

The background features a complex network of blue lines and arrows. Some lines are solid, while others are dashed. The arrows point in various directions, creating a sense of movement and connectivity. The overall design is modern and technical, fitting the theme of data analysis.

# STATISTICA E ANALISI DEI DATI

## Capitolo 7 – Clustering Non Gerarchico

---

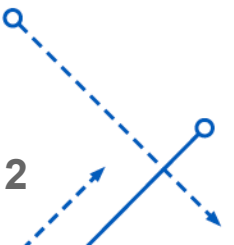
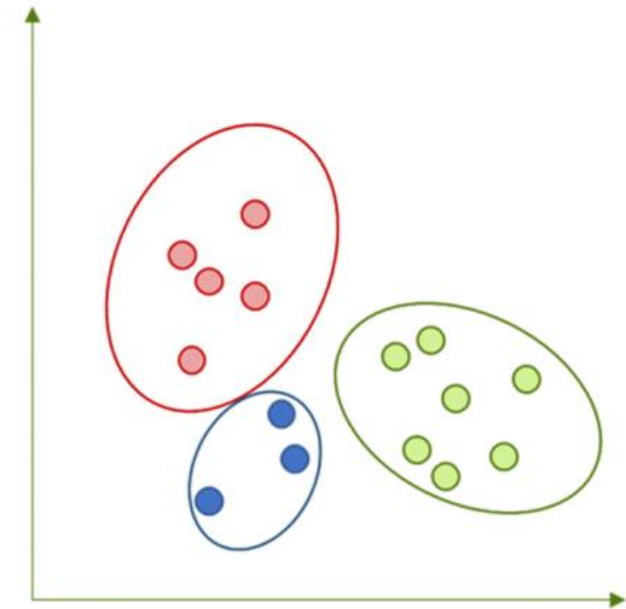
Dott. Stefano Cirillo  
Dott. Luigi Di Biasi

a.a. 2025-2026

# Clustering Non-Gerarchico

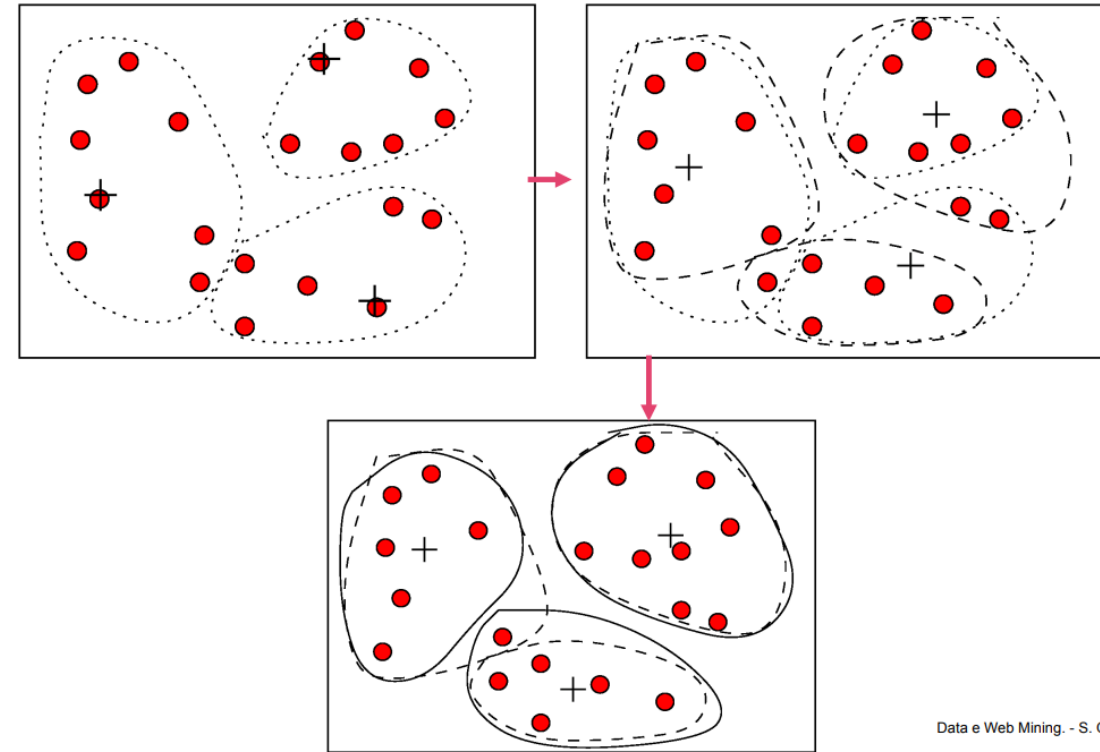
- I **metodi non-gerarchici di clustering** l'obiettivo finale dei metodi non gerarchici è quello di ottenere un'unica partizione degli  $n$  individui di partenza in cluster
- Svantaggi:
  - La scelta a priori del numero di cluster
  - La scelta a priori di parametri per la determinazione automatica del loro numero
- Vantaggio:
  - non è possibile riallocare gli individui che sono stati già classificati ad un livello precedente dell'analisi

Non-hierarchical



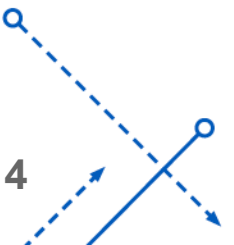
# K-Means

- Le fasi del K-means sono:
  - **Inizializzazione**: assegna in modo casuale le posizioni iniziali dei  $k$  cluster
  - **Assegnazione**: per ogni "sample" di input, scopri quale cluster è più vicino ad esso e assegnalo a quel cluster
  - **Aggiornamento**: per ogni cluster, imposta la sua nuova posizione sulla base della media di tutti i campioni di input ad esso assegnati
  - **Iterazione**: ripetere i passaggi 2 e 3 finché le posizioni dei cluster non cambiano.



# K-Means

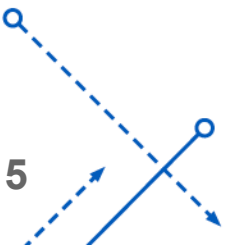
- Il metodo più utilizzato prende il nome di **k-means** ed è dovuto a *Hartigan e Wong*
- **Step 1:**
  - Fissare a priori il numero  $k$  di cluster specificando i punti di riferimento iniziali (scegliendo in maniera opportuna alcuni individui, o unità, o prendendo la configurazione determinata con una tecnica gerarchica) che inducono una prima partizione provvisoria;
- **Step 2:**
  - Considerare tutti gli individui e attribuire ciascuno di essi al cluster individuato dal punto di riferimento da cui ha distanza minore;
- **Step 3:**
  - Calcolare il centroide di ognuno dei  $k$  gruppi così ottenuti. Tali centroidi costituiscono i punti di riferimento per i nuovi cluster;



# K-Means

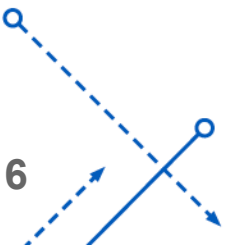
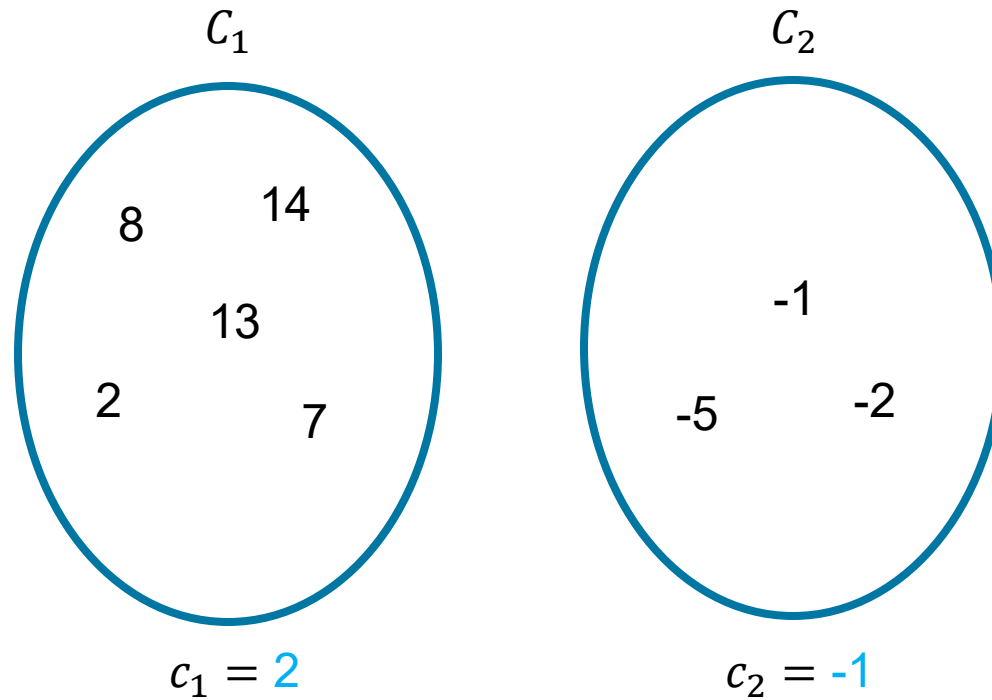
---

- **Step 4:**
  - Valutare la distanza di ogni unità da ogni centroide ottenuto al passo precedente. Se la distanza minima non è ottenuta in corrispondenza del centroide del gruppo di appartenenza, allora si procede a spostare l'individuo presso il cluster che ha il centroide più vicino
- **Step 5:**
  - Ricalcolare i centroidi dei  $k$  gruppi così ottenuti.
- **Step 6:**
  - Ripetere il procedimento a partire dal punto (4) fino a che i centroidi non subiscono ulteriori modifiche rispetto all'iterazione precedente
  - Si procede così iterativamente a spostamenti successivi fino a raggiungere una configurazione stabile, ossia gli individui all'interno di ogni cluster non cambiano al ripetersi del procedimento



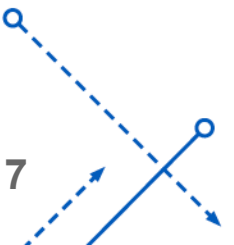
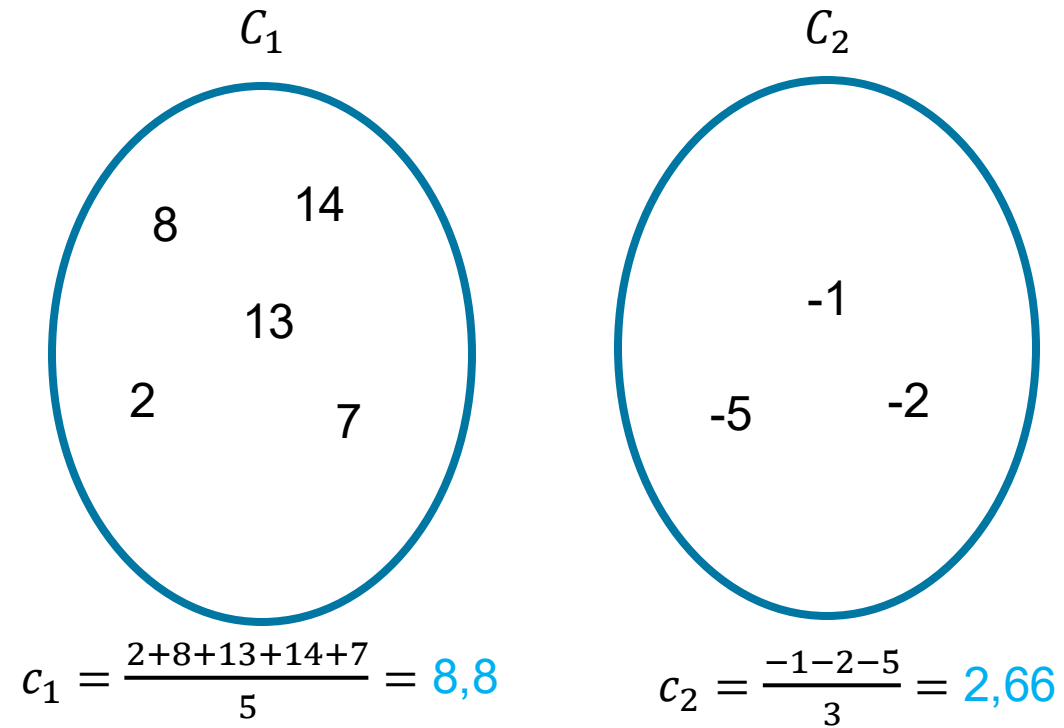
# K-Means – Iterazione 1

- X: 2, -1, 13, -5, 8, -2, 7, 14
- k: 2



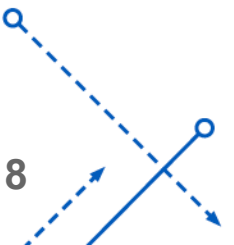
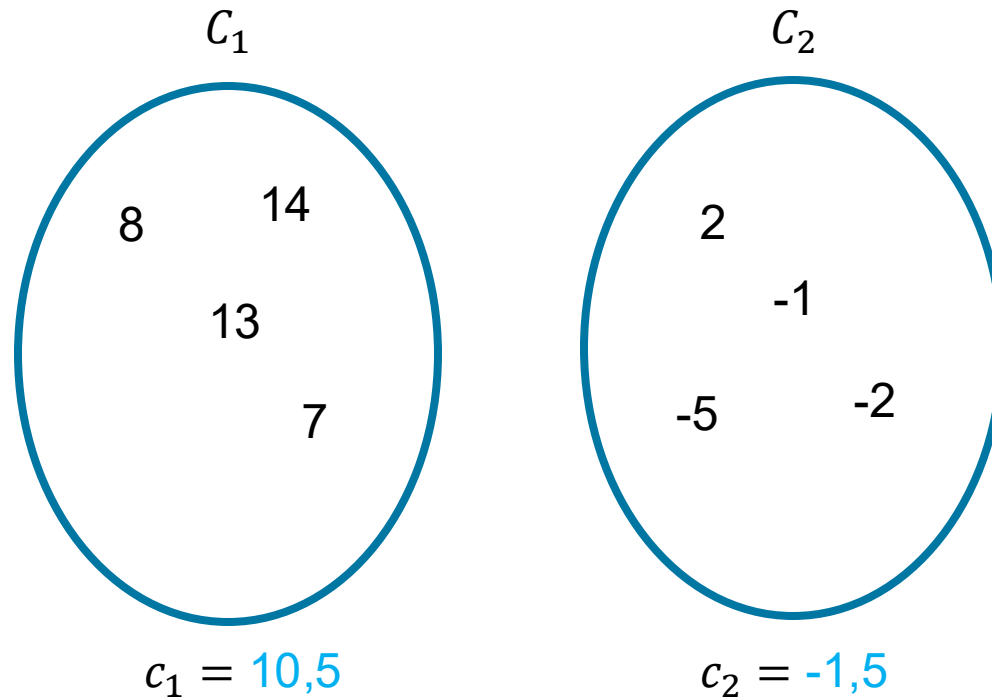
# K-Means – Iterazione 2

- X: 2, -1, 13, -5, 8, -2, 7, 14
- k: 2



# K-Means – Iterazione 3

- X: 2, -1, 13, -5, 8, -2, 7, 14
- k: 2





# K-Means (R)

- L'analisi con il metodo K-Means si effettua in R mediante la funzione

**kmeans**(X, centers, iter.max = N, nstart = M)

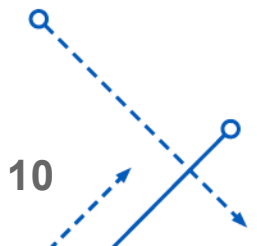
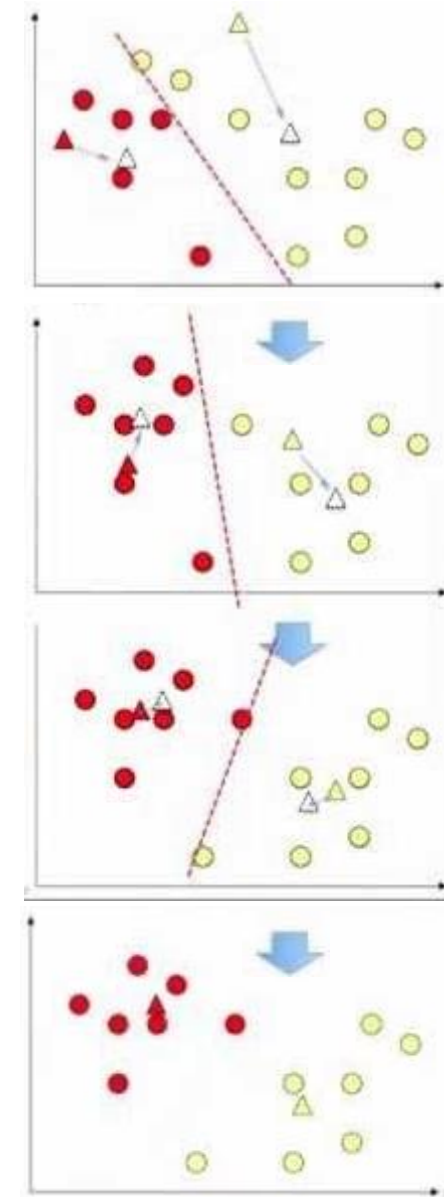
dove

- **X** è la matrice dei dati;
- **centers** è il numero dei cluster che si vogliono identificare o un vettore di lunghezza pari al numero di cluster contenente un insieme di centroidi iniziali dei cluster.
  - Nel primo caso, ossia se è numero intero, l'algoritmo sceglie casualmente i punti di riferimento e tale insieme è utilizzato per individuare la partizione iniziale
  - Nel secondo caso, i centroidi iniziali possono essere derivati effettuando preliminarmente un'analisi di tipo gerarchico con il metodo del centroide
- **iter.max** è il massimo numero di iterazioni permesse
- **nstart** fornisce il numero di volte in cui ripetere la procedura di scelta casuale dei punti di riferimento, nel caso in cui centers è il numero



# K-Means (R)

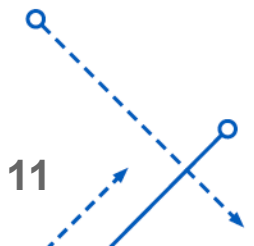
- **Nota:** Si nota che nell'algoritmo K-Means non occorre calcolare la matrice iniziale delle distanze (o dei quadrati delle distanze) così come invece si richiede nei metodi gerarchici
- La funzione **kmeans()** produce come output una lista, i cui elementi sono:
  - un vettore di interi che indica il cluster di allocazione di ogni individuo (**\$cluster**)
  - una matrice che contiene i centroidi dei cluster (**\$center**)
  - un vettore contenente le misure di non omogeneità statistica calcolate all'interno di ognuno dei cluster; tali valori dipendono dall'omogeneità interna e dalla numerosità del gruppo (**\$withinss**)
  - dimensione dei gruppi (**\$size**)
- La misura di non omogeneità statistica complessiva all'interno dei vari cluster (*within*) è quindi la somma delle misure di non omogeneità statistica di ognuno dei cluster



# Esempio K-Means

- Consideriamo la seguente matrice contenente due caratteristiche  $C_1$  e  $C_2$  osservate per 8 individui  $I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8$
- In base all'analisi di tipo gerarchico si è giunti alla conclusione che un numero ragionevole di cluster è 2
  - Usiamo 2 come scelta del numero di cluster da considerare all'inizio dell'analisi con il metodo K-Means
- Consideriamo tre differenti scelte iniziali:
  - i) Scelta casuale dei punti di riferimento;
  - ii) Ripetizione della procedura di scelta casuale dei punti di riferimento;
  - iii) Scelta dei centroidi come punti di riferimento.

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$



# Esempio K-Means

- Scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> km<-kmeans(X,center=2,iter.max=10,nstart=1)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2
1	1.0	1
2	5.2	4

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

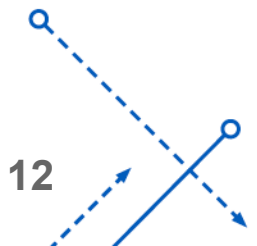
Within cluster sum of squares by cluster:

```
[1] 4.0 10.8
(between_SS / total_SS = 77.1 %)
```

Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "
    tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$



# Esempio K-Means (i)

- Scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> km<-kmeans(X,center=2,iter.max=10,nstart=1)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2	
1	1.0	1	Coordinate centroide $G_1$
2	5.2	4	Coordinate centroide $G_2$

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

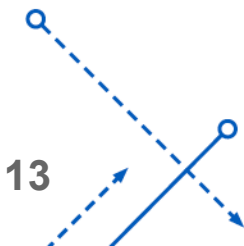
Within cluster sum of squares by cluster:

```
[1] 4.0 10.8
(between_SS / total_SS = 77.1 %)
```

Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "
    tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$



# Esempio K-Means (i)

- Scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> km<-kmeans(X,center=2,iter.max=10,nstart=1)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2
1	1.0	1
2	5.2	4

Coordinate centroide  $G_1$

Coordinate centroide  $G_2$

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

$G_1 = \{I_1, I_2, I_3\}$

$G_2 = \{I_4, I_5, I_6, I_7, I_8\}$

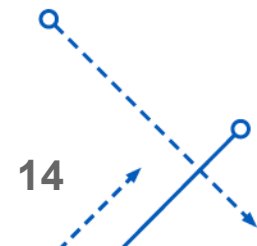
Within cluster sum of squares by cluster:

```
[1] 4.0 10.8
(between_SS / total_SS = 77.1 %)
```

Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "
    tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$



# Esempio K-Means (i)

- Scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> km<-kmeans(X,center=2,iter.max=10,nstart=1)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2
1	1.0	1
2	5.2	4

Coordinate centroide  $G_1$

Coordinate centroide  $G_2$

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

$G_1 = \{I_1, I_2, I_3\}$

$G_2 = \{I_4, I_5, I_6, I_7, I_8\}$

Within cluster sum of squares by cluster:

[1] 4.0 10.8

(between\_SS / total\_SS = 77.1 %)

Misure di non omogeneità statistica associate ai cluster

Available components:

[1]	"cluster"	"centers"	"totss"	"withinss"	"
	tot.withinss	"betweenss"	"size"		
[8]	"iter"	"ifault"			

$$\frac{\text{tr } B}{\text{tr } T} = \frac{49.95}{64.75} = 0.7714$$

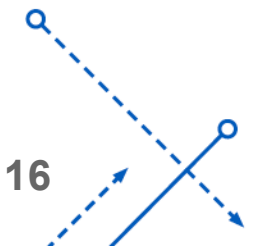
- Dove la misura di non omogeneità totale è 64.75 (dall'analisi precedente)
- La misura di non omogeneità tra i cluster è  $64.75 - 14.8 = 49.95$

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$

# Esempio K-Means (i)

- L'oggetto restituito dalla funzione *kmeans* contiene i seguenti valori:

```
> km$cluster
I1 I2 I3 I4 I5 I6 I7 I8 → Partizioni cluster
 1  1  1  2  2  2  2  2
>
> km$centers
  c1 c2
1 1.0  1 → Coordinate centroide  $G_1$ 
2 5.2  4 → Coordinate centroide  $G_2$ 
>
> km$totss
[1] 64.75 → misura di non omogeneità totale è 64.75
>
> km$withinss
[1]  4.0 10.8 → Misure di non omogeneità statistica associate ai cluster  $G_1$  e  $G_2$ 
>
> km$tot.withinss
[1] 14.8 → Misura di non omogeneità interne ai cluster (within)
>
> km$betweenss
[1] 49.95 → Misura di non omogeneità tra i cluster (between)
>
> km$size
[1] 3 5 → Dimensioni Partizioni/cluster
```





# Esempio K-Means (ii)

- Ripetizione della procedura di scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> kmeans(X, centers=2,iter.max=10, nstart=8)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2
1	1.0	1
2	5.2	4

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

Within cluster sum of squares by cluster:

```
[1] 4.0 10.8
(between_SS / total_SS = 77.1 %)
```

Available components:

[1]	"cluster"	"centers"	"totss"	"withinss"	"
	tot.withinss	"betweenss"	"size"		
[8]	"iter"	"ifault"			

**nstart** fornisce il numero di volte in cui ripetere la procedura di scelta casuale dei centroidi

Perché servono più avvii?

- K-Means non garantisce di trovare la soluzione ottimale globale.
- Dipende da **come vengono scelti casualmente i centroidi iniziali**

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$

# Esempio K-Means (ii)

## Ripetizione della procedura di scelta casuale dei punti di riferimento

```
> X<-data.frame(c1=c(0,1,2,4,5,4,6,7),c2=c(0,1,2,3,3,4,5,5))
> row.names(X)<-c("I1","I2","I3","I4","I5","I6","I7","I8")
>
> kmeans(X, centers=2,iter.max=10, nstart=8)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

c1 c2

1	1.0	1
2	5.2	4

Coordinate centroide  $G_1$

Coordinate centroide  $G_2$

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8
1	1	1	2	2	2	2	2

$$G_1 = \{I_1, I_2, I_3\}$$

$$G_2 = \{I_4, I_5, I_6, I_7, I_8\}$$

Within cluster sum of squares by cluster:

[1] 4.0 10.8

(between\_SS / total\_SS = 77.1 %)

Misure di non omogeneità statistica associate ai cluster

$$\frac{\text{tr } B}{\text{tr } T} = \frac{49.95}{64.75} = 0.7714$$

Available components:

[1]	"cluster"	"centers"	"totss"	"withinss"	"
	tot.withinss	"betweenss"	"size"		
[8]	"iter"	"ifault"			

Perché servono più avvii?

- K-Means non garantisce di trovare la soluzione ottimale globale.
- Dipende da **come vengono scelti casualmente i centroidi iniziali**

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$

- Dove la misura di non omogeneità totale è 64.75 (dall'analisi precedente)
- La misura di non omogeneità tra i cluster è  $64.75 - 14.8 = 49.95$

# Esempio K-Means (iii)

- Scelta dei centroidi come punti di riferimento

- Consideriamo:

- la matrice delle distanze euclidee
- la matrice  $d^2$  contenente i quadrati delle distanze euclidee

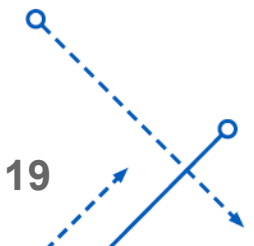
$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$

```
> d<-dist(X,method="euclidean",diag=TRUE,upper=TRUE)
> d2<-d^2
```

- **Utilizziamo i centroidi dei due cluster ottenuti** con tecnica gerarchica del centroide utilizzando la funzione *aggregate()*

```
> tree <- hclust(d2, method = "centroid")
>
> taglio<-cutree(tree, k =2, h =NULL)
> tagliolist<-list(taglio)
> centroidiIniziali<-aggregate(X, tagliolist, mean)[,-1]
> centroidiIniziali # visualizza i centroidi iniziali
   c1 c2
1 1.0  1
2 5.2  4
```

- **Nota:** Possiamo utilizzare centroidi generati randomicamente o secondo altre strategie



# Esempio K-Means (iii)

- Applichiamo il metodo K-Means con i centroidi definiti:

```
> km<-kmeans(X, centers = centroidiIniziali, iter.max = 10)
> km # visualizza i risultati ottenuti con kmeans
K-means clustering with 2 clusters of sizes 3, 5
```

Cluster means:

	c1	c2	
1	1.0	1	Coordinate centroide $G_1$
2	5.2	4	Coordinate centroide $G_2$

Clustering vector:

I1	I2	I3	I4	I5	I6	I7	I8	
1	1	1	2	2	2	2	2	$G_1 = \{I_1, I_2, I_3\}$ $G_2 = \{I_4, I_5, I_6, I_7, I_8\}$

Within cluster sum of squares by cluster:

```
[1] 4.0 10.8
(between_SS / total_SS = 77.1 %)
```

Misure di non omogeneità statistica associate ai cluster

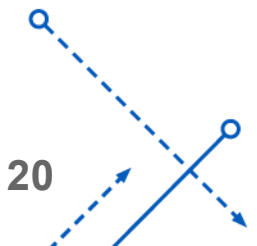
Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "
    tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

$$\frac{\text{tr } B}{\text{tr } T} = \frac{49.95}{64.75} = 0.7714$$

- Dove la misura di non omogeneità totale è 64.75 (dall'analisi precedente)
- La misura di non omogeneità tra i cluster è  $64.75 - 14.8 = 49.95$

$$X = \begin{matrix} & C_1 & C_2 \\ \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \\ I_7 \\ I_8 \end{matrix} & \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \\ 4 & 3 \\ 5 & 3 \\ 4 & 4 \\ 6 & 5 \\ 7 & 5 \end{pmatrix} \end{matrix}$$



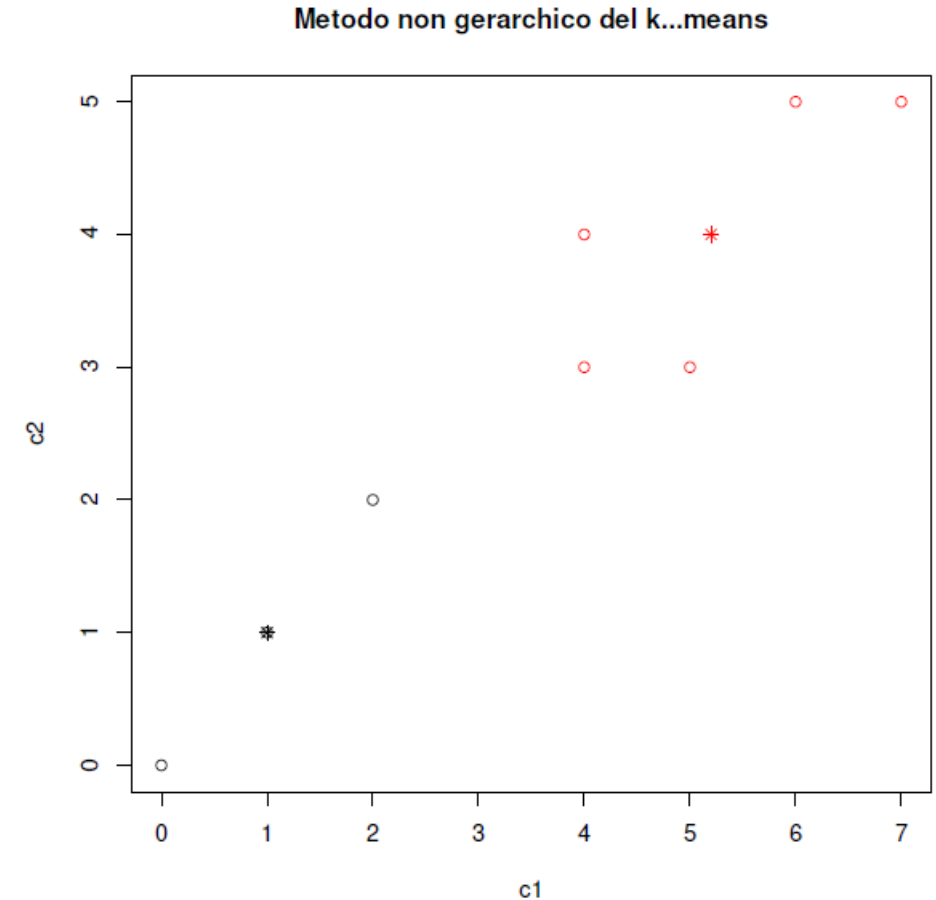
# Esempio K-Means (iii)

- Rappresentiamo i cluster generati dal metodo K-Means:

```
> plot(X, col = km$cluster, main = "Metodo non gerarchico del k-  
means")  
> points(km$center, col = 1:2, pch = 8, cex=1)
```

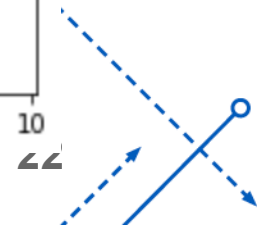
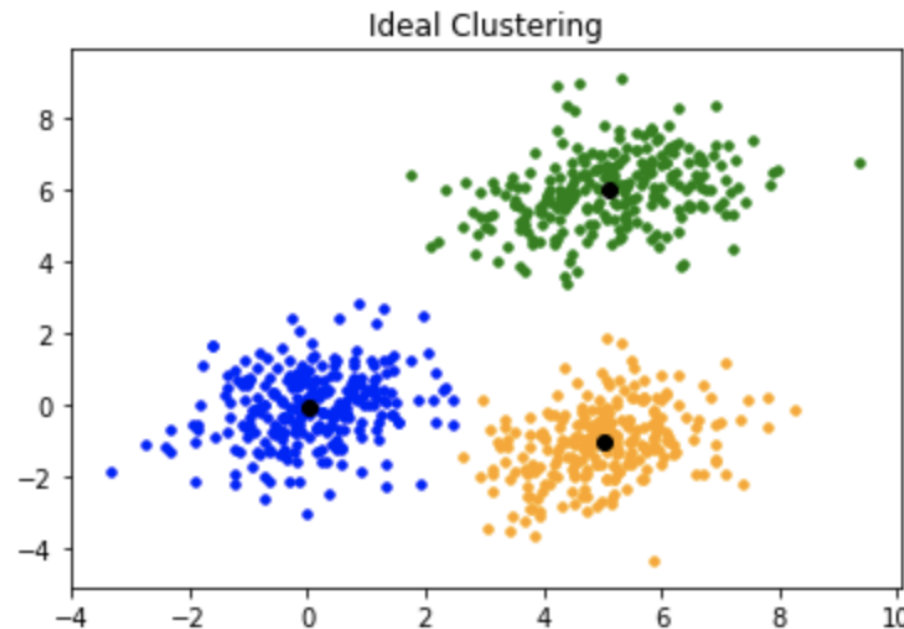
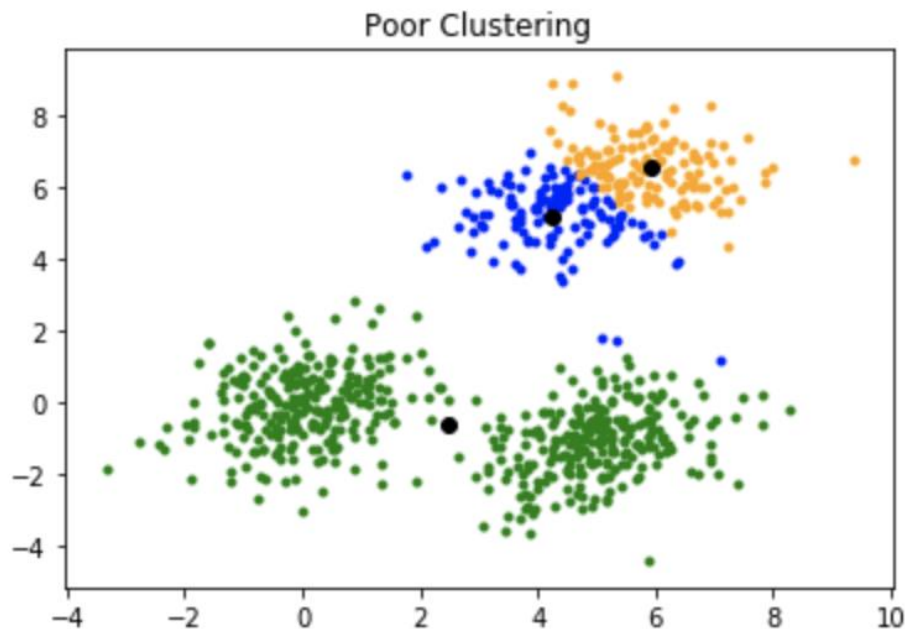
dove

- **km** è l'oggetto generato dalla funzione `kmeans()`
- **col()** individua i colori da associare ai differenti cluster
- **pch** controlla il tipo di carattere da utilizzare
- **cex** la grandezza del testo e dei simboli generati



# K-Means++

- Uno svantaggio dell'algoritmo K-means è che è sensibile all'inizializzazione dei centroidi
  - Se un centroide viene inizializzato per essere un punto "lontano", potrebbe semplicemente finire senza punti associati e più di un cluster potrebbe finire per essere collegato a un singolo centroide.
  - Allo stesso modo, più di un centroide potrebbe essere inizializzato nello stesso cluster, con conseguente scarso clustering



# K-Means++

- La differenza principale tra **K-means** e **K-means++** risiede nel metodo di inizializzazione dei centri (centroidi) dei cluster
  - **K-means**: I centroidi iniziali sono scelti **casualmente** dall'insieme di dati. Questo può portare a risultati diversi ogni volta che si esegue l'algoritmo e aumentare il rischio di convergenza verso minimi locali, producendo risultati sub-ottimali
  - **K-means++**: Implementa una tecnica di inizializzazione migliorata, selezionando i centroidi iniziali in modo mirato per **massimizzare la distanza tra i punti iniziali**. Il primo centro viene scelto a caso, mentre i successivi vengono selezionati con una **probabilità proporzionale** alla distanza quadrata dal centro già scelto
- Questa distinzione ha un impatto significativo sulle prestazioni e sull'accuratezza del clustering

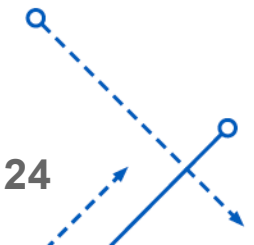


# Selezione dei Centroidi nel K-Means++

- **Passo 1:** Selezione del Primo Centroide
  - Il primo centroide  $c_1$  è scelto **a caso** tra i punti dati  $x$  in  $X$
- **Passo 2:** Calcolo delle Distanze al Quadrato
  - Per ogni punto  $x \in X$ , calcola la **distanza al quadrato**  $D(x)^2$  rispetto al centroide più vicino tra quelli già selezionati
- **Passo 3:** Selezione Probabilistica dei Centroidi Successivi
  - Per ogni nuovo centroide  $c_1$ , seleziona un punto  $x \in X$  con una probabilità proporzionale alla distanza  $D(x)^2$

$$P(x) = \frac{D(x)^2}{\sum_{x' \in X} D(x')^2}$$

- **Passo 4:** Iterazione del Processo
  - Ripetere il Passo 3 finché non si selezionano  $k$  centroidi.



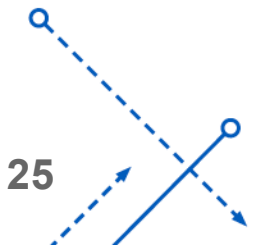


# Esempio K-Means++

- Consideriamo i seguenti dati:

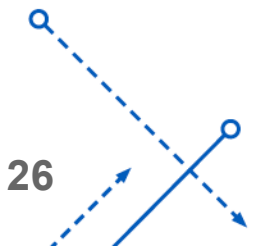
$$X = [1, 2, 3, 12, 13, 14] \quad k: 2$$

- Step 1:** K-means++ sceglie il **primo centroide a caso** tra i punti dati
  - Supponiamo che il primo centroide scelto sia  $c_1 = 3$
- Step 2:** Calcoliamo la distanza di ciascun punto dato dal centroide già scelto  $c_1$ :
  - $d(1,3) = |1-3| = 2$
  - $d(2,3) = |2-3| = 1$
  - $d(3,3) = |3-3| = 0$
  - $d(12,3) = |12-3| = 9$
  - $d(13,3) = |13-3| = 10$
  - $d(14,3) = |14-3| = 11$



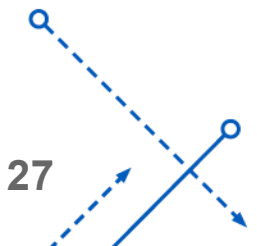
# Esempio K-Means++

- K-means++ assegna una **probabilità proporzionale** al quadrato della distanza per scegliere il prossimo centroide
  - Punti più distanti dal centroide esistente hanno maggiori probabilità di essere scelti.
- **Step 3:** Calcoliamo le distanze al quadrato per ciascun punto e la loro somma:
  - $d(1,3) = |1-3| = 2$
  - $d(2,3) = |2-3| = 1$
  - $d(3,3) = |3-3| = 0$
  - $d(12,3) = |12-3| = 9$
  - $d(13,3) = |13-3| = 10$
  - $d(14,3) = |14-3| = 11$



# Esempio K-Means++

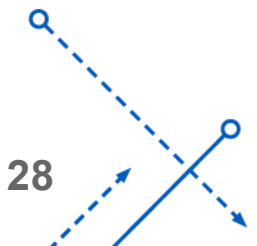
- K-means++ assegna una **probabilità proporzionale** al quadrato della distanza per scegliere il prossimo centroide
  - Punti più distanti dal centroide esistente hanno maggiori probabilità di essere scelti.
- **Step 3:** Calcoliamo le distanze al quadrato per ciascun punto e la loro somma:
  - $d(1,3) = |1-3| = 2$                       -  $D(x)^2 = d(1,3)^2 = 2^2 = 4$
  - $d(2,3) = |2-3| = 1$                       -  $D(x)^2 = d(2,3)^2 = 1^2 = 1$
  - $d(3,3) = |3-3| = 0$                       -  $D(x)^2 = d(3,3)^2 = 0^2 = 0$
  - $d(12,3) = |12-3| = 9$                       -  $D(x)^2 = d(12,3)^2 = 9^2 = 81$
  - $d(13,3) = |13-3| = 10$                       -  $D(x)^2 = d(13,3)^2 = 10^2 = 100$
  - $d(14,3) = |14-3| = 11$                       -  $D(x)^2 = d(14,3)^2 = 11^2 = 121$



# Esempio K-Means++

- K-means++ assegna una **probabilità proporzionale** al quadrato della distanza per scegliere il prossimo centroide
  - Punti più distanti dal centroide esistente hanno maggiori probabilità di essere scelti.
- **Step 3:** Calcoliamo le distanze al quadrato per ciascun punto e la loro somma:
  - $d(1,3) = |1-3| = 2$
  - $d(2,3) = |2-3| = 1$
  - $d(3,3) = |3-3| = 0$
  - $d(12,3) = |12-3| = 9$
  - $d(13,3) = |13-3| = 10$
  - $d(14,3) = |14-3| = 11$
  - $D(x)^2 = d(1,3)^2 = 2^2 = 4$
  - $D(x)^2 = d(2,3)^2 = 1^2 = 1$
  - $D(x)^2 = d(3,3)^2 = 0^2 = 0$
  - $D(x)^2 = d(12,3)^2 = 9^2 = 81$
  - $D(x)^2 = d(13,3)^2 = 10^2 = 100$
  - $D(x)^2 = d(14,3)^2 = 11^2 = 121$

$$\sum_{x' \in X} D(x')^2 = 4 + 1 + 0 + 81 + 100 + 121 = \mathbf{307}$$



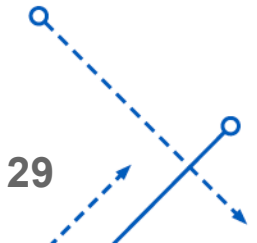
# Esempio K-Means++

- K-means++ assegna una **probabilità proporzionale** al quadrato della distanza per scegliere il prossimo centroide
  - Punti più distanti dal centroide esistente hanno maggiori probabilità di essere scelti.

- **Step 3:** Calcoliamo le distanze al quadrato per ciascun punto e la loro somma:

- $d(1,3) =  1-3  = 2$	- $D(x)^2 = d(1,3)^2 = 2^2 = 4$	- $P(1) = \frac{4}{307}$
- $d(2,3) =  2-3  = 1$	- $D(x)^2 = d(2,3)^2 = 1^2 = 1$	- $P(2) = \frac{1}{307}$
- $d(3,3) =  3-3  = 0$	- $D(x)^2 = d(3,3)^2 = 0^2 = 0$	- $P(3) = \frac{0}{307}$
- $d(12,3) =  12-3  = 9$	- $D(x)^2 = d(12,3)^2 = 9^2 = 81$	- $P(12) = \frac{81}{307}$
- $d(13,3) =  13-3  = 10$	- $D(x)^2 = d(13,3)^2 = 10^2 = 100$	- $P(13) = \frac{100}{307}$
- $d(14,3) =  14-3  = 11$	- $D(x)^2 = d(14,3)^2 = 11^2 = 121$	- $P(14) = \frac{121}{307}$

$$\sum_{x' \in X} D(x')^2 = 4 + 1 + 0 + 81 + 100 + 121 = \mathbf{307}$$

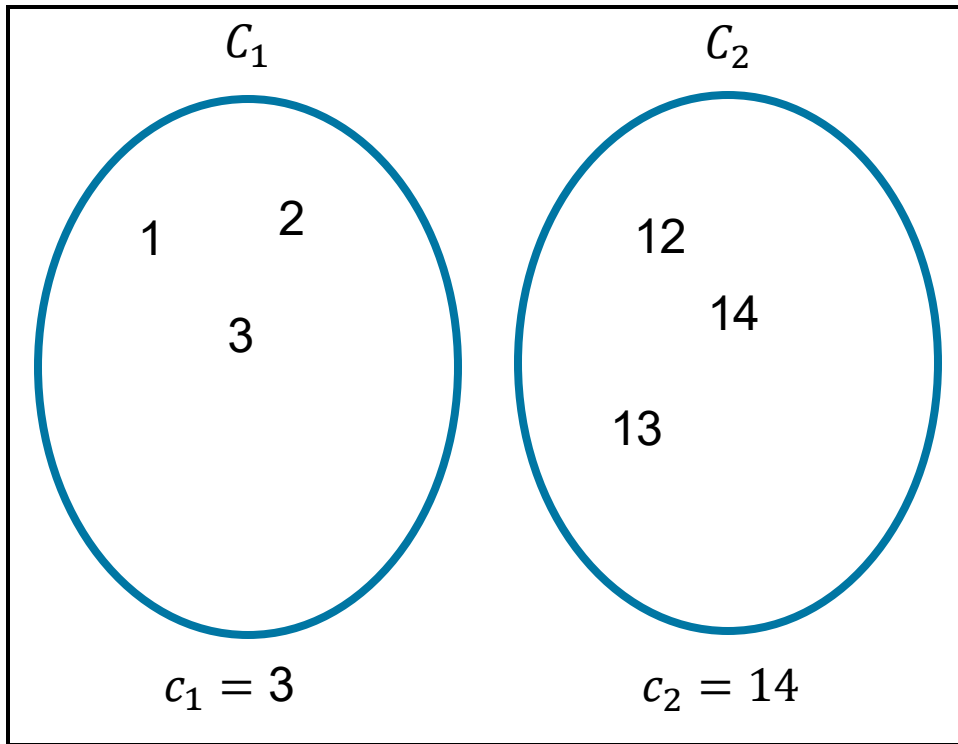


# K-Means

- Ora che abbiamo i centroidi iniziali **3** e **14**, procediamo con l'algoritmo K-means normale:

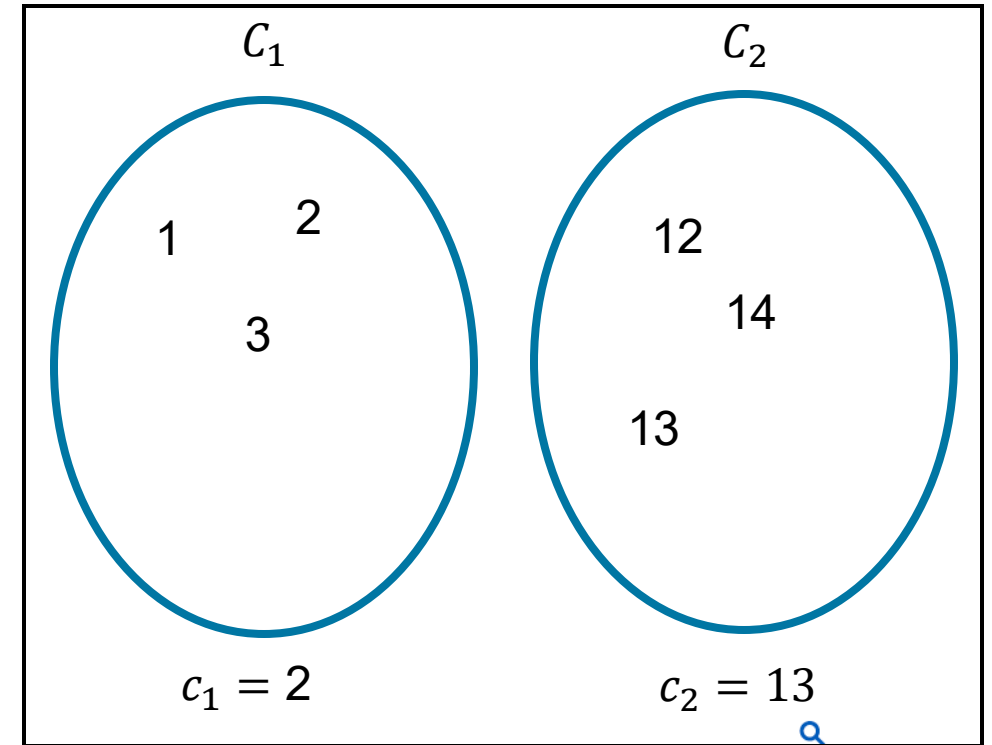
$X = [1, 2, 3, 12, 13, 14]$

$k: 2$



Step 1

K-means



Step 2 – Aggiornamento Centroidi

Ripetiamo i passaggi finché i centroidi non si stabilizzano

# STATISTICA E ANALISI DEI DATI

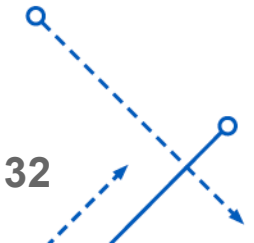
Numero ottimale di Cluster

# Elbow Method

- Il metodo dell'Elbow è una tecnica utilizzata per determinare il numero ottimale di cluster nel clustering
- L'idea di base è quella di tracciare la somma delle distanze al **quadrato intra-cluster** (inertia o Within-cluster sum of squares, WCSS) per diversi valori di  $K$  (il numero di cluster) e scegliere il valore di  $K$  dove l'inertia inizia a ridursi a un ritmo più lento, formando un **gomito** nel grafico
  - Per un insieme di punti  $X=\{x_1, x_2, \dots, x_n\}$  suddivisi in  $K$  cluster  $C_1, C_2, \dots, C_K$  la within-cluster sum of squares (WCSS) è definita come:

$$WCSS(K) = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

- $\|x_i - \mu_k\|^2$  è la **distanza euclidea** al quadrato tra un punto  $x_i$  nel cluster  $C_k$  e il suo centroide  $\mu_k$
- $\mu_k$  è il **centroide** del cluster  $C_k$  ed è uguale a  $\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$
- $K$  è il numero di cluster





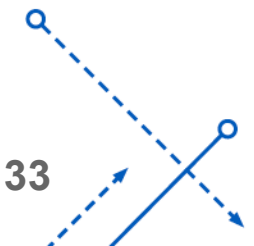
# Elbow Method

- L'obiettivo del metodo si basa sulla minimizzazione della WCSS rispetto al numero di cluster
- Supponendo di considerare un insieme di punti  $X = \{x_1, x_2, \dots, x_n\}$ , dove ogni  $x_i$  è un dato nel nostro spazio e  $K$  rappresenta il numero di cluster, il problema di minimizzazione è:

$$\min_{C_1, \dots, C_K} \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

- **Come funziona il metodo di Elbow**

- 1. Eseguire il clustering:** Si applica il clustering  $K$ -means con un numero variabile di cluster  $K$  (tipicamente da 1 a un massimo ragionevole), calcolando per ogni  $K$  la somma delle distanze quadrate dei punti dai loro centroidi (nota come *Within-Cluster Sum of Squares* o WCSS).
- 2. Calcolare la WCSS:** Per ogni valore di  $K$ , si calcola la WCSS come misura della compattezza del clustering
- 3. Tracciare la curva WCSS:** Si rappresenta la WCSS in funzione di  $K$
- 4. Identificare il “gomito” della curva:** Il gomito (o *elbow*) rappresenta il punto oltre il quale l'aggiunta di ulteriori cluster non porta a un miglioramento significativo della WCSS. Questo punto è considerato il valore ottimale di  $K$  perché **bilancia la compattezza e la semplicità dell'algoritmo**



# Elbow Method in R

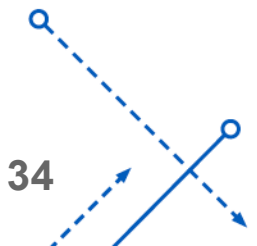
- Vogliamo valutare il numero di cluster ottimale esaminando il grafico dell'Elbow method utilizzando il K-Means
- Consideriamo il dataset Iris

