UFRN
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

IMD
INSTITUTO
METRÓPOLE
DIGITAL

Big Data - Foundations and Applications

Lesson #8 - Network Analysis II

Ivanovitch Silva

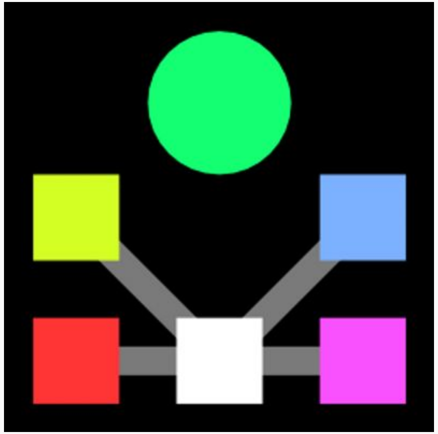
July, 2017

Agenda

- Importing data
- KONECT
- NetworkX and Pandas
- Subgraphs
- Node centrality evaluation

Previously on last class (...)

Importing data from KONECT



http://konect.uni-koblenz.de/

Handbook of Network Analysis: https://goo.gl/9YJ7g6

KONECT (the Koblenz Network Collection) is a project to collect large network datasets of all types in order to perform research in network science and related fields:

- social networks
- hyperlink networks
- authorship networks
- physical networks
- interaction networks
- communication networks.

KONECT: Twitter (ICWSM)

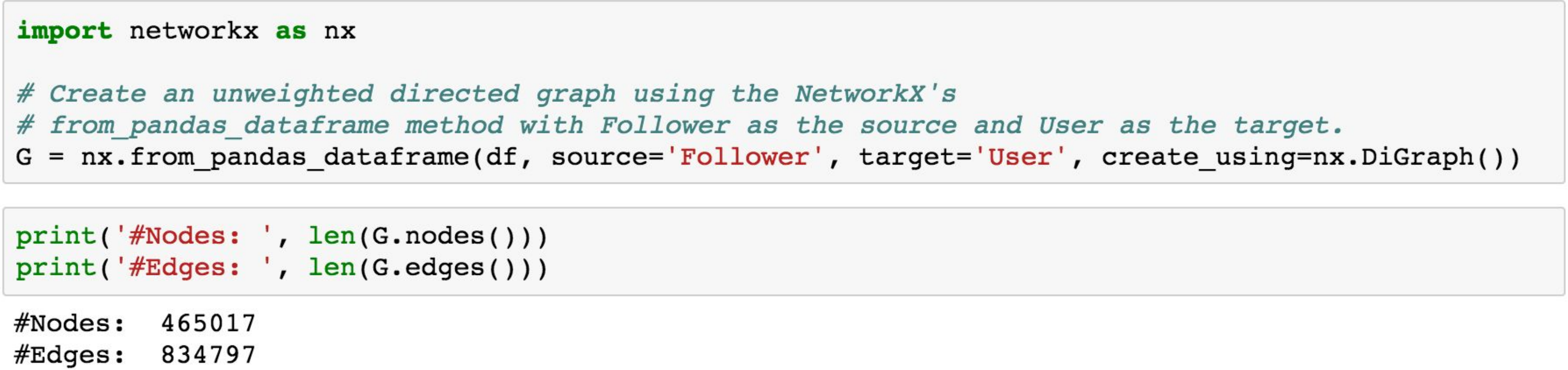
http://konect.uni-koblenz.de/networks/munmun_twitter_social

Network description: this is the directed network containing information about who follows whom on Twitter. Nodes represent users and an edge shows that the left user follows the right one.

- Format
  - Directed
- Edge weights
  - Unweighted

- Size
  - 465,017 vertices (users)
- Volume
  - 834,797 edges (follows)

NetworkX and Pandas

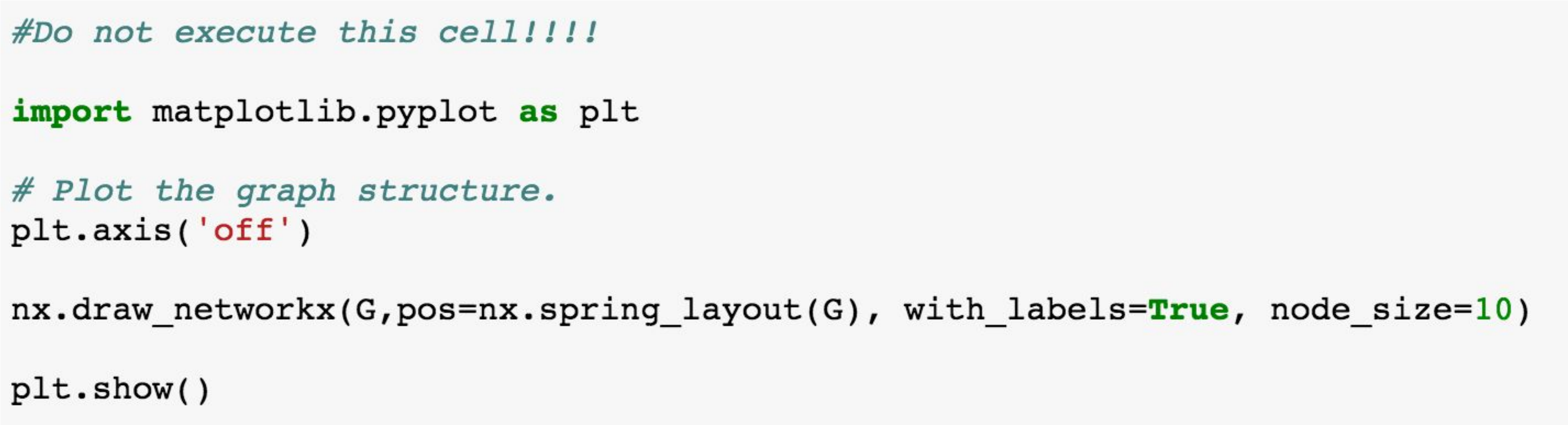


```python
import networkx as nx

# Create an unweighted directed graph using the NetworkX's
# from_pandas_dataframe method with Follower as the source and User as the target.
G = nx.from_pandas_dataframe(df, source='Follower', target='User', create_using=nx.DiGraph())
```

```python
print('#Nodes: ', len(G.nodes()))
print('#Edges: ', len(G.edges()))
```

```
#Nodes:   465017
#Edges:   834797
```

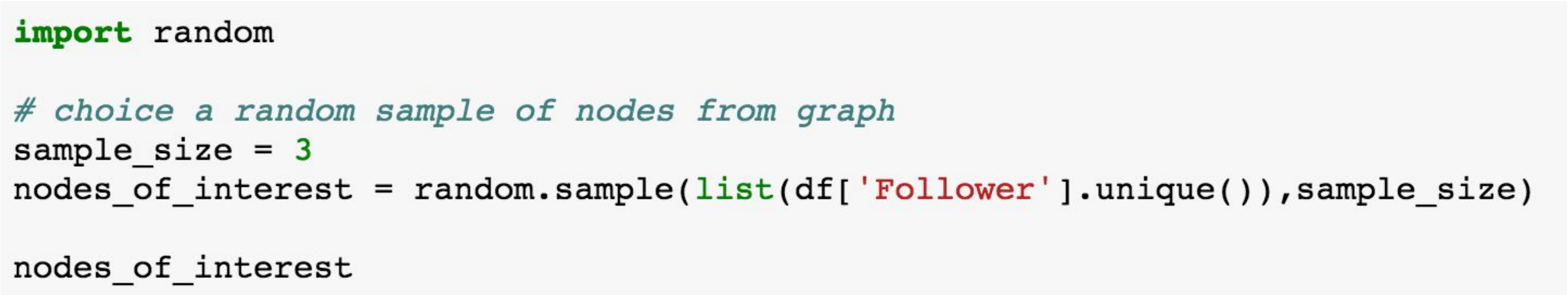
Take it easy!!! ~1M edges!!!



```
#Do not execute this cell!!!!

import matplotlib.pyplot as plt

# Plot the graph structure.
plt.axis('off')

nx.draw_networkx(G,pos=nx.spring_layout(G), with_labels=True, node_size=10)

plt.show()
```

Subgraphs: visualize portions of a large graph

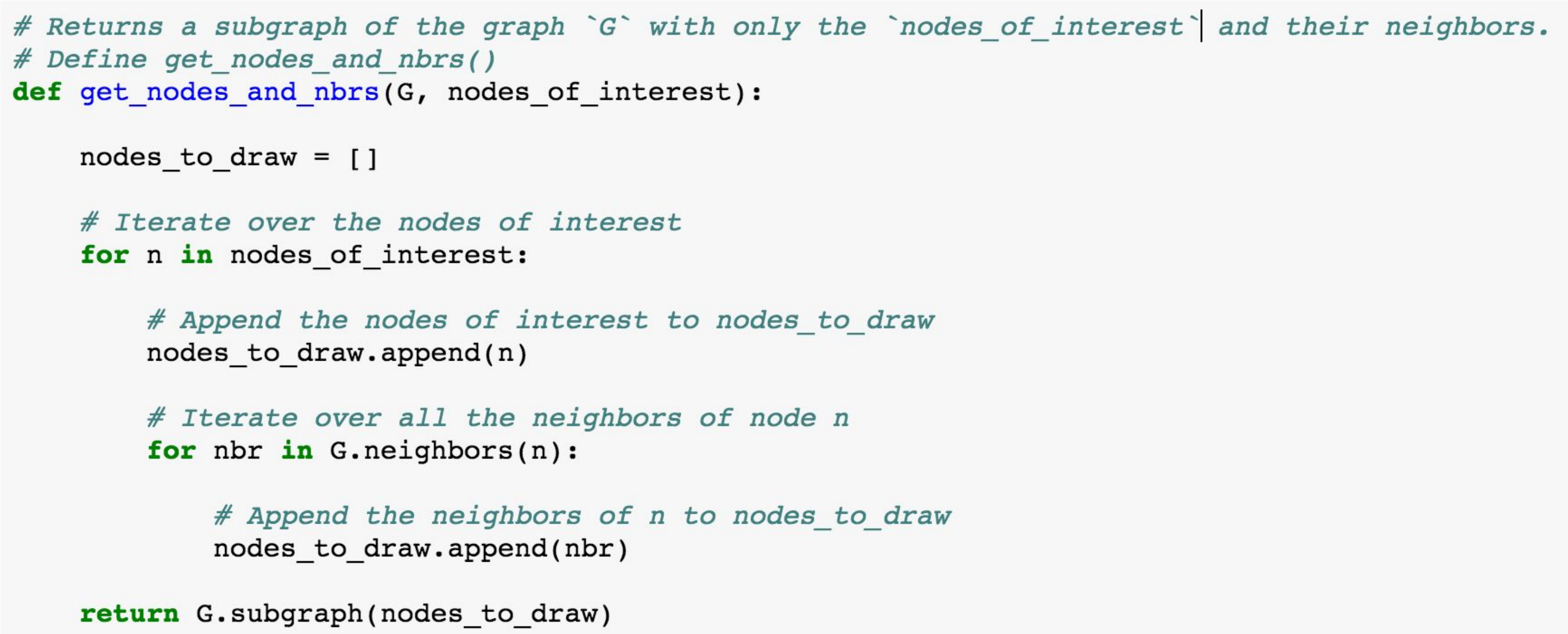
Step #1



```
import random

# choice a random sample of nodes from graph
sample_size = 3
nodes_of_interest = random.sample(list(df['Follower'].unique()),sample_size)

nodes_of_interest
```

Subgraphs: visualize portions of a large graph

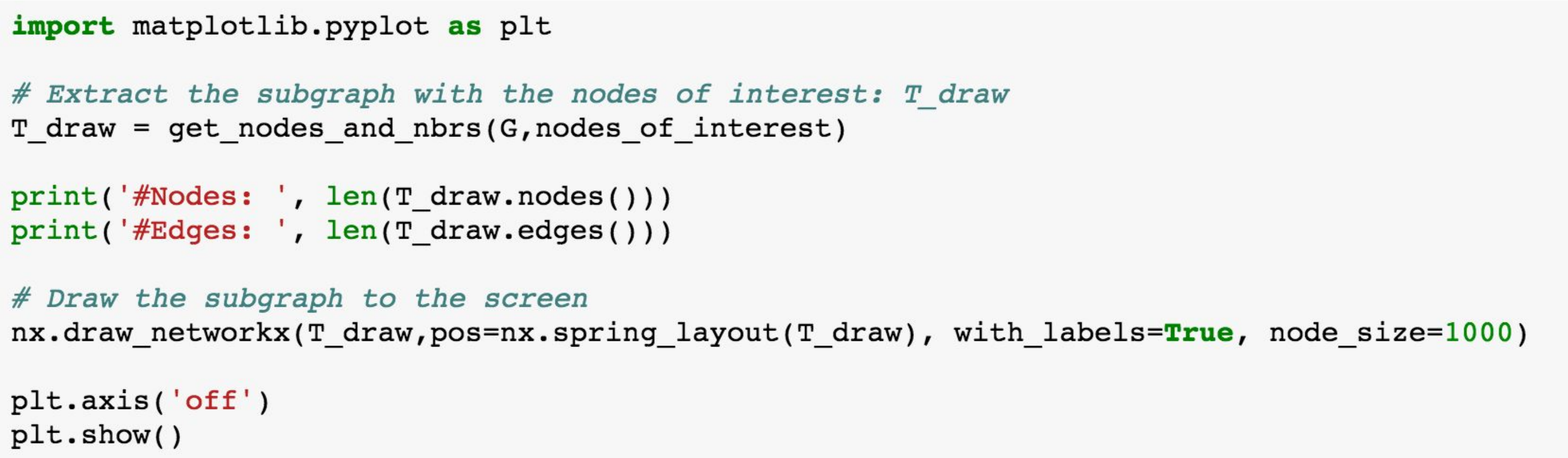
Step #2



```
# Returns a subgraph of the graph `G` with only the `nodes_of_interest` and their neighbors.
# Define get_nodes_and_nbrs()
def get_nodes_and_nbrs(G, nodes_of_interest):

    nodes_to_draw = []

    # Iterate over the nodes of interest
    for n in nodes_of_interest:

        # Append the nodes of interest to nodes_to_draw
        nodes_to_draw.append(n)

        # Iterate over all the neighbors of node n
        for nbr in G.neighbors(n):

            # Append the neighbors of n to nodes_to_draw
            nodes_to_draw.append(nbr)

    return G.subgraph(nodes_to_draw)
```

Subgraphs: visualize portions of a large graph

Step #3

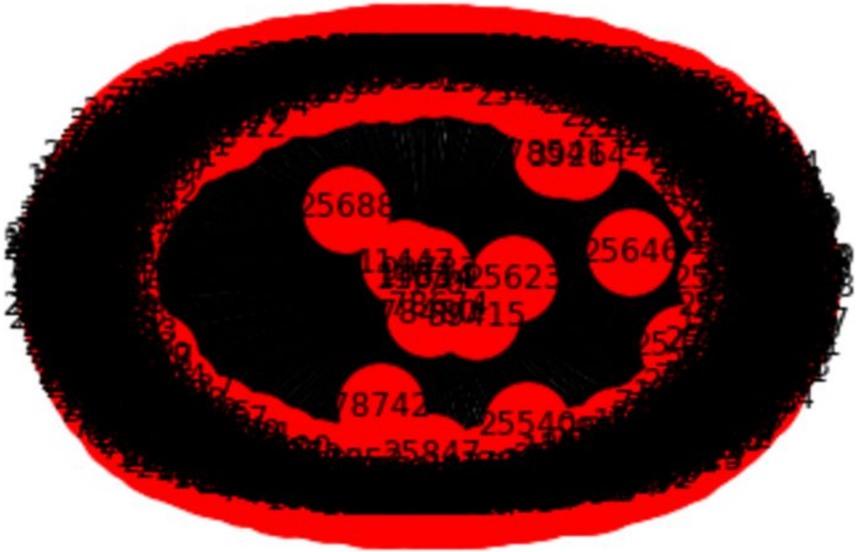


```
import matplotlib.pyplot as plt

# Extract the subgraph with the nodes of interest: T_draw
T_draw = get_nodes_and_nbrs(G,nodes_of_interest)

print('#Nodes: ', len(T_draw.nodes()))
print('#Edges: ', len(T_draw.edges()))

# Draw the subgraph to the screen
nx.draw_networkx(T_draw,pos=nx.spring_layout(T_draw), with_labels=True, node_size=1000)

plt.axis('off')
plt.show()
```

Subgraphs: visualize portions of a large graph

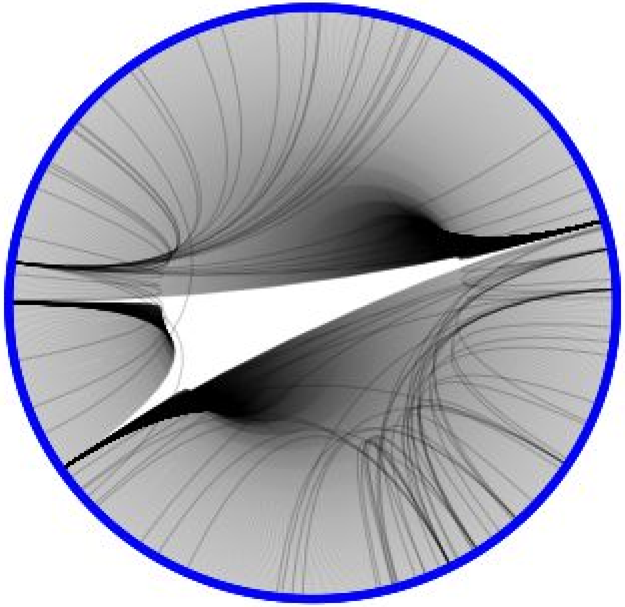
Step #4

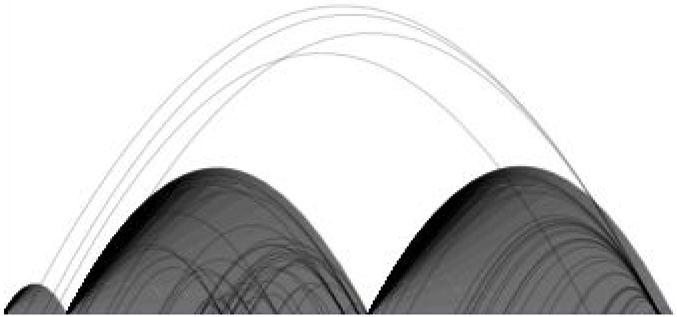
```
#Nodes:   1477
#Edges:   1575
```



Subgraphs: visualize portions of a large graph

Step #5



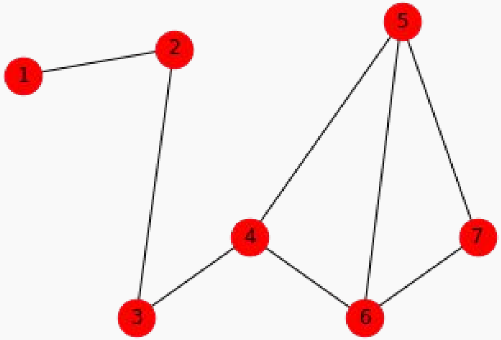


Node centrality evaluation

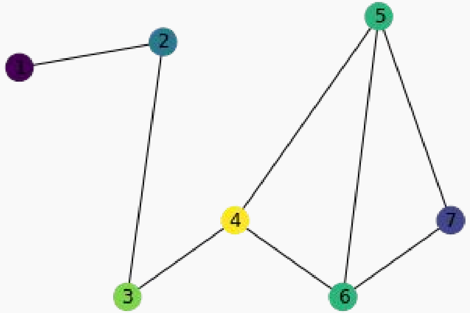
You'll learn about ways of identifying nodes that are important in a network. In doing so, you'll be introduced to more advanced concepts in network analysis.

- Degree centrality
  - Number of connections
- Closeness centrality
  - An important node is typically close to, and can communicate quickly with, the other nodes in the network.
- Betweenness centrality
  - It is a measure of the influence a node has over the spread of information through the network.
- Eigenvector centrality
  - An important node is connected to important neighbors

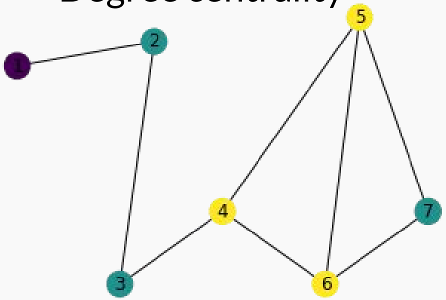
Node centrality evaluation

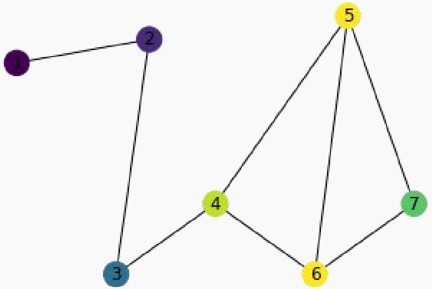


Closeness centrality

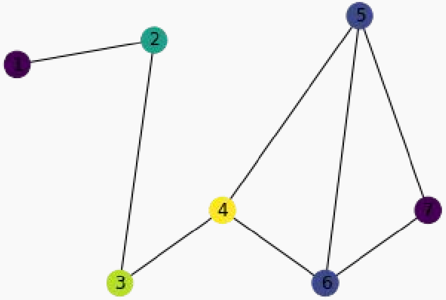


Degree centrality





Eigenvector centrality



Betweenness centrality

https://goo.gl/h7Ij9A

Reference

- http://setosa.io/ev/eigenvectors-and-eigenvalues/
- https://github.com/ericmjl/Network-Analysis-Made-Simple/blob/master/2-networkx-basics-instructor.ipynb
- http://nbviewer.jupyter.org/github/sarguido/networkx-tutorial/blob/master/notebooks/tutorial.ipynb
- https://www.slideshare.net/SarahGuido/network-theory-pycon
- https://github.com/sarguido/networkx-tutorial