Annapoornima S

225229101

Lab : 07 - Link Prediction of future connections in Facebook

Step-1:

In [1]:

```
import pandas as pd
import numpy as np
import random
import networkx as nx
from tqdm import tqdm
import re
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
```

Step-2:

In [2]:

```
with open("fb-pages-food.nodes", "r+", encoding="utf-8") as f:
    fb_nodes = f.read().splitlines()
with open("fb-pages-food.edges", "r+", encoding="utf-8") as f:
    fb_links = f.read().splitlines()
len(fb_nodes), len(fb_links)
```

Out[2]:

```
(621, 2102)
```

In [3]:

```
node_list_1 = []
node_list_2 = []
for i in tqdm(fb_links):
    node_list_1.append(i.split(',')[0])
    node_list_2.append(i.split(',')[1])
fb_df = pd.DataFrame({'node_1': node_list_1, 'node_2': node_list_2})
```

```
100%|██████████████████████████████████████████████████████████████████████████████
| 2102/2102 [00:00<00:00, 2102653.71it/s]
```

In [4]:

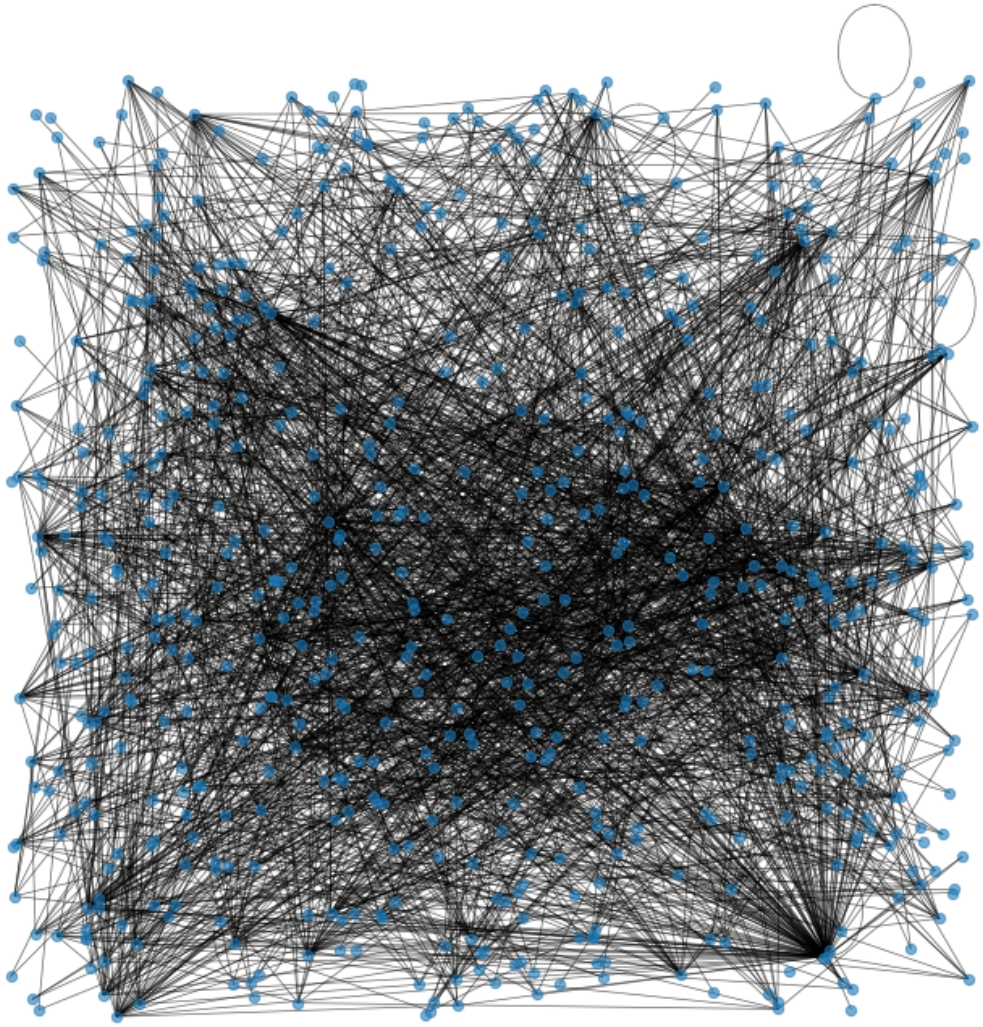
```
fb_df.head()
```

Out[4]:

	node_1	node_2
0	0	276
1	0	58
2	0	132
3	0	603
4	0	398

In [5]:

```
G = nx.from_pandas_edgelist(fb_df, "node_1", "node_2", create_using=nx.Graph())
plt.figure(figsize=(10,10))
pos = nx.random_layout(G, seed=23)
nx.draw(G, with_labels=False,  pos = pos, node_size = 40, alpha = 0.6, width = 0.7)
plt.show()
```



Step-3:

In [6]:

```
node_list = node_list_1 + node_list_2
node_list = list(dict.fromkeys(node_list))
adj_G = nx.to_numpy_matrix(G, nodelist = node_list)
```

In [7]:

```
adj_G.shape
```

Out[7]:

```
(620, 620)
```

In [8]:

```
all_unconnected_pairs = []
offset = 0
for i in tqdm(range(adj_G.shape[0])):
    for j in range(offset,adj_G.shape[1]):
        if i != j:
            if nx.shortest_path_length(G, str(i), str(j)) <=2:
                if adj_G[i,j] == 0:
                    all_unconnected_pairs.append([node_list[i],node_list[j]])
    offset = offset + 1
```

```
100%|██████████| 620/620 [00:04<00:00, 136.84it/s]
```

In [9]:

```
len(all_unconnected_pairs)
```

Out[9]:

```
19018
```

In [10]:

```
node_1_unlinked = [i[0] for i in all_unconnected_pairs]
node_2_unlinked = [i[1] for i in all_unconnected_pairs]

data = pd.DataFrame({'node_1':node_1_unlinked,'node_2':node_2_unlinked})
data['link'] = 0
```

In [11]:

```
initial_node_count = len(G.nodes)
fb_df_temp = fb_df.copy()
omissible_links_index = []
for i in tqdm(fb_df.index.values):
    G_temp = nx.from_pandas_edgelist(fb_df_temp.drop(index = i), "node_1", "node_2", create
    if (nx.number_connected_components(G_temp) == 1) and (len(G_temp.nodes) == initial_node
        omissible_links_index.append(i)
        fb_df_temp = fb_df_temp.drop(index = i)
```

```
100%|██████████| 2102/2102 [00:03<00:00, 559.95it/s]
```

In [12]:

```
len(omissible_links_index)
```

Out[12]:

```
1483
```

In [13]:

```
fb_df_ghost = fb_df.loc[omissible_links_index]
fb_df_ghost['link'] = 1
data = data.append(fb_df_ghost[['node_1', 'node_2', 'link']], ignore_index=True)
```

```
C:\Users\online.CSCENTER\AppData\Local\Temp\ipykernel_12024\3242015871.py:7:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  data = data.append(fb_df_ghost[['node_1', 'node_2', 'link']], ignore_index=
True)
```

In [14]:

```
data['link'].value_counts()
```

Out[14]:

```
0    19018
1     1483
Name: link, dtype: int64
```

Step-4:

In [15]:

```
fb_df_partial = fb_df.drop(index=fb_df_ghost.index.values)
G_data = nx.from_pandas_edgelist(fb_df_partial, "node_1","node_2", create_using=nx.Graph())
```

Step-5:

In [19]:

```
from node2vec import Node2Vec
node2vec = Node2Vec(G_data, dimensions=100, walk_length=16, num_walks=50)
n2w_model = node2vec.fit(window=7, min_count=1)
```

```
Computing transition probabilities:   0%|          | 0/620 [00:00<?, ?it/s]

Generating walks (CU: 1): 100%|
| 50/50 [00:00<00:00, 50.75it/s]
```

In [21]:

```
x = [(n2w_model.wv[str(i)]+n2w_model.wv[str(j)]) for i,j in zip(data['node_1'], data['node_
```

In [22]:

```
xtrain, xtest, ytrain, ytest = train_test_split(np.array(x), data['link'], test_size = 0.3,
```

In [23]:

```
lr = LogisticRegression(class_weight="balanced")
lr.fit(xtrain, ytrain)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
```

Out[23]:

```
LogisticRegression(class_weight='balanced')
```

In [24]:

```
predictions = lr.predict_proba(xtest)
```

In [25]:

```
roc_auc_score(ytest, predictions[:,1])
```

Out[25]:

```
0.7900162913034287
```

In [28]:

```
import lightgbm as lgbm
train_data = lgbm.Dataset(xtrain, ytrain)
test_data = lgbm.Dataset(xtest, ytest)

parameters = {
    'objective': 'binary',
    'metric': 'auc',
    'is_unbalance': 'true',
    'feature_fraction': 0.5,
    'bagging_fraction': 0.5,
    'bagging_freq': 20,
    'num_threads' : 2,
    'seed' : 76
}
model = lgbm.train(parameters,train_data,valid_sets=test_data,num_boost_round=1000,early_st
```

```
C:\Users\online.CSCENTER\AppData\Roaming\Python\Python39\site-packages\lig
htgbm\engine.py:181: UserWarning: 'early_stopping_rounds' argument is depr
ecated and will be removed in a future release of LightGBM. Pass 'early_st
opping()' callback via 'callbacks' argument instead.
  _log_warning("'early_stopping_rounds' argument is deprecated and will be
removed in a future release of LightGBM. "

[LightGBM] [Info] Number of positive: 1062, number of negative: 13288
[LightGBM] [Warning] Auto-choosing col-wise multi-threading, the overhead
of testing was 0.003141 seconds.
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 25500
[LightGBM] [Info] Number of data points in the train set: 14350, number of
used features: 100
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.074007 -> initscore=-2.5
26707
[LightGBM] [Info] Start training from score -2.526707
[1]     valid_0's auc: 0.737591
Training until validation scores don't improve for 20 rounds
```

In [ ]: