & Supervised
- Used for both Classification & Regression
- Decision is made based on K-Number of neighborhood points
- NN is **easy to implement**. Only 2 parameters required i.e. the value of K and the distance function (e.g. Euclidean or Manhattan etc.)
- **Does not work with large dataset:** In large datasets, the cost of calculating the distance between the new point and each existing points is huge which degrades the performance of the algorithm.

# Example for Nearest Neighbor Classification

| Name | Age | Gender | Sport |
|---|---|---|---|
| Ajay | 32 | M | Football |
| Mark | 40 | M | Neither |
| Sara | 16 | F | Cricket |
| Zaira | 34 | F | Cricket |
| Sachin | 55 | M | Neither |
| Rahul | 40 | M | Cricket |
| Pooja | 20 | F | Neither |
| Smith | 15 | M | Cricket |
| Laxmi | 55 | M | Football |
| Machael | 15 | M | Football |

| | | | |
|---|---|---|---|
| **Angelina** | **5** | **F** | **?** |

# Example for Nearest Neighbor Classification

- Assume that we are using Euclidean Distance with value of K as 3.

- Distance between *angelina* (5,1) and *Ajay* (32,0) is

$$D=\sqrt{(5-32)^2+(1-0)^2}$$

$$=\sqrt{27^2+1^2}=\sqrt{729+1}=27.02$$

# Example for Nearest Neighbor Classification

| Name | Age | Gender | Distance | Sport |
|---|---|---|---|---|
| Ajay | 32 | 0 | 27.02 | Football |
| Mark | 40 | 0 | | Neither |
| Sara | 16 | 1 | | Cricket |
| Zaira | 34 | 1 | | Cricket |
| Sachin | 55 | 0 | | Neither |
| Rahul | 40 | 0 | | Cricket |
| Pooja | 20 | 1 | | Football |
| Smith | 15 | 0 | | Cricket |
| Laxmi | 55 | 0 | | Football |
| Machael | 15 | 0 | | Football |

# Example for Nearest Neighbor Classification

| Name | Age | Gender | Distance | Sport |
|---|---|---|---|---|
| Ajay | 32 | 0 | 27.02 | Football |
| Mark | 40 | 0 | 35.01 | Neither |
| Sara | 16 | 1 | 11.00 | Cricket |
| Zaira | 34 | 1 | 29.00 | Cricket |
| Sachin | 55 | 0 | 50.01 | Neither |
| Rahul | 40 | 0 | 35.01 | Cricket |
| Pooja | 20 | 1 | 15.00 | Football |
| Smith | 15 | 0 | 10.00 | Cricket |
| Laxmi | 55 | 0 | 50.00 | Football |
| Machael | 15 | 0 | 10.05 | Football |

# Example for Nearest Neighbor Classification

- K=3 in the example :
- So the 3 nearest neighbors are

| | | | | |
|---|---|---|---|---|
| Sara | 16 | 1 | 11.00 | Cricket |
| Smith | 15 | 0 | 10.00 | Cricket |
| Machael | 15 | 0 | 10.05 | Football |

- Majority Voting Rule:

*Angelina belongs to class of Cricket*

# Example for Nearest Neighbor Classification

- Let K=5 in the example :
- So the 5 nearest neighbors are

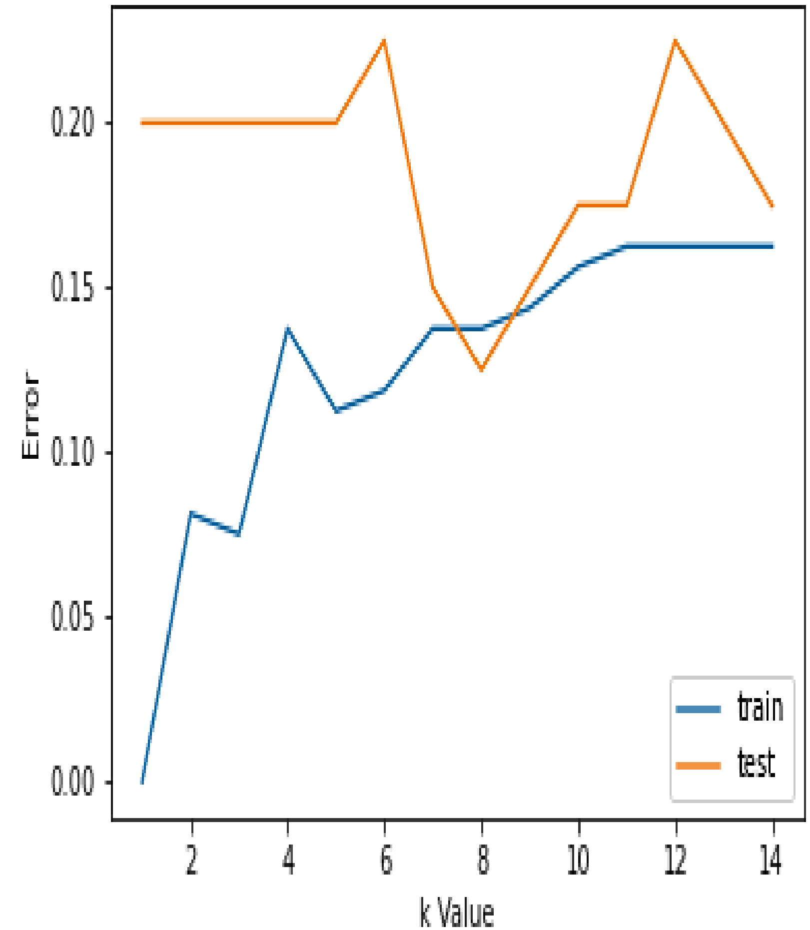
| | | | | |
|---|---|---|---|---|
| Sara | 16 | 1 | 11.00 | Cricket |
| Smith | 15 | 0 | 10.00 | Cricket |
| Machael | 15 | 0 | 10.05 | Football |
| Pooja | 20 | 1 | 15.00 | Football |
| Ajay | 32 | 0 | 27.02 | Football |

- Majority Voting Rule:

**Angelina belongs to class of Football**

# K-Nearest Neighbor Classification

- If K is too small, sensitive to noise points
- If K is too large, neighborhood may include points from other classes
- Thumb rule: **K< sqrt(n)** where n is number of samples
- To choose the correct K value use error curves.



# KNN Algorithm

**Input:** Dataset of n instances, K, Distance measure, new instance T

**Output:** Predicted Label for the given new instance T

**Method:**

1. For each instance x in the dataset
   a. Calculate the distance between T and x.
   b. Add the distance and the index of the x to an ordered collection.
2. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
3. Pick the first K entries from the sorted collection.
4. Get the labels of the selected K entries.
5. **If Regression**, return the **mean** of the K labels.
6. **If Classification**, return the **mode** of the K labels.

# Important points in KNN Classification

- Takes more time if number of dimensions in the dataset is more
  - Use dimensionality reduction techniques and feature selection techniques to reduce the number of dimensions.

- How to handle noise in the Data
  - Increase the K value

- Relation between K value and Bias & Variance
  - If K is small bias is less(as k is more bias increases)
  - As k decreases variance increases

# Topics

- Introduction
- Nearest Neighbor Classification
- Distance based Clustering
  - Partitioning Clustering
    - K-Means algorithm,
    - Clustering around medoids,
  - Hierarchical Clustering.

# What is Clustering?

- **Cluster:** A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- **Clustering**
  - Finding similarities between data according to the characteristics found in the data and
  - grouping similar data objects into clusters
- **Unsupervised learning**: no predefined class labels for the data

# Applications of Clustering

- **Information retrieval:** document clustering
- **Land use:** Identification of areas of similar land use in an earth observation database
- **Marketing:** Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs

# Applications of Clustering

- Image Segmentation (Clustering the pixels)

# Quality: What Is Good Clustering?

- A **good clustering** method will produce high quality clusters
  - **high <u>intra-Class or Intra-Cluster</u> similarity**: cohesive within clusters
  - **low <u>Inter-class or Inter-Cluster</u> similarity**: distinctive between clusters

- The **quality of a clustering method** depends on
  - the similarity measure used by the method
  - Process used for clustering, and

# Measure the Quality of Clustering

- **Similarity / Dissimilarity metric**
  - Dissimilarity is expressed in terms of a distance function, typically metric: $d(i, j)$
  - The definitions of distance functions are usually rather different for categorical and continuous attributes.
  - Weights should be associated with different variables based on applications and data semantics

# Distance Measures-for Numerical data

- **Properties of Distance Measures:**
  - for all objects A and B, dist(A, B) $\geq$ 0, and dist(A, B) = dist(B, A)
  - for any object A, dist(A, A) = 0
- **Common Distance Measures:**

$$X = \langle x_1, x_2, \cdots, x_n \rangle \qquad Y = \langle y_1, y_2, \cdots, y_n \rangle$$

  - Manhattan distance:

$$dist(X,Y) = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_n - y_n|$$

  - Euclidean distance:

$$dist(X,Y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2}$$

**Can be normalized to make values fall between 0 and 1.**

  - Cosine similarity:

$$dist(X,Y) = 1 - sim(X,Y)$$

$$sim(X,Y) = \frac{\sum_i (x_i \times y_i)}{\sqrt{\sum_i x_i^2 \times \sum_i y_i^2}}$$

# Major Clustering Approaches

- **Partitioning approach**:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: K-Means, K-Medoids, CLARANS
- **Hierarchical approach**:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- **Density-based approach**:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue
- **Grid-based approach**:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE

# Partitioning Algorithms: Basic Concept

- **Given *k*, Partitioning a database *D* of *n* objects into a set of *K* clusters** that optimizes the chosen partitioning criterion
  - **Global optimal**: exhaustively enumerate all partitions
  - **Heuristic methods:** *k-means* and *k-medoids* algorithms
  - ***k-Means*** **:** Each cluster is represented by the center of the cluster
  - ***k-Medoids*** **or PAM (Partition Around Medoids)** : Each cluster is represented by one of the objects in the cluster

# Partitioning Algorithms: Basic Concept

- **Partitioning method:**

**Partitioning a database *D* of *n* objects into a set of *K* clusters, such that the sum of squared distances is minimized**

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, c_i)^2,$$

**where $c_i$ is the centroid of cluster $C_i$**

# *K-Means* Clustering Method

**Algorithm 8.1:** KMeans$(D, K)$ – $K$-means clustering using Euclidean distance $\text{Dis}_2$.

**Input** : data $D \subseteq \mathbb{R}^d$; number of clusters $K \in \mathbb{N}$.

**Output** : $K$ cluster means $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$.

randomly initialise $K$ vectors $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$;

**repeat**

  assign each $\mathbf{x} \in D$ to $\arg\min_j \text{Dis}_2(\mathbf{x}, \mu_j)$;

  **for** $j = 1$ to $K$ **do**

    $D_j \leftarrow \{\mathbf{x} \in D | \mathbf{x} \text{ assigned to cluster } j\}$;

    $\mu_j = \frac{1}{|D_j|} \sum_{\mathbf{x} \in D_j} \mathbf{x}$;

  **end**

**until** no change in $\mu_1, \ldots, \mu_K$;

**return** $\mu_1, \ldots, \mu_K$;

# K-Means Algorithm: Example

Partition the following data points into 2 clusters

| Data | X | Y |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 1 | 2 |
| 4 | 2 | 2 |
| 5 | 3 | 3 |
| 6 | 6 | 6 |
| 7 | 6 | 8 |
| 8 | 5 | 7 |
| 9 | 7 | 5 |
| 10 | 4 | 5 |

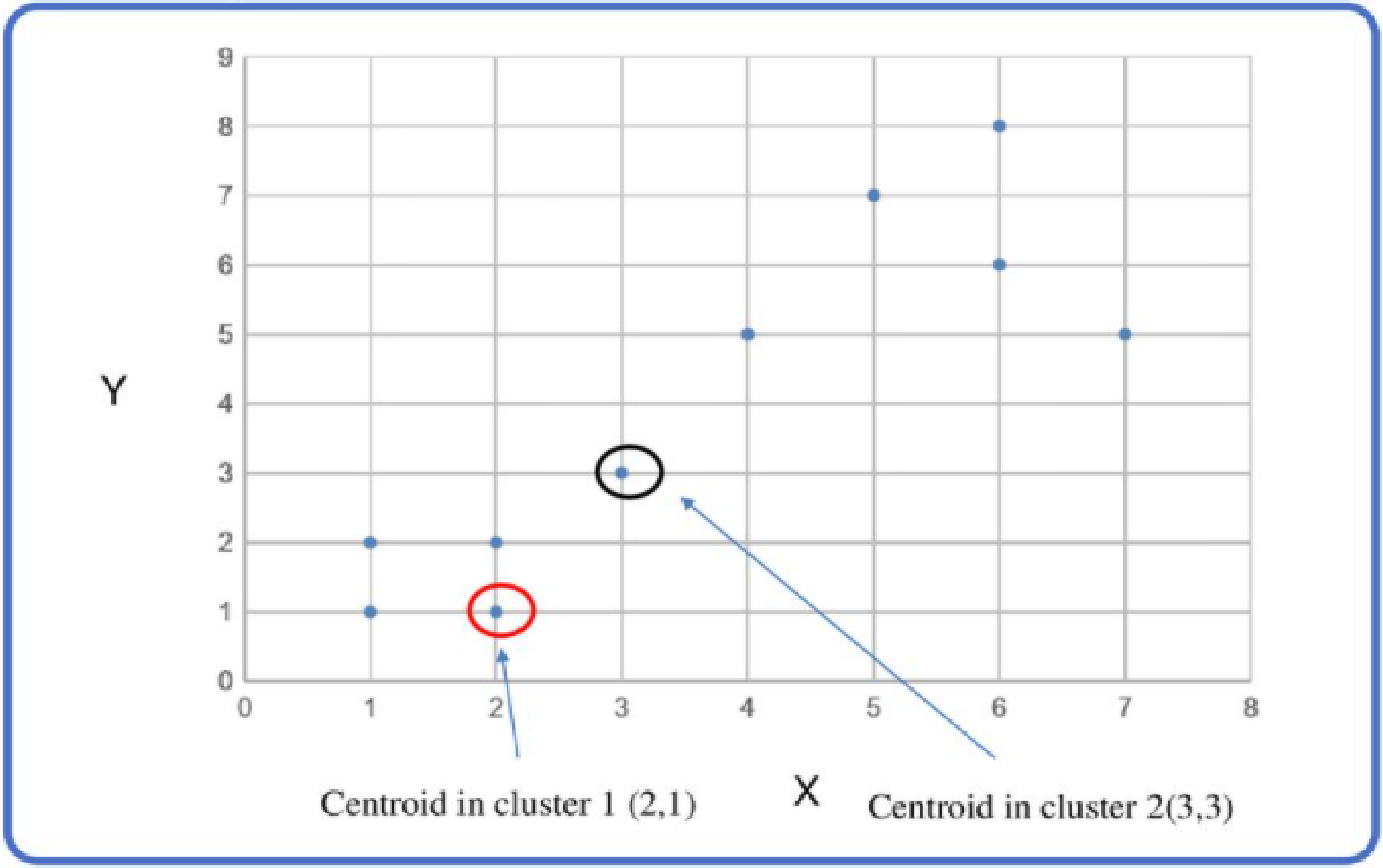
# K-Means Algorithm: Example

Randomly select 2 data points as cluster centers

| Data | X | Y |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 1 | 2 |
| 4 | 2 | 2 |
| 5 | 3 | 3 |
| 6 | 6 | 6 |
| 7 | 6 | 8 |
| 8 | 5 | 7 |
| 9 | 7 | 5 |
| 10 | 4 | 5 |

# K-Means Algorithm: Example

- First, randomly set a point as centroid point
- For example, $k = 2$



# K-Means Algorithm: Example

- Calculate the distance between the centroid and each point

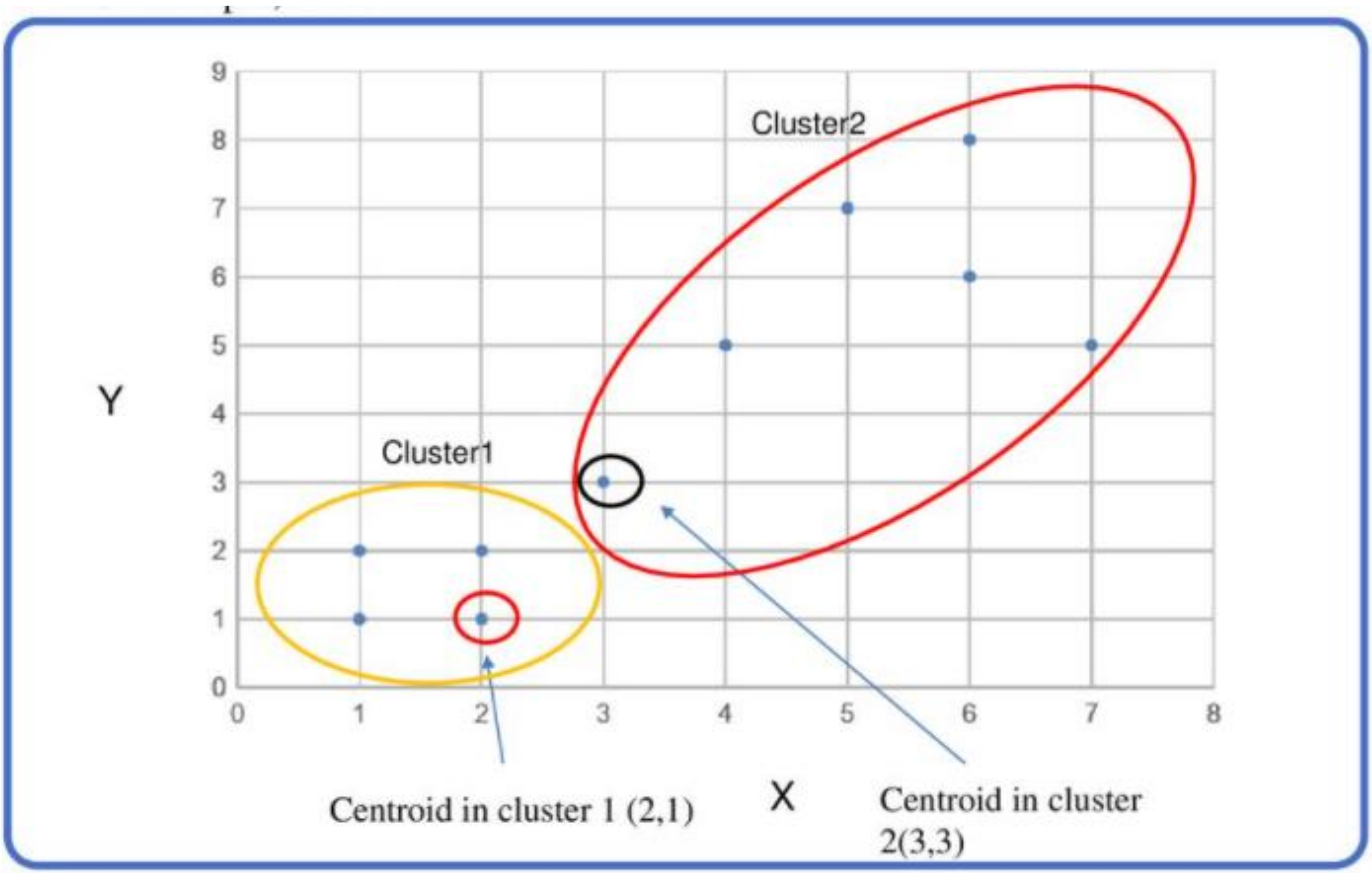
| Data | X | Y | Distance between centroid(2,1) and point in cluster 1 | Compare | Distance between centroid(3,3) and point in cluster 2 | Cluster |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | | | | |
| 2 | 2 | 1 | | | | |
| 3 | 1 | 2 | | | | |
| 4 | 2 | 2 | | | | |
| 5 | 3 | 3 | | | | |
| 6 | 6 | 6 | | | | |
| 7 | 6 | 8 | | | | |
| 8 | 5 | 7 | | | | |
| 9 | 7 | 5 | | | | |
| 10 | 4 | 5 | | | | |

# K-Means Algorithm: Example

- Calculate the distance between the centroid and each point

| Data | X | Y | Distance between centroid(2,1) and point in cluster 1 | Compare | Distance between centroid(3,3) and point in cluster 2 | Cluster |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | | < | | 1 |
| 2 | 2 | 1 | | < | | 1 |
| 3 | 1 | 2 | | < | | 1 |
| 4 | 2 | 2 | | < | | 1 |
| 5 | 3 | 3 | | > | | 2 |
| 6 | 6 | 6 | | > | | 2 |
| 7 | 6 | 8 | | > | | 2 |
| 8 | 5 | 7 | | > | | 2 |
| 9 | 7 | 5 | | > | | 2 |
| 10 | 4 | 5 | | > | | 2 |

# K-Means Algorithm: Example



# K-Means Algorithm: Example

## K-means Clustering – Centroid update step

- In this step, the centroids are recomputed by taking the mean of all data points assigned to that centroid's cluster

| Data | X | Y | Cluster |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 2 | 1 | 1 |
| 3 | 1 | 2 | 1 |
| 4 | 2 | 2 | 1 |
| 5 | 3 | 3 | 2 |
| 6 | 6 | 6 | 2 |
| 7 | 6 | 8 | 2 |
| 8 | 5 | 7 | 2 |
| 9 | 7 | 5 | 2 |
| 10 | 4 | 5 | 2 |

New centroid(cluster 1)

$$=\left(\frac{1+2+1+2}{4}, \frac{1+1+2+2}{4}\right)$$

$$= (1.5,\ 1.5)$$

New centroid(cluster 2)

$$=\left(\frac{3+6+6+5+7+4}{6}, \frac{3+6+8+7+5+5}{6}\right)$$

$$= (5.1,\ 5.6)$$

| Cluster | New Centroid | Data Index |
|---|---|---|
| 1 | (1.5, 1.5) | 1,2,3,4 |
| 2 | (5.1, 5.6) | 5,6,7,8,9,10 |

# K-means Clustering – Data assignment step

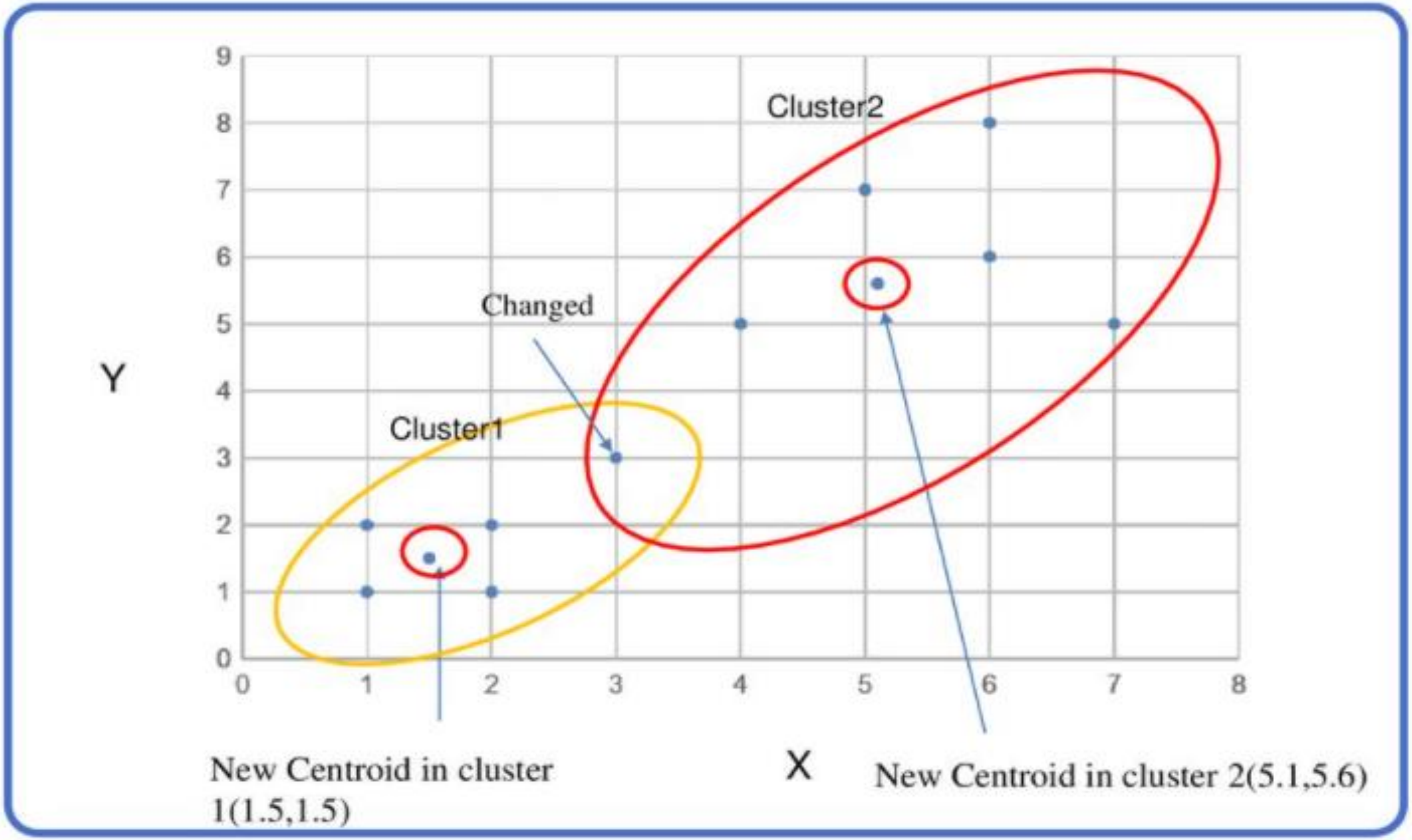
- Calculate the distance between the centroid and each point

| Data | X | Y | Distance between centroid(1.5,1.5) and point in cluster 1 | Compare | Distance between centroid(5.1,5.6) and point in cluster 2 | Cluster |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | | < | | 1 |
| 2 | 2 | 1 | | < | | 1 |
| 3 | 1 | 2 | | < | | 1 |
| 4 | 2 | 2 | | < | | 1 |
| 5 | 3 | 3 | | < | | 2 |
| 6 | 6 | 6 | | > | | 2 |
| 7 | 6 | 8 | | > | | 2 |
| 8 | 5 | 7 | | > | | 2 |
| 9 | 7 | 5 | | > | | 2 |
| 10 | 4 | 5 | | > | | 2 |

