Abhishek Patwardhan

D17A - 57

SMA Experiment 6

==> Importing Dependencies

```
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

==> Loading Dataset

```
df = pd.read_csv("Reddit.csv")
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 286561 entries, 0 to 286560
Data columns (total 4 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   SOURCE_SUBREDDIT  286561 non-null  object
 1   TARGET_SUBREDDIT  286561 non-null  object
 2   POST_ID           286561 non-null  object
 3   TIMESTAMP         286561 non-null  object
dtypes: object(4)
memory usage: 8.7+ MB
```

```
df['SOURCE_SUBREDDIT'].value_counts()
```

```
subredditdrama        4665
circlebroke           2358
shitliberalssay       1968
outoftheloop          1958
copypasta             1824
                      ...
highqualityreviews       1
sefiefythis              1
testcaseforcss           1
tahrox                   1
mildlynomil              1
Name: SOURCE_SUBREDDIT, Length: 27863, dtype: int64
```

==> Visualizing the Graph

Selecting a random 1% of data for visualisation purposes

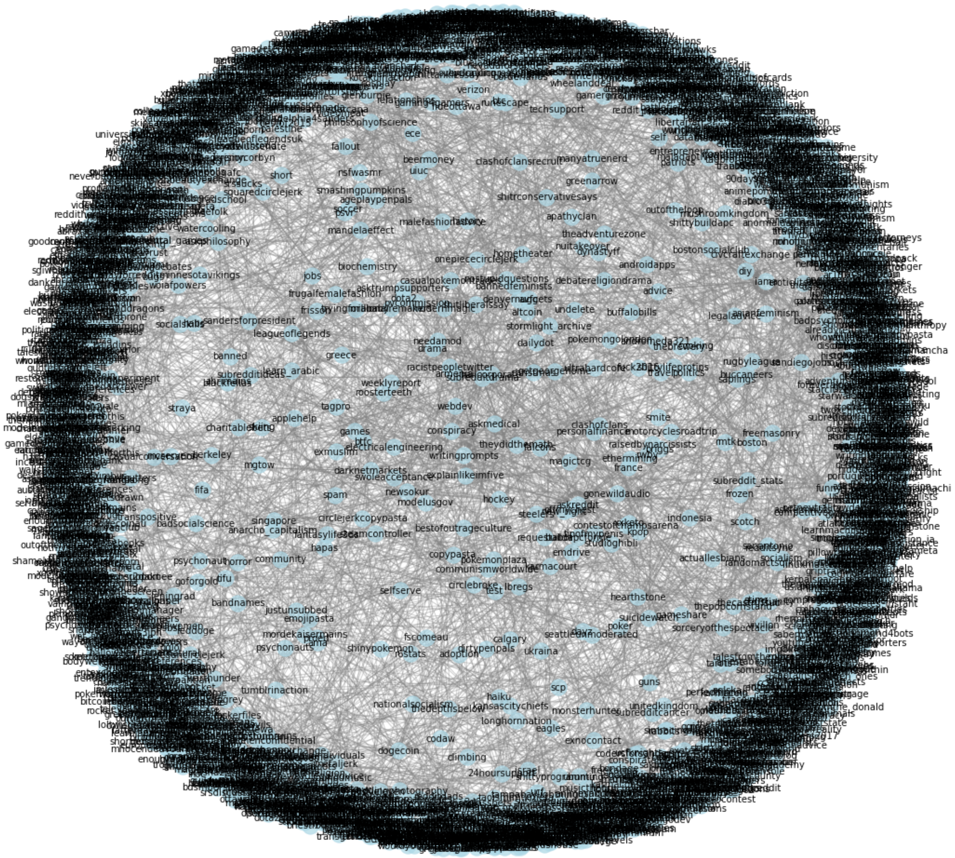
```
# Select a random 10% of the data
df1 = df.sample(frac=0.01, random_state=42)
```

```
# Create a DiGraph object
G = nx.DiGraph()
```

```
nodes = set(df1['SOURCE_SUBREDDIT']).union(set(df1['TARGET_SUBREDDIT']))
for node in nodes:
 G.add_node(node)
```

```
for _, row in df1.iterrows():
 G.add_edge(row['SOURCE_SUBREDDIT'], row['TARGET_SUBREDDIT'])
```

```
%%time
plt.figure(figsize=(20, 20))
pos = nx.spring_layout(G, k=0.5)
nx.draw_networkx_nodes(G, pos, node_color='lightblue', node_size=300, alpha=0.7)
nx.draw_networkx_edges(G, pos, edge_color='gray', alpha=0.4)
nx.draw_networkx_labels(G, pos, font_size=10, font_family='sans-serif')
plt.axis('off')
plt.show()
```



```
CPU times: user 59.6 s, sys: 926 ms, total: 1min
Wall time: 1min
```

==> Finding degrees and degree centrality of each node for the entire data

```
g = nx.DiGraph()
# Add nodes to the graph
gnodes = set(df['SOURCE_SUBREDDIT']).union(set(df['TARGET_SUBREDDIT']))
for gnode in gnodes:
 g.add_node(gnode)
# Add edges to the graph
for _, g_row in df.iterrows():
 g.add_edge(g_row['SOURCE_SUBREDDIT'], g_row['TARGET_SUBREDDIT'])
```

```
deg_cent = pd.DataFrame()


deg_cent['node'] = [node for (node, val) in nx.degree(g)]
deg_cent['degree'] = [val for (node, val) in nx.degree(g)]


deg_cent['centrality'] = nx.degree_centrality(g).values()


deg_cent['eigenCentrality'] = nx.eigenvector_centrality(g).values()


deg_cent.head(5)
```

	node	degree	centrality	eigenCentrality
0	gtaa	11	0.000307	3.482767e-04
1	openandhonest	4	0.000112	8.315722e-06
2	reckful	5	0.000140	1.724627e-03
3	waggansw	1	0.000028	8.569079e-15
4	lojban	6	0.000168	6.431221e-06

==> Most influential node

```
deg_cent[deg_cent.centrality == deg_cent.centrality.max()]
```

	node	degree	centrality	eigenCentrality
9462	askreddit	2524	0.070552	0.265185

==> Most important connection

```
deg_cent[deg_cent.eigenCentrality == deg_cent.eigenCentrality.max()]
```

	node	degree	centrality	eigenCentrality
9462	askreddit	2524	0.070552	0.265185

==> Betweeness Centrality Issue

The nx.betweenness_centrality function in NetworkX can take time to run if there is a huge number of edges and nodes in the graph. The time complexity of betweenness centrality is O(nm), where n is the number of nodes and m is the number of edges in the graph.