Load Package

In [1]:
```
import pandas as pd
import numpy as np
import os
import networkx as nx
from node2vec import Node2Vec
```

In [2]:
```
print (os.getcwd())
os.chdir('D:/OneDrive/ASU/2021 Spring/Applied Project/ASU_Applied_Project_2021/Data')
print (os.getcwd())
```

```
C:\Users\Jinhang Jiang
D:\OneDrive\ASU\2021 Spring\Applied Project\ASU_Applied_Project_2021\Data
```

Load Data and Explore

In [3]:
```
data = pd.read_csv("networkanalysis2.csv")
```

In [4]:
```
data.head(6)
```

Out[4]:

	Celebrity	Usernames
0	Kerwin Frost	jamiedevlin999
1	Kerwin Frost	neighborgang
2	Kerwin Frost	jothvm
3	Kerwin Frost	nostylist2900
4	Kerwin Frost	New_Age_Dryer
5	Kerwin Frost	janspirit

In [5]:
```
print(*data.Celebrity.unique(),sep="\n")
```

```
Kerwin Frost
Beyonce
Zoe Saldana
Karlie Kloss
Yara Sayeh Shahidi
naeun
Pharrell Williams
Adriene Mishler
BlackPink
NinjasHyper
BadBunny
JERRY LORENZO
CHINAE ALEXANDER
ALLY LOVE
```

In [6]:
```
data.shape
```

Out[6]:
```
(2142, 2)
```

In [7]:
```
print("Number of Celebrities: %0.0f" %len(data.Celebrity.unique()))
print("Number of Users: %0.0f" %len(data.Usernames.unique()))
```

```
Number of Celebrities: 14
Number of Users: 2014
```

In [139]:
```
print("The percentage of unique values: {:.2%}".format(len(data.Usernames.unique())/len(data.Usernames)))
```

```
The percentage of unique values: 94.02%
```

In [100]:
```
data.Celebrity.value_counts()
```

Out[100]:
```
Kerwin Frost          211
Beyonce               202
NinjasHyper           197
ALLY LOVE             191
JERRY LORENZO         180
Zoe Saldana           174
BadBunny              173
Yara Sayeh Shahidi    161
naeun                 151
Karlie Kloss          147
BlackPink             131
CHINAE ALEXANDER      129
Pharrell Williams      74
Adriene Mishler        21
Name: Celebrity, dtype: int64
```

In [134]:
```
from itertools import combinations
cel_names = list(data.Celebrity.unique())
output = []
for (cel1, cel2) in combinations(cel_names, 2):
    fans_overlap = num_of_fans_overlap(cel1, cel2, data,'Celebrity','Usernames')
    temp = (cel1, cel2, fans_overlap)
    output.append(tuple(temp))
print(((len(cel_names)-1)*len(cel_names))/2)
print(len(output))
```

```
91.0
91
```

Generate Adjacency Matrix

In [13]:
```
#Create matrix
matrix = pd.get_dummies(data.set_index('Usernames')['Celebrity'].astype(str)).max(level=0).sort_index()
```

In [14]:
```
matrix.iloc[0:5,0:5]
```

Out[14]:

Usernames	ALLY LOVE	Adriene Mishler	BadBunny	Beyonce	BlackPink
-Fashion-News-	0	0	0	0	0
-Sportswear-	0	0	0	0	0
-en-	1	1	0	0	0
-lastmanstan-	1	0	0	0	0
0LoveRainbow0	1	0	0	0	0

In [15]:
```
#matrix.to_csv("dummy_matrix.csv")
```

In [16]:
```
cel_matrix = np.asmatrix(matrix)
cel_matrix_transpose = cel_matrix.transpose()
final_matrix = cel_matrix_transpose.dot(cel_matrix)

network_table = pd.DataFrame(final_matrix)
print(network_table.iloc[0:5,0:5])
print(network_table.shape)
```

```
     0   1    2    3    4
0  191   3    0    3    0
1    3  21    0    2    0
2    0   0  173    0    0
3    3   2    0  202    3
4    0   0    0    3  131
(14, 14)
```