Changed

# K-Means Algorithm: Example

- Now, repeat calculating distance between new centroid and each point



# K-Means Algorithm: Example

## K-means Clustering – Centroid update step

- In this step, the centroids are recomputed by taking the mean of all data points assigned to that centroid's cluster

| Data | X | Y | Cluster |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 2 | 1 | 1 |
| 3 | 1 | 2 | 1 |
| 4 | 2 | 2 | 1 |
| 5 | 3 | 3 | 2 |
| 6 | 6 | 6 | 2 |
| 7 | 6 | 8 | 2 |
| 8 | 5 | 7 | 2 |
| 9 | 7 | 5 | 2 |
| 10 | 4 | 5 | 2 |

New centroid(cluster 1)

$$=\left(\frac{1+2+1+2+3}{5}, \frac{1+1+2+2+3}{5}\right)$$

$$= (1.8,\ 1.6)$$

New centroid(cluster 2)

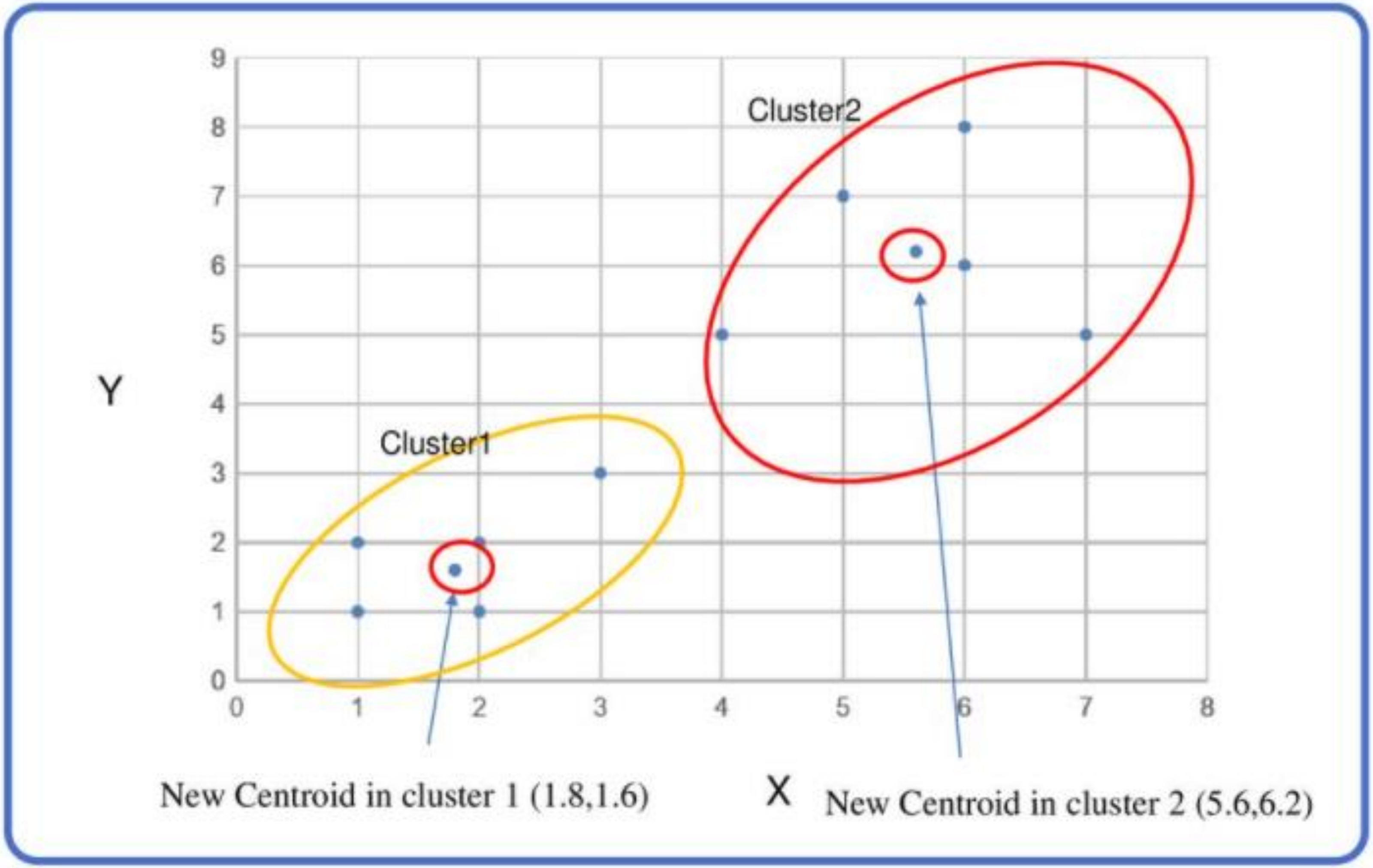
$$=\left(\frac{6+6+5+7+4}{5}, \frac{6+8+7+5+5}{5}\right)$$

$$= (5.6,\ 6.2)$$

| Cluster | New Centroid | Data Index |
|---|---|---|
| 1 | (1.8, 1.6) | 1,2,3,4,5 |
| 2 | (5.6, 6.2) | 6,7,8,9,10 |

# K-Means Algorithm: Example

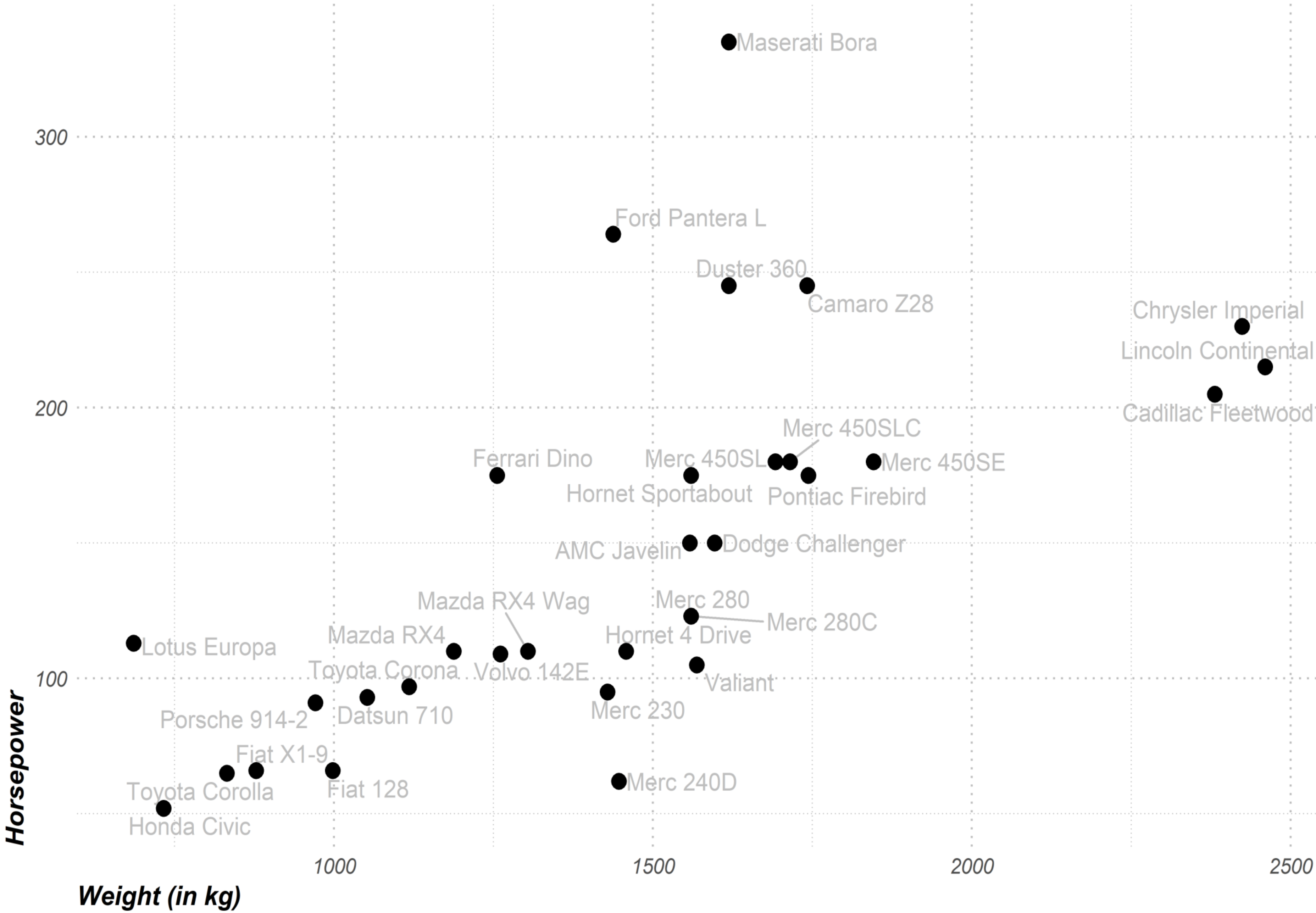
- Now, repeat calculating distance between new centroid and each point



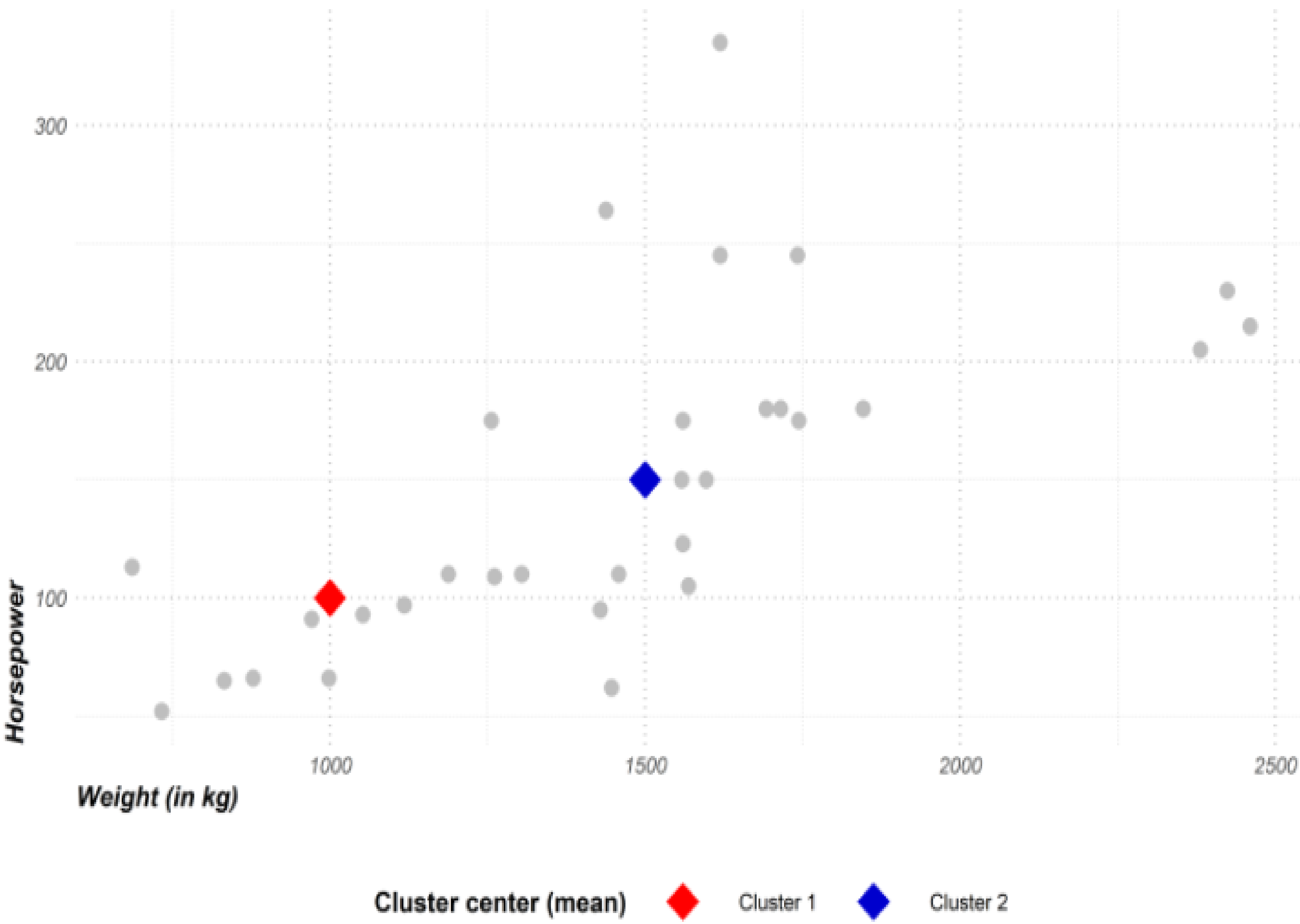
# Example 2 of K-means Clustering

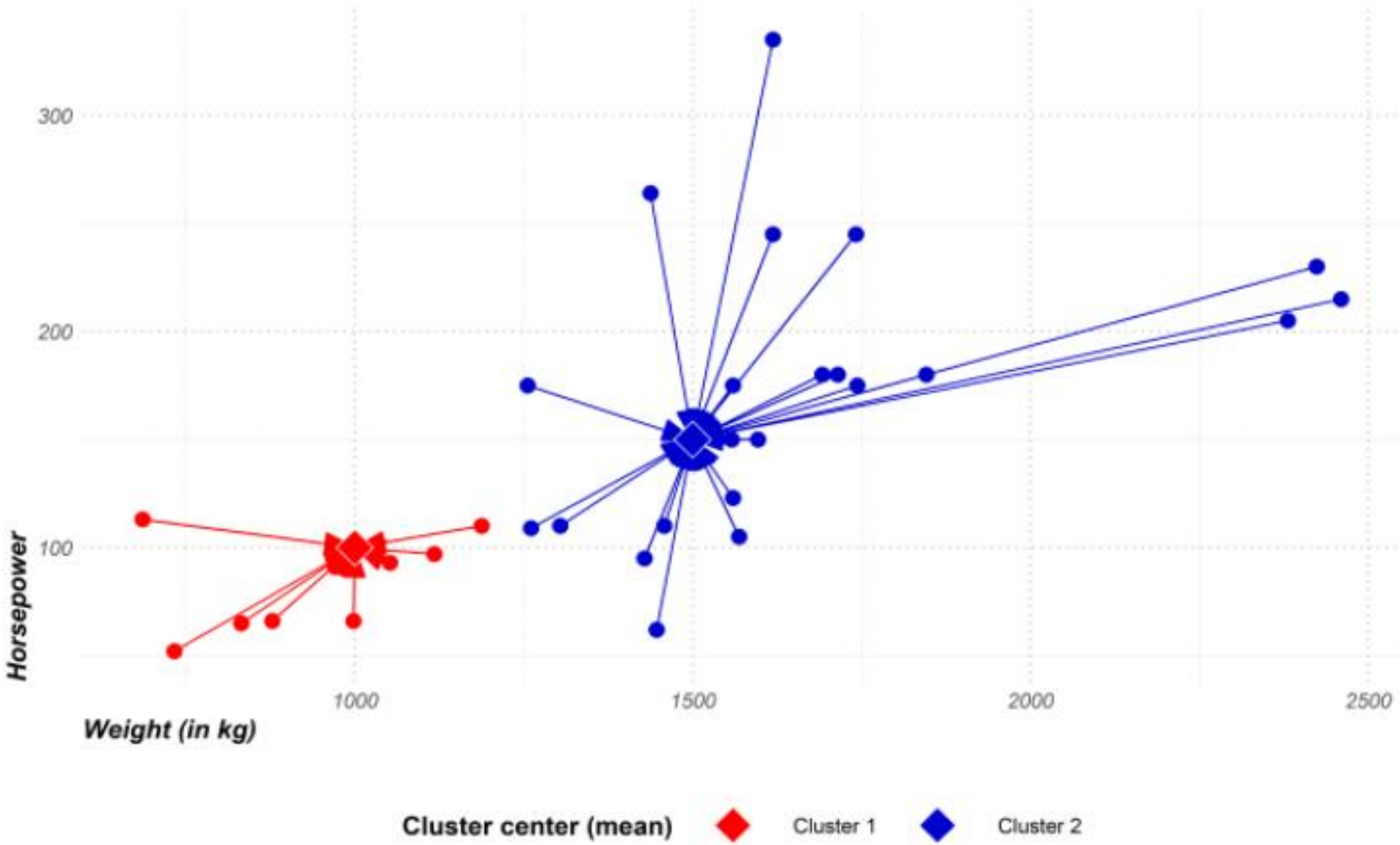
- dataset containing information on 32 cars.

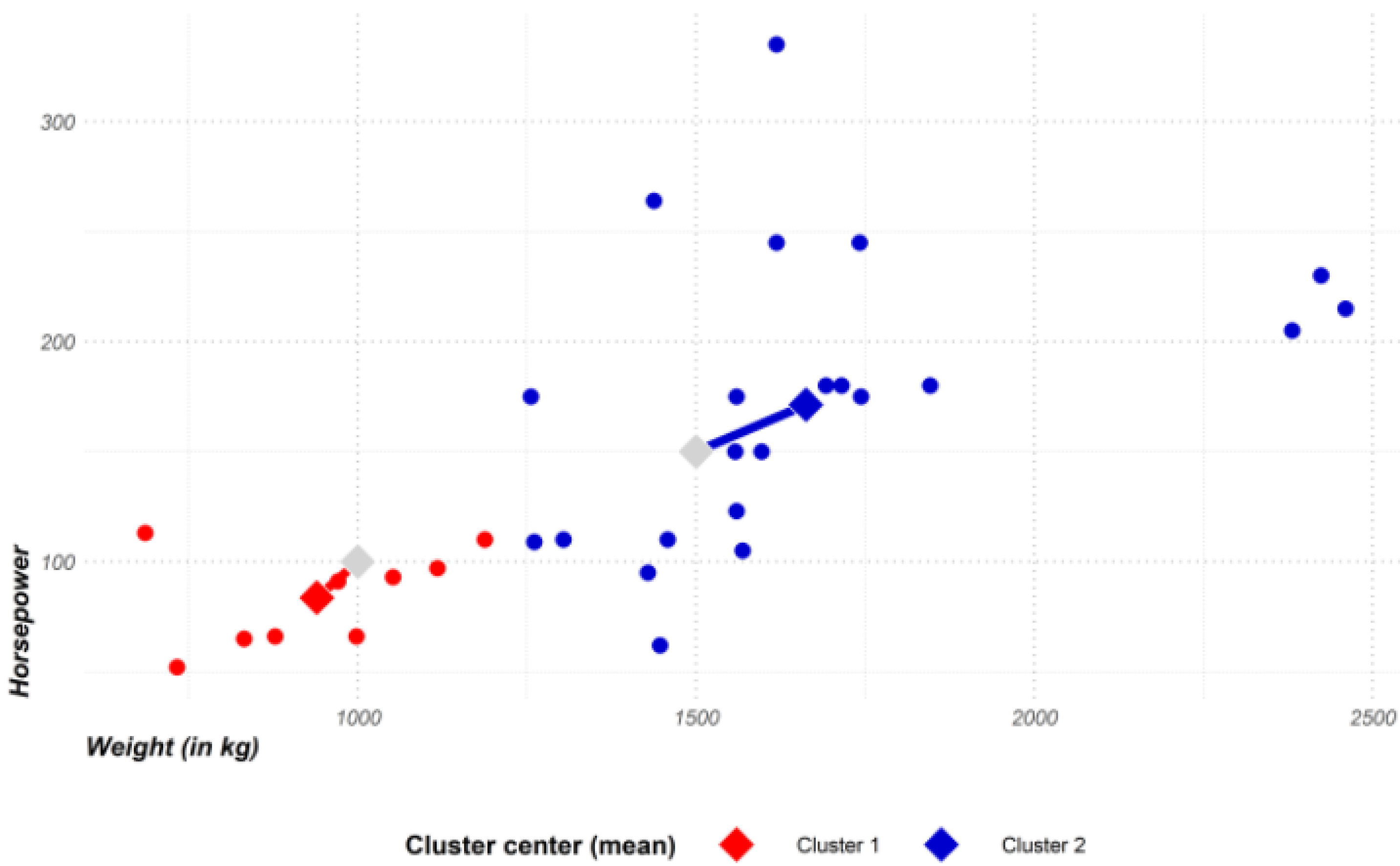
- We can consider each car a separate observation.
- For each observation, we have its weight and its horsepower.



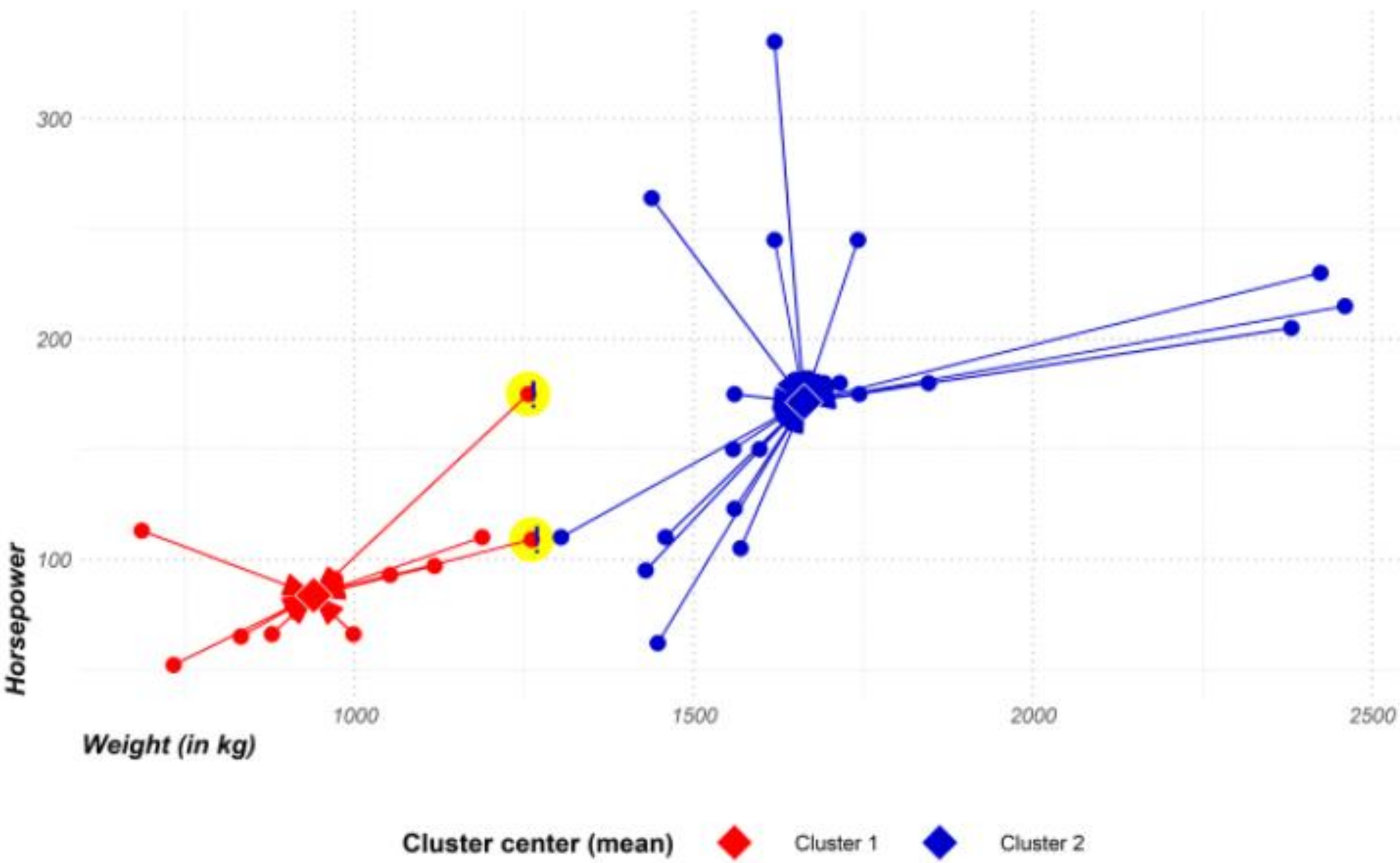
Maserati Bora
Ford Pantera L
Duster 360
Camaro Z28
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Ferrari Dino
Merc 450SL
Merc 450SE
Hornet Sportabout
Pontiac Firebird
AMC Javelin
Dodge Challenger
Mazda RX4 Wag
Merc 280
Merc 280C
Lotus Europa
Mazda RX4
Hornet 4 Drive
Toyota Corona
Volvo 142E
Valiant
Porsche 914-2
Datsun 710
Merc 230
Fiat X1-9
Toyota Corolla
Fiat 128
Merc 240D
Honda Civic
300
200
100
1000
1500
2000
2500
Horsepower
Weight (in kg)


