Distancias

In [4]:
```
from scipy.spatial import distance_matrix
import pandas as pd
```

In [5]:
```
data = pd.read_csv("../../Data-Sets/datasets/movies/movies.csv", sep=";")
```

In [6]:
```
data
```

Out[6]:

	user_id	star_wars	lord_of_the_rings	harry_potter
0	1	1.2	4.9	2.1
1	2	2.1	8.1	7.9
2	3	7.4	3.0	9.9
3	4	5.6	0.5	1.8
4	5	1.5	8.3	2.6
5	6	2.5	3.7	6.5
6	7	2.0	8.2	8.5
7	8	1.8	9.3	4.5
8	9	2.6	1.7	3.1
9	10	1.5	4.7	2.3

In [7]:
```
movies = data.columns.values.tolist()[1:]
movies
```

Out[7]:
```
['star_wars', 'lord_of_the_rings', 'harry_potter']
```

In [8]:
```
dd1 = distance_matrix(data[movies], data[movies], p=1)
dd2 = distance_matrix(data[movies], data[movies], p=2)
dd10 = distance_matrix(data[movies], data[movies], p=10)
```

In [9]:
```
def dm_to_df(dd, col_name):
    import pandas as pd
    return pd.DataFrame(dd, index=col_name, columns=col_name)
```

In [10]:
```
dm_to_df(dd1, data["user_id"])
```

Out[10]:

user_id	1	2	3	4	5	6	7	8	9	10
user_id										
1	0.0	9.9	15.9	9.1	4.2	6.9	10.5	7.4	5.6	0.7
2	9.9	0.0	12.4	17.2	6.1	6.2	0.8	4.9	11.7	9.6

user_id	1	2	3	4	5	6	7	8	9	10
user_id										
3	15.9	12.4	0.0	12.4	18.5	9.0	12.0	17.3	12.9	15.2
4	9.1	17.2	12.4	0.0	12.7	11.0	18.0	15.3	5.5	8.8
5	4.2	6.1	18.5	12.7	0.0	9.5	6.5	3.2	8.2	3.9
6	6.9	6.2	9.0	11.0	9.5	0.0	7.0	8.3	5.5	6.2
7	10.5	0.8	12.0	18.0	6.5	7.0	0.0	5.3	12.5	10.2
8	7.4	4.9	17.3	15.3	3.2	8.3	5.3	0.0	9.8	7.1
9	5.6	11.7	12.9	5.5	8.2	5.5	12.5	9.8	0.0	4.9
10	0.7	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0

In [11]:
```
dm_to_df(dd2, data["user_id"])
```

Out[11]:

user_id	1	2	3	4	5	6	7	8
user_id								
1	0.000000	6.685058	10.143471	6.229767	3.449638	4.742362	7.244998	5.047772
2	6.685058	0.000000	7.622336	10.354709	5.337602	4.634652	0.616441	3.618011
3	10.143471	7.622336	0.000000	8.666026	10.779147	6.004998	7.626270	10.010494
4	6.229767	10.354709	8.666026	0.000000	8.848164	6.476110	10.823123	9.958414
5	3.449638	5.337602	10.779147	8.848164	0.000000	6.113101	5.921993	2.167948
6	4.742362	4.634652	6.004998	6.476110	6.113101	0.000000	4.949747	5.987487
7	7.244998	0.616441	7.626270	10.823123	5.921993	4.949747	0.000000	4.153312
8	5.047772	3.618011	10.010494	9.958414	2.167948	5.987487	4.153312	0.000000
9	3.633180	8.015610	8.424369	3.482815	6.709694	3.945884	8.471718	7.769170
10	0.412311	6.578754	9.770363	5.890671	3.612478	4.431704	7.137226	5.107837

In [12]:
```
dm_to_df(dd10, data["user_id"])
```

Out[12]:

user_id	1	2	3	4	5	6	7	8	9
user_id									
1	0.000000	5.801514	7.875189	4.715803	3.400000	4.400003	6.400850	4.401025	3.200085
2	5.801514	0.000000	5.582463	7.680689	5.300000	4.400005	0.600000	3.400010	6.435159
3	7.875189	5.582463	0.000000	8.100007	7.408914	4.912532	5.689450	6.570254	6.820602
4	4.715803	7.680689	8.100007	0.000000	7.801255	4.717102	7.873307	8.800205	3.000101
5	3.400000	5.300000	7.408914	7.801255	0.000000	4.681464	5.900000	1.900310	6.600000
6	4.400003	4.400005	4.912532	4.717102	4.681464	0.000000	4.500135	5.600019	3.401683
7	6.400850	0.600000	5.689450	7.873307	5.900000	4.500135	0.000000	4.000001	6.595259

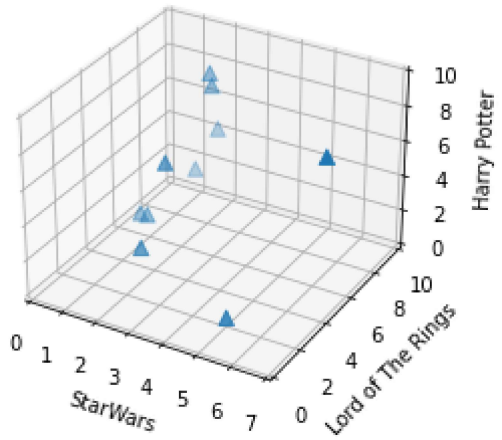
user_id	1	2	3	4	5	6	7	8	9	
user_id										
8	4.401025	3.400010	6.570254	8.800205	1.900310	5.600019	4.000001	0.000000	7.600000	4.6
9	3.200085	6.435159	6.820602	3.000101	6.600000	3.401683	6.595259	7.600000	0.000000	3.(
10	0.301025	5.603800	7.658364	4.450759	3.600000	4.200000	6.202035	4.600288	3.000014	0.(

Se nota que al incrementar el p la distancia entre puntos disminuye.

