

```
In [1]: import networkx as nx
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import operator
```

Reading the dataset

```
In [2]: actor_edges = pd.read_csv('actor_edges2.csv')
actors_key = pd.read_csv('actors_key.csv')
```

Creating the network using the edge list

Since each edge is the link between every pairs of actors, the network is undirected

```
In [3]: G = nx.from_pandas_edgelist(actor_edges, 'from', 'to', 'weight', create_using = nx.Graph)
```

Question a

Betweenness Centrality

```
In [4]: d = dict(nx.betweenness centrality(G, weight = 'weight'))
betweenness centrality_ranked = sorted(d.items(), key = operator.itemgetter(1), reverse = True)
for i in range(5):
    print("Actor with the top {} betweenness centrality is actor {}, with the corresponding betweenness centrality equal to {}.
```

Actor with the top 0 betweenness centrality is actor 4, with the corresponding betweenness centrality equal to 0.08.

Actor with the top 1 betweenness centrality is actor 47, with the corresponding betweenness centrality equal to 0.07.

Actor with the top 2 betweenness centrality is actor 27, with the corresponding betweenness centrality equal to 0.07.

Actor with the top 3 betweenness centrality is actor 10, with the corresponding betweenness centrality equal to 0.06.

Actor with the top 4 betweenness centrality is actor 13, with the corresponding betweenness centrality equal to 0.05.

Closeness Centrality

```
In [21]: d = dict(nx.closeness centrality(G))
closeness centrality_ranked = sorted(d.items(), key = operator.itemgetter(1), reverse = True)
for i in range(5):
    print("Actor with the top {} closeness centrality is actor {}, with the corresponding closeness centrality equal to {}.
```

Actor with the top 0 closeness centrality is actor 28, with the corresponding closeness centrality equal to 0.67.

Actor with the top 1 closeness centrality is actor 4, with the corresponding closeness centrality equal to 0.67.

Actor with the top 2 closeness centrality is actor 1, with the corresponding closeness centrality equal to 0.65.

Actor with the top 3 closeness centrality is actor 5, with the corresponding closeness centrality equal to 0.65.

Actor with the top 4 closeness centrality is actor 27, with the corresponding closeness centrality equal to 0.65.

The 5 actors with the largest betweenness centrality does not necessarily have to be the same as the 5 actors with the largest closeness centrality.

While both the criteria analyze the centrality of the network, the actor with the highest betweenness centrality is someone who is on the shortest path linking other actors, and the actor with the highest closeness centrality is someone who directly collaborates with other actors.

For example, we could assume three actors, A,B,C, collaborated once in a movie, while A has collaborated many time with Group 1, and B has collaborated many time with Group 2. Actors in Group 1 have never collaborated with actors in Group 2 before. B might have high betweenness centrality since B links Group 1 and Group 2, but B might have a low closeness centrality.

Question b

While there are many algorithms to detect the communities within a network, for this question I used Girvan-Newman method, which finds and removes the edge with the highest betweenness repeatedly.

1. Define the function to find the edge with the highest betweenness centrality

```
In [6]: def find_the_edge_with_highest_betweenness centrality(G):
        d = dict(nx.edge_betweenness centrality(G, weight = 'weight'))
        between centrality_ranked = sorted(d.items(), key = operator.itemgetter(1), reverse = True)
        return between centrality_ranked[0][0]
```

1. Create a copy of the original network

```
In [7]: C = G.copy()
```

1. Split the network by removing the edge with the highest betweenness. By constantly trying, I found that until I divided the network into 29 communities, the network is composed of 1 giant communities and multiple communities which have only 1 or 2 actors.

```
In [8]: while nx.number_connected_components(C) < 29 : # Dividing the network into 4 communities
        removed_edge = find_the_edge_with_highest_betweenness centrality(C)
        C.remove_edge(removed_edge[0], removed_edge[1])
        communities = list(nx.connected_components(C))
```

1. Define the largest and second largest communities

```
In [9]: larest_community = communities[0]
        second_larest_community = communities[1]
```