300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2

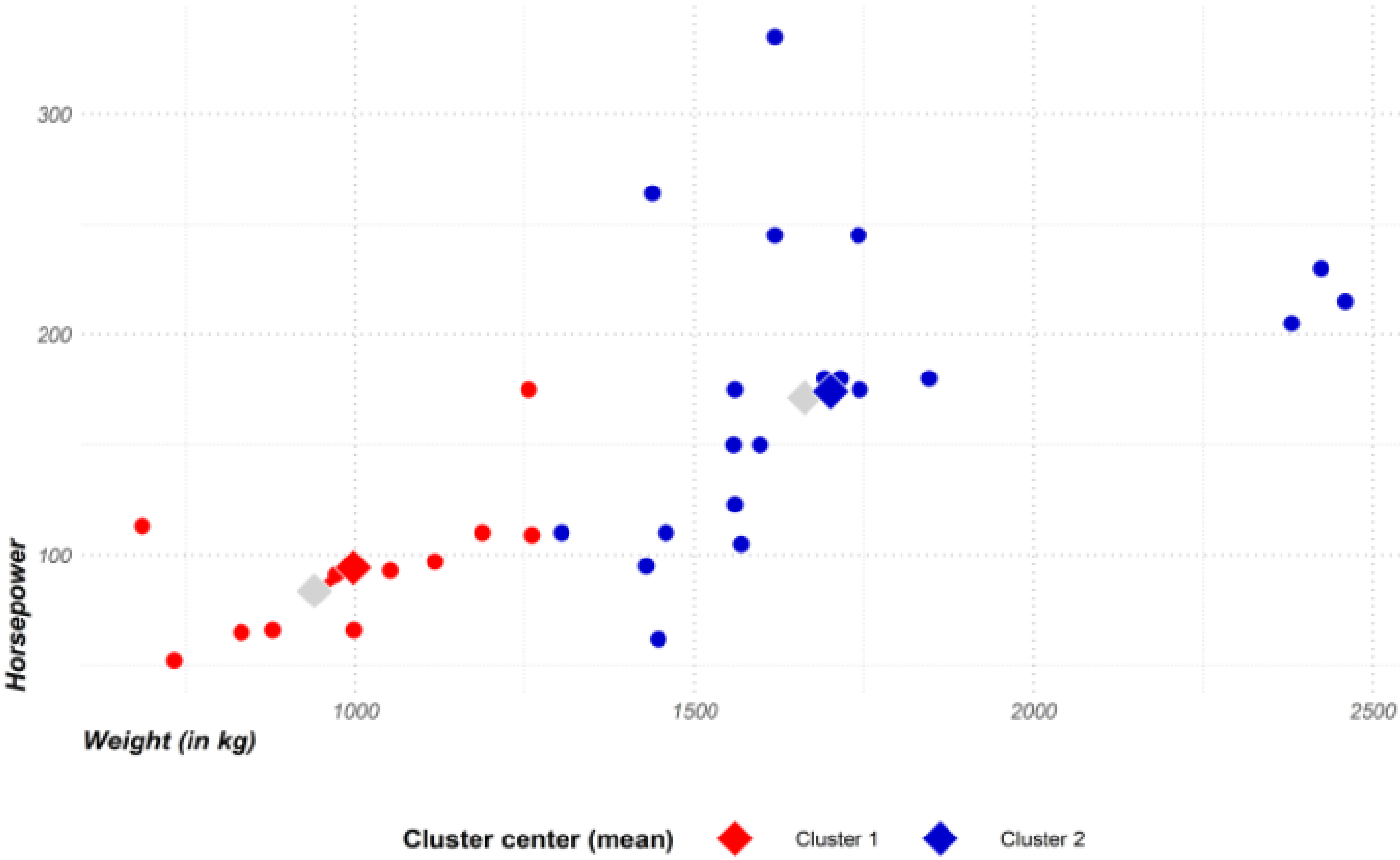

300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2



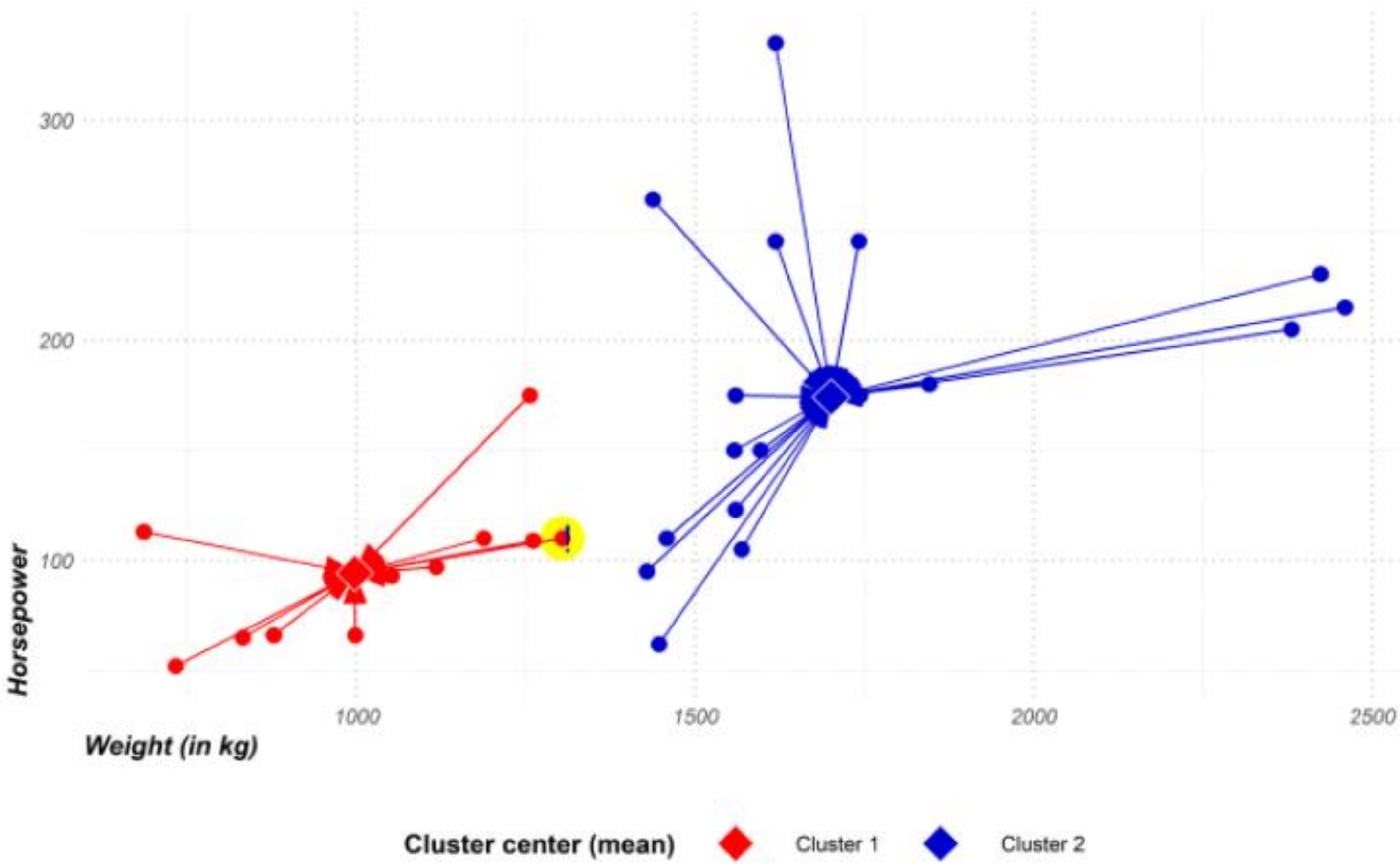
300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2