In [13]:
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

In [14]:
```
fig = plt.figure()
ax = fig.add_subplot(1,1,1, projection = "3d")
ax.set_zlim(0, 10)
ax.set_ylim(0, 10)
ax.set_xlim(0, 7)
ax.set_xlabel('StarWars')
ax.set_ylabel('Lord of The Rings')
ax.set_zlabel('Harry Potter')
ax.scatter(xs= data["star_wars"], ys=data["lord_of_the_rings"], zs=data["harry_potte
```

Out[14]:
```
<mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x1b50a3e2e50>
```



Distancias

In [15]:
```
df=dm_to_df(dd1, data["user_id"])
df
```

Out[15]:

user_id	1	2	3	4	5	6	7	8	9	10
user_id										
1	0.0	9.9	15.9	9.1	4.2	6.9	10.5	7.4	5.6	0.7
2	9.9	0.0	12.4	17.2	6.1	6.2	0.8	4.9	11.7	9.6
3	15.9	12.4	0.0	12.4	18.5	9.0	12.0	17.3	12.9	15.2
4	9.1	17.2	12.4	0.0	12.7	11.0	18.0	15.3	5.5	8.8

user_id	1	2	3	4	5	6	7	8	9	10
user_id										
5	4.2	6.1	18.5	12.7	0.0	9.5	6.5	3.2	8.2	3.9
6	6.9	6.2	9.0	11.0	9.5	0.0	7.0	8.3	5.5	6.2
7	10.5	0.8	12.0	18.0	6.5	7.0	0.0	5.3	12.5	10.2
8	7.4	4.9	17.3	15.3	3.2	8.3	5.3	0.0	9.8	7.1
9	5.6	11.7	12.9	5.5	8.2	5.5	12.5	9.8	0.0	4.9
10	0.7	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0

In [16]:
```
z=[]
```

In [17]:
```
df[11] = df[1]+df[10] #Creamos la columna 11 con la suma de las dos columnas, 1 y 10
df.loc[11] = df.loc[1]+df.loc[10] # Creamos la fila 11 nuevamente con la suma de fil
z.append([1,10,0.7,2])#id1, id2, d, n_elemenmtos_en_cluster -> Dato 11
df
```

Out[17]:

user_id	1	2	3	4	5	6	7	8	9	10	11
user_id											
1	0.0	9.9	15.9	9.1	4.2	6.9	10.5	7.4	5.6	0.7	0.7
2	9.9	0.0	12.4	17.2	6.1	6.2	0.8	4.9	11.7	9.6	19.5
3	15.9	12.4	0.0	12.4	18.5	9.0	12.0	17.3	12.9	15.2	31.1
4	9.1	17.2	12.4	0.0	12.7	11.0	18.0	15.3	5.5	8.8	17.9
5	4.2	6.1	18.5	12.7	0.0	9.5	6.5	3.2	8.2	3.9	8.1
6	6.9	6.2	9.0	11.0	9.5	0.0	7.0	8.3	5.5	6.2	13.1
7	10.5	0.8	12.0	18.0	6.5	7.0	0.0	5.3	12.5	10.2	20.7
8	7.4	4.9	17.3	15.3	3.2	8.3	5.3	0.0	9.8	7.1	14.5
9	5.6	11.7	12.9	5.5	8.2	5.5	12.5	9.8	0.0	4.9	10.5
10	0.7	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0	0.7
11	0.7	19.5	31.1	17.9	8.1	13.1	20.7	14.5	10.5	0.7	1.4

In [18]:
```
for i in df.columns.values.tolist():
    df.loc[11][i] = min(df.loc[1][i], df.loc[10][i])# Reemplaza por el mínimo que ha
    df.loc[i][11] = min(df.loc[i][1], df.loc[i][10])
df
```

Out[18]:

user_id	1	2	3	4	5	6	7	8	9	10	11
user_id											
1	0.0	9.9	15.9	9.1	4.2	6.9	10.5	7.4	5.6	0.7	0.0
2	9.9	0.0	12.4	17.2	6.1	6.2	0.8	4.9	11.7	9.6	9.6
3	15.9	12.4	0.0	12.4	18.5	9.0	12.0	17.3	12.9	15.2	15.2

user_id	1	2	3	4	5	6	7	8	9	10	11
user_id											
4	9.1	17.2	12.4	0.0	12.7	11.0	18.0	15.3	5.5	8.8	8.8
5	4.2	6.1	18.5	12.7	0.0	9.5	6.5	3.2	8.2	3.9	3.9
6	6.9	6.2	9.0	11.0	9.5	0.0	7.0	8.3	5.5	6.2	6.2
7	10.5	0.8	12.0	18.0	6.5	7.0	0.0	5.3	12.5	10.2	10.2
8	7.4	4.9	17.3	15.3	3.2	8.3	5.3	0.0	9.8	7.1	7.1
9	5.6	11.7	12.9	5.5	8.2	5.5	12.5	9.8	0.0	4.9	4.9
10	0.7	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0	0.0
11	0.0	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0	0.0