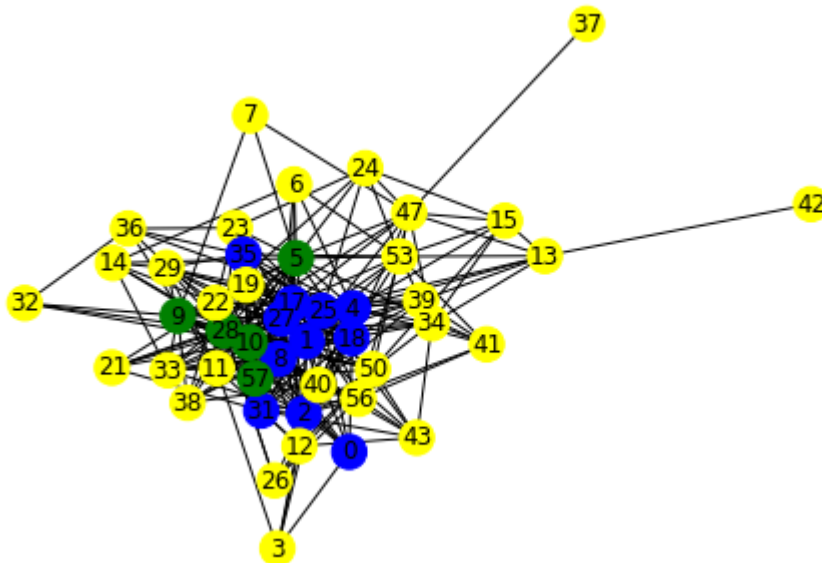
1. Define the color for the nodes in the largest community, the nodes in the second largest communities, and other nodes

```
In [10]: color_map = []
        for node in G:
            if node in larest_community:
                color_map.append('blue')
            elif node in second_larest_community:
                color_map.append('green')
```

```
else:
    color_map.append('yellow')
```

1. Draw the nodes with the corresponding communities

```
In [11]: nx.draw(G, node_color=color_map, with_labels=True)
```



Question c

From the result of question b, if we use Girvan-Newman method to split the network, then the largest community is composed of actors 0, 1, 2, 4, 8, 17, 18, 25, 27, 31, 35, and the second largest community is composed of actor 5, 9, 10, 28, 57, while other actors form other small communities.

The table below is the information for those actor in the largest community

```
In [19]: actors_key[actors_key['id'].isin(largest_community)]
```

Out[19]:	id	name	movies_95_04	main_genre	genres
1774	17	Székely B., Miklós	21	Drama	Comedy:1, Crime:3, Drama:7, Family:1, Music:1, Musi...
2001	25	Bezeredy, Zoltán	20	Drama	Comedy:5, Drama:6, NULL:4, Romance:2, Short:2, War:1
3278	4	Rajhona, Édém	20	Drama	Action:1, Adventure:2, Animation:1, Comedy:2, Crim...
4129	2	Kollai, Ferenc	15	Drama	Action:1, Comedy:4, Crime:1, Drama:4, Family:1, NUL...
4439	35	Pogány, Judit	12	Drama	Adventure:1, Comedy:2, Crime:2, Drama:4, Musical:1...
6480	1	Reviczky, Gábor	17	Comedy	Comedy:7, Crime:2, Fantasy:1, Musical:1, NULL:2, Ro...
6703	27	Göspör, Sándor	18	Comedy	Comedy:6, Crime:1, Drama:4, NULL:5, Romance:1, Short:1

	id	name	movies_95_04	main_genre	genres
8795	31	Eperjes, Károly	14	Drama	Adventure:1,Comedy:2,Crime:1,Drama:6,NULL:4
9431	0	Tordy, Gábor	10	Comedy	Adventure:1,Animation:1,Comedy:2,Family:1,NULL...
13209	18	Kiss, Jenő (II)	13	Drama	Action:1,Comedy:3,Drama:3,Family:1,Mystery:1,N...
16017	8	Dengyel, Iván	16	Drama	Animation:1,Comedy:1,Crime:1,Drama:6,Mystery:1...

The table below is the information for those actor in the second largest community

```
In [18]: actors_key[actors_key['id'].isin(second_largest_community)]
```

```
Out[18]:
```

	id	name	movies_95_04	main_genre	genres
6024	5	Csúja, Imre	26	Drama	Action:2,Animation:1,Comedy:4,Drama:7,Fantasy:...
6189	10	Cserna, Antal	19	Drama	Adventure:1,Animation:1,Comedy:2,Drama:5,Fanta...
9837	9	Gál, Balázs	17	Drama	Animation:1,Comedy:2,Drama:3,Music:1,NULL:6,Ro...
10700	28	Mucsi, Zoltán	31	Comedy	Action:1,Adventure:1,Comedy:6,Crime:2,Drama:5,...
15756	57	Nagy-Közlézy, Eszter	16	Drama	Comedy:2,Drama:5,Music:1,NULL:5,Romance:2,Short:1

From the two tables above, it appears that the main genres of actors in both communities are Drama and Comedy, therefore, they have collaborated more than other actors do. Also, it appears that the average number of movies made by actors in the largest community is less than that in the second largest community. So naturally, the actors in the largest community collaborated less than the actors in the second largest community.

In conclusion, the communities are split regarding to the main genre and the number of movies made during 1995 and 2004.