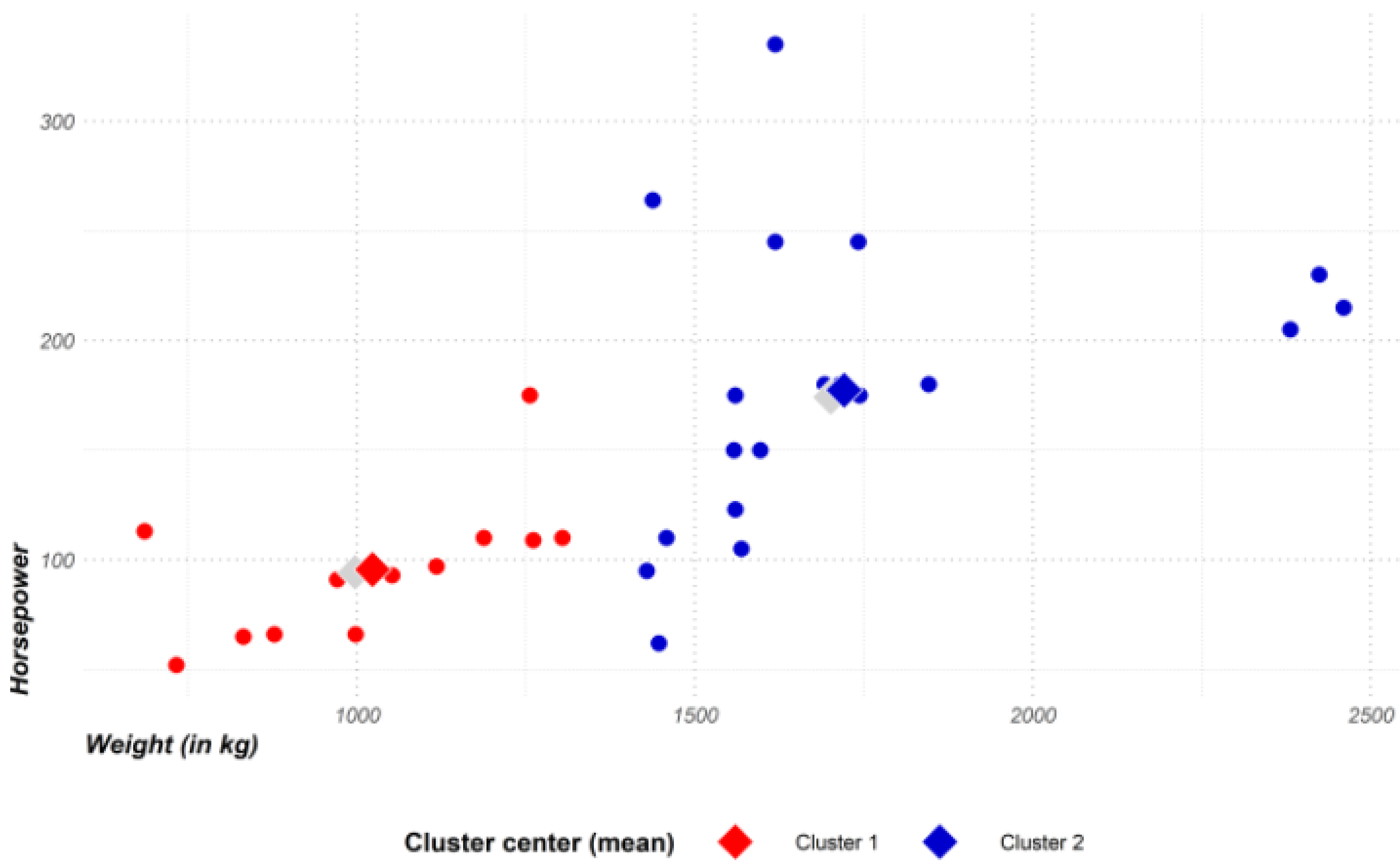


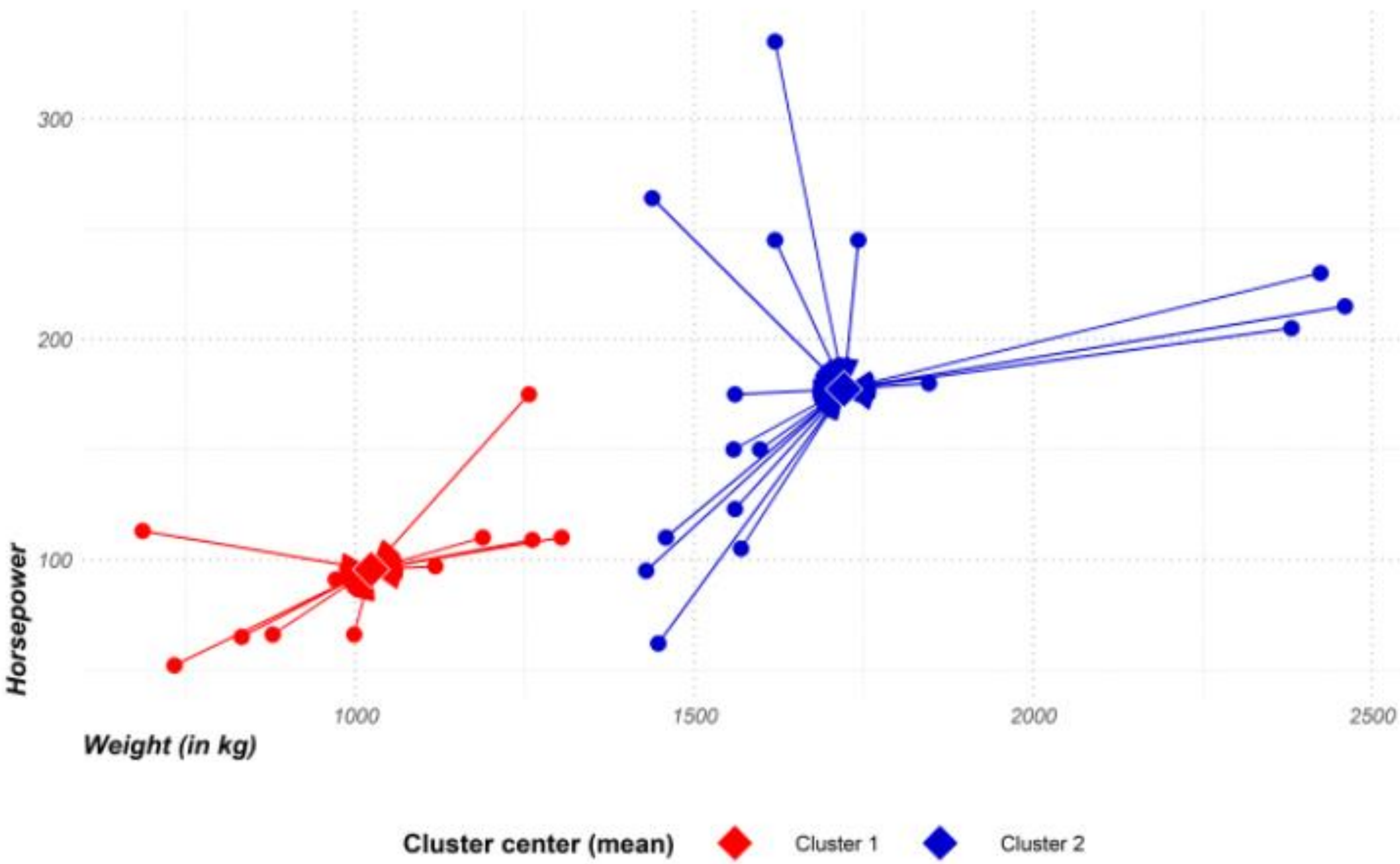
300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2



300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2


300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2


300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2



300
200
100
Horsepower
1000
1500
2000
2500
Weight (in kg)
Cluster center (mean)
Cluster 1
Cluster 2

# Comments on the *K-Means* Method

- **Strength:**
  - *Efficient*: $O(tkn)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations. Normally, $k$, $t$ << $n$.

- **Weakness**
  - Applicable only to objects in a continuous n-dimensional space
  - Need to specify *k,* the *number* of clusters, in advance (there are ways to determine the best k)
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes(concave Shape)*

# Variations of the *K-Means* Method

- Most of the variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: *k-modes*
  - Replacing means of clusters with <u>modes</u>
  - Using new dissimilarity measures to deal with categorical objects
  - Using a <u>frequency</u>-based method to update modes of clusters

# K-Medoids Method

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data

- **K-Medoids:** Instead of taking the **mean** value of the objects in a cluster as a cluster center, **medoids** can be used, which is the **most centrally located** object in a cluster

# The K-Medoids Clustering Method

**Algorithm 8.2:** KMedoids($D, K$, Dis) – $K$-medoids clustering using arbitrary distance metric Dis.

**Input** : data $D \subseteq \mathscr{X}$; number of clusters $K \in \mathbb{N}$;
distance metric $\text{Dis} : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$.

**Output** : $K$ medoids $\mu_1, \ldots, \mu_K \in D$, representing a predictive clustering of $\mathscr{X}$.

randomly pick $K$ data points $\mu_1, \ldots, \mu_K \in D$;

**repeat**

    assign each $\mathbf{x} \in D$ to $\arg\min_j \text{Dis}(\mathbf{x}, \mu_j)$;

    **for** $j = 1$ to $k$ **do**

        $D_j \leftarrow \{\mathbf{x} \in D | \mathbf{x} \text{ assigned to cluster } j\}$;

        $\mu_j = \arg\min_{\mathbf{x} \in D_j} \sum_{\mathbf{x}' \in D_j} \text{Dis}(\mathbf{x}, \mathbf{x}')$;

    **end**

**until** no change in $\mu_1, \ldots, \mu_K$;

**return** $\mu_1, \ldots, \mu_K$;

# Hierarchical Clustering

- While partitioning methods meet the basic clustering requirement of organizing a set of objects into a number of exclusive groups, in some situations **we may want to partition our data into groups at different levels such as in a hierarchy**.

- A **hierarchical clustering method** works by grouping data objects into a hierarchy or "tree" (called as Dendogram) of clusters.

**Definition 8.4 (Dendrogram).** *Given a data set D, a* dendrogram *is a binary tree with the elements of D at its leaves. An internal node of the tree represents the subset of elements in the leaves of the subtree rooted at that node. The level of a node is the distance between the two clusters represented by the children of the node. Leaves have level 0.*

# Hierarchical Clustering

- Use distance matrix as clustering criteria.

| | $g_1$ | $g_2$ | $g_3$ | $g_4$ | $g_5$ | $g_6$ | $g_7$ | $g_8$ | $g_9$ | $g_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $g_1$ | 0.0 | 8.1 | 9.2 | 7.7 | 9.3 | 2.3 | 5.1 | 10.2 | 6.1 | 7.0 |
| $g_2$ | 8.1 | 0.0 | 12.0 | 0.9 | 12.0 | 9.5 | 10.1 | 12.8 | 2.0 | 1.0 |
| $g_3$ | 9.2 | 12.0 | 0.0 | 11.2 | 0.7 | 11.1 | 8.1 | 1.1 | 10.5 | 11.5 |
| $g_4$ | 7.7 | 0.9 | 11.2 | 0.0 | 11.2 | 9.2 | 9.5 | 12.0 | 1.6 | 1.1 |
| $g_5$ | 9.3 | 12.0 | 0.7 | 11.2 | 0.0 | 11.2 | 8.5 | 1.0 | 10.6 | 11.6 |
| $g_6$ | 2.3 | 9.5 | 11.1 | 9.2 | 11.2 | 0.0 | 5.6 | 12.1 | 7.7 | 8.5 |
| $g_7$ | 5.1 | 10.1 | 8.1 | 9.5 | 8.5 | 5.6 | 0.0 | 9.1 | 8.3 | 9.3 |
| $g_8$ | 10.2 | 12.8 | 1.1 | 12.0 | 1.0 | 12.1 | 9.1 | 0.0 | 11.4 | 12.4 |
| $g_9$ | 6.1 | 2.0 | 10.5 | 1.6 | 10.6 | 7.7 | 8.3 | 11.4 | 0.0 | 1.1 |
| $g_{10}$ | 7.0 | 1.0 | 11.5 | 1.1 | 11.6 | 8.5 | 9.3 | 12.4 | 1.1 | 0.0 |

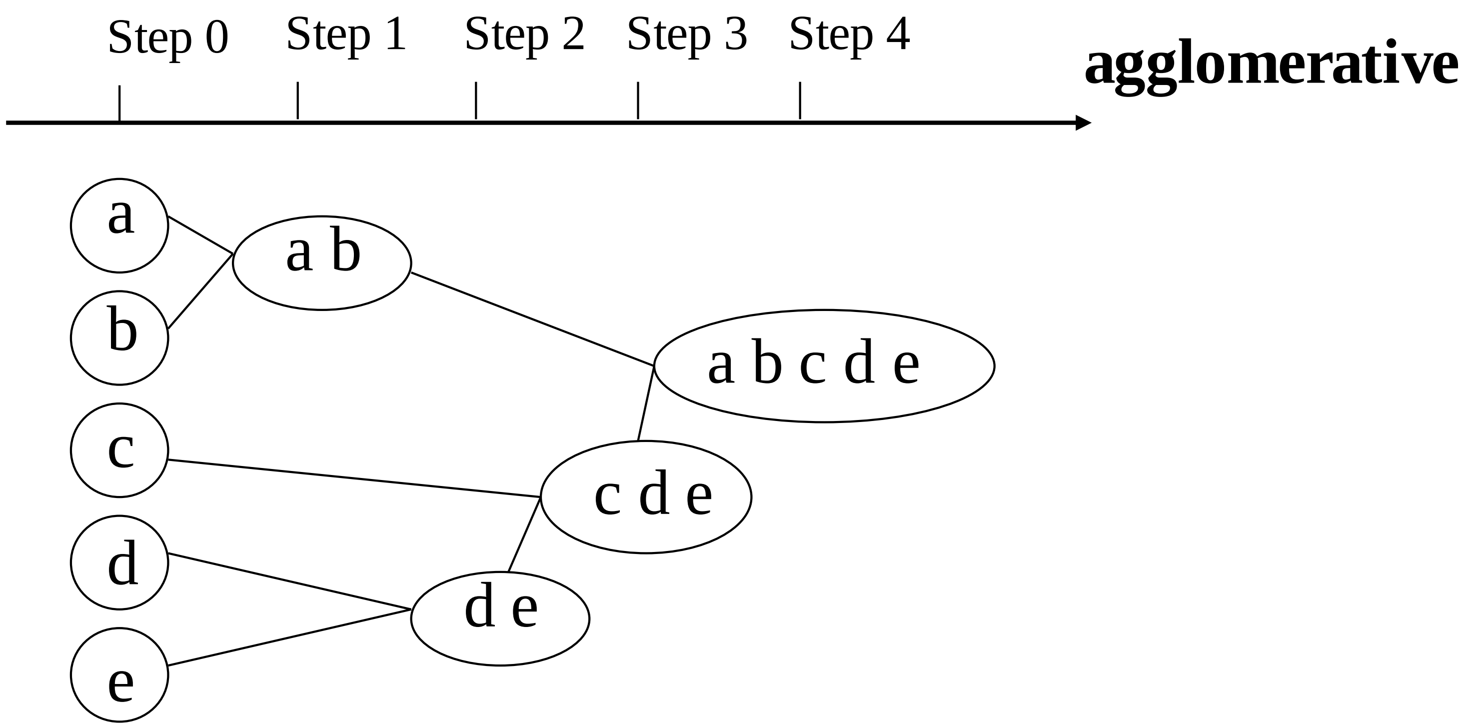
- This method does not require the number of clusters ***k*** as an input, but **needs a termination condition**

# Hierarchical Clustering

- The **most important point** in Hierarchical clustering methods is regarding the **selection of merge or split points.**
- Such a decision is critical, because once a group of objects is merged or split, the process at the next step will operate on the newly generated clusters.
- **It will neither undo what was done previously, nor perform object swapping between clusters.**
- Thus, merge or split decisions, if not well chosen, may lead to low-quality clusters

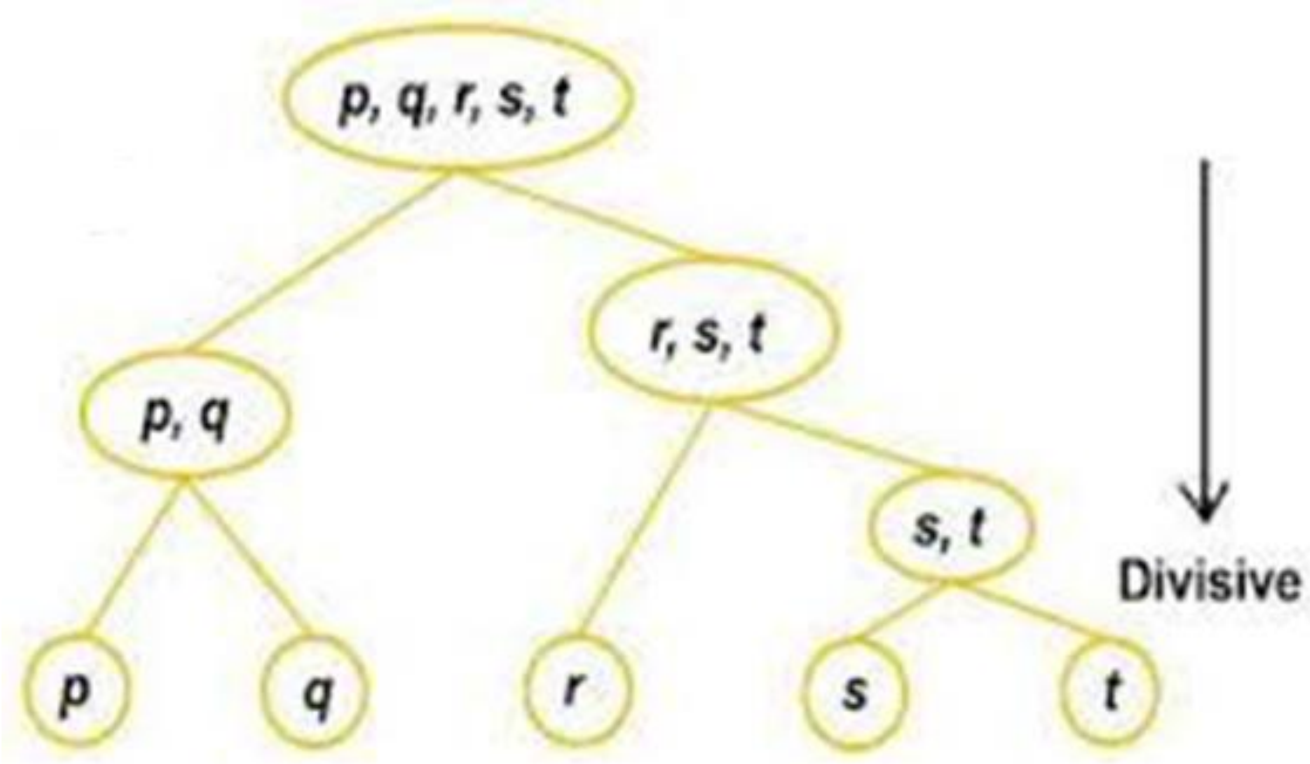
# Hierarchical Clustering

- Two types of hierarchical Clustering



# Hierarchical Clustering

- Two types of hierarchical Clustering



# Distance between Clusters

**Definition 8.5 (Linkage function).** *A* linkage function $L : 2^{\mathscr{X}} \times 2^{\mathscr{X}} \to \mathbb{R}$ *calculates the distance between arbitrary subsets of the instance space, given a distance metric* $\mathrm{Dis} : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$.

The most common linkage functions are as follows:

| | |
|---|---|
| Single linkage | defines the distance between two clusters as the *smallest* pairwise distance between elements from each cluster. |
| Complete linkage | defines the distance between two clusters as the *largest* pointwise distance. |
| Average linkage | defines the cluster distance as the *average* pointwise distance. |
| Centroid linkage | defines the cluster distance as the point distance between the cluster means. |

# Distance between Clusters

These linkage functions can be defined mathematically as follows:

$$L_{\text{single}}(A, B) = \min_{x \in A, y \in B} \text{Dis}(x, y)$$

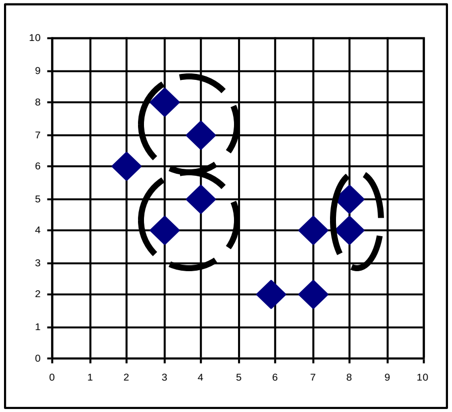
$$L_{\text{complete}}(A, B) = \max_{x \in A, y \in B} \text{Dis}(x, y)$$

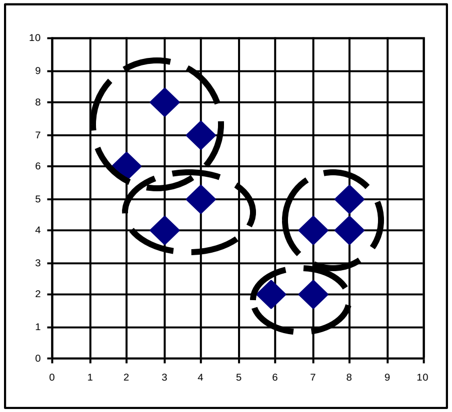
$$L_{\text{average}}(A, B) = \frac{\sum_{x \in A, y \in B} \text{Dis}(x, y)}{|A| \cdot |B|}$$

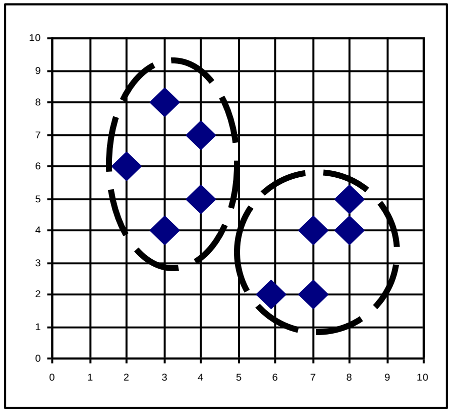
$$L_{\text{centroid}}(A, B) = \text{Dis}\left(\frac{\sum_{x \in A} x}{|A|}, \frac{\sum_{y \in B} y}{|B|}\right)$$

# AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw
- Implemented in statistical packages, e.g., Splus
- Use the **single-link** method and the dissimilarity matrix
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster

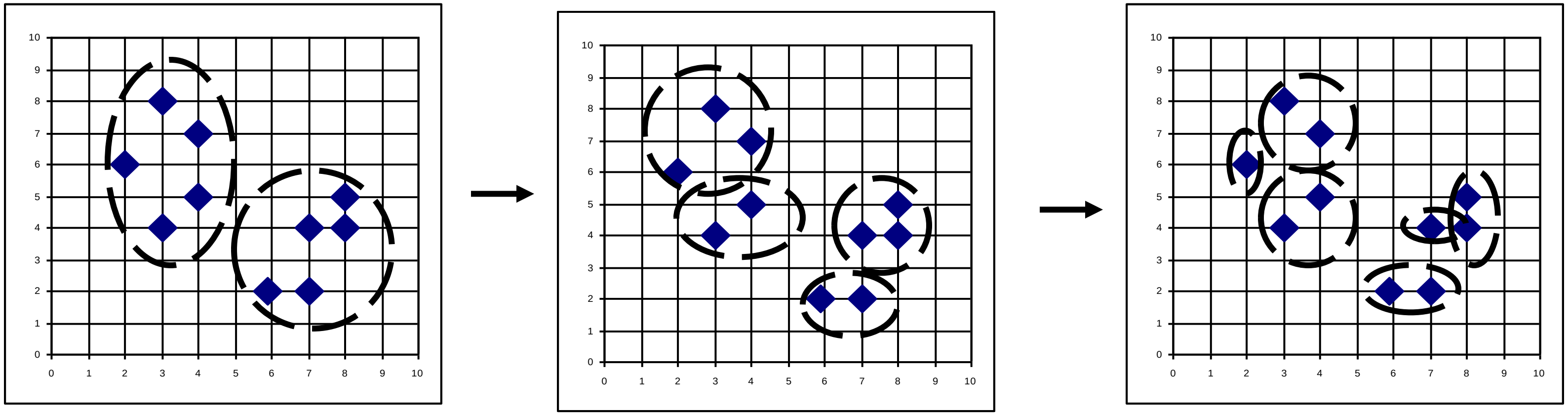


# DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



# (Agglomerative Clustering)

**Algorithm 8.4:** HAC($D, L$) – Hierarchical agglomerative clustering.

**Input** : data $D \subseteq \mathscr{X}$; linkage function $L : 2^{\mathscr{X}} \times 2^{\mathscr{X}} \to \mathbb{R}$ defined in terms of distance metric.

**Output** : a dendrogram representing a descriptive clustering of $D$.

```
initialise clusters to singleton data points;
create a leaf at level 0 for every singleton cluster;
repeat
    find the pair of clusters X, Y with lowest linkage l, and merge;
    create a parent of X, Y at level l;
until all data points are in one cluster;
return the constructed binary tree with linkage levels;
```

# Example for AGNES (Agglomerative Clustering)

- Assume that you have given a set of 6 data tuples or objects named A, B, C,D,E and F.

| | X1 | X2 |
|---|---|---|
| A | 1 | 1 |
| B | 1.5 | 1.5 |
| C | 5 | 5 |
| D | 3 | 4 |
| E | 4 | 4 |
| F | 3 | 3.5 |

- The following is the distance matrix fc

| Dist | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0.00 | 0.71 | 5.66 | 3.61 | 4.24 | 3.20 |
| B | 0.71 | 0.00 | 4.95 | 2.92 | 3.54 | 2.50 |
| C | 5.66 | 4.95 | 0.00 | 2.24 | 1.41 | 2.50 |
| D | 3.61 | 2.92 | 2.24 | 0.00 | 1.00 | 0.50 |
| E | 4.24 | 3.54 | 1.41 | 1.00 | 0.00 | 1.12 |
| F | 3.20 | 2.50 | 2.50 | 0.50 | 1.12 | 0.00 |

| Dist | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0.00 | 0.71 | 5.66 | 3.61 | 4.24 | 3.20 |
| B | 0.71 | 0.00 | 4.95 | 2.92 | 3.54 | 2.50 |
| C | 5.66 | 4.95 | 0.00 | 2.24 | 1.41 | 2.50 |
| D | 3.61 | 2.92 | 2.24 | 0.00 | 1.00 | 0.50 |
| E | 4.24 | 3.54 | 1.41 | 1.00 | 0.00 | 1.12 |
| F | 3.20 | 2.50 | 2.50 | 0.50 | 1.12 | 0.00 |

| Dist | A | B | C | D, F | E |
|---|---|---|---|---|---|
| A | 0.00 | 0.71 | 5.66 | ? | 4.24 |
| B | 0.71 | 0.00 | 4.95 | ? | 3.54 |
| C | 5.66 | 4.95 | 0.00 | ? | 1.41 |
| D, F | ? | ? | ? | 0.00 | ? |
| E | 4.24 | 3.54 | 1.41 | ? | 0.00 |

Using the input distance matrix, distance between cluster (D, F) and cluster A is computed as

$$d_{(D,F)\mapsto A} = \min\left(d_{DA}, d_{FA}\right) = \min\left(3.61, 3.20\right) = 3.20$$

| Dist | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0.00 | 0.71 | 5.66 | 3.61 | 4.24 | 3.20 |
| B | 0.71 | 0.00 | 4.95 | 2.92 | 3.54 | 2.50 |
| C | 5.66 | 4.95 | 0.00 | 2.24 | 1.41 | 2.50 |
| D | 3.61 | 2.92 | 2.24 | 0.00 | 1.00 | 0.50 |
| E | 4.24 | 3.54 | 1.41 | 1.00 | 0.00 | 1.12 |
| F | 3.20 | 2.50 | 2.50 | 0.50 | 1.12 | 0.00 |

Distance between cluster (D, F) and cluster B is

$$d_{(D,F)\mapsto B} = \min\left(d_{DB}, d_{FB}\right) = \min\left(2.92, 2.50\right) = 2.50$$

Similarly, distance between cluster (D, F) and cluster C is

$$d_{(D,F)\mapsto C} = \min\left(d_{DC}, d_{FC}\right) = \min\left(2.24, 2.50\right) = 2.24$$

Finally, distance between cluster E and cluster (D, F) is calculated as

$$d_{E\to(D,F)} = \min\left(d_{ED}, d_{EF}\right) = \min\left(1.00, 1.12\right) = 1.00$$

| Dist | A | B | C | D, F | E |
|---|---|---|---|---|---|
| A | 0.00 | 0.71 | 5.66 | **3.20** | 4.24 |
| B | 0.71 | 0.00 | 4.95 | **2.50** | 3.54 |
| C | 5.66 | 4.95 | 0.00 | **2.24** | 1.41 |
| D, F | **3.20** | **2.50** | **2.24** | 0.00 | **1.00** |
| E | 4.24 | 3.54 | 1.41 | **1.00** | 0.00 |

| Dist | A,B | C | (D, F) | E |
|---|---|---|---|---|
| A,B | 0 | ? | ? | ? |
| C | ? | 0 | 2.24 | 1.41 |
| (D, F) | ? | 2.24 | 0 | 1.00 |
| E | ? | 1.41 | 1.00 | 0 |

Using the input distance matrix (size 6 by 6), distance between cluster C and cluster (D, F) is computed as $d_{C\rightarrow(A,B)} = \min(d_{CA}, d_{CB}) = \min(5.66, 4.95) = 4.95$

Distance between cluster (D, F) and cluster (A, B) is the minimum distance between all objects involves in the two clusters

$$d_{(D,F)\rightarrow(A,B)} = \min(d_{DA}, d_{DB}, d_{FA}, d_{FB}) = \min(3.61, 2.92, 3.20, 2.50) = 2.50$$

Similarly, distance between cluster E and (A, B) is

$$d_{E\rightarrow(A,B)} = \min(d_{EA}, d_{EB}) = \min(4.24, 3.54) = 3.54$$

Then the updated distance matrix is

**Min Distance (Single Linkage)**

| Dist | A,B | C | (D, F) | E |
|---|---|---|---|---|
| A,B | 0 | 4.95 | 2.50 | 3.54 |
| C | **4.95** | 0 | 2.24 | 1.41 |
| (D, F) | **2.50** | 2.24 | 0 | 1.00 |
| E | **3.54** | 1.41 | 1.00 | 0 |

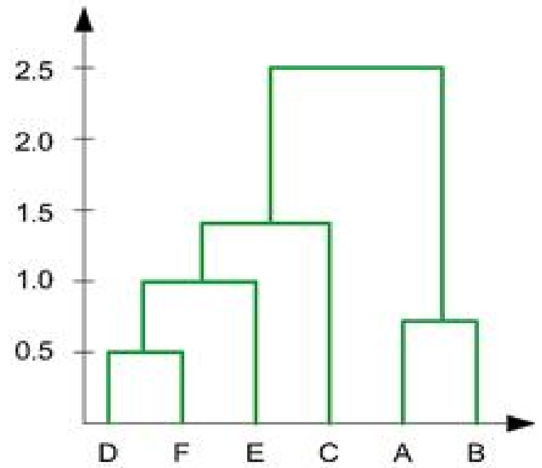
## Min Distance (Single Linkage)

| Dist | (A,B) | C | (D, F), E |
|---|---|---|---|
| (A,B) | 0.00 | 4.95 | 2.50 |
| C | 4.95 | 0.00 | 1.41 |
| (D, F), E | **2.50** | **1.41** | 0.00 |

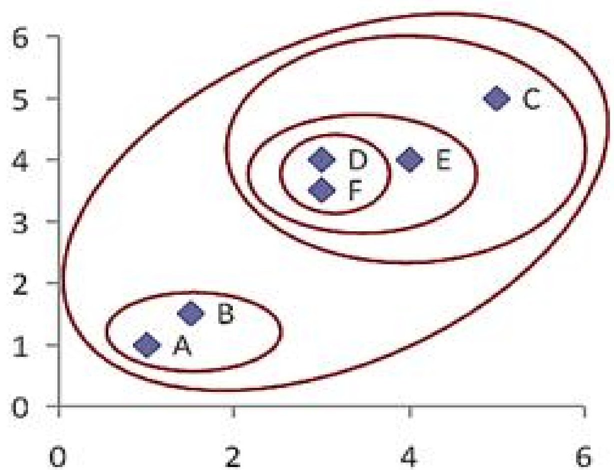
## Min Distance (Single Linkage)

| Dist | (A,B) | (D, F), E),C |
|---|---|---|
| (A,B) | 0.00 | 2.50 |
| ((D, F), E),C | **2.50** | 0.00 |

# Dendogram Representation

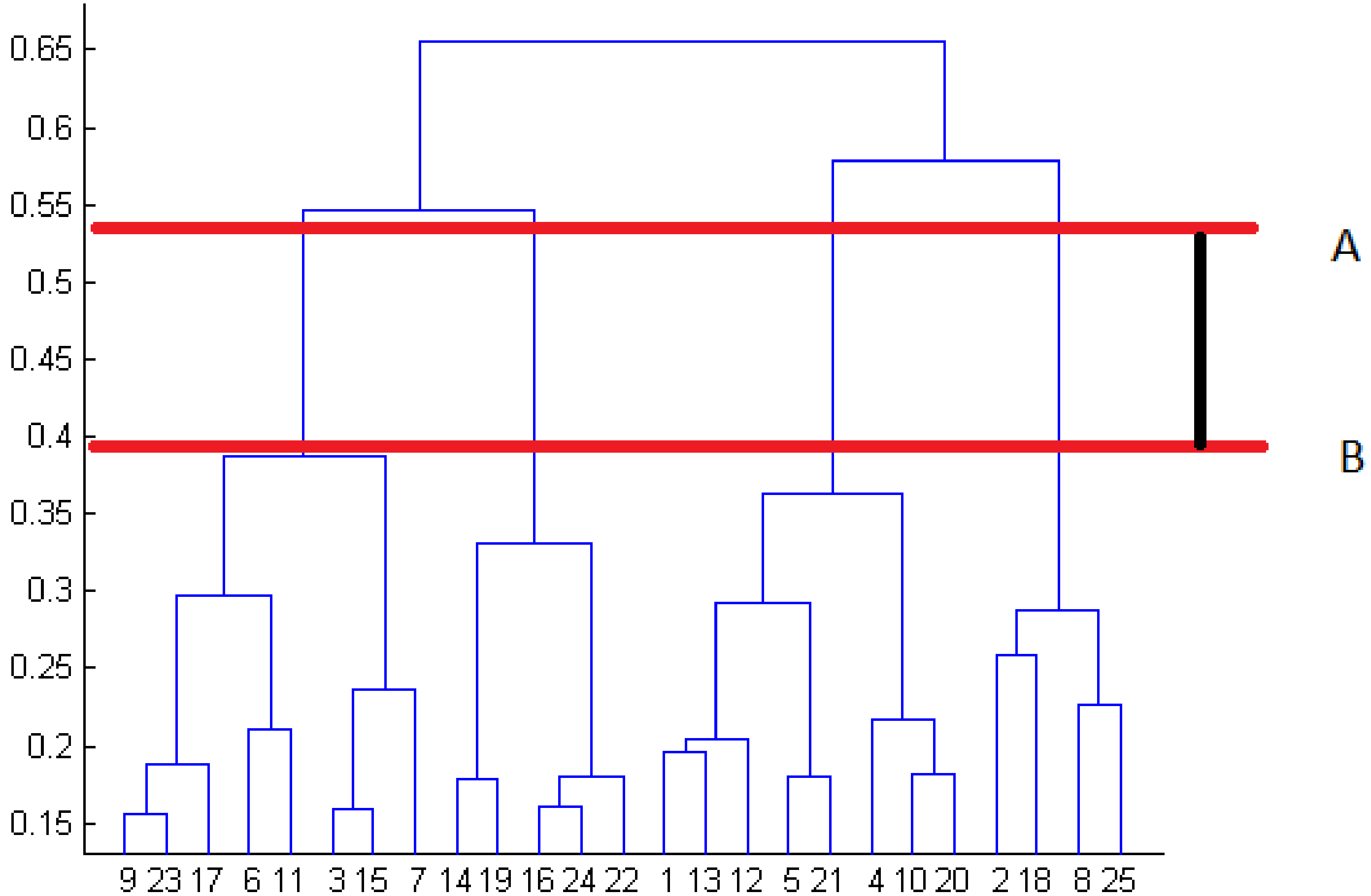



The hierarchy is given as (((D, F), E),C), (A,B). We can also plot the clustering hierarchy into XY space




| | X1 | X2 |
|---|---|---|
| A | 1 | 1 |
| B | 1.5 | 1.5 |
| C | 5 | 5 |
| D | 3 | 4 |
| E | 4 | 4 |
| F | 3 | 3.5 |

# Deciding the Best Number of Clusters



# Example for Divisive Hierarchical Clustering

## Divisive Clustering Example

The following is an example of Divisive Clustering.

| Distance | a | b | c | d | e |
|---|---|---|---|---|---|
| a | 0 | 2 | 6 | 10 | 9 |
| b | 2 | 0 | 5 | 9 | 8 |
| c | 6 | 5 | 0 | 4 | 5 |
| d | 10 | 9 | 4 | 0 | 3 |
| e | 9 | 8 | 5 | 3 | 0 |

**Step 1.** Split whole data into 2 clusters
Which is dissimilar more with other members? (in Average)
a to others: mean(2,6,10,9) =6.75
b to others: mean(2,5,9,8) =6.0
c to others: mean(6,5,4,5) =5.0
d to others: mean(10,9,4,3) =6.5
e to others: mean(9,8,5,3) =6.25

**Step 2:** Because a has more dissimilar to others split a into separate cluster.
Recheck the remaining objects

| | $\alpha$ =distance to the old party | $\beta$ =distance to the new party |
|---|---|---|
| b | $\frac{5+9+8}{3} = 7.33$ | 2 |
| c | $\frac{5+4+5}{3} = 4.67$ | 6 |
| d | $\frac{9+4+3}{3} = 5.33$ | 10 |
| e | $\frac{8+5+3}{3} = 5.33$ | 9 |

# Example for Divisive Hierarchical Clustering

Cluster 1: {a,b}
Cluster 2: {c,d,e}

**Step 3:** Choose a current cluster and split it as in Step 1.
split the cluster with the largest number of members
split the cluster with the largest diameter

| cluster | diameter |
|---|---|
| {a,b} | 2 |
| {c,d,e} | 5 |

Split the chosen cluster as in Step 1.

# Example for Practice

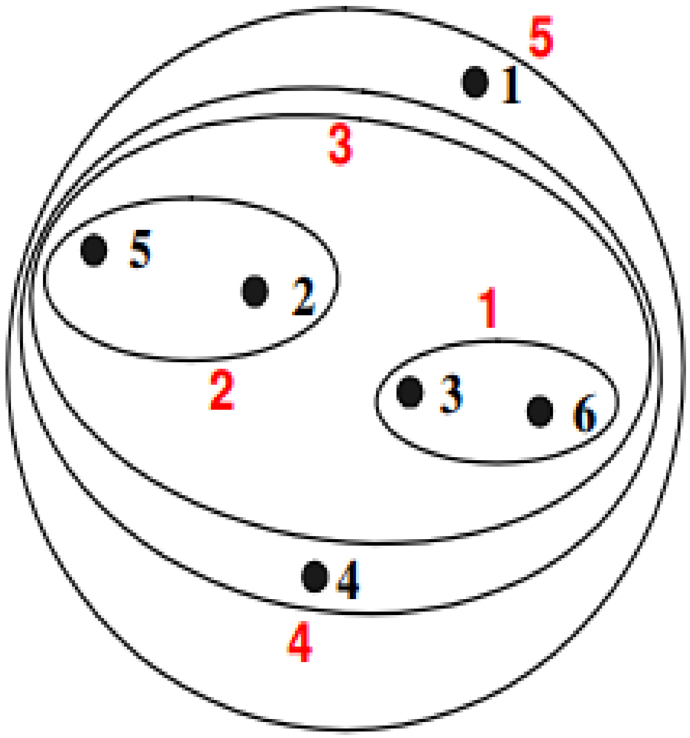
| point | x coordinate | y coordinate |
|---|---|---|
| p1 | 0.4005 | 0.5306 |
| p2 | 0.2148 | 0.3854 |
| p3 | 0.3457 | 0.3156 |
| p4 | 0.2652 | 0.1875 |
| p5 | 0.0789 | 0.4139 |
| p6 | 0.4548 | 0.3022 |

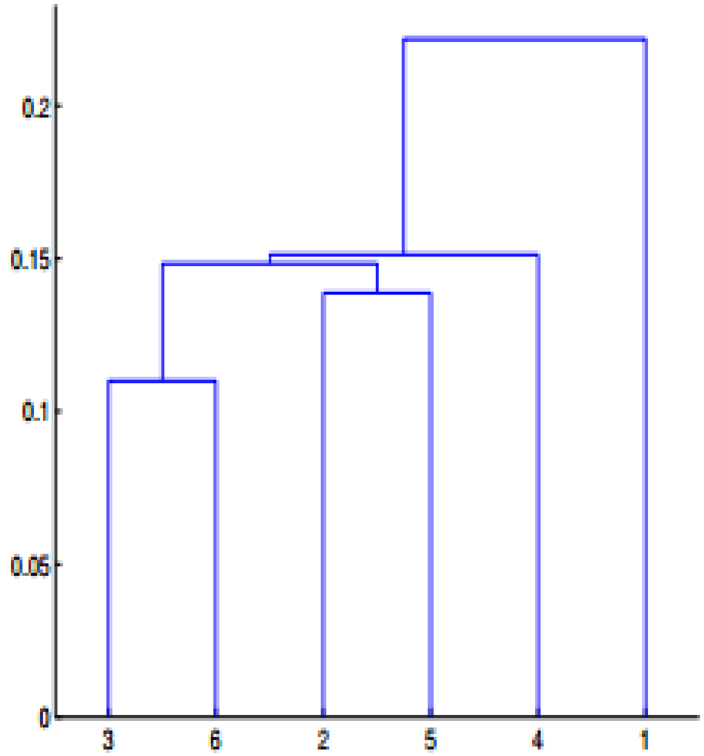
**Table** : X-Y coordinates of six points.

| | p1 | p2 | p3 | p4 | p5 | p6 |
|---|---|---|---|---|---|---|
| p1 | 0.0000 | 0.2357 | 0.2218 | 0.3688 | 0.3421 | 0.2347 |
| p2 | 0.2357 | 0.0000 | 0.1483 | 0.2042 | 0.1388 | 0.2540 |
| p3 | 0.2218 | 0.1483 | 0.0000 | 0.1513 | 0.2843 | 0.1100 |
| p4 | 0.3688 | 0.2042 | 0.1513 | 0.0000 | 0.2932 | 0.2216 |
| p5 | 0.3421 | 0.1388 | 0.2843 | 0.2932 | 0.0000 | 0.3921 |
| p6 | 0.2347 | 0.2540 | 0.1100 | 0.2216 | 0.3921 | 0.0000 |

**Table** : Distance Matrix for Six Points

# Answer for Example 2



# Topics

- Introduction
- Nearest Neighbor Classification
- Distance based Clustering
  - Partitioning Clustering
    - K-Means algorithm
    - Clustering around medoids
  - Hierarchical Clustering
    - Agnes
    - Diana


Thank You