```
# Carico il dataset Iris
data(iris)
iris_data <- iris[, -5] # Rimuovo la colonna delle specie

# Calcolare la WCSS per diversi valori di K
wcss <- sapply(1:10, function(k){
  kmeans(iris_data, centers = k, nstart = 10)$tot.withinss
})

# Traccio il grafico del metodo dell'Elbow
plot(1:10, wcss, type="b", pch = 19, frame = FALSE,
     xlab="Numero di Cluster K",
     ylab="Within-cluster Sum of Squares (WCSS)",
     main="Metodo dell'Elbow - Dataset Iris")

# Aggiungo una linea per visualizzare meglio i risultati
abline(v = 3, lty = 2, col = "red")
```



# Elbow Method in R

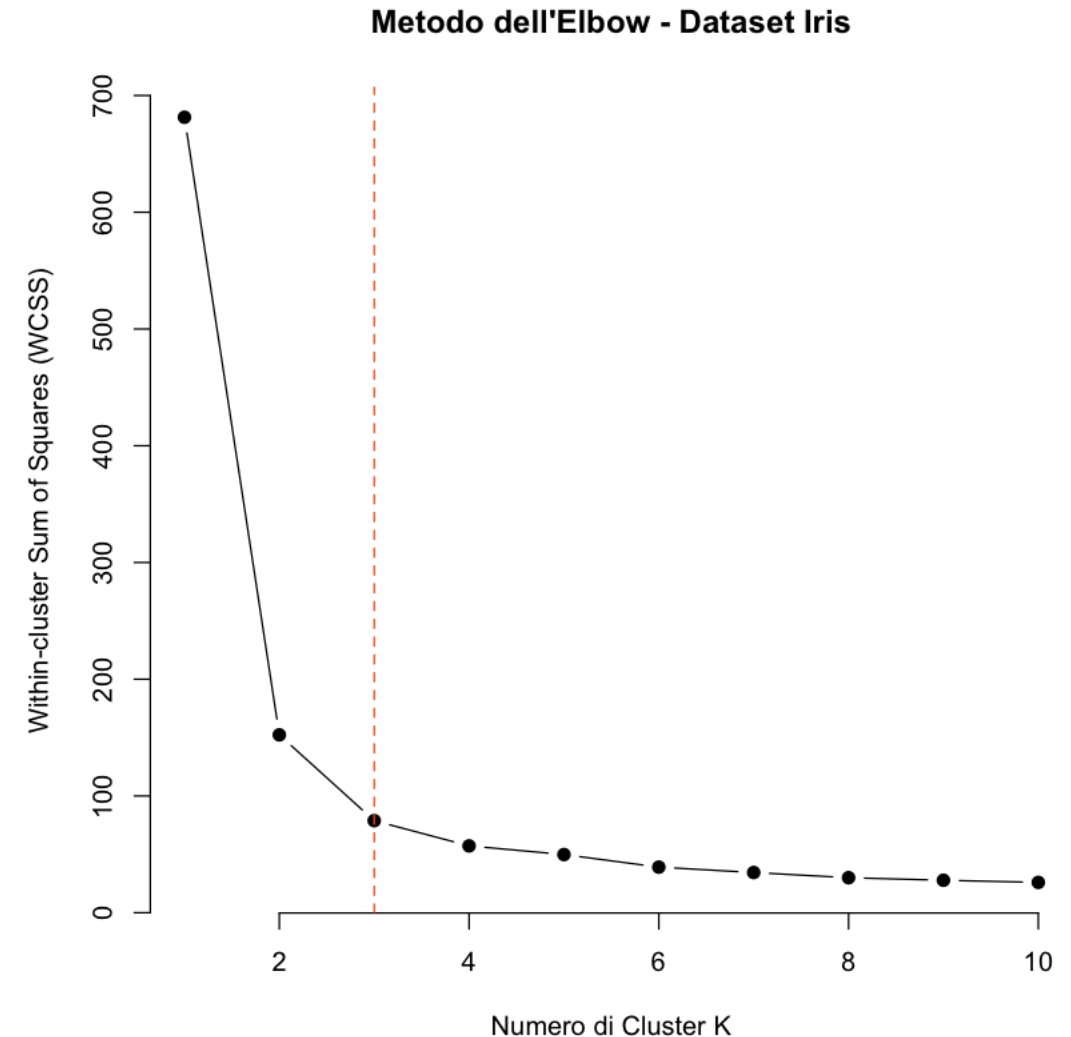
- Vogliamo valutare il numero di cluster ottimale esaminando il grafico dell'Elbow Method utilizzando il K-Means
- Consideriamo il dataset Iris

```
# Carico il dataset Iris
data(iris)
iris_data <- iris[, -5] # Rimuovo la colonna delle specie

# Calcolare la WCSS per diversi valori di K
wcsc <- sapply(1:10, function(k){
  kmeans(iris_data, centers = k, nstart = 10)$tot.withinss
})

# Traccio il grafico del metodo dell'Elbow
plot(1:10, wcsc, type="b", pch = 19, frame = FALSE,
     xlab="Numero di Cluster K",
     ylab="Within-cluster Sum of Squares (WCSS)",
     main="Metodo dell'Elbow - Dataset Iris")

# Aggiungo una linea per visualizzare meglio i risultati
abline(v = 3, lty = 2, col = "red")
```

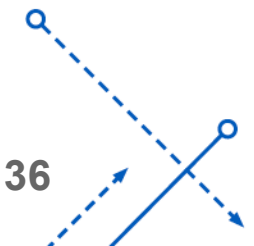


# Silhouette Method

- Il metodo della silhouette è una tecnica utilizzata per valutare la **qualità di un clustering**, aiutando a determinare quanto bene un punto è assegnato al proprio cluster rispetto agli altri cluster
  - Per un insieme di punti  $X = \{x_1, x_2, \dots, x_n\}$  suddivisi in  $K$  cluster  $C_1, C_2, \dots, C_K$  la **silhouette** di un singolo punto  $x_i$  è definita come:

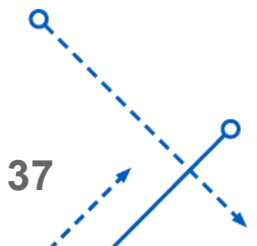
$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max(a(x_i), b(x_i))}$$

- $a(x_i)$  la **distanza media** tra  $x_i$  e tutti gli altri punti nel suo stesso cluster  $C_k$  (**intra-cluster**)
  - Questa distanza rappresenta quanto il punto  $x_i$  è "vicino" ai punti del suo cluster e misura quindi la compattezza interna
- $b(x_i)$  è la **distanza media** tra  $x_i$  e tutti gli altri punti del cluster più vicino a  $x_i$  (**inter-cluster**)



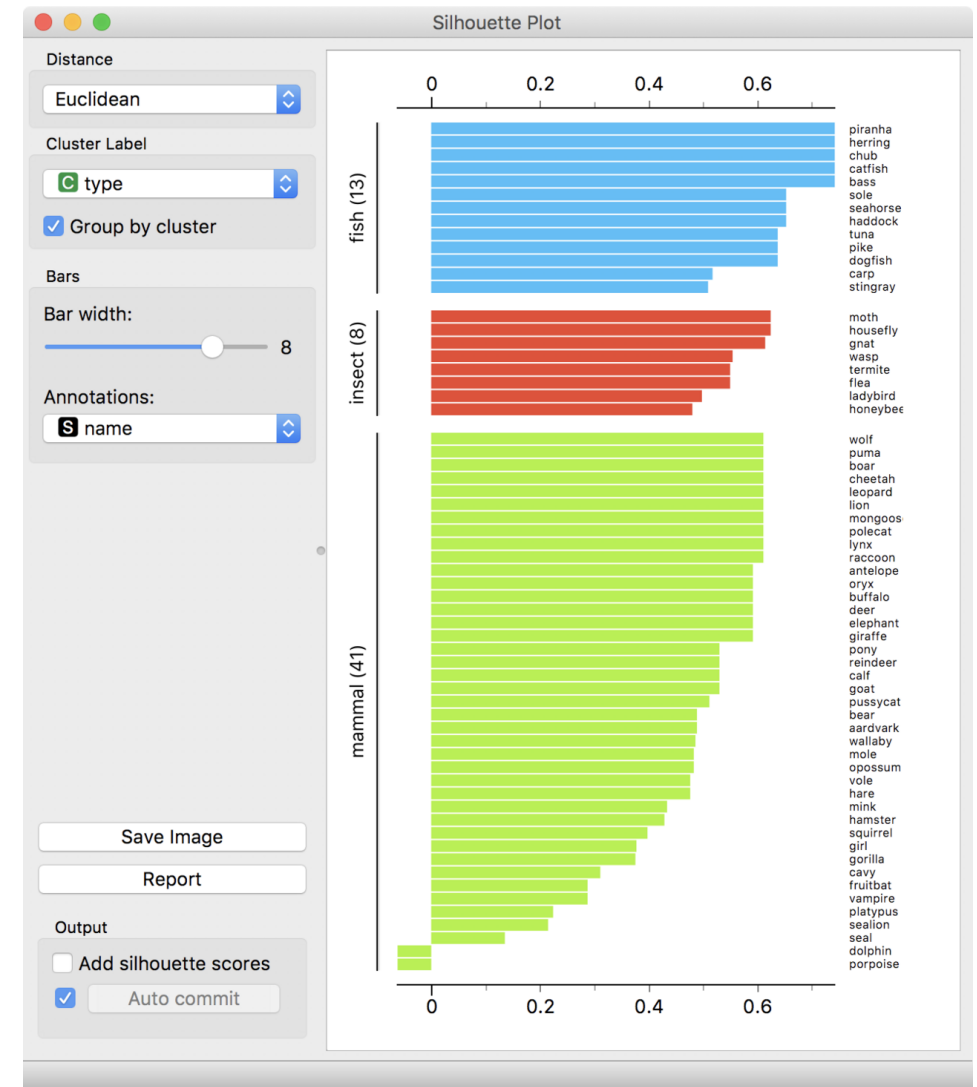
# Silhouette Method

- Fornisce una misura numerica che va da -1 a 1:
  - $s(x_i) \approx 1$ : Il punto  $x_i$  è ben assegnato al proprio cluster, poiché la distanza intra-cluster  $a(x_i)$  è molto inferiore rispetto alla distanza al cluster più vicino  $b(x_i)$
  - $s(x_i) \approx 0$ : Il punto si trova vicino ai confini tra due cluster, poiché  $a(x_i) \approx b(x_i)$
  - $s(x_i) \approx -1$ : Il punto potrebbe essere assegnato al cluster sbagliato, poiché la distanza al cluster più vicino  $b(x_i)$  è inferiore alla distanza intra-cluster  $a(x_i)$
- La silhouette media di tutti i punti di un dataset fornisce una stima globale della bontà del clustering
  - Se la silhouette media è alta (vicina a 1), il **clustering è considerato buono**,
  - Silhouette medie basse o negative indicano un **clustering di scarsa qualità**



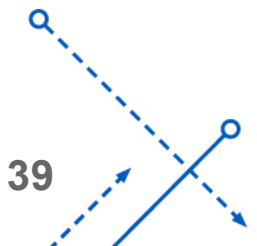
# Silhouette Method per la scelta del numero di cluster

- Il metodo della silhouette può essere utilizzato per determinare il numero ottimale di cluster  $K$ 
  - Calcola il clustering per un intervallo di possibili valori di  $k$  (ad esempio, da 2 a 10)
  - Calcola il coefficiente di silhouette medio per ogni valore di  $k$ 
    - La silhouette media è data dalla media dei coefficienti di silhouette di tutti i punti nel dataset per quel numero di cluster
  - Scegli il numero di cluster  $K$  che **massimizza** la silhouette media
    - Il **valore ottimale** di  $k$  è quello che rende i cluster più distinti e meglio separati



# Esempio Silhouette

- Consideriamo i seguenti dati:  $X = [1, 2, 3, 12, 13, 14]$   $k: 2$
- Supponiamo che l'algoritmo di  $K$ -means divida i punti in due cluster:
  - Cluster 1:  $[1, 2, 3]$
  - Cluster 2:  $[12, 13, 14]$
- Calcoliamo i centroidi di ciascun cluster:
  - Centroide di Cluster 1:  $c_1 = \frac{1+2+3}{3} = 2$  Centroide di Cluster 2:  $c_2 = \frac{12+13+14}{3} = 13$
- Calcolo  $a(1)$  e  $b(1)$ :
  - Distanza media di 1 dagli altri punti del proprio cluster  $[2, 3]$  e dai punti nell'altro cluster  $[12, 13, 14]$ :
$$a(1) = \frac{|1-2| + |1-3|}{2} = \frac{1+2}{2} = 1.5 \qquad b(1) = \frac{|1-12| + |1-13| + |1-14|}{3} = \frac{11+12+13}{3} = 12$$
  - Coefficiente di silhouette  $s(1)$ :
$$s(1) = \frac{b(1) - a(1)}{\max(a(1), b(1))} = \frac{12 - 1.5}{12} = 0.875$$



# Esempio Silhouette

- Calcolo  $a(2)$  e  $b(2)$ :

$$a(2) = \frac{|2 - 1| + |2 - 3|}{2} = \frac{1 + 1}{2} = 1 \qquad b(2) = \frac{|2 - 12| + |2 - 13| + |2 - 14|}{3} = \frac{10 + 11 + 12}{3} = 11$$

$$s(2) = \frac{b(2) - a(2)}{\max(a(2), b(2))} = \frac{11 - 1}{11} = 0.909$$

- Calcolo  $a(3)$  e  $b(3)$ :

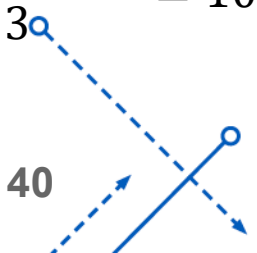
$$a(3) = \frac{|3 - 1| + |3 - 2|}{2} = \frac{2 + 1}{2} = 1.5 \qquad b(3) = \frac{|3 - 12| + |3 - 13| + |3 - 14|}{3} = \frac{9 + 10 + 11}{3} = 10$$

$$s(3) = \frac{b(3) - a(3)}{\max(a(3), b(3))} = \frac{10 - 1.5}{10} = 0.85$$

- Calcolo  $a(12)$  e  $b(12)$ :

$$a(12) = \frac{|12 - 13| + |12 - 14|}{2} = \frac{1 + 2}{2} = 1.5 \qquad b(12) = \frac{|12 - 1| + |12 - 2| + |12 - 3|}{3} = \frac{11 + 10 + 9}{3} = 10$$

$$s(12) = \frac{b(12) - a(12)}{\max(a(12), b(12))} = \frac{10 - 1.5}{10} = 0.85$$





# Esempio Silhouette

- Calcolo  $a(13)$  e  $b(13)$ :

$$a(13) = \frac{|13 - 12| + |13 - 14|}{2} = \frac{1 + 1}{2} = 1 \quad b(13) = \frac{|13 - 1| + |13 - 2| + |13 - 3|}{3} = \frac{12 + 11 + 10}{3} = 11$$

$$s(13) = \frac{b(13) - a(13)}{\max(a(13), b(13))} = \frac{11 - 1}{11} = 0.909$$

- Calcolo  $a(14)$  e  $b(14)$ :

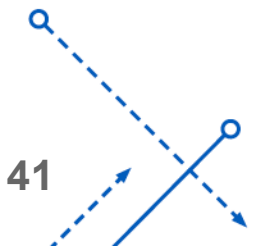
$$a(14) = \frac{|14 - 12| + |14 - 13|}{2} = \frac{2 + 1}{2} = 1.5 \quad b(14) = \frac{|14 - 1| + |14 - 2| + |14 - 3|}{3} = \frac{13 + 12 + 11}{3} = 11$$

$$s(14) = \frac{b(14) - a(14)}{\max(a(14), b(14))} = \frac{11 - 1.5}{11} = 0.875$$

- Coefficiente medio di silhouette**

- Il coefficiente medio di silhouette per l'intero dataset è la media dei coefficienti silhouette di tutti i punti:

$$silhouette\_media = \frac{0.875 + 0.909 + 0.85 + 0.85 + 0.909 + 0.875}{6} \approx 0.878$$



# Silhouette Method per la scelta del numero di cluster

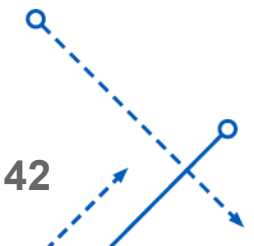
```
# Installa i pacchetti se non sono già installati  
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)  
  
# Carica le librerie necessarie  
library(cluster)
```



Installazione ed import della libreria



per il metodo della silhouette



# Silhouette Method per la scelta del numero di cluster

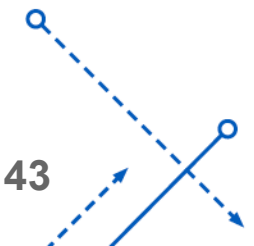
```
# Installa i pacchetti se non sono già installati  
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)  
  
# Carica le librerie necessarie  
library(cluster)  
  
# Rimuoviamo la colonna delle specie per usare solo i dati numerici  
data(iris)  
iris_data <- iris[, -5]
```



Installazione ed import della libreria  
per il metodo della silhouette



Caricamento dei dati



# Silhouette Method per la scelta del numero di cluster

```
# Installa i pacchetti se non sono già installati
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)

# Carica le librerie necessarie
library(cluster)

# Rimuoviamo la colonna delle specie per usare solo i dati numerici
data(iris)
iris_data <- iris[, -5]

# Funzione per calcolare il coefficiente di silhouette medio per diversi k
silhouette_analysis <- function(k) {
  km <- kmeans(iris_data, centers = k, nstart = 25)
  sil <- silhouette(km$cluster, dist(iris_data))
  mean(sil[, 3]) # Restituisce il coefficiente di silhouette medio
}
```

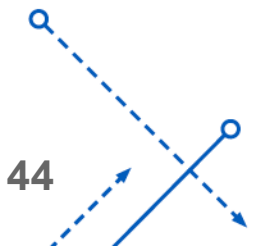
Installazione ed import della libreria  
per il metodo della silhouette

Caricamento dei dati

Definisco una funzione che esegue il  
clustering, calcola la silhouette e restituisce  
il coefficiente di silhouette medio

Prendo la 3° colonna perché i coefficienti  
di silhouette sono presenti nella terza  
colonna del dataframe risultante

[1] "silhouette"	cluster	neighbor	sil_width
[1,]	4	2	0.32270479
[2,]	2	4	0.50004048
[3,]	2	4	0.52801522
[4,]	2	4	0.58448659
[5,]	4	2	0.30139998
[6,]	4	2	0.52274284
[7,]	2	4	0.38383296
[8,]	4	2	0.04299262



# Silhouette Method per la scelta del numero di cluster

```
# Installa i pacchetti se non sono già installati
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)

# Carica le librerie necessarie
library(cluster)

# Rimuoviamo la colonna delle specie per usare solo i dati numerici
data(iris)
iris_data <- iris[, -5]

# Funzione per calcolare il coefficiente di silhouette medio per diversi k
silhouette_analysis <- function(k) {
  km <- kmeans(iris_data, centers = k, nstart = 25)
  sil <- silhouette(km$cluster, dist(iris_data))
  mean(sil[, 3]) # Restituisce il coefficiente di silhouette medio
}

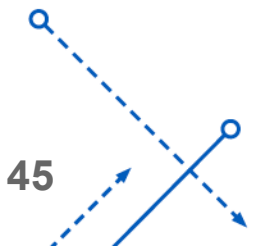
# Eseguiamo il calcolo per k da 2 a 10 cluster
k_values <- 2:10
silhouette_scores <- sapply(k_values, silhouette_analysis)
```

Installazione ed import della libreria  
per il metodo della silhouette

Caricamento dei dati

Definisco una funzione che esegue il  
clustering, calcola la silhouette e restituisce  
il coefficiente di silhouette medio

Eseguo la funzione considerando un numero  
variabile di cluster



# Silhouette Method per la scelta del numero di cluster

```
# Installa
if (!requi

# Carica l
library(cl

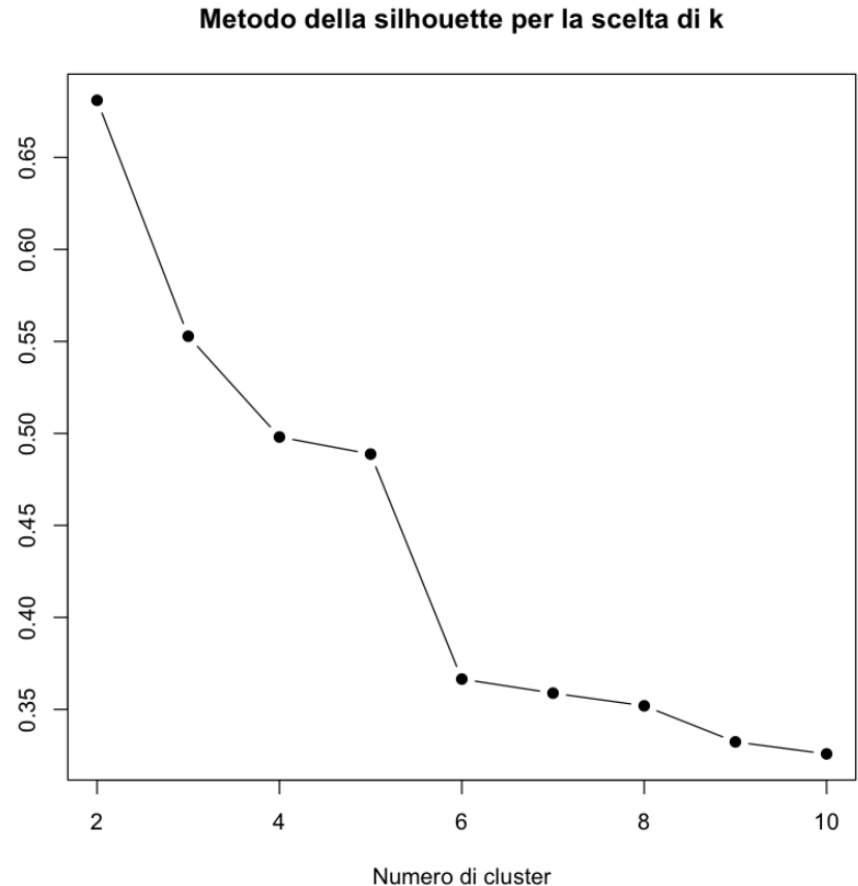
# Rimuovia
data(iris)
iris_data

# Funzione
silhouette
km <- km
sil <- s
mean(sil

}

# Eseguiam
k_values <-
silhouette

# Visualizziamo i risultati
plot(k_values, silhouette_scores, type = "b", pch = 19,
     xlab = "Numero di cluster", ylab = "Silhouette media",
     main = "Metodo della silhouette per la scelta di k")
```



=TRUE)



Installazione ed import della libreria  
per il metodo della silhouette

rici



Caricamento dei dati

r diversi k



Definisco una funzione che esegue il  
clustering, calcola la silhouette e restituisce  
il coefficiente di silhouette medio

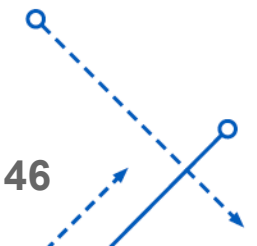
edio



Eseguo la funzione considerando un numero  
variabile di cluster



Rappresento graficamente i coefficienti di  
silhouette medi



# Silhouette Method per la scelta del numero di cluster

```
# Installa i pacchetti se non sono già installati
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)
```



Installazione ed import della libreria  
per il metodo della silhouette

```
# Carica le librerie necessarie
library(cluster)
```



```
# Rimuoviamo la colonna delle specie per usare solo i dati numerici
data(iris)
iris_data <- iris[, -5]
```



Caricamento dei dati

```
# Funzione per calcolare il coefficiente di silhouette medio per diversi k
silhouette_analysis <- function(k) {
  km <- kmeans(iris_data, centers = k, nstart = 25)
  sil <- silhouette(km$cluster, dist(iris_data))
  mean(sil[, 3]) # Restituisce il coefficiente di silhouette medio
}
```



Definisco una funzione che esegue il  
clustering, calcola la silhouette e restituisce  
il coefficiente di silhouette medio

```
# Eseguiamo il calcolo per k da 2 a 10 cluster
k_values <- 2:10
silhouette_scores <- sapply(k_values, silhouette_analysis)
```



Eseguo la funzione considerando un numero  
variabile di cluster

```
# Visualizziamo i risultati
plot(k_values, silhouette_scores, type = "b", pch = 19,
     xlab = "Numero di cluster", ylab = "Silhouette media",
     main = "Metodo della silhouette per la scelta di k")
```

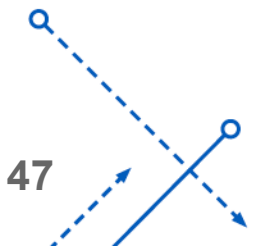


Rappresento graficamente i coefficienti di  
silhouette medi

```
# Il numero ottimale di cluster è quello con la silhouette media più alta
optimal_k <- which.max(silhouette_scores)
cat("Il numero ottimale di cluster è:", k_values[optimal_k], "\n")
```



Estraggo il coefficiente di  
silhouette medio più grande



# Silhouette Method per la scelta del numero di cluster

```
# Installa i pacchetti se non sono già installati
if (!require(cluster)) install.packages("cluster", dependencies=TRUE)
```

```
# Carica le librerie
library(cluster)
```

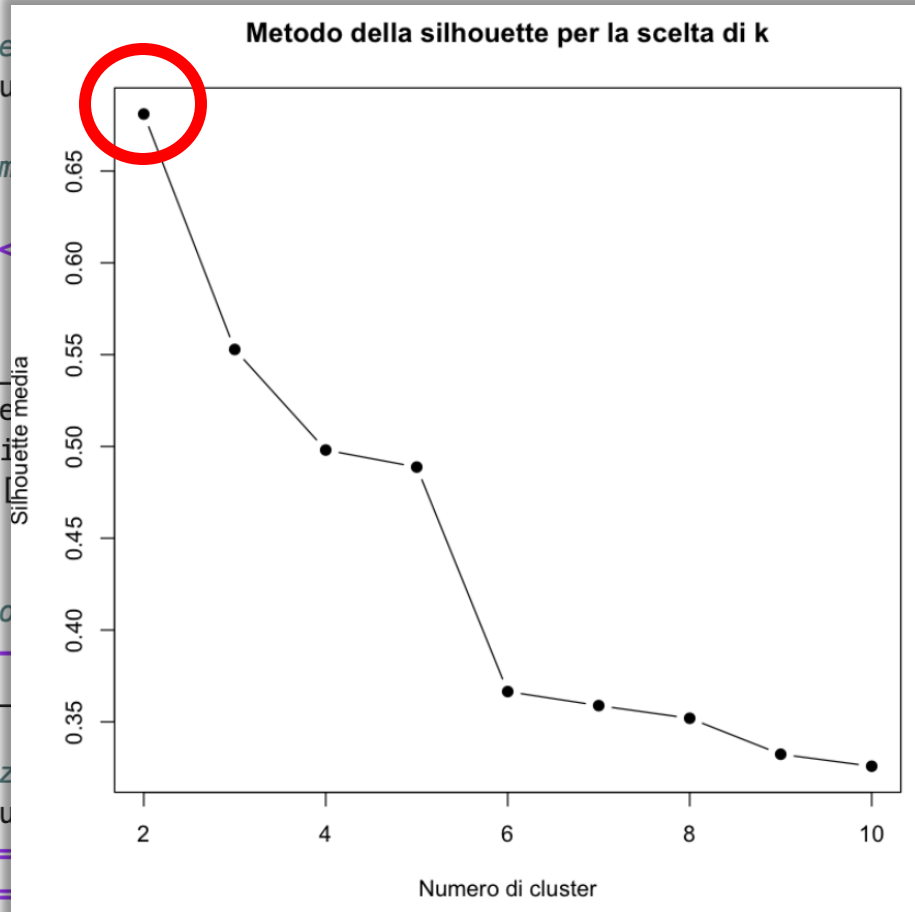
```
# Rimuovi i dati originali
data(iris)
iris_data <- iris
```

```
# Funzione per calcolare la silhouette media per diversi k
silhouette_km <- function(k) {
  km <- kmeans(iris_data, k)
  sil <- silhouette(km$cluster, km$centers)
  mean(sil[, "silhouette"])
}
```

```
# Eseguiamo la funzione per diversi valori di k
k_values <- 2:10
silhouette_scores <- silhouette_km(k_values)
```

```
# Visualizziamo i risultati
plot(k_values, silhouette_scores,
     xlab="Numero di cluster",
     ylab="Silhouette media",
     main="Metodo della silhouette per la scelta di k")
```

```
# Il numero ottimale di cluster è quello con la silhouette media più alta
optimal_k <- which.max(silhouette_scores)
cat("Il numero ottimale di cluster è:", k_values[optimal_k], "\n")
```



Installazione ed import della libreria  
per il metodo della silhouette

Caricamento dei dati

Definisco una funzione che esegue il  
clustering, calcola la silhouette e restituisce  
il coefficiente di silhouette medio

Eseguo la funzione considerando un numero  
variabile di cluster

Rappresento graficamente i coefficienti di  
silhouette medi

Il numero ottimale di cluster è: 2



# STATISTICA E ANALISI DEI DATI

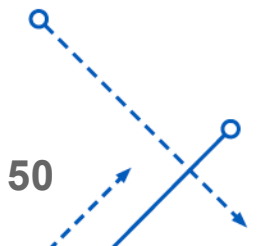
Valutazione Bontà dei Cluster

# Within-Cluster Sum of Squares

- La **Within-Cluster Sum of Squares** (WCSS) rappresenta la coesione interna del cluster, con un valore minore che indica che i punti sono più vicini al centroide, il che suggerisce una buona compattezza del clustering
- Dato un numero  $k$  di cluster  $C_1, \dots, C_K$ , con centroidi  $\bar{x}_1, \dots, \bar{x}_K = \mu_1, \dots, \mu_K$ , lo scarto quadratico medio (o [inerzia totale](#)) è:

$$\text{WCSS} = \sum_{j=1}^k \sum_{x \in C_j} d(x - \mu_j)^2$$

- Dove  $d(x_j - \mu_j)^2$  è la distanza euclidea tra il punto  $x$  e il centroide  $\mu_j$  del cluster  $C_j$



# Within-Cluster Sum of Squares

- Applicata ai cluster:

$$WCSS = \sum_{j=1}^k \sum_{x \in C_j} d(x - \mu_j)^2$$

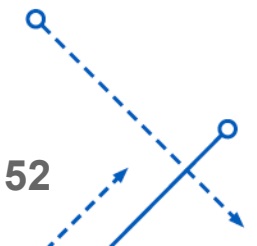
- **Valore Basso**: Un valore basso di WCSS indica che i punti all'interno dei cluster sono vicini ai loro centroidi, suggerendo che i cluster sono ben separati e compatti
  - Ciò significa che il clustering ha funzionato bene, con punti simili raggruppati insieme
- **Valore Alto**: Un valore alto di WCSS indica che i punti sono più distanti dai loro centroidi, suggerendo che i cluster potrebbero essere sparsi e meno distinti
  - Questo può indicare che il numero di cluster scelto non è appropriato o che i dati non presentano una chiara struttura clusterizzata

# Between-Cluster Sum of Squares

- La **between-cluster sum of squares** (o **BCSS**) è una misura che quantifica la separazione tra i cluster trovati e si calcola come la somma delle distanze al quadrato tra i centroidi dei cluster e il centroide generale (centroide dell'intero dataset):

$$\text{BCSS} = \sum_{j=1}^k |C_j| * d(\mu_j - \mu)^2$$

- $k$  è il numero di cluster
- $|C_j|$  è il numero di punti nel cluster  $C_j$
- $\mu_j$  è il centroide del cluster  $C_j$
- $\mu$  è il centroide globale dell'intero dataset calcolato come  $\frac{\sum_{i=1}^N \mu_i}{N}$  con  $N$  numero totale di punti nel dataset
- $d(\mu - \mu_j)^2$  è la distanza euclidea tra il centroide globale  $\mu$  e il centroide  $\mu_j$



# Between-Cluster Sum of Squares

- La **between-cluster sum of squares** (o **BCSS**)

$$\text{BCSS} = \sum_{j=1}^k |C_j| * d(\mu_j - \mu)^2$$

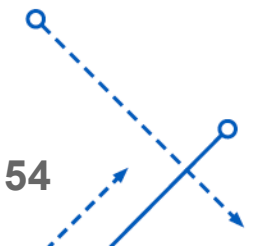
- **Valore Basso**: Quando il valore di BCSS è **basso**, significa che i centroidi dei cluster sono vicini al centroide globale, indicando che i cluster non sono ben separati e sono **più sovrapposti** tra loro
  - Il clustering potrebbe non essere ottimale, in quanto i cluster sono poco distinti.
- **Valore Alto**: Quando il valore di BCSS è alto, significa che i centroidi dei cluster sono molto **distanti** dal centroide globale, indicando che i cluster sono ben separati tra loro
  - Questo è un buon segno in clustering, in quanto i cluster risultano distinti e separati.

# Indice di Calinski-Harabasz

- L'**indice di Calinski-Harabasz** (CH Index), è una misura utilizzata per valutare la qualità del clustering
  - Misura il bilanciamento tra la **coesione interna** (compattezza dei cluster) e la **separazione tra i cluster**
- L'indice di Calinski-Harabasz si calcola come il rapporto tra la **Between-Cluster Sum of Squares (BCSS)** e la **Within-Cluster Sum of Squares (WCSS)**, moltiplicato per il numero di punti nel dataset e diviso per il numero di cluster:

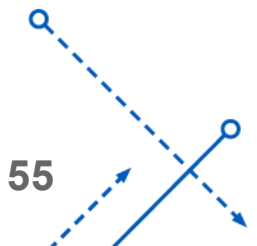
$$CH = \frac{BCSS/(k - 1)}{WCSS/(N - k)}$$

- $k$  è il numero di cluster
- $N$  è il numero totale di punti nel dataset



# Indice di Calinski-Harabasz

- Interpretazione dell'indice di Calinski-Harabasz:
  - **Valore Alto:** Un valore più alto dell'indice di Calinski-Harabasz indica una **buona qualità del clustering**, poiché implica che i cluster sono compatti (basso WCSS) e ben separati (alto BCSS)
    - Un buon clustering avrà una separazione significativa tra i cluster e una bassa varianza all'interno di ciascun cluster
  - **Valore basso:** Un valore basso dell'indice di Calinski-Harabasz suggerisce che i cluster sono poco separati e/o troppo dispersivi
    - In altre parole, i punti sono troppo distanti dal centroide del proprio cluster, o i cluster stessi **non sono sufficientemente distinti**.



# Indice di Calinski-Harabasz

- Interpretazione dell'indice di Calinski-Harabasz:
  - **Valore Alto:** Un valore più alto dell'indice di Calinski-Harabasz indica una **buona qualità del clustering**, poiché implica che i cluster sono compatti (basso WCSS) e ben separati (alto BCSS)
    - Un buon clustering avrà una separazione significativa tra i cluster e una bassa varianza all'interno di ciascun cluster
  - **Valore basso:** Un valore basso dell'indice di Calinski-Harabasz suggerisce che i cluster sono poco separati e/o troppo dispersivi
    - In altre parole, i punti sono troppo distanti dal centroide del proprio cluster, o i cluster stessi **non sono sufficientemente distinti**.
- Limiti:
  - l'indice di Calinski-Harabasz può essere sensibile alla presenza di outlier o alla forma dei cluster.
  - In caso di cluster con forme complesse (non sferiche), la metrica potrebbe non riflettere correttamente la qualità del clustering



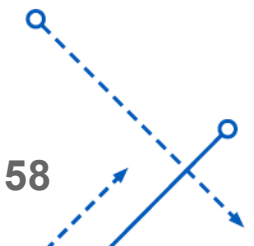
# Esempio WCSS, BCSS e Calinski-Harabasz

- Calcoliamo la **BCSS** (Between-Cluster Sum of Squares), la **WCSS** (Within-Cluster Sum of Squares) e l'indice di **Calinski-Harabasz** dati i punti  $X=[1,2,3,12,13,14]$  e  $k=2$ :
- Supponiamo che l'algoritmo di  $K$ -means divida i punti in due cluster:
  - Cluster 1  $C_1 = [1,2,3]$  Cluster 2  $C_2 = [12,13,14]$
- Calcoliamo i centroidi di ciascun cluster:
  - Centroide di Cluster 1:  $\mu_1 = \frac{1+2+3}{3} = 2$  Centroide di Cluster 2:  $\mu_2 = \frac{12+13+14}{3} = 13$
  - Centroide globale:  $\mu = \frac{1+2+3+12+13+14}{6} = \frac{45}{6} = 7.5$



# Esempio WCSS, BCSS e Calinski-Harabasz

- Calcoliamo la **BCSS** (Between-Cluster Sum of Squares), la **WCSS** (Within-Cluster Sum of Squares) e l'indice di **Calinski-Harabasz** dati i punti  $X=[1,2,3,12,13,14]$  e  $k=2$ :
- Supponiamo che l'algoritmo di  $K$ -means divida i punti in due cluster:
  - Cluster 1  $C_1 = [1,2,3]$  Cluster 2  $C_2 = [12,13,14]$
- Calcoliamo i centroidi di ciascun cluster:
  - Centroide di Cluster 1:  $\mu_1 = \frac{1+2+3}{3} = 2$  Centroide di Cluster 2:  $\mu_2 = \frac{12+13+14}{3} = 13$
  - Centroide globale:  $\mu = \frac{1+2+3+12+13+14}{6} = \frac{45}{6} = 7.5$
- **Calcolo la WCSS (Within-Cluster Sum of Squares):**
  - Per il Cluster  $C_1$ :  $WCSS_1 = (1-2)^2 + (2-2)^2 + (3-2)^2 = 1 + 0 + 1 = 2$
  - Per il Cluster  $C_2$ :  $WCSS_2 = (12-13)^2 + (13-13)^2 + (14-13)^2 = 1 + 0 + 1 = 2$
  - Quindi, la **WCSS totale** è:  
 $WCSS = WCSS_1 + WCSS_2 = 2 + 2 = 4$



# Esempio WCSS, BCSS e Calinski-Harabasz

- **Calcolo la BCSS (Between-Cluster Sum of Squares):**

- Distanza tra il centroide  $\mu_1 = 2$  e il centroide globale  $\mu = 7.5$ :  $d(\mu_j - \mu)^2 = (2 - 7.5)^2 = 30.25$
- Distanza tra il centroide  $\mu_2 = 13$  e il centroide globale  $\mu = 7.5$ :  $d(\mu_j - \mu)^2 = (13 - 7.5)^2 = 30.25$

- Per il Cluster  $C_1$ :  $BCSS_1 = 3 * 30.25 = 90.75$

- Per il Cluster  $C_2$ :  $BCSS_2 = 3 * 30.25 = 90.75$

- Quindi, la **BCSS totale** è:

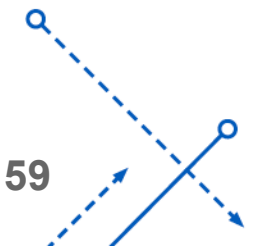
$$BCSS = BCSS_1 + BCSS_2 = 90.75 + 90.75 = 181.5$$

- **Calcolo l'indice di Calinski-Harabasz:**

- Ora possiamo calcolare l'indice di Calinski-Harabasz:

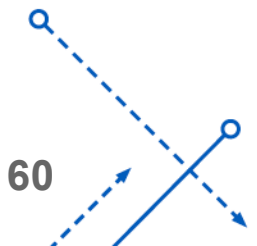
$$CH = \frac{BCSS/(k - 1)}{WCSS/(N - k)} = \frac{181.5/(2 - 1)}{4/(6 - 2)} = 45.375$$

- Un valore più elevato dell'indice di Calinski-Harabasz suggerisce che il clustering è di buona qualità, in quanto i cluster sono ben separati e compatti



# Esempio (ii) WCSS, BCSS e Calinski-Harabasz

- Calcoliamo la **BCSS** (Between-Cluster Sum of Squares), la **WCSS** (Within-Cluster Sum of Squares) e l'indice di **Calinski-Harabasz** dati i punti  $X = [[1,2,3,12,13,14],[5,6,7,18,19,21]]$  e  $k=2$ :
  - Supponiamo che l'algoritmo di  $K$ -means divida i punti in due cluster:
    - Cluster 1  $C_1 = [(1,5),(2,6),(3,7)]$  Cluster 2  $C_2 = [(12,18),(13,19),(14,21)]$
  - Calcoliamo i centroidi di ciascun cluster:
    - Centroide di Cluster 1:  $\mu_1 = \left(\frac{1+2+3}{3}, \frac{5+6+7}{3}\right) = \left(\frac{6}{3}, \frac{18}{3}\right) = (2, 6)$
    - Centroide di Cluster 2:  $\mu_2 = \left(\frac{12+13+14}{3}, \frac{18+19+21}{3}\right) = \left(\frac{39}{3}, \frac{58}{3}\right) = (13, 19.33)$
- Centroide globale:  $\mu = \left(\frac{1+2+3+12+13+14}{6}, \frac{5+6+7+18+19+21}{6}\right) = \left(\frac{45}{6}, \frac{76}{6}\right) = (7.5, 12.67)$



# Esempio (ii) WCSS, BCSS e Calinski-Harabasz

- Calcolo la WCSS (Within-Cluster Sum of Squares):

- Per il Cluster  $C_1$ :

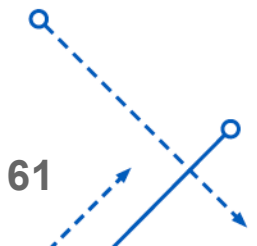
$$\begin{aligned}WCSS_1 &= ((1 - 2)^2 + (5 - 6)^2) + ((2 - 2)^2 + (6 - 6)^2) + ((3 - 2)^2 + (7 - 6)^2) \\ &= (1 + 1) + (0 + 0) + (1 + 1) = 4\end{aligned}$$

- Per il Cluster  $C_2$ :

$$\begin{aligned}WCSS_2 &= ((12 - 13)^2 + (18 - 19.33)^2) + ((13 - 13)^2 + (19 - 19.33)^2) + ((14 - 13)^2 + (21 - 19.33)^2) \\ &= (1 + 1.77) + (0 + 0.11) + (1 + 2.79) = 6.67\end{aligned}$$

- Quindi, la **WCSS totale** è:

$$WCSS = WCSS_1 + WCSS_2 = 4 + 6.67 = 10.67$$



# Esempio (ii) WCSS, BCSS e Calinski-Harabasz

- **Calcolo la BCSS (Between-Cluster Sum of Squares):**

- Distanza tra il centroide  $\mu_1 = (2, 6)$  e il centroide globale  $\mu = (7.5, 12.67)$ :

$$d(\mu_j - \mu)^2 = (2 - 7.5)^2 + (6 - 12.67)^2 = (-5.5)^2 + (-6.67)^2 = 30.25 + 44.49 = 74.74$$

- Distanza tra il centroide  $\mu_2 = (13, 19.33)$  e il centroide globale  $\mu = (7.5, 12.67)$ :

$$d(\mu_j - \mu)^2 = (13 - 7.5)^2 + (19.33 - 12.67)^2 = (-5.5)^2 + (-6.66)^2 = 30.25 + 44.35 = 74.6$$

- Per il Cluster  $C_1$ :  $BCSS_1 = 3 * 74.74 = 224.22$

- Per il Cluster  $C_2$ :  $BCSS_2 = 3 * 74.6 = 223.8$

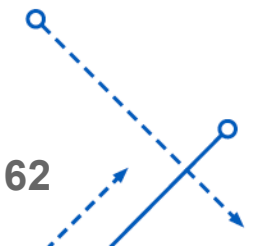
- Quindi, la **BCSS totale** è:

$$BCSS = BCSS_1 + BCSS_2 = 224.22 + 223.8 = 448.02$$

- **Calcolo l'indice di Calinski-Harabasz:**

$$CH = \frac{BCSS/(k - 1)}{WCSS/(N - k)} = \frac{448.02/(2 - 1)}{10.67/(6 - 2)} = 168.06$$

- Un valore più elevato dell'indice di Calinski-Harabasz suggerisce che il clustering è di buona qualità, in quanto i cluster sono ben separati e compatti



# WCSS, BCSS e Calinski-Harabasz in R

```
# Dati
X <- matrix(c(1, 2, 3, 12, 13, 14,
              5, 6, 7, 18, 19, 21),
            nrow = 2, byrow = TRUE)

# Trasponi i dati per avere righe come punti
X <- t(X)

# KMeans clustering con k = 2
set.seed(42)
kmeans_result <- kmeans(X, centers = 2)

print(kmeans_result)

# Centroidi dei cluster
centroids <- kmeans_result$centers
```



K-means clustering with 2 clusters of sizes 3, 3

Cluster means:

	[,1]	[,2]
1	2	6.00000
2	13	19.33333

Clustering vector:

```
[1] 1 1 1 2 2 2
```



# WCSS, BCSS e Calinski-Harabasz in R

```
# Dati
X <- matrix(c(1, 2, 3, 12, 13, 14,
              5, 6, 7, 18, 19, 21),
            nrow = 2, byrow = TRUE)

# Trasponi i dati per avere righe come punti
X <- t(X)

# KMeans clustering con k = 2
set.seed(42)
kmeans_result <- kmeans(X, centers = 2)

print(kmeans_result)

# Centroidi dei cluster
centroids <- kmeans_result$centers

# Calcolare WCSS (Within-Cluster Sum of Squares)
wcss <- sum(kmeans_result$withinss)

# Calcolare BCSS (Between-Cluster Sum of Squares)
global_centroid <- colMeans(X)
cat("global_centroid", global_centroid, "\n")

bcss <- 0
for (i in 1:2) {
  cluster_points <- X[kmeans_result$cluster == i, ]
  cluster_size <- nrow(cluster_points)
  cluster_centroid <- centroids[i, ]
  bcsc <- bcsc + cluster_size * sum((cluster_centroid - global_centroid)^2)
}
```

K-means clustering with 2 clusters of sizes 3, 3

Cluster means:

	[,1]	[,2]
1	2	6.00000
2	13	19.33333

Clustering vector:

[1] 1 1 1 2 2 2

Sommiamo i valori di within, che rappresentano la somma delle distanze quadrate all'interno di ciascun cluster

Calcoliamo il centroide globale

Calcoliamo la somma delle distanze quadrate tra i centroidi dei cluster e il centroide globale, ponderata per la dimensione di ciascun cluster.



# WCSS, BCSS e Calinski-Harabasz in R

```
# Calcolare l'indice di Calinski-Harabasz
n <- nrow(X)
k <- length(unique(kmeans_result$cluster))
ch_index <- (bcss / (k - 1)) / (wcsc / (n - k))

# Risultati
cat("WCSS:", wcsc, "\n")
cat("BCSS:", bcsc, "\n")
cat("Calinski-Harabasz Index:", ch_index, "\n")
```



```
global_centroid 7.5 12.66667
WCSS: 10.66667
BCSS: 448.1667
Calinski-Harabasz Index: 168.0625
```