In [17]:
```
## append index name
Celebrity = list(data.Celebrity.unique())
Celebrity.sort()

network_table.index = Celebrity
network_table.columns = Celebrity
network_table
```

Out[17]:

	ALLY LOVE	Adriene Mishler	BadBunny	Beyonce	BlackPink	CHINAE ALEXANDER	JERRY LORENZO	Karlie Kloss	Kerwin Frost	NinjasHyper	Pharrell Williams	Yara Sayeh Shahidi
ALLY LOVE	191	3	0	3	0	6	2	3	0	0	5	6
Adriene Mishler	3	21	0	2	0	3	3	2	0	0	2	3
BadBunny	0	0	173	0	0	1	0	1	0	0	1	1
Beyonce	3	2	0	202	3	2	3	9	0	0	6	9
BlackPink	0	0	0	3	131	0	0	0	1	0	0	0
CHINAE ALEXANDER	6	3	1	2	0	129	2	2	1	0	3	7
JERRY LORENZO	2	3	0	3	0	2	180	1	1	0	3	4
Karlie Kloss	3	2	1	9	0	2	1	147	0	1	5	10
Kerwin Frost	0	0	0	0	1	1	1	0	211	0	1	1
NinjasHyper	0	0	0	0	0	0	0	1	0	197	1	0
Pharrell Williams	5	2	1	6	0	3	3	5	1	1	74	8
Yara Sayeh Shahidi	6	3	1	9	0	7	4	10	1	0	8	161
Zoe Saldana	3	1	0	12	1	2	1	27	0	0	5	17
naeun	2	2	0	3	8	3	2	0	0	0	2	2

In [18]:
```
matrix.to_csv('frequency_matrix2.csv')
network_table.to_csv('network_table2.csv')
```

Fit NetworkX

In [19]:
```
#network_table = pd.read_csv('network_table1.csv', index_col=0)
```

In [20]:
```
network_table
```

Out[20]:

	ALLY LOVE	Adriene Mishler	BadBunny	Beyonce	BlackPink	CHINAE ALEXANDER	JERRY LORENZO	Karlie Kloss	Kerwin Frost	NinjasHyper	Pharrell Williams	Yara Sayeh Shahidi
ALLY LOVE	191	3	0	3	0	6	2	3	0	0	5	6
Adriene Mishler	3	21	0	2	0	3	3	2	0	0	2	3
BadBunny	0	0	173	0	0	1	0	1	0	0	1	1
Beyonce	3	2	0	202	3	2	3	9	0	0	6	9
BlackPink	0	0	0	3	131	0	0	0	1	0	0	0
CHINAE ALEXANDER	6	3	1	2	0	129	2	2	1	0	3	7
JERRY LORENZO	2	3	0	3	0	2	180	1	1	0	3	4
Karlie Kloss	3	2	1	9	0	2	1	147	0	1	5	10
Kerwin Frost	0	0	0	0	1	1	1	0	211	0	1	1
NinjasHyper	0	0	0	0	0	0	0	1	0	197	1	0
Pharrell Williams	5	2	1	6	0	3	3	5	1	1	74	8
Yara Sayeh Shahidi	6	3	1	9	0	7	4	10	1	0	8	161
Zoe Saldana	3	1	0	12	1	2	1	27	0	0	5	17
naeun	2	2	0	3	8	3	2	0	0	0	2	2

In [21]:
```
#pip install --upgrade networkx
```

In [50]:
```
np_matrix = np.matrix(network_table)
np_matrix
```

Out[50]:
```
matrix([[191,   3,   0,   3,   0,   6,   2,   3,   0,   0,   5,   6,   3,
           2],
        [  3,  21,   0,   2,   0,   3,   3,   2,   0,   0,   2,   3,   1,
           2],
        [  0,   0, 173,   0,   0,   1,   0,   1,   0,   0,   1,   1,   0,
           0],
        [  3,   2,   0, 202,   3,   2,   3,   9,   0,   0,   6,   9,  12,
           3],
        [  0,   0,   0,   3, 131,   0,   0,   0,   1,   0,   0,   0,   1,
           8],
        [  6,   3,   1,   2,   0, 129,   2,   2,   1,   0,   3,   7,   2,
           3],
        [  2,   3,   0,   3,   0,   2, 180,   1,   1,   0,   3,   4,   1,
           2],
        [  3,   2,   1,   9,   0,   2,   1, 147,   0,   1,   5,  10,  27,
           0],
        [  0,   0,   0,   0,   1,   1,   1,   0, 211,   0,   1,   1,   0,
           0],
        [  0,   0,   0,   0,   0,   0,   0,   1,   0, 197,   1,   0,   0,
           0],
        [  5,   2,   1,   6,   0,   3,   3,   5,   1,   1,  74,   8,   5,
           2],
        [  6,   3,   1,   9,   0,   7,   4,  10,   1,   0,   8, 161,  17,
           2],
        [  3,   1,   0,  12,   1,   2,   1,  27,   0,   0,   5,  17, 174,
           0],
        [  2,   2,   0,   3,   8,   3,   2,   0,   0,   0,   2,   2,   0,
         151]], dtype=uint8)
```

In [86]:
```
node2vec = Node2Vec(graph, dimensions=128, walk_length=80, num_walks=10, workers=4)
model = node2vec.fit(window=10, min_count=1)
```

```
Computing transition probabilities: 100%|████████████████████████████████████████
█████| 14/14 [00:00<00:00, 2383.32it/s]
```

In [76]:
```
#model.wv.save_word2vec_format('embedding2.csv')
```

In [87]:

```
vocab, vectors = model.wv.vocab, model.wv.vectors

# get node name and embedding vector index.
name_index = np.array([(v[0], v[1].index) for v in vocab.items()])

# init dataframe using embedding vectors and set index as node name
df =  pd.DataFrame(vectors[name_index[:,1].astype(int)])
df.index = name_index[:,0]
```

In [88]:
```
df.index = df.index.astype('int64')
df.index
df = df.sort_index(axis=0,ascending=True)
df
```

Out[88]:
```
           0         1         2         3         4         5         6         7         8         9  ...       118       119
 0  0.326913  0.471571  0.083203  0.031402 -0.010628 -0.023912  0.063921 -0.097126 -0.189452  0.003003  ... -0.171120  0.176966
 1  0.178629  0.269867 -0.028111  0.083830 -0.137122  0.025222 -0.069708 -0.070570  0.026597  0.055278  ... -0.148554  0.042999
 2 -0.234911  0.153243  0.069050  0.081673 -0.753533 -0.487369 -0.796651  0.191836  0.217514 -0.140537  ...  0.035442 -0.264339
 3  0.330528  0.466208  0.190072 -0.045266 -0.191760 -0.184798 -0.006684 -0.049817 -0.172090  0.131152  ... -0.198641  0.058706
 4  0.241027  0.173419 -0.115924  0.001182 -0.364212 -0.074330 -0.251910 -0.259114  0.477806  0.236056  ... -0.175344 -0.316474
 5  0.046100  0.234730  0.003506  0.047189 -0.184279 -0.061933 -0.211761  0.012216  0.108383  0.010814  ... -0.113881  0.032860
 6  0.233969  0.385906  0.118609  0.011564 -0.136997 -0.008664 -0.003212 -0.105238 -0.161327  0.064653  ... -0.220058  0.101051
 7  0.244061  0.391757  0.086530  0.035902 -0.140783 -0.064211 -0.018344 -0.044024 -0.158008  0.058264  ... -0.154134  0.106935
 8 -0.036308 -0.073083 -0.653143  0.475408 -0.712173  0.365094 -0.859598  0.582446  0.462809  0.274421  ... -0.505584 -0.260074
 9 -0.027901  0.090799 -0.704487  0.691049  0.493449  0.791593  0.347704 -0.736355  0.707216 -0.419034  ...  0.405868  0.390749
10  0.166735  0.261806 -0.010940  0.079094 -0.194773 -0.018241 -0.142166  0.046986 -0.031154  0.068653  ... -0.184787  0.024560
11  0.274355  0.399355  0.041073  0.062148 -0.100946 -0.017217 -0.017766 -0.022574 -0.120697  0.059744  ... -0.197649  0.127755
12  0.332378  0.447453  0.101442  0.009290 -0.200168 -0.087849 -0.036698 -0.013969 -0.205062  0.130509  ... -0.242428  0.058356
13  0.179751  0.126084 -0.159199  0.093563 -0.267558  0.052427 -0.237505 -0.097700  0.342827  0.168560  ... -0.223391 -0.186987

14 rows × 128 columns
```

In [89]:
```
celebrity_names = network_table.columns
```

In [90]:
```
df.index = celebrity_names
```

In [91]:
```
df
```

Out[91]:

	0	1	2	3	4	5	6	7	8	9	...	118	
ALLY LOVE	0.326913	0.471571	0.083203	0.031402	-0.010628	-0.023912	0.063921	-0.097126	-0.189452	0.003003	...	-0.171120	0
Adriene Mishler	0.178629	0.269867	-0.028111	0.083830	-0.137122	0.025222	-0.069708	-0.070570	0.026597	0.055278	...	-0.148554	0
BadBunny	-0.234911	0.153243	0.069050	0.081673	-0.753533	-0.487369	-0.796651	0.191836	0.217514	-0.140537	...	0.035442	-0
Beyonce	0.330528	0.466208	0.190072	-0.045266	-0.191760	-0.184798	-0.006684	-0.049817	-0.172090	0.131152	...	-0.198641	0
BlackPink	0.241027	0.173419	-0.115924	0.001182	-0.364212	-0.074330	-0.251910	-0.259114	0.477806	0.236056	...	-0.175344	-0
CHINAE ALEXANDER	0.046100	0.234730	0.003506	0.047189	-0.184279	-0.061933	-0.211761	0.012216	0.108383	0.010814	...	-0.113881	0
JERRY LORENZO	0.233969	0.385906	0.118609	0.011564	-0.136997	-0.008664	-0.003212	-0.105238	-0.161327	0.064653	...	-0.220058	0
Karlie Kloss	0.244061	0.391757	0.086530	0.035902	-0.140783	-0.064211	-0.018344	-0.044024	-0.158008	0.058264	...	-0.154134	0
Kerwin Frost	-0.036308	-0.073083	-0.653143	0.475408	-0.712173	0.365094	-0.859598	0.582446	0.462809	0.274421	...	-0.505584	-0
NinjasHyper	-0.027901	0.090799	-0.704487	0.691049	0.493449	0.791593	0.347704	-0.736355	0.707216	-0.419034	...	0.405868	0
Pharrell Williams	0.166735	0.261806	-0.010940	0.079094	-0.194773	-0.018241	-0.142166	0.046986	-0.031154	0.068653	...	-0.184787	0
Yara Sayeh Shahidi	0.274355	0.399355	0.041073	0.062148	-0.100946	-0.017217	-0.017766	-0.022574	-0.120697	0.059744	...	-0.197649	0
Zoe Saldana	0.332378	0.447453	0.101442	0.009290	-0.200168	-0.087849	-0.036698	-0.013969	-0.205062	0.130509	...	-0.242428	0
naeun	0.179751	0.126084	-0.159199	0.093563	-0.267558	0.052427	-0.237505	-0.097700	0.342827	0.168560	...	-0.223391	-0

14 rows × 128 columns

In [92]:
```
df.to_csv("node2vec2.csv")
```

In [ ]: