Sean Shiverick 10.07.16

Node Centrality Assignment
This assignment explores node centrality to get an intuitive feel for what the various centrality metrics tell us about the nodes in the graph. This assignment uses the Dolphin social network. Download the graph and load it as a networkx graph.

```
# Load Dophins graph
dolphins = nx.read_graphml('dolphins.graphml')
print(nx.info(dolphins))
Name:
Type: Graph
Number of nodes: 62
Number of edges: 159
Average degree:   5.1290
```

Centrality in Networkx
Networkx has several functions available for calculating the centralities of the nodes in the graph. There are functions for eigenvector, katz, closeness, betweenness, degree, etc. For a full list you can visit the documentation page. The functions take a graph as an argument and return a dictionary with nodes as keys and the centrality as values. This is convenient for us because we can set these as attributes for the nodes in the graph using the set_node_attributes function. For example:

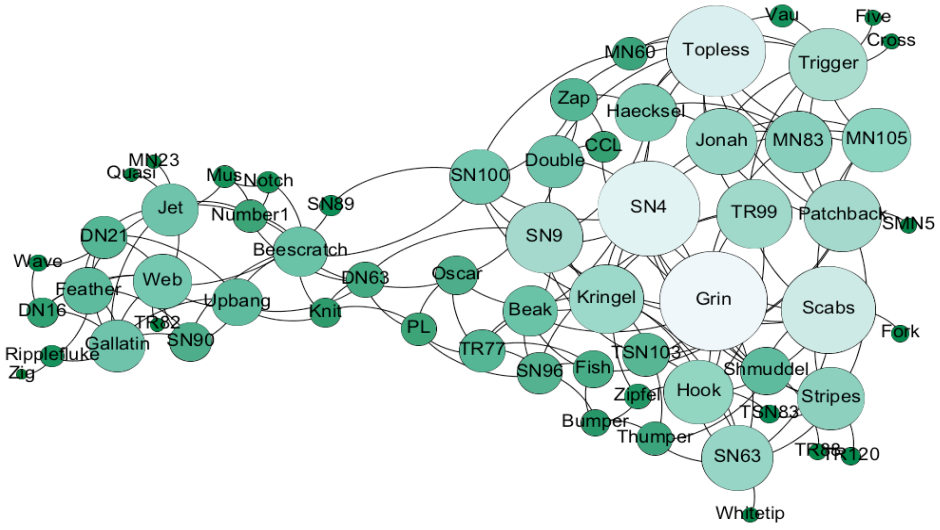
```
## Get the eigenvector centralities for all the nodes
centralities = nx.eigenvector_centrality(dolphins)
# Set attributes of nodes to include centralities; arguments: <graph> <attribute key> <values>
# Where <values> is a dictionary with keys=nodes
nx.set_node_attributes(dolphins, 'eigenvector', centralities)      # refer to node's attributes in graph
print(dolphins.node['3.0']['eigenvector'])

katz_centrality = nx.katz_centrality(dolphins)
nx.set_node_attributes(dolphins, 'katz', katz_centrality)
print(dolphins.node['3.0']['katz'])
0.09402349837042716

closeness_centrality = nx.closeness_centrality(dolphins)
nx.set_node_attributes(dolphins, 'closeness', closeness_centrality)
print(dolphins.node['3.0']['eigenvector'])
0.07933449207476471

betweenness_centrality = nx.betweenness_centrality(dolphins)
nx.set_node_attributes(dolphins, 'betweenness', betweenness_centrality)
print(dolphins.node['3.0']['betweenness'])
0.0023737965131407756

degree_centrality = nx.degree_centrality(dolphins)
nx.set_node_attributes(dolphins, 'degreectr', degree_centrality)
print(dolphins.node['3.0']['degreectr'])
0.04918032786885246

print(dolphins.node['3.0'])
{'r': 0, 'b': 0, 'g': 0, 'betweenness': 0.0023737965131407756, 'size': 10.0, 'katz':
0.09402349837042716, 'degreectr': 0.04918032786885246, 'x': 88.314964, 'label': 'CCL',
'closeness': 0.30808080808080807, 'y': 389.9351, 'eigenvector': 0.07933449207476471}

nx.write_gexf(dolphins, "dolphins.gexf")
```

What to submit: Turn in a PDF that contains your short responses and the visualizations for each of the following questions. **Keep the node location the same** for your graph visualizations.

Picking the right Dolphins
Answer the following questions:

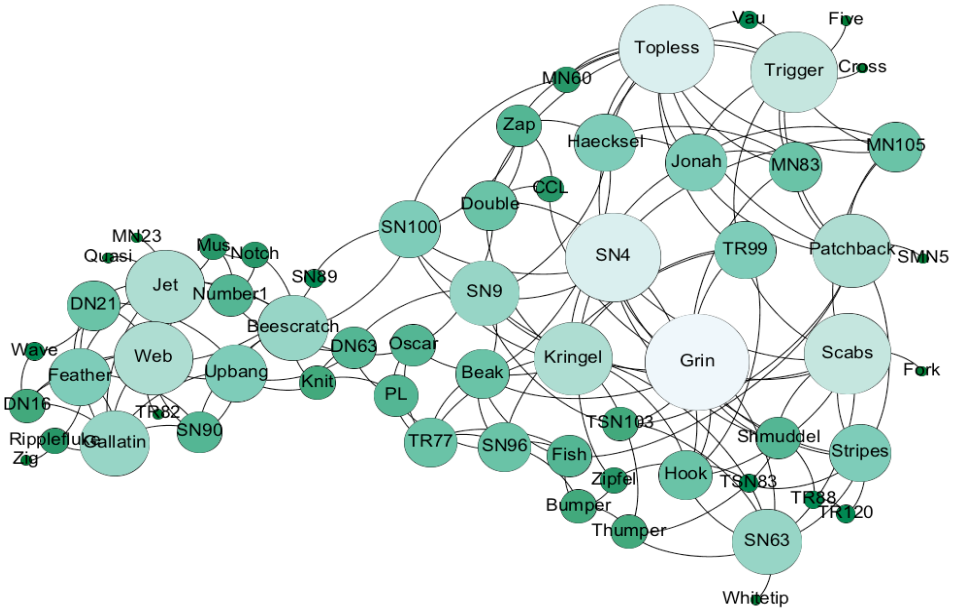
(1) Popularity Contest

For the first question, I chose Katz centrality to measure the most popular dolphins in the network. As indicated in the graph below, the real centers of attraction in the dolphin pod were Grin, SN4, Scabs, and Topless. The Katz centrality provides a result similar to the eigenvector centrality: "a node's importance is the sum of its neighbors' importance". A problem with the eigenvector centrality is that sometimes it only identifies hubs, but does not provide much information for most of the other nodes. The Katz centrality solves this problem by including a free term ("beta", given to every node) and returns a result in which every node will have a non-zero value for centrality. The Katz centrality works nicely for directed or non-directed graphs, whereas the eigenvector centrality does not work well with directed graphs.



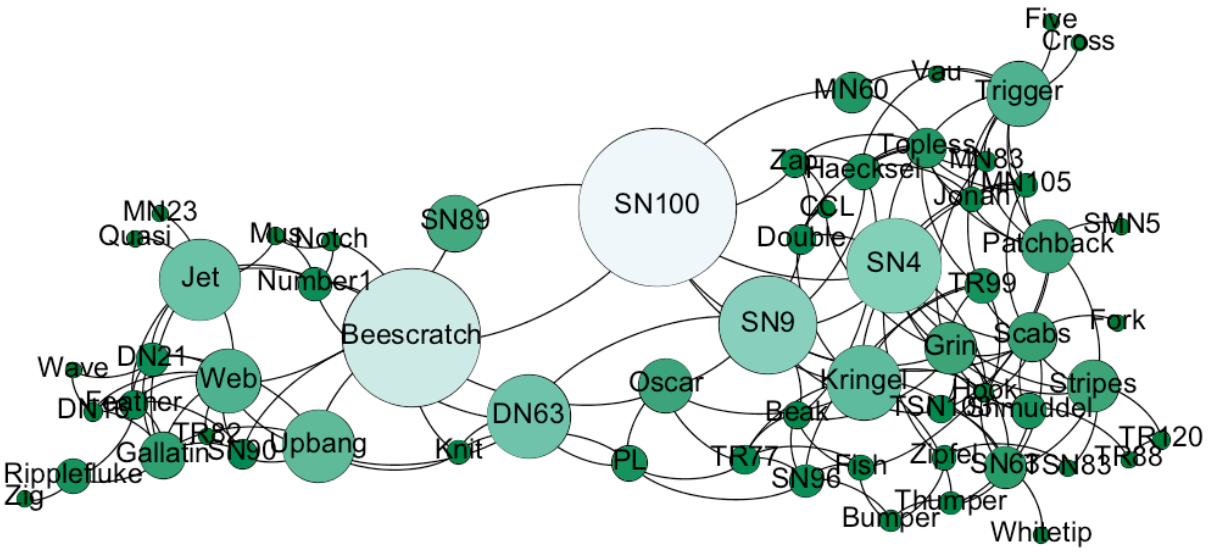
(2) Relay

In considering how dolphins pass information efficiently along the shortest-paths, I chose the degree measure of centrality as the most appropriate because the key ideas of degree centrality are "*spreading communication, in a process that affects neighbors, in parallel*". As the graph below shows, the most important dolphins for relaying message among their neighbors in the network were Grin, SN4, Scabs, Topless, and Trigger. These are the high degree dolphins with the most connections to others in network.



(3) Gossip

For identifying the dolphins that are in the best position for getting all the best gossip from around the pod, I chose the '*Betweenness*' measure of centrality as it identifies the nodes that connect clusters and mediates as many communication paths as possible. This approach picks every random pair of nodes in the network and calculate the shortest path. In this case, the majority of shortest paths had to travel through dolphin SN100 and Beescratch. Betweenness is interesting because it provides a very different measure of importance compared to the other measures of centrality described above. The only dolphin which seemed central in all three measures, *Katz centrality, degree, and betweenness* was SN4.


Five
Cross
Vau
MN60
Trigger
Topless
Zap
Plaecksel
MN83
MN105
Johan
SN100
CCL
Double
SN4
SMN5
Patchback
MN23
Quasi
Mus
Notch
SN89
Jet
Number1
Beescratch
TR99
SN9
Fork
Scabs
Grin
DN21
Wave
Web
Oscar
Kringel
Stripes
Hook
Feather
Beak
Shmuddel
DN63
TR120
Gallatin
Upbang
Knit
PL
TR77
Fish
Zipfel
SN63
SN83
TR88
SN96
Ripplefluke
Zig
Thumper
Bumper
Whitetip