In [19]:
```
df = df.drop([1,10])
df = df.drop([1,10], axis = 1)
df
```

Out[19]:

user_id	2	3	4	5	6	7	8	9	11
user_id									
2	0.0	12.4	17.2	6.1	6.2	0.8	4.9	11.7	9.6
3	12.4	0.0	12.4	18.5	9.0	12.0	17.3	12.9	15.2
4	17.2	12.4	0.0	12.7	11.0	18.0	15.3	5.5	8.8
5	6.1	18.5	12.7	0.0	9.5	6.5	3.2	8.2	3.9
6	6.2	9.0	11.0	9.5	0.0	7.0	8.3	5.5	6.2
7	0.8	12.0	18.0	6.5	7.0	0.0	5.3	12.5	10.2
8	4.9	17.3	15.3	3.2	8.3	5.3	0.0	9.8	7.1
9	11.7	12.9	5.5	8.2	5.5	12.5	9.8	0.0	4.9
11	9.6	15.2	8.8	3.9	6.2	10.2	7.1	4.9	0.0

In [23]:
```
df[12] = df[2]+df[7] #Creamos la columna 11 con la suma de las dos columnas, 1 y 10
df.loc[12] = df.loc[2]+df.loc[7] # Creamos la fila 11 nuevamente con la suma de fila
z.append([2,7,0.8,2])#id1, id2, d, n_elemenmtos_en_cluster -> Dato 12
for i in df1.columns.values.tolist():
    df.loc[12][i] = min(df.loc[2][i], df.loc[7][i])# Reemplaza por el mínimo que hay
    df.loc[i][12] = min(df.loc[i][2], df.loc[i][7])
df = df.drop([2,7])
df = df.drop([2,7], axis = 1)
```

In [24]:
```
dm_to_df(dd1, data["user_id"])
df=dm_to_df(dd1, data["user_id"])
df1=df
```

In [25]:
```
#Función de automatización de la matriz de distancias.
def Distance_matriz(df1):
    z=[]
```

```
    long = len(df1)
    n=long+1
    d_min_flag=999999
    while(len(df1)!=1):
        for j in (df1.columns.values.tolist()[0:]):
            for k in (df1.index[0:]):
                if (k != j):
                    print("J= ", j, " - K=", k)
                    d_min = round(df1[j][k],1)
                    if (d_min_flag >= d_min):
                        if (d_min_flag == d_min):
                            print("IF iGUAL--------------")
                            d_min_flag = d_min
                            zz = int(k)
                        else:
                            print("IF3 DIFERENTE------------")
                            d_min_flag = d_min
                            x = int(j)
                            y = int(k)
                            zz = x
        df1[n] = df1[x]+df1[y]
        df1.loc[n] = df1.loc[x]+df1.loc[y]
        if(x != zz):
            print("ELIMINACIÓN CON Z")
            print("X: ", x, " Y: ", y , " Z: ", zz)
            z.append([x,zz,y,d_min_flag,3])
            for l in df1.columns.values.tolist():
                df1.loc[n][l] = min(df1.loc[x][l], df1.loc[zz][l], df1.loc[y][l])
                df1.loc[l][n] = min(df1.loc[l][x], df1.loc[l][zz], df1.loc[l][y])
            df2 = df1.drop([x,y,zz])
            df1 = df2
            df2 = df1.drop([x,y,zz], axis = 1)
            df1 = df2
        elif(x == zz):
            print("ELIMINACIÓN SIN Z")
            print("X: ", x, " Y: ", y , " Z: ", zz)
            z.append([x,y,d_min_flag,2])
            for l in df1.columns.values.tolist():
                df1.loc[n][l] = min(df1.loc[y][l], df1.loc[x][l])
                df1.loc[l][n] = min(df1.loc[l][y], df1.loc[l][x])
            df2 = df1.drop([x,y])
            df1 = df2
            df2 = df1.drop([x,y], axis = 1)
            df1 = df2
        print ("Distancia mínima: ", d_min_flag)
        print ("")
        print(df2)
        d_min_flag = 999999
        n +=1
        print("-----------------------------------------------------------------")
    return (df2,z)
```

In [26]:

```
df2, z = Distance_matriz(df1)
```

```
J=  1  - K= 2
IF3 DIFERENTE------------
J=  1  - K= 3
J=  1  - K= 4
IF3 DIFERENTE------------
J=  1  - K= 5
IF3 DIFERENTE------------
J=  1  - K= 6
```

```
J=  1  - K= 7
J=  1  - K= 8
J=  1  - K= 9
J=  1  - K= 10
IF3 DIFERENTE------------
J=  2  - K= 1
J=  2  - K= 3
J=  2  - K= 4
J=  2  - K= 5
J=  2  - K= 6
J=  2  - K= 7
J=  2  - K= 8
J=  2  - K= 9
J=  2  - K= 10
J=  3  - K= 1
J=  3  - K= 2
J=  3  - K= 4
J=  3  - K= 5
J=  3  - K= 6
J=  3  - K= 7
J=  3  - K= 8
J=  3  - K= 9
J=  3  - K= 10
J=  4  - K= 1
J=  4  - K= 2
J=  4  - K= 3
J=  4  - K= 5
J=  4  - K= 6
J=  4  - K= 7
J=  4  - K= 8
J=  4  - K= 9
J=  4  - K= 10
J=  5  - K= 1
J=  5  - K= 2
J=  5  - K= 3
J=  5  - K= 4
J=  5  - K= 6
J=  5  - K= 7
J=  5  - K= 8
J=  5  - K= 9
J=  5  - K= 10
J=  6  - K= 1
J=  6  - K= 2
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 5
J=  6  - K= 7
J=  6  - K= 8
J=  6  - K= 9
J=  6  - K= 10
J=  7  - K= 1
J=  7  - K= 2
J=  7  - K= 3
J=  7  - K= 4
J=  7  - K= 5
J=  7  - K= 6
J=  7  - K= 8
J=  7  - K= 9
J=  7  - K= 10
J=  8  - K= 1
J=  8  - K= 2
J=  8  - K= 3
J=  8  - K= 4
J=  8  - K= 5
```

```
J=  8  - K= 6
J=  8  - K= 7
J=  8  - K= 9
J=  8  - K= 10
J=  9  - K= 1
J=  9  - K= 2
J=  9  - K= 3
J=  9  - K= 4
J=  9  - K= 5
J=  9  - K= 6
J=  9  - K= 7
J=  9  - K= 8
J=  9  - K= 10
J=  10  - K= 1
IF iGUAL--------------
J=  10  - K= 2
J=  10  - K= 3
J=  10  - K= 4
J=  10  - K= 5
J=  10  - K= 6
J=  10  - K= 7
J=  10  - K= 8
J=  10  - K= 9
ELIMINACIÓN SIN Z
X:  1  Y:  10  Z:  1
Distancia mínima:  0.7

user_id    2     3     4     5     6     7     8     9     11
user_id
2         0.0  12.4  17.2   6.1   6.2   0.8   4.9  11.7   9.6
3        12.4   0.0  12.4  18.5   9.0  12.0  17.3  12.9  15.2
4        17.2  12.4   0.0  12.7  11.0  18.0  15.3   5.5   8.8
5         6.1  18.5  12.7   0.0   9.5   6.5   3.2   8.2   3.9
6         6.2   9.0  11.0   9.5   0.0   7.0   8.3   5.5   6.2
7         0.8  12.0  18.0   6.5   7.0   0.0   5.3  12.5  10.2
8         4.9  17.3  15.3   3.2   8.3   5.3   0.0   9.8   7.1
9        11.7  12.9   5.5   8.2   5.5  12.5   9.8   0.0   4.9
11        9.6  15.2   8.8   3.9   6.2  10.2   7.1   4.9   0.0
------------------------------------------------------------
J=  2  - K= 3
IF3 DIFERENTE------------
J=  2  - K= 4
J=  2  - K= 5
IF3 DIFERENTE------------
J=  2  - K= 6
J=  2  - K= 7
IF3 DIFERENTE------------
J=  2  - K= 8
J=  2  - K= 9
J=  2  - K= 11
J=  3  - K= 2
J=  3  - K= 4
J=  3  - K= 5
J=  3  - K= 6
J=  3  - K= 7
J=  3  - K= 8
J=  3  - K= 9
J=  3  - K= 11
J=  4  - K= 2
J=  4  - K= 3
J=  4  - K= 5
J=  4  - K= 6
J=  4  - K= 7
J=  4  - K= 8
```

```
J=  4  - K= 9
J=  4  - K= 11
J=  5  - K= 2
J=  5  - K= 3
J=  5  - K= 4
J=  5  - K= 6
J=  5  - K= 7
J=  5  - K= 8
J=  5  - K= 9
J=  5  - K= 11
J=  6  - K= 2
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 5
J=  6  - K= 7
J=  6  - K= 8
J=  6  - K= 9
J=  6  - K= 11
J=  7  - K= 2
IF iGUAL--------------
J=  7  - K= 3
J=  7  - K= 4
J=  7  - K= 5
J=  7  - K= 6
J=  7  - K= 8
J=  7  - K= 9
J=  7  - K= 11
J=  8  - K= 2
J=  8  - K= 3
J=  8  - K= 4
J=  8  - K= 5
J=  8  - K= 6
J=  8  - K= 7
J=  8  - K= 9
J=  8  - K= 11
J=  9  - K= 2
J=  9  - K= 3
J=  9  - K= 4
J=  9  - K= 5
J=  9  - K= 6
J=  9  - K= 7
J=  9  - K= 8
J=  9  - K= 11
J=  11  - K= 2
J=  11  - K= 3
J=  11  - K= 4
J=  11  - K= 5
J=  11  - K= 6
J=  11  - K= 7
J=  11  - K= 8
J=  11  - K= 9
ELIMINACIÓN SIN Z
X:  2  Y:  7  Z:  2
Distancia mínima:  0.8

user_id    3     4     5     6     8     9     11    12
user_id
3         0.0  12.4  18.5   9.0  17.3  12.9  15.2  12.0
4        12.4   0.0  12.7  11.0  15.3   5.5   8.8  17.2
5        18.5  12.7   0.0   9.5   3.2   8.2   3.9   6.1
6         9.0  11.0   9.5   0.0   8.3   5.5   6.2   6.2
8        17.3  15.3   3.2   8.3   0.0   9.8   7.1   4.9
9        12.9   5.5   8.2   5.5   9.8   0.0   4.9  11.7
11       15.2   8.8   3.9   6.2   7.1   4.9   0.0   9.6
```

```
12       12.0  17.2   6.1   6.2   4.9  11.7   9.6   0.0
------------------------------------------------------------
J=  3  - K= 4
IF3 DIFERENTE------------
J=  3  - K= 5
J=  3  - K= 6
IF3 DIFERENTE------------
J=  3  - K= 8
J=  3  - K= 9
J=  3  - K= 11
J=  3  - K= 12
J=  4  - K= 3
J=  4  - K= 5
J=  4  - K= 6
J=  4  - K= 8
J=  4  - K= 9
IF3 DIFERENTE------------
J=  4  - K= 11
J=  4  - K= 12
J=  5  - K= 3
J=  5  - K= 4
J=  5  - K= 6
J=  5  - K= 8
IF3 DIFERENTE------------
J=  5  - K= 9
J=  5  - K= 11
J=  5  - K= 12
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 5
J=  6  - K= 8
J=  6  - K= 9
J=  6  - K= 11
J=  6  - K= 12
J=  8  - K= 3
J=  8  - K= 4
J=  8  - K= 5
IF iGUAL--------------
J=  8  - K= 6
J=  8  - K= 9
J=  8  - K= 11
J=  8  - K= 12
J=  9  - K= 3
J=  9  - K= 4
J=  9  - K= 5
J=  9  - K= 6
J=  9  - K= 8
J=  9  - K= 11
J=  9  - K= 12
J=  11  - K= 3
J=  11  - K= 4
J=  11  - K= 5
J=  11  - K= 6
J=  11  - K= 8
J=  11  - K= 9
J=  11  - K= 12
J=  12  - K= 3
J=  12  - K= 4
J=  12  - K= 5
J=  12  - K= 6
J=  12  - K= 8
J=  12  - K= 9
J=  12  - K= 11
ELIMINACIÓN SIN Z
```

```
X:  5  Y:  8  Z:  5
Distancia mínima:  3.2

user_id    3     4     6     9     11    12    13
user_id
3         0.0  12.4   9.0  12.9  15.2  12.0  17.3
4        12.4   0.0  11.0   5.5   8.8  17.2  12.7
6         9.0  11.0   0.0   5.5   6.2   6.2   8.3
9        12.9   5.5   5.5   0.0   4.9  11.7   8.2
11       15.2   8.8   6.2   4.9   0.0   9.6   3.9
12       12.0  17.2   6.2  11.7   9.6   0.0   4.9
13       17.3  12.7   8.3   8.2   3.9   4.9   0.0
------------------------------------------------------------
J=  3  - K= 4
IF3 DIFERENTE------------
J=  3  - K= 6
IF3 DIFERENTE------------
J=  3  - K= 9
J=  3  - K= 11
J=  3  - K= 12
J=  3  - K= 13
J=  4  - K= 3
J=  4  - K= 6
J=  4  - K= 9
IF3 DIFERENTE------------
J=  4  - K= 11
J=  4  - K= 12
J=  4  - K= 13
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 9
IF iGUAL--------------
J=  6  - K= 11
J=  6  - K= 12
J=  6  - K= 13
J=  9  - K= 3
J=  9  - K= 4
IF iGUAL--------------
J=  9  - K= 6
IF iGUAL--------------
J=  9  - K= 11
IF3 DIFERENTE------------
J=  9  - K= 12
J=  9  - K= 13
J=  11  - K= 3
J=  11  - K= 4
J=  11  - K= 6
J=  11  - K= 9
IF iGUAL--------------
J=  11  - K= 12
J=  11  - K= 13
IF3 DIFERENTE------------
J=  12  - K= 3
J=  12  - K= 4
J=  12  - K= 6
J=  12  - K= 9
J=  12  - K= 11
J=  12  - K= 13
J=  13  - K= 3
J=  13  - K= 4
J=  13  - K= 6
J=  13  - K= 9
J=  13  - K= 11
IF iGUAL--------------
```

```
J=  13  - K= 12
ELIMINACIÓN SIN Z
X:  11  Y:  13  Z:  11
Distancia mínima:  3.9

user_id    3     4     6     9     12    14
user_id
3         0.0  12.4   9.0  12.9  12.0  15.2
4        12.4   0.0  11.0   5.5  17.2   8.8
6         9.0  11.0   0.0   5.5   6.2   6.2
9        12.9   5.5   5.5   0.0  11.7   4.9
12       12.0  17.2   6.2  11.7   0.0   4.9
14       15.2   8.8   6.2   4.9   4.9   0.0
------------------------------------------------------------
J=  3  - K= 4
IF3 DIFERENTE------------
J=  3  - K= 6
IF3 DIFERENTE------------
J=  3  - K= 9
J=  3  - K= 12
J=  3  - K= 14
J=  4  - K= 3
J=  4  - K= 6
J=  4  - K= 9
IF3 DIFERENTE------------
J=  4  - K= 12
J=  4  - K= 14
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 9
IF iGUAL--------------
J=  6  - K= 12
J=  6  - K= 14
J=  9  - K= 3
J=  9  - K= 4
IF iGUAL--------------
J=  9  - K= 6
IF iGUAL--------------
J=  9  - K= 12
J=  9  - K= 14
IF3 DIFERENTE------------
J=  12  - K= 3
J=  12  - K= 4
J=  12  - K= 6
J=  12  - K= 9
J=  12  - K= 14
IF iGUAL--------------
J=  14  - K= 3
J=  14  - K= 4
J=  14  - K= 6
J=  14  - K= 9
IF iGUAL--------------
J=  14  - K= 12
IF iGUAL--------------
ELIMINACIÓN CON Z
X:  9  Y:  14  Z:  12
Distancia mínima:  4.9

user_id    3     4     6     15
user_id
3         0.0  12.4   9.0  12.0
4        12.4   0.0  11.0   5.5
6         9.0  11.0   0.0   5.5
15       12.0   5.5   5.5   0.0
```

```
------------------------------------------------------------
J=  3  - K= 4
IF3 DIFERENTE------------
J=  3  - K= 6
IF3 DIFERENTE------------
J=  3  - K= 15
J=  4  - K= 3
J=  4  - K= 6
J=  4  - K= 15
IF3 DIFERENTE------------
J=  6  - K= 3
J=  6  - K= 4
J=  6  - K= 15
IF iGUAL--------------
J=  15  - K= 3
J=  15  - K= 4
IF iGUAL--------------
J=  15  - K= 6
IF iGUAL--------------
ELIMINACIÓN CON Z
X:  4  Y:  15  Z:  6
Distancia mínima:  5.5

user_id   3    16
user_id
3        0.0  9.0
16       9.0  0.0
------------------------------------------------------------
J=  3  - K= 16
IF3 DIFERENTE------------
J=  16  - K= 3
IF iGUAL--------------
ELIMINACIÓN SIN Z
X:  3  Y:  16  Z:  3
Distancia mínima:  9.0

user_id   17
user_id
17       0.0
------------------------------------------------------------
```

In [27]:
```
z
```

Out[27]:
```
[[1, 10, 0.7, 2],
 [2, 7, 0.8, 2],
 [5, 8, 3.2, 2],
 [11, 13, 3.9, 2],
 [9, 12, 14, 4.9, 3],
 [4, 6, 15, 5.5, 3],
 [3, 16, 9.0, 2]]
```

In [28]:
```
df2
```

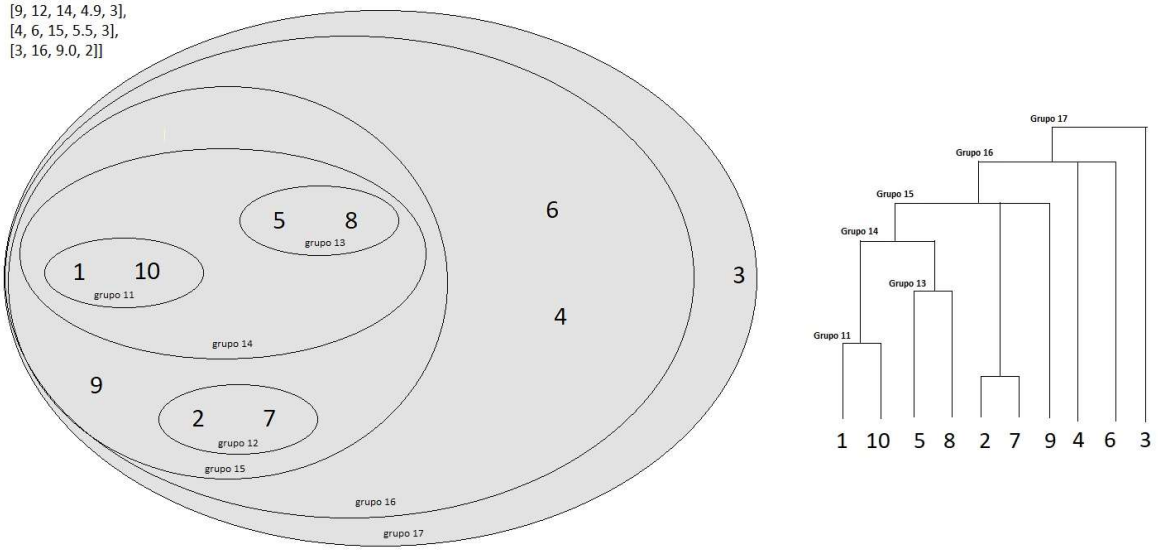
Out[28]:

user_id	17
user_id	
17	0.0

In [29]:
```
from IPython.display import Image
Image(filename="C:/Users/Kevin/Desktop/Kevin/1-Cursos/1-Machine Learning/CURSO UDEMY
```

Out[29]:

```
[[1, 10, 0.7, 2],
 [2, 7, 0.8, 2],
 [5, 8, 3.2, 2],
 [11, 13, 3.9, 2],
 [9, 12, 14, 4.9, 3],
 [4, 6, 15, 5.5, 3],
 [3, 16, 9.0, 2]]
```



Clustering Jerárquico

In [30]:

```
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
```

In [31]:

```
movies
```

Out[31]:

```
['star_wars', 'lord_of_the_rings', 'harry_potter']
```

In [32]:

```
data[movies]
```

Out[32]:

	star_wars	lord_of_the_rings	harry_potter
0	1.2	4.9	2.1
1	2.1	8.1	7.9
2	7.4	3.0	9.9
3	5.6	0.5	1.8
4	1.5	8.3	2.6
5	2.5	3.7	6.5
6	2.0	8.2	8.5
7	1.8	9.3	4.5
8	2.6	1.7	3.1
9	1.5	4.7	2.3

In [45]:

```
Z = linkage(data[movies], "ward") #El método ward minimiza los cuadrados de las vari
Z
```

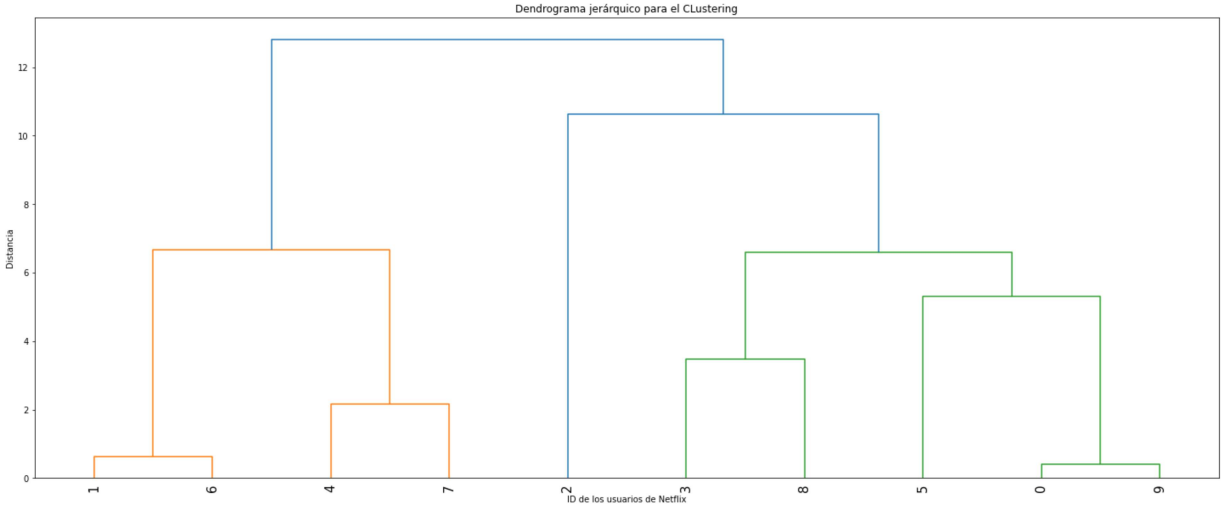
Out[45]:

```
array([[ 0.        ,  9.        ,  0.41231056,  2.        ],
```

```
       [ 1.        ,  6.        ,  0.6164414 ,  2.        ],
       [ 4.        ,  7.        ,  2.16794834,  2.        ],
       [ 3.        ,  8.        ,  3.48281495,  2.        ],
       [ 5.        , 10.        ,  5.2943366 ,  3.        ],
       [13.        , 14.        ,  6.59317829,  5.        ],
       [11.        , 12.        ,  6.66408283,  4.        ],
       [ 2.        , 15.        , 10.62355873,  6.        ],
       [16.        , 17.        , 12.8156935 , 10.        ]])
```

In [52]:

```
plt.figure(figsize=(25,10))
plt.title("Dendrograma jerárquico para el CLustering")
plt.xlabel("ID de los usuarios de Netflix")
plt.ylabel("Distancia")
dendrogram(Z, leaf_rotation=90.0, leaf_font_size=15)
plt.show()
```



In [53]:

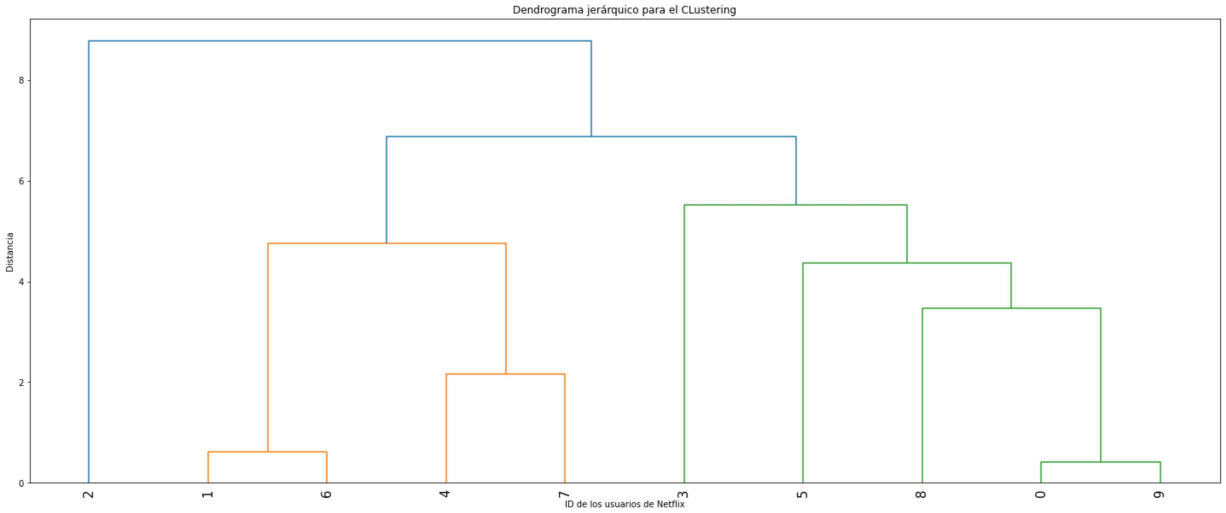
```
Z = linkage(data[movies], "average") #El método average es de promedio
Z
```

Out[53]:

```
array([[ 0.        ,  9.        ,  0.41231056,  2.        ],
       [ 1.        ,  6.        ,  0.6164414 ,  2.        ],
       [ 4.        ,  7.        ,  2.16794834,  2.        ],
       [ 8.        , 10.        ,  3.46355712,  3.        ],
       [ 5.        , 13.        ,  4.37331672,  4.        ],
       [11.        , 12.        ,  4.75772945,  4.        ],
       [ 3.        , 14.        ,  5.51984068,  5.        ],
       [15.        , 16.        ,  6.87496749,  9.        ],
       [ 2.        , 17.        ,  8.78305268, 10.        ]])
```

In [54]:

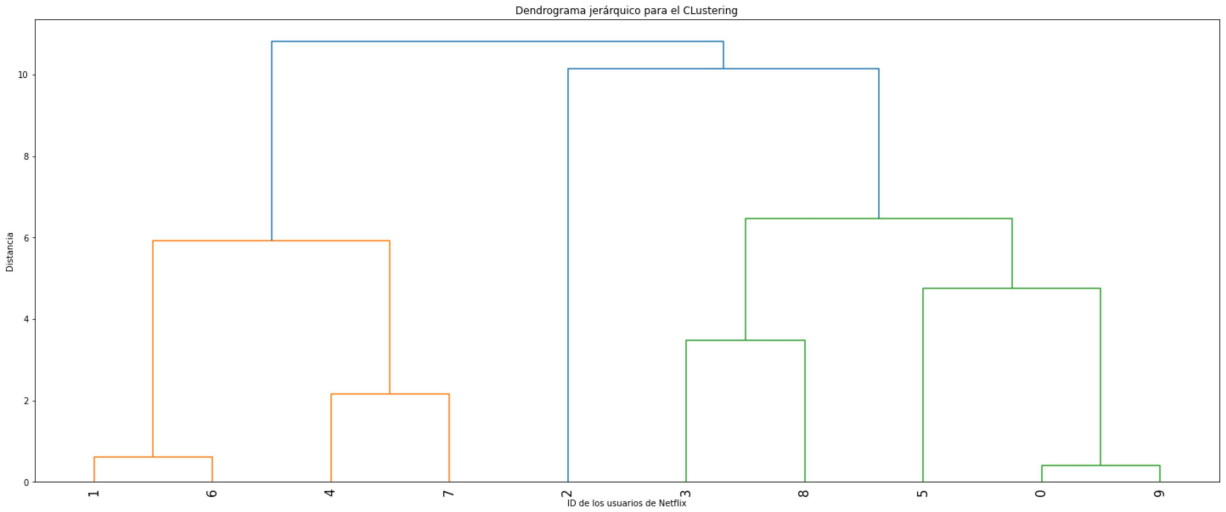
```
plt.figure(figsize=(25,10))
plt.title("Dendrograma jerárquico para el CLustering")
plt.xlabel("ID de los usuarios de Netflix")
plt.ylabel("Distancia")
dendrogram(Z, leaf_rotation=90.0, leaf_font_size=15)
plt.show()
```



In [55]:
```
Z = linkage(data[movies], "complete") # El enlace es muy similar al de ward
Z
```

Out[55]:
```
array([[ 0.        ,  9.        ,  0.41231056,  2.        ],
       [ 1.        ,  6.        ,  0.6164414 ,  2.        ],
       [ 4.        ,  7.        ,  2.16794834,  2.        ],
       [ 3.        ,  8.        ,  3.48281495,  2.        ],
       [ 5.        , 10.        ,  4.74236228,  3.        ],
       [11.        , 12.        ,  5.92199291,  4.        ],
       [13.        , 14.        ,  6.47610994,  5.        ],
       [ 2.        , 16.        , 10.14347081,  6.        ],
       [15.        , 17.        , 10.82312339, 10.        ]])
```

In [56]:
```
plt.figure(figsize=(25,10))
plt.title("Dendrograma jerárquico para el CLustering")
plt.xlabel("ID de los usuarios de Netflix")
plt.ylabel("Distancia")
dendrogram(Z, leaf_rotation=90.0, leaf_font_size=15)
plt.show()
```



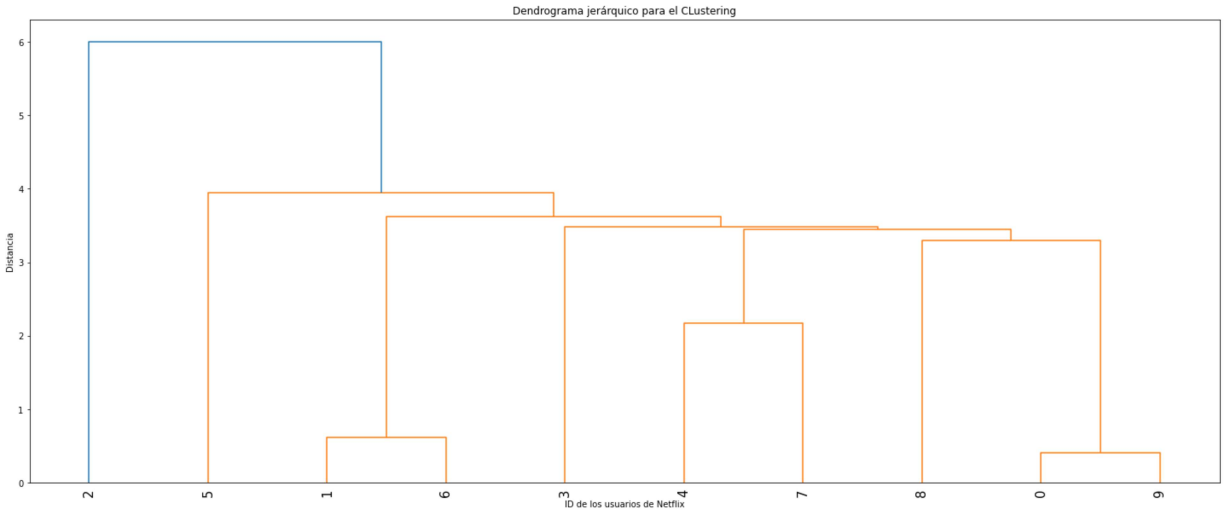
In [57]:
```
Z = linkage(data[movies], "single") # El enlace single es el simple
Z
```

Out[57]:
```
array([[ 0.        ,  9.        ,  0.41231056,  2.        ],
       [ 1.        ,  6.        ,  0.6164414 ,  2.        ],
       [ 4.        ,  7.        ,  2.16794834,  2.        ],
       [ 8.        , 10.        ,  3.29393382,  3.        ],
       [12.        , 13.        ,  3.44963766,  5.        ],
```

```
[ 3.        , 14.        ,  3.48281495,  6.        ],
[11.        , 15.        ,  3.6180105 ,  8.        ],
[ 5.        , 16.        ,  3.94588393,  9.        ],
[ 2.        , 17.        ,  6.00499792, 10.        ]])
```

In [58]:

```
plt.figure(figsize=(25,10))
plt.title("Dendrograma jerárquico para el CLustering")
plt.xlabel("ID de los usuarios de Netflix")
plt.ylabel("Distancia")
dendrogram(Z, leaf_rotation=90.0, leaf_font_size=15)
plt.show()
```



In [ ]: