Degree_Centrality_Analysis

May 24, 2022

1 (Node) Degree Centrality Analysis

1.1 Introduction

In this notebook, we will do an analysis on the degree centrality of our network, that was scrapped from Reddit.

To get started, let us first state some definitions and facts, which were taken from the Slides of Dr. Khayati from the University of Fribourg. We will denote such pieces of information within a quote block.

> Definition Centrality measures are used to define the central node which: - is important and/or powerful - has an influential/advantageous position on the network

To analyse our network for centrality makes a lot of sense. Since our network represents the interaction between different reddit users, we can analyse the network for influental users.

> There most common centrality measures are: - Based on degree: Degree centrality - Based on geodesics: Closeness and Betweenness - Recursive: Eigenvector and PageRank

We will treat in this document the analysis based on degree. Please refer to the report for the PageRang analysis and more.

Node Degree Centrality First, let us refer to slide 7 of SL03: > Idea: A central actor is the one that has many connections. > > Definition: > - The degree of centrality $d(v_i)$ of a node v_i is the number of ties that vi has in the network. > - In social media networks, $d(v_i)$ represents the number of friends person v_i has.

In our network, $d(v_i)$ doesn't represent the number of friends, since Reddit doesn't have any notion of friends. In our data model, $d(v_i)$ rerpresents the number of different users who commented on user v_i.

Also, let us note here that our data model is a *directed* graph. This means that we need to define the notion of centrality a bit more rigorous. From the slides, we have that:

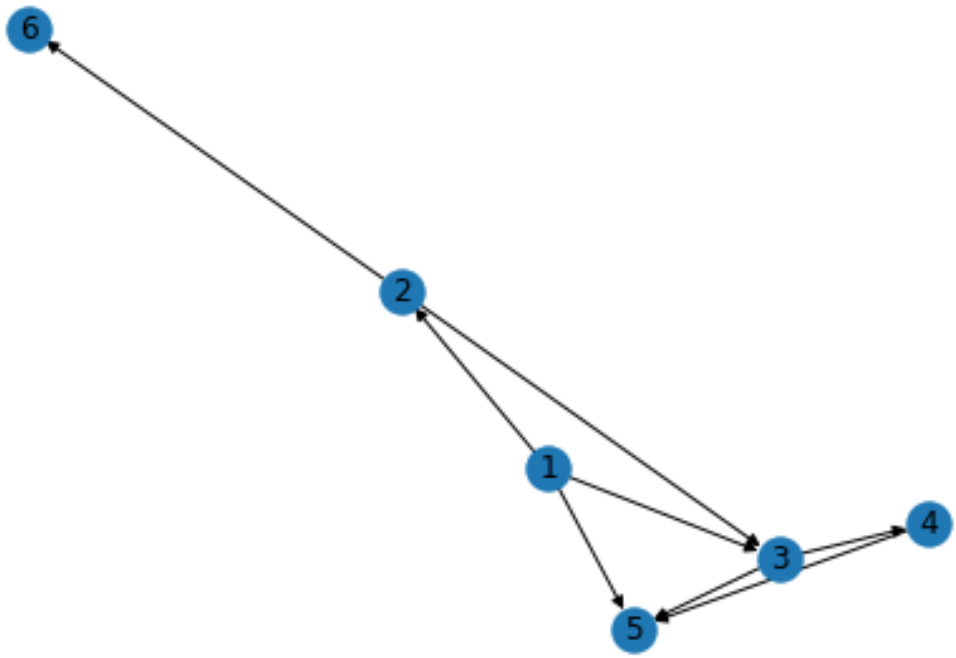
> In directed graphs, the degree centrality value is computed using the in-degree, the out-degree, or the combination.

In this analysis, we will do an analysis on both: the sum of the degrees, the in-degrees and the out-degrees.

1.2 The degree preprocessing

We will consider for the computations here a small graph. We will load our social media network at a later point.

[2]:
```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.DiGraph()
G.add_edge(1, 2)
G.add_edge(1, 3)
G.add_edge(1, 5)
G.add_edge(2, 3)
G.add_edge(3, 4)
G.add_edge(4, 5)
G.add_edge(3, 5)
G.add_edge(2, 6)

nx.draw(G, with_labels=True)
```



We suppose here also that the graph to be analysed has been loaded into a `nx.Graph` or `nx.DiGraph` object. To make computations easier and most importantly **faster**, we will use here `pandas` instead of an own implementation.

First, we have to preprocess our node degrees into a pandas `DataFrame`. We will consider now only

the case for a `nx.DiGraph`.

[3]:
```
import pandas as pd

degrees = [dict(G.degree()), dict(G.in_degree()), dict(G.out_degree())]

#prepare our data to load into pandas
data = {}
for k in degrees[0].keys():
    #by creating tuples of (id, degree, in_degree, out_degree)
    data[k] = tuple(d[k] for d in degrees)

df = pd.DataFrame.from_dict(data, columns=["degree", "in_degree",␣
 ↪"out_degree"], orient='index')
df.index.name = "node"

df
```

[3]:

node	degree	in_degree	out_degree
1	3	0	3
2	3	1	2
3	4	2	2
5	3	3	0
4	2	1	1
6	1	1	0

1.3 Normalizing the degree centrality

As it's often useful to normalize data, we do this here with three different techniques.

Normalization by the maximum possible degree

$$\tilde{d}(v_i) = \frac{d(v_i)}{n-1} \tag{1}$$

[4]:
```
n = G.number_of_nodes()
df['deg_centr_max_poss_degree'] = df['degree']/(n-1)
df
```

[4]:

node	degree	in_degree	out_degree	deg_centr_max_poss_degree
1	3	0	3	0.6
2	3	1	2	0.6
3	4	2	2	0.8
5	3	3	0	0.6
4	2	1	1	0.4
6	1	1	0	0.2

Normalization by the maximum degree

$$d^{\max}(v_i) = \frac{d(v_i)}{\max_{j,j\in\{1,n\}} d(v_j)} \tag{2}$$

[5]:
```
max_degree = df['degree'].max()
df['deg_centr_max_degree'] = df['degree']/max_degree
df
```

[5]:

node	degree	in_degree	out_degree	deg_centr_max_poss_degree	deg_centr_max_degree
1	3	0	3	0.6	0.75
2	3	1	2	0.6	0.75
3	4	2	2	0.8	1.00
5	3	3	0	0.6	0.75
4	2	1	1	0.4	0.50
6	1	1	0	0.2	0.25

Normalization by the degree sum

$$d^{sum}(v_i) = \frac{d(v_i)}{\sum_{j,j\in\{1,n\}} d(v_j)} = \frac{d(v_i)}{2\times|E|} = \frac{d(v_i)}{2\times m}$$

[6]:
```
df['deg_centr_degree_sum'] = df['degree']/G.number_of_edges()
df
```

[6]:

node	degree	in_degree	out_degree	deg_centr_max_poss_degree	deg_centr_max_degree	deg_centr_degree_sum
1	3	0	3	0.6	0.75	0.375
2	3	1	2	0.6	0.75	
3	4	2	2	0.8	1.00	
5	3	3	0	0.6	0.75	
4	2	1	1	0.4	0.50	
6	1	1	0	0.2	0.25	

2	0.75	0.375
3	1.00	0.500
5	0.75	0.375
4	0.50	0.250
6	0.25	0.125

1.4 Graph Degree Centrality

The centrality of a graph is a degree vector V that contains the degree centrality of all nodes. Let A be the adjacency matrix of a undirected graph and let $U \in \mathbb{R}^n$ be the ones vector (all ones). Then,

$$V = A \cdot U$$

[7]:
```
A = nx.to_numpy_matrix(G)
U = [1]*n
V = A.dot(U)
V
```

[7]:
```
matrix([[3., 2., 2., 0., 1., 0.]])
```

1.5 Code Summary: A class

We can summarize this snippets into a small class.

[8]:
```
class DegreeCentrality:

    def __init__(self):
        pass

    def compute_degree_centrality(self, G):
        self.n = G.number_of_nodes()
        self.G = G

        #load graph
        if isinstance(G, nx.DiGraph):
            self.load_digraph(G)
        else:
            self.load_graph(G)

        #compute degree stats
        self.df['deg_centr_max_poss_degree'] = self.df['degree']/(self.n-1)

        max_degree = self.df['degree'].max()
        self.df['deg_centr_max_degree'] = self.df['degree']/max_degree

        deg_sum = self.df['degree'].max()
        self.df['deg_centr_max_degree'] = self.df['degree']/max_degree
```

```
        self.df['deg_centr_degree_sum'] = self.df['degree']/G.number_of_edges()


        if self.digraph:
            self.df['deg_centr_max_degree_out'] = self.df['out_degree']/
↪max_degree
            self.df['deg_centr_max_degree_in'] = self.df['in_degree']/max_degree

            self.df['deg_centr_max_poss_degree_out'] = self.df['out_degree']/
↪(self.n-1)
            self.df['deg_centr_max_poss_degree_in'] = self.df['in_degree']/
↪(self.n-1)

            self.df['deg_centr_degree_sum_out'] = self.df['out_degree']/G.
↪number_of_edges()
            self.df['deg_centr_degree_sum_in'] = self.df['in_degree']/G.
↪number_of_edges()


        return self.df


    def load_digraph(self, G):
        self.digraph = True
        degrees = [dict(G.degree()), dict(G.in_degree()), dict(G.out_degree())]
        data = {}
        for k in degrees[0].keys():
            data[k] = tuple(d[k] for d in degrees)

        self.df = pd.DataFrame.from_dict(data, columns=["degree", "in_degree",␣
↪"out_degree"], orient='index')


    def load_graph(self, G):
        self.digraph = False
        self.df = pd.DataFrame.from_dict(dict(G.degree()), columns=["degree"],␣
↪orient='index')


    def graph_degree_centrality(self):
        return nx.to_numpy_matrix(G).dot([1]*self.n)
```

1.6 Application on our network

For this, we will now load our network data. We have multiple data models. We're going to choose the `deep_link_no_merge` model. Please refer to the report for more details.

[9]:
```
edges = pd.read_csv('../data/csv/edges_deep_link_no_merge.csv')
G = nx.from_pandas_edgelist(edges, create_using=nx.DiGraph(), source='source',␣
↪target='target', edge_attr=['score', 'weight', 'time', 'sub'])
```

[10]:
```
dc = DegreeCentrality()
df = dc.compute_degree_centrality(G)
df
```

[10]:

	degree	in_degree	out_degree	\
Technical-Reason-324	9	7	2	
huttonrtyf	315	296	19	
Xifajk	6	4	2	
DeadSol	39	10	29	
KRONOSDOUBLE	4	0	4	
…	…	…	…	
grimreaperxc	1	0	1	
crexcent	1	0	1	
leirmunumms	1	0	1	
Topacogluahmet	3	0	3	
zarathustratetnuldi	3	3	0	

	deg_centr_max_poss_degree	deg_centr_max_degree	\
Technical-Reason-324	0.000203	0.004839	
huttonrtyf	0.007097	0.169355	
Xifajk	0.000135	0.003226	
DeadSol	0.000879	0.020968	
KRONOSDOUBLE	0.000090	0.002151	
…	…	…	
grimreaperxc	0.000023	0.000538	
crexcent	0.000023	0.000538	
leirmunumms	0.000023	0.000538	
Topacogluahmet	0.000068	0.001613	
zarathustratetnuldi	0.000068	0.001613	

	deg_centr_degree_sum	deg_centr_max_degree_out	\
Technical-Reason-324	0.000054	0.001075	
huttonrtyf	0.001894	0.010215	
Xifajk	0.000036	0.001075	
DeadSol	0.000235	0.015591	
KRONOSDOUBLE	0.000024	0.002151	
…	…	…	
grimreaperxc	0.000006	0.000538	
crexcent	0.000006	0.000538	
leirmunumms	0.000006	0.000538	
Topacogluahmet	0.000018	0.001613	
zarathustratetnuldi	0.000018	0.000000	

```
                      deg_centr_max_degree_in  deg_centr_max_poss_degree_out  \
Technical-Reason-324                 0.003763                       0.000045
huttonrtyf                           0.159140                       0.000428
Xifajk                               0.002151                       0.000045
DeadSol                              0.005376                       0.000653
KRONOSDOUBLE                         0.000000                       0.000090
…                                         …                              …
grimreaperxc                         0.000000                       0.000023
crexcent                             0.000000                       0.000023
leirmunumms                          0.000000                       0.000023
Topacogluahmet                       0.000000                       0.000068
zarathustratetnuldi                  0.001613                       0.000000

                      deg_centr_max_poss_degree_in  deg_centr_degree_sum_out  \
Technical-Reason-324                      0.000158                  0.000012
huttonrtyf                                0.006669                  0.000114
Xifajk                                    0.000090                  0.000012
DeadSol                                   0.000225                  0.000174
KRONOSDOUBLE                              0.000000                  0.000024
…                                              …                         …
grimreaperxc                              0.000000                  0.000006
crexcent                                  0.000000                  0.000006
leirmunumms                               0.000000                  0.000006
Topacogluahmet                            0.000000                  0.000018
zarathustratetnuldi                       0.000068                  0.000000

                      deg_centr_degree_sum_in
Technical-Reason-324                 0.000042
huttonrtyf                           0.001780
Xifajk                               0.000024
DeadSol                              0.000060
KRONOSDOUBLE                         0.000000
…                                         …
grimreaperxc                         0.000000
crexcent                             0.000000
leirmunumms                          0.000000
Topacogluahmet                       0.000000
zarathustratetnuldi                  0.000018

[44386 rows x 12 columns]
```

Now, let's sort the data by some column

[11]:
```
s = df.sort_values(by='deg_centr_max_poss_degree', ascending=False)
s['deg_centr_max_poss_degree'].plot()
s
```

[11]:

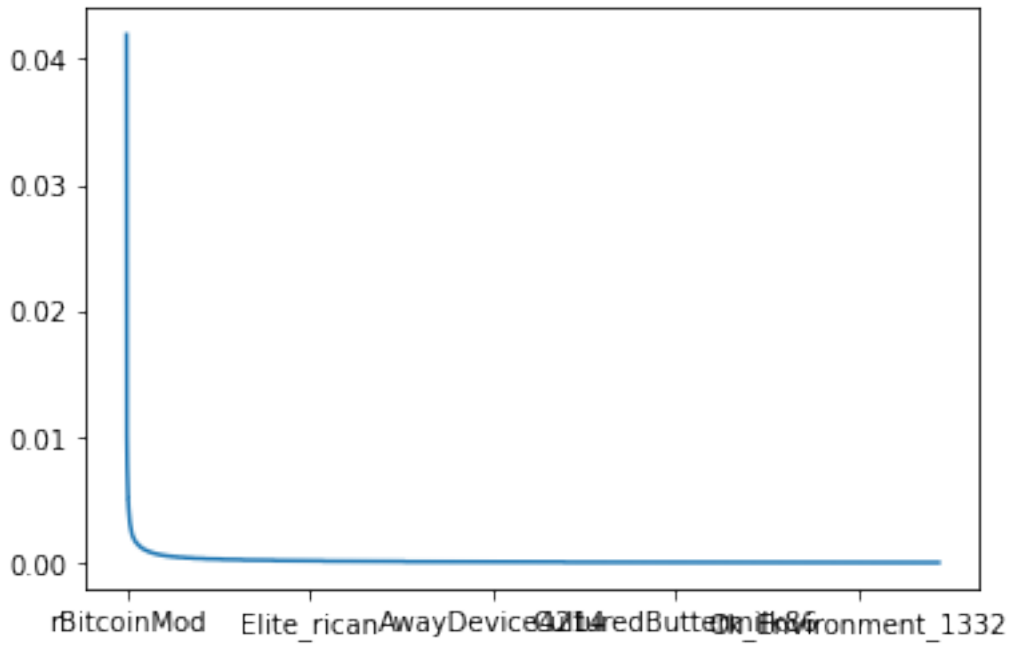
	degree	in_degree	out_degree	deg_centr_max_poss_degree	\
rBitcoinMod	1860	1860	0	0.041906	
AutoModerator	1391	124	1267	0.031339	
42points	978	973	5	0.022034	
bitusher	795	271	524	0.017911	
Reahvenz	776	775	1	0.017483	
…	…	…	…	…	
apartImpeach87	1	0	1	0.000023	
Pako1986	1	0	1	0.000023	
Chemical_Cutthroat	1	0	1	0.000023	
nomadicshorty	1	0	1	0.000023	
Furukawa_Cali	1	0	1	0.000023	

	deg_centr_max_degree	deg_centr_degree_sum	\
rBitcoinMod	1.000000	0.011186	
AutoModerator	0.747849	0.008366	
42points	0.525806	0.005882	
bitusher	0.427419	0.004781	
Reahvenz	0.417204	0.004667	
…	…	…	
apartImpeach87	0.000538	0.000006	
Pako1986	0.000538	0.000006	
Chemical_Cutthroat	0.000538	0.000006	
nomadicshorty	0.000538	0.000006	
Furukawa_Cali	0.000538	0.000006	

	deg_centr_max_degree_out	deg_centr_max_degree_in	\
rBitcoinMod	0.000000	1.000000	
AutoModerator	0.681183	0.066667	
42points	0.002688	0.523118	
bitusher	0.281720	0.145699	
Reahvenz	0.000538	0.416667	
…	…	…	
apartImpeach87	0.000538	0.000000	
Pako1986	0.000538	0.000000	
Chemical_Cutthroat	0.000538	0.000000	
nomadicshorty	0.000538	0.000000	
Furukawa_Cali	0.000538	0.000000	

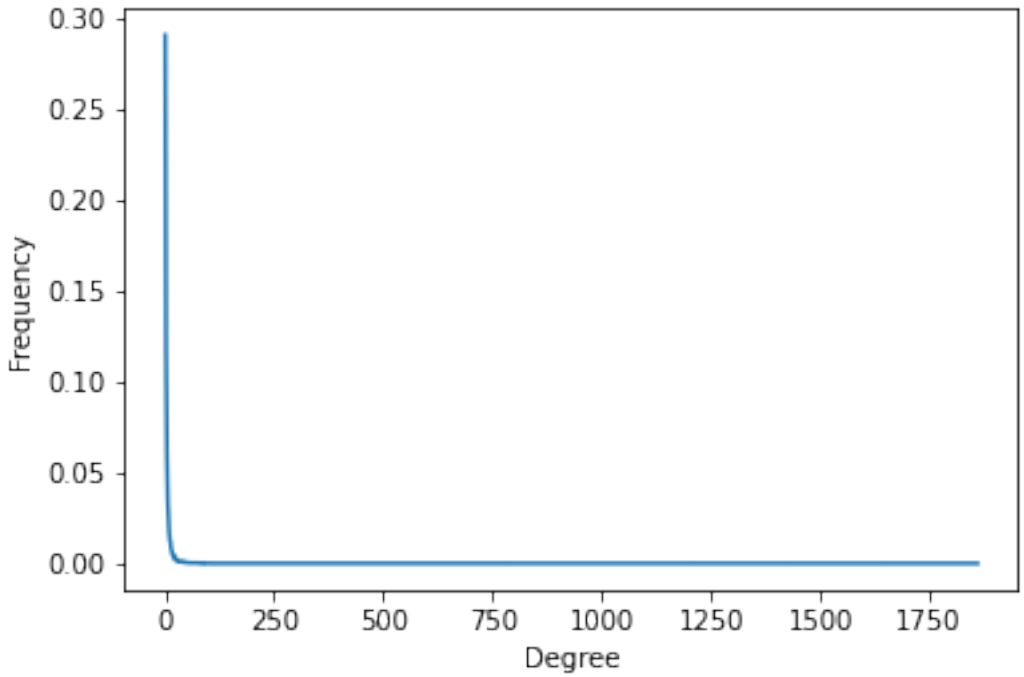
	deg_centr_max_poss_degree_out	\
rBitcoinMod	0.000000	
AutoModerator	0.028546	
42points	0.000113	
bitusher	0.011806	
Reahvenz	0.000023	
…	…	
apartImpeach87	0.000023	

```
Pako1986                                  0.000023
Chemical_Cutthroat                        0.000023
nomadicshorty                             0.000023
Furukawa_Cali                             0.000023

                   deg_centr_max_poss_degree_in  deg_centr_degree_sum_out  \
rBitcoinMod                            0.041906                  0.000000
AutoModerator                          0.002794                  0.007620
42points                               0.021922                  0.000030
bitusher                               0.006106                  0.003151
Reahvenz                               0.017461                  0.000006
…                                         …                         …
apartImpeach87                         0.000000                  0.000006
Pako1986                               0.000000                  0.000006
Chemical_Cutthroat                     0.000000                  0.000006
nomadicshorty                          0.000000                  0.000006
Furukawa_Cali                          0.000000                  0.000006

                   deg_centr_degree_sum_in
rBitcoinMod                       0.011186
AutoModerator                     0.000746
42points                          0.005852
bitusher                          0.001630
Reahvenz                          0.004661
…                                    …
apartImpeach87                    0.000000
Pako1986                          0.000000
Chemical_Cutthroat                0.000000
nomadicshorty                     0.000000
Furukawa_Cali                     0.000000

[44386 rows x 12 columns]
```



[19]:
```
import numpy as np
def plot_degree_histogram(g, normalized=True):
    aux_y = nx.degree_histogram(g)
    aux_x = np.arange(0, len(aux_y)).tolist()
    n_nodes = g.number_of_nodes()
    if normalized:
        for i in range(len(aux_y)):
            aux_y[i] = aux_y[i]/n_nodes
    return aux_x, aux_y
```

[21]:
```
import seaborn as sns
x,y = plot_degree_histogram(G)
ax=sns.lineplot(x=x[1:],y=y[1:])
ax.set(xlabel="Degree", ylabel="Frequency")
plt.savefig("../data/images/deg_dist.pdf")
```



We can clearly see that the nodes follow a power law distribution.

We could now filter the users by some criteria about their node degree. E.g.

[14]:
```
s[s['deg_centr_max_degree'] > 0.2]
```

[14]:

	degree	in_degree	out_degree	deg_centr_max_poss_degree	\
rBitcoinMod	1860	1860	0	0.041906	
AutoModerator	1391	124	1267	0.031339	
42points	978	973	5	0.022034	
bitusher	795	271	524	0.017911	
Reahvenz	776	775	1	0.017483	
CoinCorner_Sam	697	633	64	0.015704	
coinbasesupport	688	191	497	0.015501	
NiceDoctorBeam	669	320	349	0.015073	
KiwipOrn	628	224	404	0.014149	
liquid_at	613	177	436	0.013811	
jakkkmotivator	567	567	0	0.012775	
Nada_Lives	555	232	323	0.012504	
Egon_1	544	498	46	0.012256	
Bitcoin__Hodler	535	405	130	0.012054	
igadjeed	521	128	393	0.011738	
Mallardshead	516	333	183	0.011626	
frank__costello	502	316	186	0.011310	
KAX1107	478	474	4	0.010769	

ShotBot	465	458	7	0.010477
Deep-Art3195	447	420	27	0.010071
Perleflamme	424	94	330	0.009553
sylsau	421	354	67	0.009485
Fiach_Dubh	414	393	21	0.009327
slvbtc	414	382	32	0.009327
thefullmcnulty	412	265	147	0.009282
buzzbooz	395	339	56	0.008899
hyperinflationUSA	395	352	43	0.008899
ohnoh18	384	381	3	0.008652
Bear_Hammer_99	379	376	3	0.008539
jessquit	376	192	184	0.008471
Ok_Aerie3546	376	190	186	0.008471
BashCo	374	275	99	0.008426

	deg_centr_max_degree	deg_centr_degree_sum	\
rBitcoinMod	1.000000	0.011186	
AutoModerator	0.747849	0.008366	
42points	0.525806	0.005882	
bitusher	0.427419	0.004781	
Reahvenz	0.417204	0.004667	
CoinCorner_Sam	0.374731	0.004192	
coinbasesupport	0.369892	0.004138	
NiceDoctorBeam	0.359677	0.004023	
KiwipOrn	0.337634	0.003777	
liquid_at	0.329570	0.003687	
jakkkmotivator	0.304839	0.003410	
Nada_Lives	0.298387	0.003338	
Egon_1	0.292473	0.003272	
Bitcoin__Hodler	0.287634	0.003218	
igadjeed	0.280108	0.003133	
Mallardshead	0.277419	0.003103	
frank__costello	0.269892	0.003019	
KAX1107	0.256989	0.002875	
ShotBot	0.250000	0.002797	
Deep-Art3195	0.240323	0.002688	
Perleflamme	0.227957	0.002550	
sylsau	0.226344	0.002532	
Fiach_Dubh	0.222581	0.002490	
slvbtc	0.222581	0.002490	
thefullmcnulty	0.221505	0.002478	
buzzbooz	0.212366	0.002376	
hyperinflationUSA	0.212366	0.002376	
ohnoh18	0.206452	0.002309	
Bear_Hammer_99	0.203763	0.002279	
jessquit	0.202151	0.002261	
Ok_Aerie3546	0.202151	0.002261	

```
BashCo                          0.201075              0.002249

                  deg_centr_max_degree_out  deg_centr_max_degree_in  \
rBitcoinMod                       0.000000                 1.000000
AutoModerator                     0.681183                 0.066667
42points                          0.002688                 0.523118
bitusher                          0.281720                 0.145699
Reahvenz                          0.000538                 0.416667
CoinCorner_Sam                    0.034409                 0.340323
coinbasesupport                   0.267204                 0.102688
NiceDoctorBeam                    0.187634                 0.172043
KiwipOrn                          0.217204                 0.120430
liquid_at                         0.234409                 0.095161
jakkkmotivator                    0.000000                 0.304839
Nada_Lives                        0.173656                 0.124731
Egon_1                            0.024731                 0.267742
Bitcoin__Hodler                   0.069892                 0.217742
igadjeed                          0.211290                 0.068817
Mallardshead                      0.098387                 0.179032
frank__costello                   0.100000                 0.169892
KAX1107                           0.002151                 0.254839
ShotBot                           0.003763                 0.246237
Deep-Art3195                      0.014516                 0.225806
Perleflamme                       0.177419                 0.050538
sylsau                            0.036022                 0.190323
Fiach_Dubh                        0.011290                 0.211290
slvbtc                            0.017204                 0.205376
thefullmcnulty                    0.079032                 0.142473
buzzbooz                          0.030108                 0.182258
hyperinflationUSA                 0.023118                 0.189247
ohnoh18                           0.001613                 0.204839
Bear_Hammer_99                    0.001613                 0.202151
jessquit                          0.098925                 0.103226
Ok_Aerie3546                      0.100000                 0.102151
BashCo                            0.053226                 0.147849

                  deg_centr_max_poss_degree_out  \
rBitcoinMod                            0.000000
AutoModerator                          0.028546
42points                               0.000113
bitusher                               0.011806
Reahvenz                               0.000023
CoinCorner_Sam                         0.001442
coinbasesupport                        0.011197
NiceDoctorBeam                         0.007863
KiwipOrn                               0.009102
liquid_at                              0.009823
```

jakkkmotivator	0.000000
Nada_Lives	0.007277
Egon_1	0.001036
Bitcoin__Hodler	0.002929
igadjeed	0.008854
Mallardshead	0.004123
frank__costello	0.004191
KAX1107	0.000090
ShotBot	0.000158
Deep-Art3195	0.000608
Perleflamme	0.007435
sylsau	0.001510
Fiach_Dubh	0.000473
slvbtc	0.000721
thefullmcnulty	0.003312
buzzbooz	0.001262
hyperinflationUSA	0.000969
ohnoh18	0.000068
Bear_Hammer_99	0.000068
jessquit	0.004146
Ok_Aerie3546	0.004191
BashCo	0.002230

	deg_centr_max_poss_degree_in	deg_centr_degree_sum_out	\
rBitcoinMod	0.041906	0.000000	
AutoModerator	0.002794	0.007620	
42points	0.021922	0.000030	
bitusher	0.006106	0.003151	
Reahvenz	0.017461	0.000006	
CoinCorner_Sam	0.014262	0.000385	
coinbasesupport	0.004303	0.002989	
NiceDoctorBeam	0.007210	0.002099	
KiwipOrn	0.005047	0.002430	
liquid_at	0.003988	0.002622	
jakkkmotivator	0.012775	0.000000	
Nada_Lives	0.005227	0.001943	
Egon_1	0.011220	0.000277	
Bitcoin__Hodler	0.009125	0.000782	
igadjeed	0.002884	0.002364	
Mallardshead	0.007503	0.001101	
frank__costello	0.007120	0.001119	
KAX1107	0.010679	0.000024	
ShotBot	0.010319	0.000042	
Deep-Art3195	0.009463	0.000162	
Perleflamme	0.002118	0.001985	
sylsau	0.007976	0.000403	
Fiach_Dubh	0.008854	0.000126	

slvbtc	0.008607	0.000192
thefullmcnulty	0.005970	0.000884
buzzbooz	0.007638	0.000337
hyperinflationUSA	0.007931	0.000259
ohnoh18	0.008584	0.000018
Bear_Hammer_99	0.008471	0.000018
jessquit	0.004326	0.001107
Ok_Aerie3546	0.004281	0.001119
BashCo	0.006196	0.000595

	deg_centr_degree_sum_in
rBitcoinMod	0.011186
AutoModerator	0.000746
42points	0.005852
bitusher	0.001630
Reahvenz	0.004661
CoinCorner_Sam	0.003807
coinbasesupport	0.001149
NiceDoctorBeam	0.001925
KiwipOrn	0.001347
liquid_at	0.001065
jakkkmotivator	0.003410
Nada_Lives	0.001395
Egon_1	0.002995
Bitcoin__Hodler	0.002436
igadjeed	0.000770
Mallardshead	0.002003
frank__costello	0.001900
KAX1107	0.002851
ShotBot	0.002754
Deep-Art3195	0.002526
Perleflamme	0.000565
sylsau	0.002129
Fiach_Dubh	0.002364
slvbtc	0.002297
thefullmcnulty	0.001594
buzzbooz	0.002039
hyperinflationUSA	0.002117
ohnoh18	0.002291
Bear_Hammer_99	0.002261
jessquit	0.001155
Ok_Aerie3546	0.001143
BashCo	0.001654

1.7 Summary

We discovered the most influential users using the node degree centrality. However, we need to keep in mind that some of these users are bots like moderator bots who automatically comment on

certain posts. To filter out these, we need more advanced techniques and most likely also a bit of ML, as the Reddit API doesn't distinguish between user or bot/moderator comments.