Proposal

September 22, 2025

The Deezer Europe Social Network dataset consists of 28,281 users and 92,752 mutual follower relationships, making it a large yet sparse network with a density of only 0.00023. On average, each user maintains about 6.56 connections, and the clustering coefficient of 0.0959 indicates a modest tendency toward small friendship circles where friends of friends are also connected. Because not all users belong to the same connected component, the largest connected component (LCC) is extracted to provide a meaningful foundation for further analysis. This dataset, with its binary gender labels, offers a valuable opportunity to explore centrality measures across categories, assess homophily, and compare how structural importance differs between groups.

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

we perform our anlysis using the library above and create different metrics to plan for our project

[2]:
```
# laod the edge files
edges = pd.read_csv('https://raw.githubusercontent.com/Raji030/
 ↪data620_assignment03_deezer_edges/refs/heads/main/deezer_europe_edges.csv')
targets = pd.read_csv('https://raw.githubusercontent.com/Raji030/
 ↪data620_assignment03_deezer_target/refs/heads/main/deezer_europe_target.csv')

# Shwo infor
edges_info = edges.info()
targets_info = targets.info()

edges_head = edges.head()
targets_head = targets.head()

edges_info, targets_info, edges_head, targets_head
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 92752 entries, 0 to 92751
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   node_1  92752 non-null  int64
 1   node_2  92752 non-null  int64
dtypes: int64(2)
```

```
memory usage: 1.4 MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28281 entries, 0 to 28280
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   id      28281 non-null  int64
 1   target  28281 non-null  int64
dtypes: int64(2)
memory usage: 442.0 KB
```

[2]:
```
(None,
 None,
    node_1  node_2
 0       0   14270
 1       0   16976
 2       0   12029
 3       0    3001
 4       0   14581,
    id  target
 0   0       0
 1   1       0
 2   2       0
 3   3       1
 4   4       0)
```

[3]:
```
# create graph from column names
G = nx.from_pandas_edgelist(edges, source = "node_1", target="node_2")
target_dict = pd.Series(targets.target.values, index=targets.id).to_dict()
nx.set_node_attributes(G , target_dict, "label")
```

The Deezer Europe Social Network comprises 28,281 users and 92,752 edges, where each edge corresponds to a mutual follower relationship. The network is highly sparse, with a density of only 0.00023, meaning that users connect to a very small fraction of all possible connections, an expected property of large-scale social platforms where relationships are formed selectively. On average, each user maintains approximately 6.56 connections, reinforcing the observation that while the network is large, individual connectivity is relatively modest. The clustering coefficient of 0.0959 suggests a moderate likelihood of triadic closure: if one user connects with two others, there is roughly a 9.6% chance that those two users are also connected. This highlights the presence of localized friendship circles or small communities within the broader structure. Because not all nodes are guaranteed to be part of the same connected component, the largest connected component (LCC) is extracted for analysis, ensuring that centrality measures and categorical comparisons are computed on the most structurally meaningful portion of the graph.

[4]:
```
# Basic Statistics
n_nodes = G.number_of_nodes()
n_edges = G.number_of_edges()
density = nx.density(G)
```

```
avg_degree = sum(dict(G.degree()).values()) / n_nodes
transitivity = nx.transitivity(G)

print(f'Number of nodes :{n_nodes}')
print(f'Number of edges; {n_edges}')
print(f'The density is : {density}')
print(f'the Average degree {avg_degree}')
print(f'transitivity: {transitivity}')
```

```
Number of nodes :28281
Number of edges; 92752
The density is : 0.00023194184729358083
the Average degree 6.559315441462466
transitivity: 0.09592226364671026
```

[5]:
```
# Extract largest connected component
largest_cc_nodes = max(nx.connected_components(G), key=len)
G_lcc = G.subgraph(largest_cc_nodes).copy()
lcc_size = len(largest_cc_nodes)
```

[6]:
```
# Build summary table
stats_df = pd.DataFrame({
    "Statistic": [
        "Number of Nodes",
        "Number of Edges",
        "Density",
        "Average Degree",
        "Transitivity (Clustering)",
        "Largest Connected Component (Nodes)"
    ],
    "Value": [
        n_nodes,
        n_edges,
        round(density, 6),
        round(avg_degree, 3),
        round(transitivity, 4),
        lcc_size
    ]
})
```

[7]:
```
stats_df
```

[7]:

	Statistic	Value
0	Number of Nodes	28281.000000
1	Number of Edges	92752.000000
2	Density	0.000232
3	Average Degree	6.559000
4	Transitivity (Clustering)	0.095900

```
5  Largest Connected Component (Nodes)  28281.000000
```

To make these statistics interpretable and provide meaningful insights, several visualizations will be incorporated into the analysis. A histogram of degree distribution, plotted on a log-log scale, will illustrate how connectivity varies across users while highlighting the existence of highly connected hubs. A boxplot of degree centrality by gender will compare connectivity between male and female users, enabling us to assess whether one group tends to maintain more connections. The bar chart of class distribution will show the proportion of users in each gender category, providing a quick assessment of balance or skew in the dataset. A pie chart of homophily will capture the tendency of users to connect with others of the same gender versus across genders. Finally, a subgraph visualization (sampling 200–300 nodes) will be produced, where nodes are colored by gender and sized by degree centrality, offering an intuitive view of the network's structure and categorical clustering.

[11]:
```
!jupyter nbconvert --to pdf Proposal.ipynb
```

```
[NbConvertApp] Converting notebook Proposal.ipynb to pdf
[NbConvertApp] Writing 33662 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | b had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 41460 bytes to Proposal.pdf
```

[ ]:
```
```