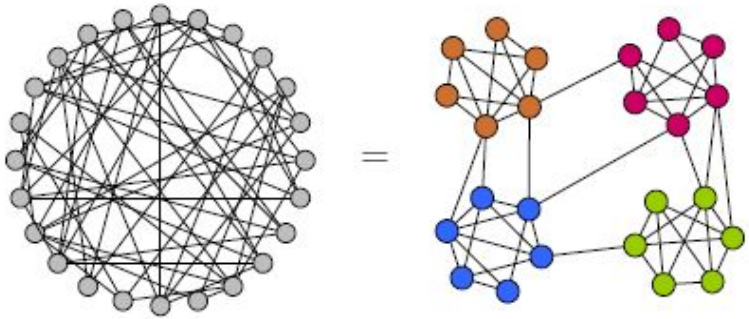
# AED Challenge

Team formation process at FME Dathaton



# Definición del Proyecto

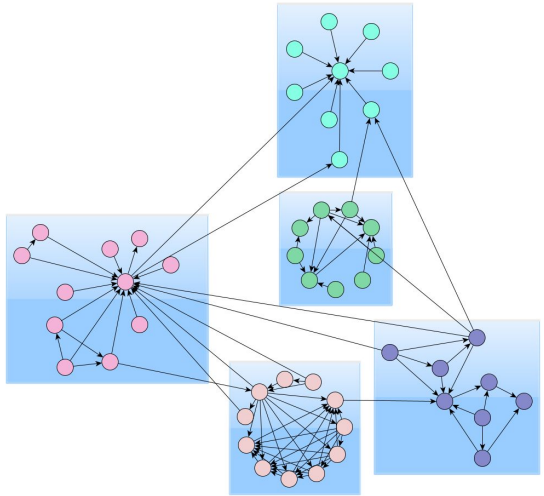
- A partir del enunciado del proyecto, podemos ver que se trata de un problema de Clustering.

- Debemos formar grupos de estudiantes perfectos.

- Consideramos la noción de "perfecto" como los estudiantes más similares entre sí.

# Formulación del Problema

- Podemos formular este problema como un problema de clustering en un grafo, donde los estudiantes son nodos y las distancias entre pares reflejan cuán similares son cada par de estudiantes.



# Formulación del Problema

- Podemos formular este problema como un problema de clustering en un grafo, donde los estudiantes son nodos y las distancias entre pares reflejan cuán similares son cada par de estudiantes.

- Para calcular las distancias, debemos usar los atributos proporcionados en el archivo JSON.

# Planteamiento Inicial - #1

- La formulación inicial fue la siguiente: la distancia $d_T$ entre dos estudiantes (dos nodos) será la suma de las distancias individuales $d_k$ correspondientes a cada atributo, donde $k \in [1, 2, \ldots m]$ siendo $m$ el número total de atributos.

- Por ejemplo, dos nodos tendrán una distancia $d_{k_1}$ asociada al atributo de `experience_level` y otra distancia $d_{k_2}$ asociada al atributo de `age`.

# Planteamiento Final - #2

- La nueva formulación propuesta consiste en asignar **pesos** a los atributos. De este modo, podemos determinar en el sistema qué atributos se consideran más importantes.

  - Pesos inferiores a 1 tendrán más impacto en que la distancia final sea pequeña e influirán en que nodos parecidos ( en este atributo ) serán más propensos a estar en el mismo cluster.

  - Pesos superiores a 1 tendrán más impacto en que la distancia final sea mayor e influirán en que nodos no parecidos (en este atributo ) serán más propensos a estar en clusters distintos.

# Mejora futura - #3

Asignación de **pesos personalizada** a los atributos por participante

Cada usuario cuando rellena el cuestionario de preferencias introduce nivel de importancia de cada preferencia

#2: Sistema asigna pesos
VS
#3: Usuario elige pesos

# Atributos del Dataset

- Dentro de los atributos proporcionados. Los podemos clasificar en tres tipos principales.
  - Atributos directos: Son atributos donde el cálculo de la distancia $d_k$ es (casi) directo.
  - Atributos de diccionario: son atributos que son diccionarios y que el cálculo de la distancia $d_k$ no es directo.
  - Atributos de texto: son atributos que son textos y que el cálculo de la distancia $d_k$ no es directo.

# Atributos Directos

- Dentro de este tipos tenemos tres subtipos.

  - Atributos Cuantitativos: mapeo uniforme al rango [0,1].

    - `age`: los valores de la edad, que inicialmente están en el rango [17,27], deben mapearse uniformemente al rango real [0,1], de manera que luego podamos calcular la distancia $d_k$ para este atributo entre cada par de estudiantes, calculada como la diferencia en valor absoluto.

    - `hacktahons_done`: Inicialmente en el rango [0,9], distribuidos uniformemente en el rango [0,1]. La distancia $d_k$ se calcula como la diferencia en valor absoluto.

# Atributos Directos

- Dentro de este tipos tenemos tres subtipos.

  - Atributos Categóricos Mapeados a Escala (mapeo a [0,1]).

    - `year_of_study`: los valores de este atributo deben asignarse dentro del rango [0,1]. Recordemos que los valores iniciales categóricos de este atributo son "1st year", "2nd year", "3rd year", "4th year", "Masters", "PhD"
      Así, se puede asignar al "1st year" el valor real más bajo y al "PhD" el valor real más alto. La distancia $d_k$ se calcula como la diferencia en valor absoluto.

# Atributos Directos

- Dentro de este tipos tenemos tres subtipos.

  - Atributos Categóricos Mapeados a Escala (mapeo a [0,1]).

    - `experience_level`: aquí los valores iniciales categóricos son "Beginner", "Intermediate" y "Advanced". Estos han sido mapeados uniformemente al rango [0,1] y la distancia $d_k$ se calcula como la diferencia en valor absoluto.

# Atributos Directos

- Dentro de este tipos tenemos tres subtipos.

  - Atributos Categóricos No Mapeados a Escala.

    - `university`: $d_k$ es 0 si son la misma universidad, y 1 en caso contrario).

    - `dietary_restrictions`: $d_k$ es 0 si coinciden, 1 en caso contrario

    - `preferred_role`: si coincide la distancia $d_k$ es 1, en caso contrario es 0. → Favorece formación de grupos con perfiles variados/complementarios.

# Atributos de Diccionario

- `programming_skills`: interpretamos los diccionarios como vectores donde cada skill es un índice (dimensión) en el vector. La distancia $d_k$ se calcula a partir de la cosine similarity.

- `interests`, `interest_in_challenges` y `preferred_languages`: Estos atributos se representan como conjuntos de valores categóricos. Para comparar la superposición entre estos conjuntos (distancia $d_k$) , se utiliza la Jaccard distance.

# Atributos de Diccionario

- `friend_registration`: usamos un checkeo binario de coincidencia. Si dos participantes tienen amigos en común, la distancia $d_k$ se establece en 0, de lo contrario, la distancia $d_k$ se establece en 1.

- `availability`: Para medir la alineación de los cronogramas de disponibilidad ($d_k$), utilizamos la distancia de Hamming, que calcula la proporción de valores diferentes entre dos diccionarios.

# Atributos de Texto

- Hemos separado estos atributos de texto en dos tipos, dependiendo del uso que les hemos dado en nuestro algoritmo.
  - Atributos de texto de clasificación: usamos la similitud entre un texto y categorías predefinidas para clasificar esa frase en una de las categorías específicas. Esta tipología solo consta de el atributos objective, para el cual creamos cuatro categorías: win, learn, meet and experience, con el fin de agrupar los objetivos de los participantes. Si la categoría objetivo entre dos participantes es la misma, entonces el componente objetivo en la distancia entre esos dos participantes es 0; mientras que en caso contrario, es 1.

# Atributos de Texto

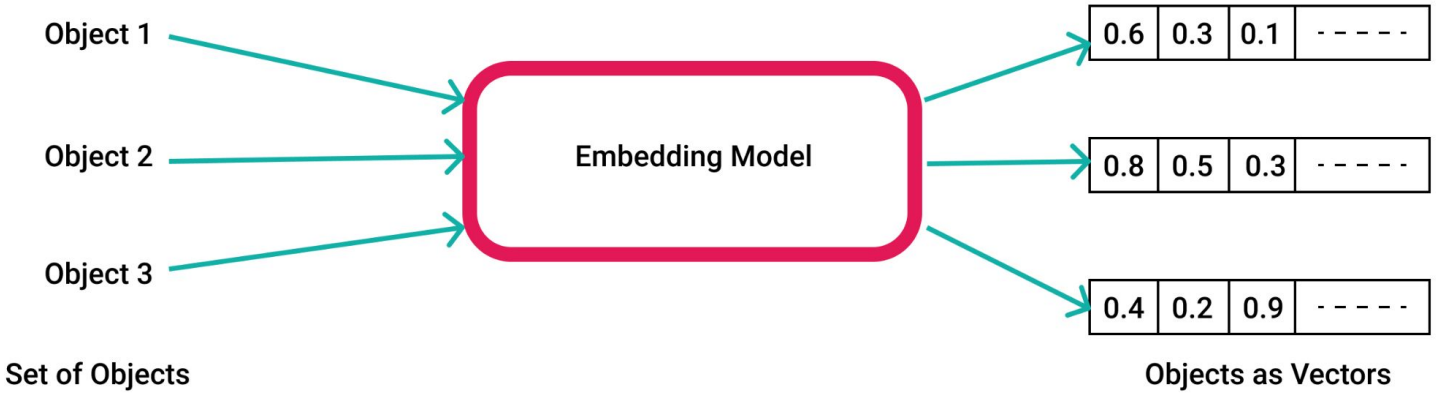
- Hemos separado estos atributos de texto en dos tipos, dependiendo del uso que les hemos dado en nuestro algoritmo.
  - El segundo tipo consta de los atributos de texto de similitud, donde utilizamos la similitud entre dos textos para obtener el valor exacto del componente objetivo entre esos dos participantes. Forman parte de esta tipología atributos como introductions, technical project, future excitement, etc.

# Atributos de Texto

- Inicialmente, consideramos usar la coincidencia de palabras o cadenas para clasificar o encontrar similitudes entre los textos, pero una reflexión más profunda sobre el tema nos mostró que, al usar esta técnica, no podíamos diferenciar contexto del texto (frases como “My objective is not to learn but to…”, homónimos, etc no era clasificados de forma apropiada). Por esta razón decidimos utilizar una técnica más potente: el embedding de texto.

# Atributos de Texto

- El embedding de texto captura el significado de una frase o párrafo como un vector, por medio del uso lo cual nos permite calcular la similitud entre dos fragmentos de texto usando la similitud de coseno entre esos dos vectores



# Algoritmo de clustering

Modelado del problema como un problema de clustering en un grafo G = (V,E) - knowledge graph - donde cada nodo tiene propiedades.

- Cada participante es un nodo
- Cada atributo de los participantes es una propiedad del nodo

- Nuestra propuesta: Algoritmo voraz
- Otras posibilidades:
  - K - means: minimiza distancia intra-cluster (entre los del mismo cluster) y maximiza distancia inter-cluster.

**Algorithm 1** Greedy Algorithm - Cluster formation

**Require:** $G = (V, E)$ is the graph of participants.
**Require:** $G$ is a complete graph with distance metric calculated for each pair of nodes.
**Require:** `cluster_size` is the max/desired cluster-group size.

$C \leftarrow \emptyset$ {$C$: set of clusters}
$U \leftarrow V$ {$U$: set of unvisited nodes}
**while** $U \neq \emptyset$ **do**
    $C_{new} \leftarrow \emptyset$ {$C_{new}$: new cluster/set of nodes}
    $u \leftarrow U$ {Extract one random unvisited node $u$ of the set $U$}
    $U = U - \{u\}$
    $C_{new} \leftarrow C_{new} + u$ {Add to the new cluster $C_{new}$}
    **while** $|C_{new}| <$ `cluster_size` $\wedge\, U \neq \emptyset$ **do**
        `closest` $\leftarrow min\_distance(u, U)$
        $C_{new} \leftarrow$ `closest`
        $U = U - \{$`closest`$\}$
    **end while**
    $C \leftarrow C + \{C_{new}\}$
**end while**
**return** $C$

# Juegos de Prueba

- Se han elaborado cuatro soluciones al problema.
  - Formulación inicial (sin pesos) con los atributos directos. (#1)
  - Formulación final (con pesos) con los atributos directos. (#2)
  - Formulación final (con pesos) con los atributos directos y atributos de diccionario. (#3)
  - Formulación final (con pesos) con los atributos directos, atributos de diccionario y atributos de texto. (#4)

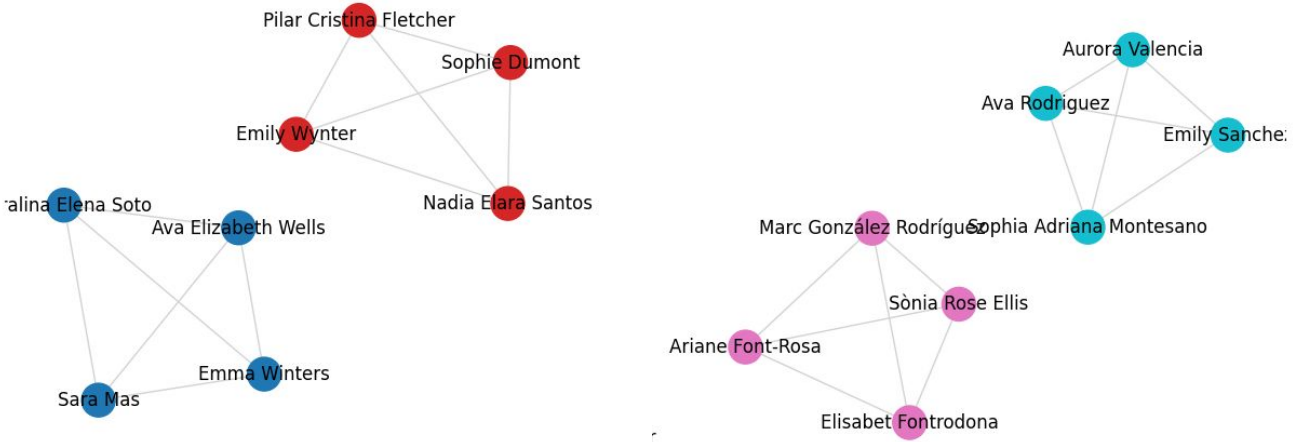
# Juegos de Prueba

- Supongamos que queremos ver los clusters (grupos) de los siguientes estudiantes:

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]

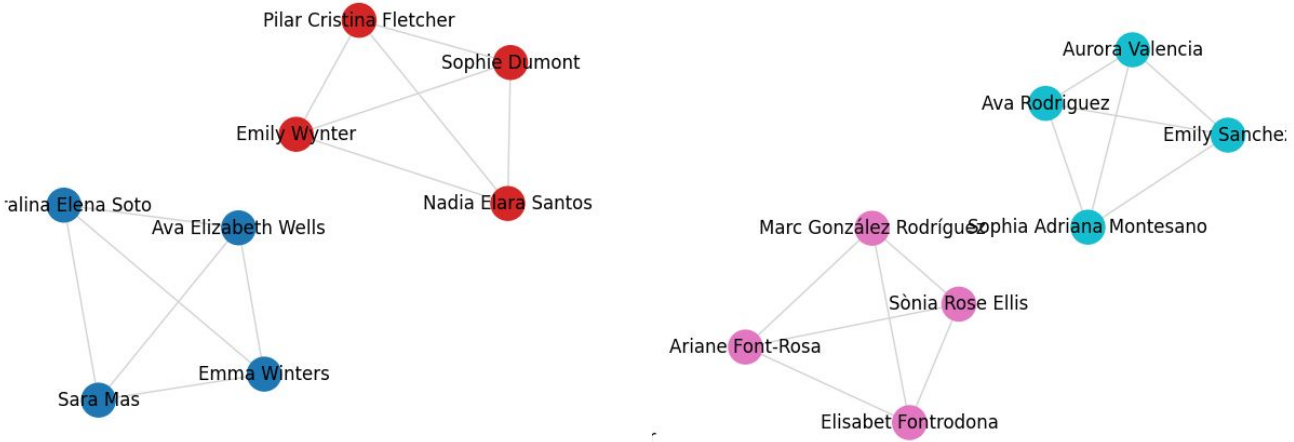
# Resultado con Solución (#1)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



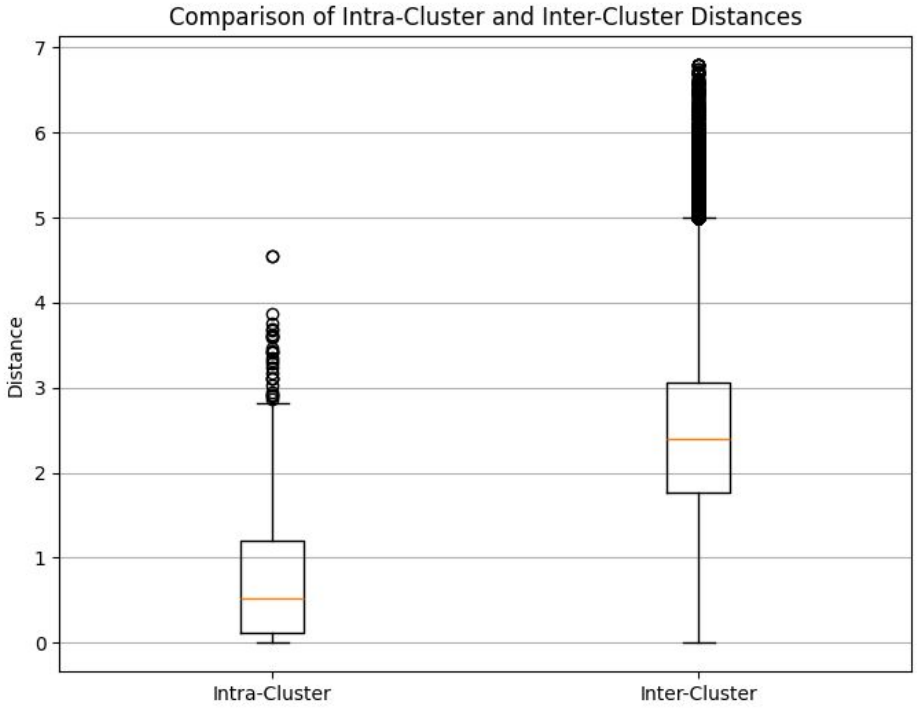
# Resultado con Solución (#1)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



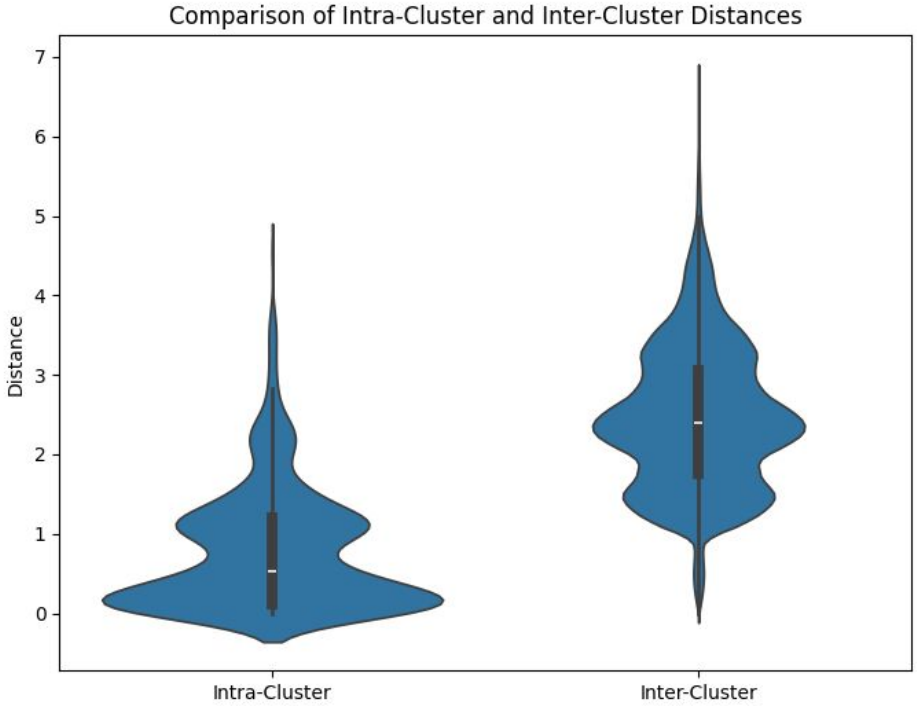
# Resultado con Solución (#1)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



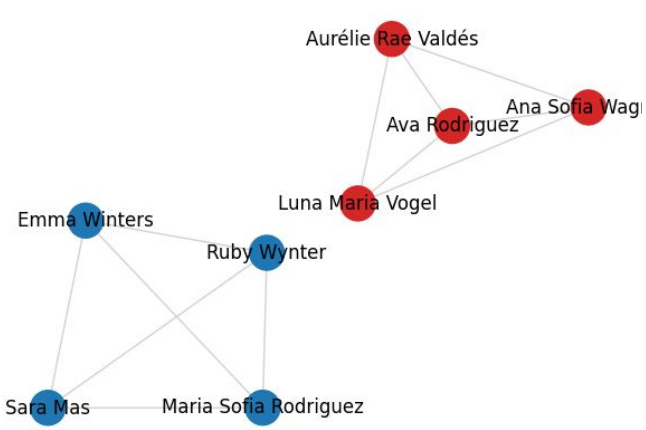
# Resultado con Solución (#1)

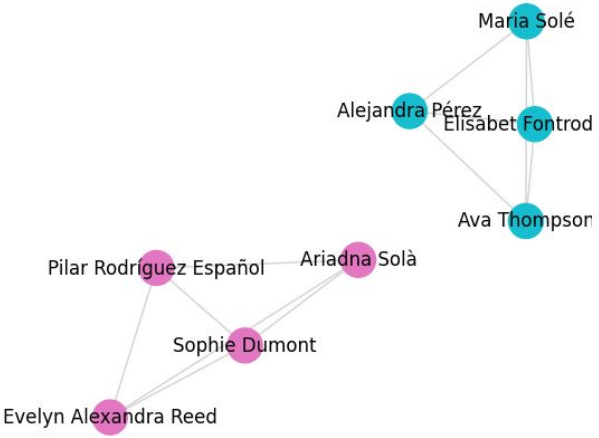
["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



# Resultado con Solución (#2)

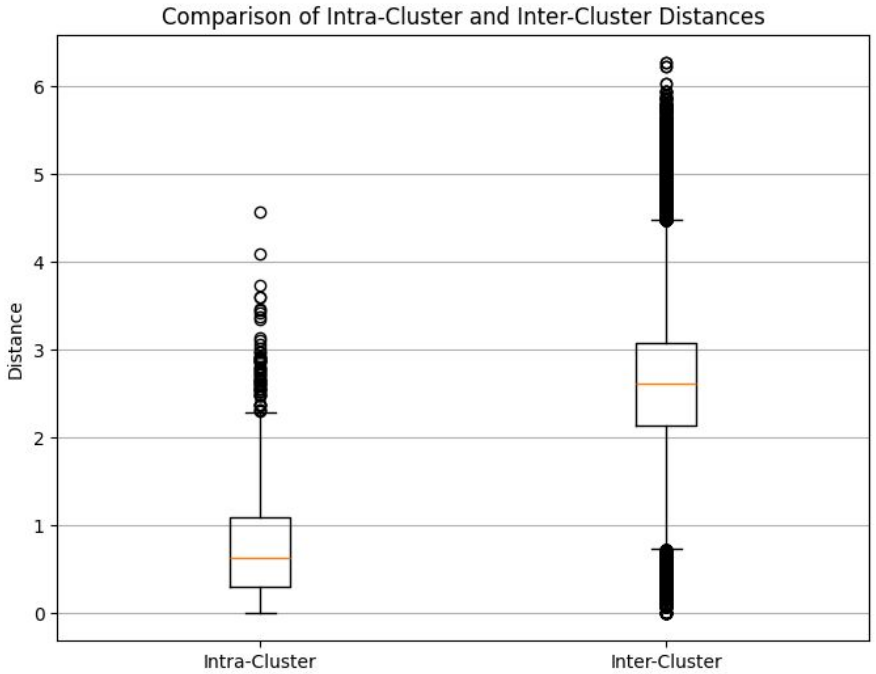
["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



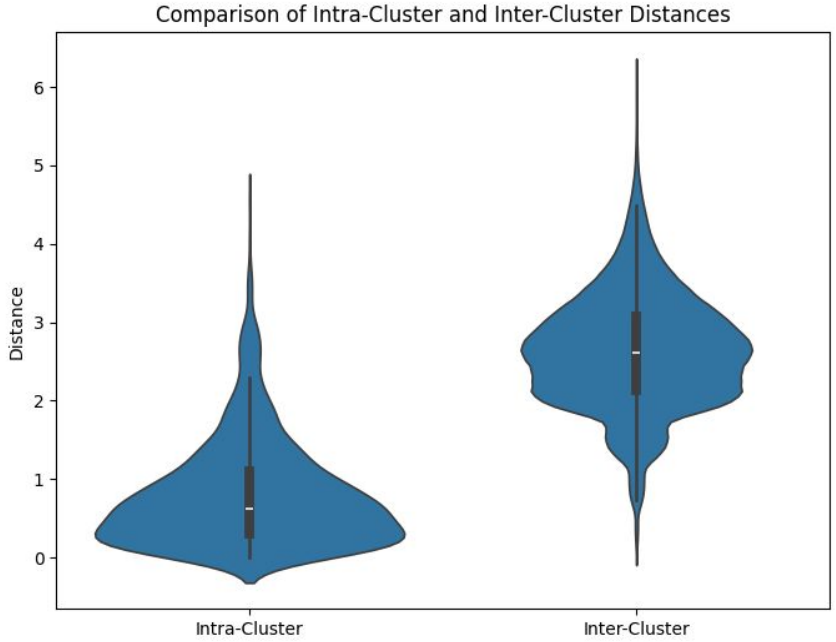
# Resultado con Solución (#2)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



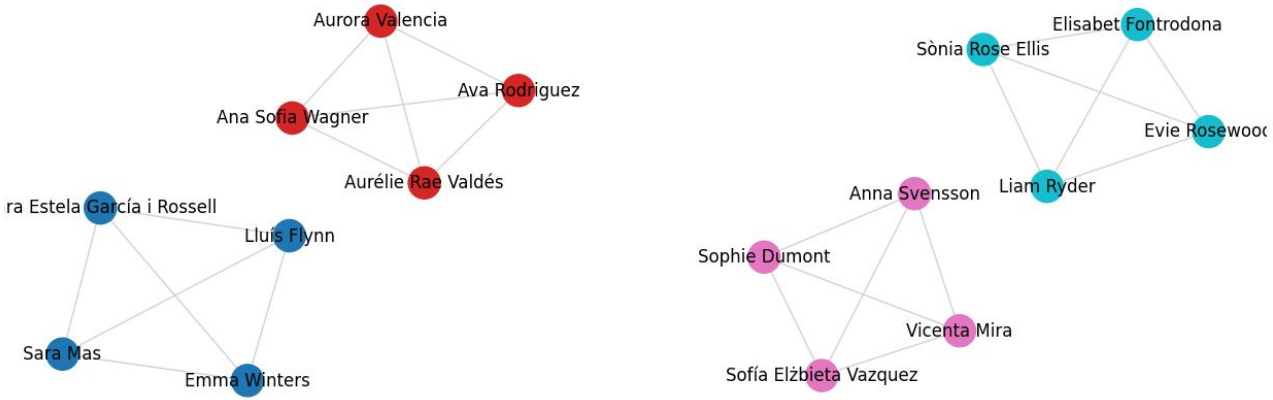
# Resultado con Solución (#2)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



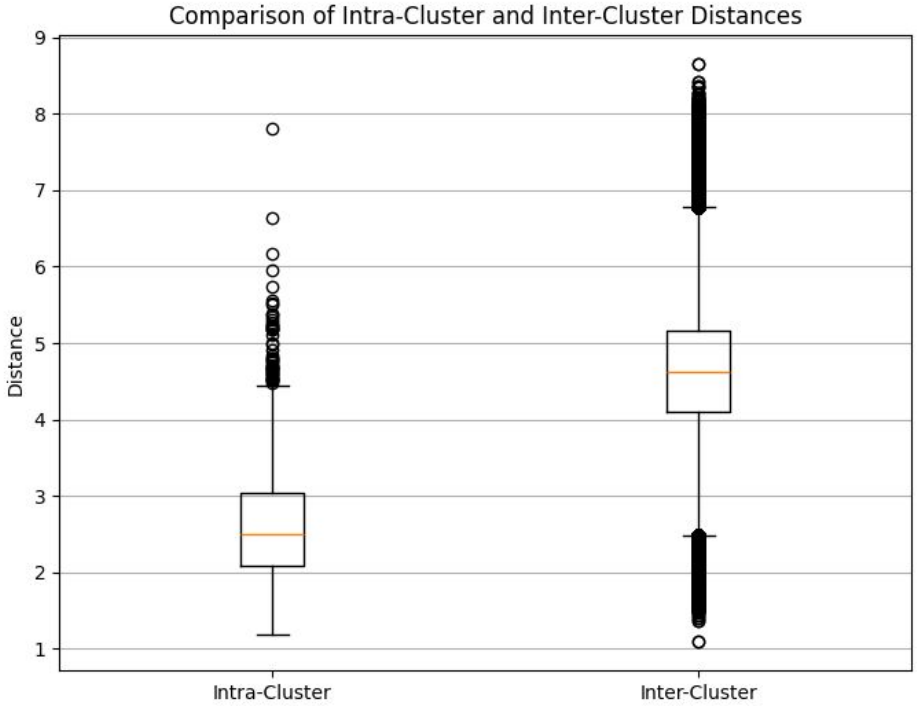
# Resultado con Solución (#3)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



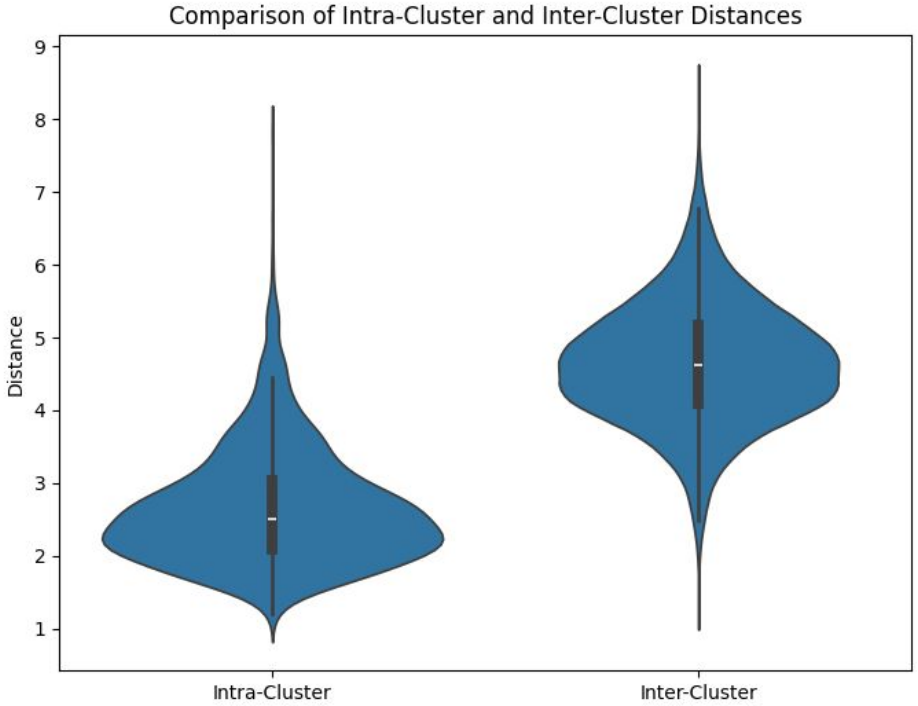
# Resultado con Solución (#3)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



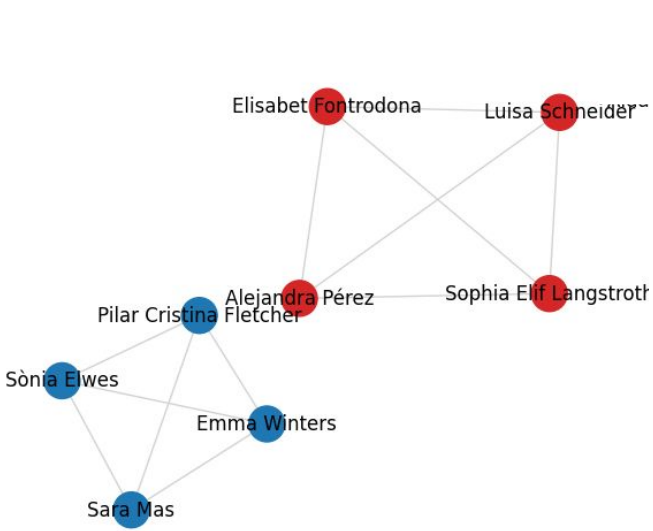
# Resultado con Solución (#3)

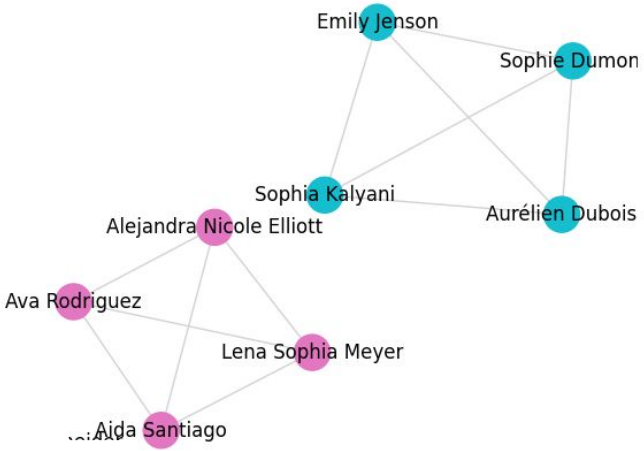
["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



# Resultado con Solución (#4)

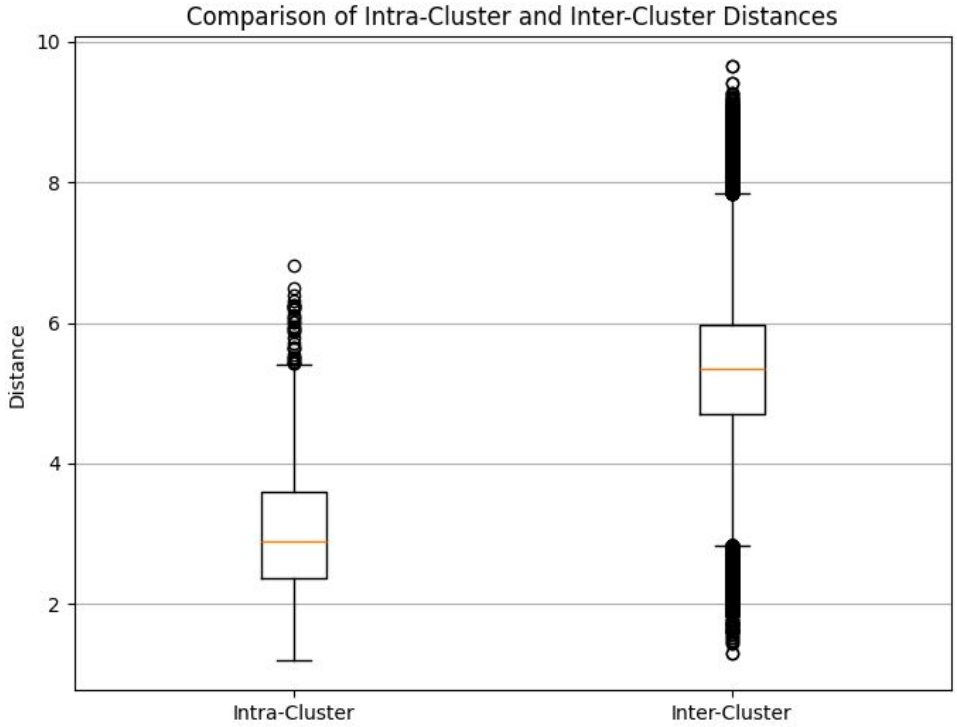
["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



# Resultado con Solución (#4)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]



# Resultado con Solución (#4)

["Emma Winters", "Sophie Dumont", "Elisabet Fontrodona", "Ava Rodriguez"]