```
! pip install networkx
! pip install plotly
! pip install colorlover
```

```
Requirement already satisfied: networkx in /usr/local/lib/python3.6/dist-packages (2.4)
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: plotly in /usr/local/lib/python3.6/dist-packages (4.1.1)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from plotly
Requirement already satisfied: retrying>=1.3.3 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: colorlover in /usr/local/lib/python3.6/dist-packages (0.3.
```

```
import networkx as nx
import pandas as pd
```

Q1. Choose a hashtag

#photography

```
df = pd.read_csv("tweets2009-06-0115.csv.zip", sep='\t', compression='zip')
```

```
print("Num of rows:", df.shape[0])
```

```
Num of rows: 3437690
```

```
df.head()
```

	date	user	tweet
0	2009-06-01 21:43:59	burtonator	No Post Title
1	2009-06-01 21:47:23	burtonator	No Post Title
2	2009-06-02 01:15:44	burtonator	No Post Title
3	2009-06-02 05:17:52	burtonator	No Post Title
4	2009-06-02 23:58:25	burtonator	No Post Title

Find the most common hash tag

```
from collections import Counter
```

```
allTweets = df["tweet"].str.cat(sep=' ')
tweetWords = [word.strip(""" ,.:'\";""").lower() for word in allTweets.split()]
hashTags = [word for word in tweetWords if word.startswith("#")]
hashTagsCounter = Counter(hashTags)
```

```
hashTagsCounter.most_common(50)
```

```
[('#iranelection', 26853),
 ('#followfriday', 16400),
 ('#jobs', 13322),
 ('#iremember', 11057),
 ('#spymaster', 10587),
 ('#ff', 10446),
 ('#squarespace', 9198),
 ('#tcot', 7691),
 ('#fb', 6107),
 ('#cnnfail', 4451),
 ('#11thcommandment', 3429),
 ('#jtv', 3317),
 ('#140mafia', 3144),
 ('#iran', 2935),
 ('#', 2895),
 ('#news', 2837),
 ('#quote', 2750),
 ('#vampirebite', 2634),
 ('#1', 2587),
 ('#bsb', 2433),
 ('#tweetmyjobs', 2086),
 ('#iphone', 1697),
 ('#lastfm', 1599),
 ('#mp2', 1589),
 ('#niley', 1528),
 ('#music', 1489),
 ('#p2', 1439),
 ('#follow', 1390),
 ('#pawpawty', 1305),
 ('#hhrs', 1256),
 ('#fail', 1246),
 ('#twitter', 1216),
 ('#tlot', 1214),
 ('#facebook', 1177),
 ('#sgp', 1151),
 ('#mashchat', 1143),
 ('#tinychat', 1111),
 ('#2', 1107),
 ('#digg', 1102),
 ('#gop', 1009),
 ('#phish', 1001),
 ('#mlb', 962),
 ('#travel', 932),
 ('#bonnaroo', 887),
 ('#twitpocalypse', 879),
 ('#iranelections', 857),
 ('#rt', 856),
 ('#zensursula', 811),
 ('#jamlegend', 790),
 ('#quotes', 756)]
```

```
pgTag = df[df["tweet"].str.lower().str.contains("#photography", na=False)].copy()
```

```
def addMentionedColumn(df):

    def mentionsList(txt):
        allWords = [word.strip(""" ,.:'\";""").lower() for word in txt.split()]
        allNames = [word.strip("@") for word in allWords if word.startswith("@")]
        uniqueNames = list(set(allNames))
```

```
        uniqueNames = list(set(allNames))
        return allNames

    df["mentioned"] = df["tweet"].apply(mentionsList)

addMentionedColumn(pgTag)

pgTag.head(50)
```

	date	user	tweet	mentioned
1147	2009-06-11 16:58:53	base10	Took a walk at lunch. My lunch-time architectu...	[]
12882	2009-06-11 17:15:44	replayphotos	Just added myself to the http://wefollow.com t...	[]
14482	2009-06-11 17:17:56	cottontw	Just added myself to the http://wefollow.com t...	[]
15171	2009-06-11 17:18:34	gospain	All I can find at the moment about the Noche d...	[]
23958	2009-06-11 17:32:48	hiway	Finally got around to setting up my #photograp...	[]
31519	2009-06-11 17:46:37	pwcarey	@murrayed Thanks again for the book recommend....	[murrayed]
34693	2009-06-11 17:51:09	karensperling	It's official! New writeup in Lexjet Focus #Pa...	[]
40725	2009-06-11 17:59:57	photocentrum	#photography Kata's PR Bags: Grab Your Gear an...	[]
41012	2009-06-11 18:00:18	photocentrum	#photography Casio Exilim EX-S12: Last update ...	[]
44876	2009-06-11 18:05:45	wayneofthedead	http://twitpic.com/74pv2 - #photography - is i...	[]
46686	2009-06-11 18:07:26	getphotograjobs	Retoucher - Hong Kong, Hong Kong (http://tinyu...	[]
55667	2009-06-11 18:20:38	svenyurgensson	Это, я хочу сказать, круто, братцы! A photogra...	[quiteuseful)]
55972	2009-06-11 18:21:00	rob_b2805	Just finished re-mounting the new lens onto a ...	[]
57992	2009-06-11 18:23:54	pr_photography	True or False :: A330 Crash Captured Onboard w...	[]
62571	2009-06-11 18:31:46	bradhuss	AMAZING Photo journalistic perspectives http:/...	[]
67808	2009-06-11	scanmyphotos	5 Things You Didn't Know About Memory Cards ht...	[popphoto]

	18:43:39		About Memory Cards ht...	
72726	2009-06-11 18:50:22	photocentrum	#photography Canon PowerShot S20 review http:/...	[]
80727	2009-06-11 19:04:50	pr_photography	Awkwaaaard!!! http://bit.ly/eFLsu #postrank #p...	[]
90271	2009-06-11 19:19:48	jason_pollock	#Photography: The Rules Of Composition (VIDEO)...	[]
92619	2009-06-11 19:22:52	ecotrotters	RT @HavePack: Top 10 Man-Made Wonders of the W...	[havepack]
99511	2009-06-11 19:35:37	thenightwriterz	Episode 005 of The Night Writerz is available,...	[]
110247	2009-06-11 19:50:00	dawnjohio	@geofollow Springfield, Illinois 62704 #ghosth...	[geofollow]
116366	2009-06-11 19:58:42	photoclubs	#photography 30 Colorful Shots Of High Speed B...	[photoframd)]
116673	2009-06-11 19:58:58	conflagratio	In case you missed it, I updated my "Aqua" Gal...	[]
117414	2009-06-11 19:59:41	cophotographer	RT @gregkb http://bit.ly/i2kkS (via @andrewhyd...	[gregkb, andrewhyde)]
118496	2009-06-11 20:00:48	pixum	Top 10 Man-Made Wonders of the World – a Photo...	[havepack)]
119944	2009-06-11 20:06:18	starstruk	Trust I seek & I find in you. Every day for us...	[fantasyparade, mojolabs, hypem, housymphony, ...
121075	2009-06-11 20:10:04	theconstruct	Take a look at my friend stacie's boudoir phot...	[]
122186	2009-06-11 20:12:18	photoclubs	#photography 30 Colorful Shots Of High Speed B...	[photoframd)]
124202	2009-06-11 20:16:45	eddyizm	#photography #etsy the duomo (firenze series) ...	[]
124929	2009-06-11 20:17:52	staycation_la	The @Hammer_Museum has a Sebastiao Salgado Pod...	[hammer_museum]
129428	2009-06-11 20:26:11	ashleyrwatts	Just added myself to the http://wefollow.com t...	[]

134487	2009-06-11 20:36:22	melaniewci	Really beautiful underwater #photography from ...	[jasondpg]
153197	2009-06-11 21:04:56	newcastleweb	Eye-Fi launch wireless internet memory card fo...	[]
154116	2009-06-11 21:07:55	gertmuurling	an oldie from paris #photography http://twitpi...	[]
172617	2009-06-11 21:42:53	lidaverner	Just added myself to the http://wefollow.com t...	[]
180823	2009-06-11 21:53:31	filllight	Added a Anhinga trying to swallow a Bream to m...	[]

```
pgTag.shape
```

```
(658, 4)
```

Q2. Build a Mention Graph

```
def mentionGraph(df):
    g = nx.Graph()

    for (index, date, user, tweet, mentionedUsers) in df.itertuples():
        for mentionedUser in mentionedUsers:
            if (user in g) and (mentionedUser in g[user]):
                g[user][mentionedUser]["numberMentions"] += 1
            else:
                g.add_edge(user, mentionedUser, numberMentions=1)

    return g

pgGraph = mentionGraph(pgTag)

# use nx.info to show the nodes and edges, also average degree
print(nx.info(pgGraph))
```

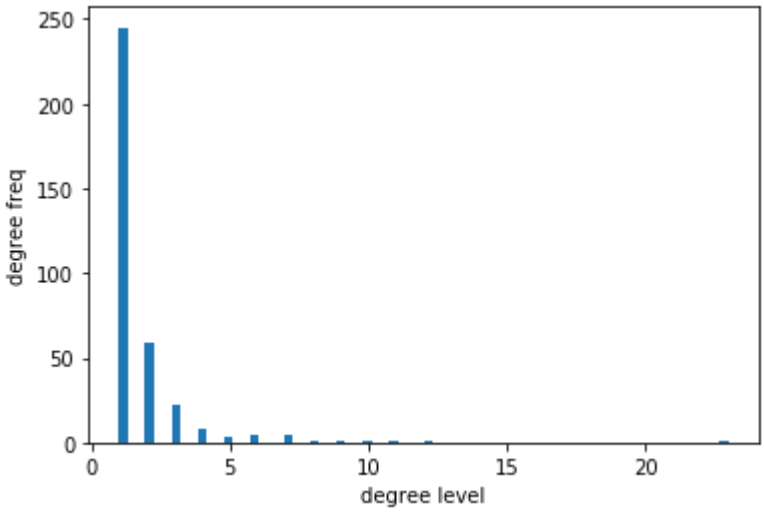
```
Name:
Type: Graph
Number of nodes: 352
Number of edges: 304
Average degree:   1.7273
```

```
print("# nodes:", len(pgGraph.nodes()))
print("# edges:", len(pgGraph.edges()))
```

```
# nodes: 352
# edges: 304
```

```
degree_ls = [val for (node, val) in pgGraph.degree()]
```

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
x = degree_ls
plt.hist(x, bins = 70)
plt.xlabel('degree level')
plt.ylabel('degree freq');
```



top 5 edges with highest weights

```
def weight_dict(G):
    weight_dict = {}
    for (u,v) in G.edges():
        edgeWidth = G[u][v]['numberMentions']
        weight_dict[(u,v)] = edgeWidth
    return weight_dict

fb_edge_dict = weight_dict(pgGraph)
```

```
top_5_edge = sorted(fb_edge_dict.items(), key=lambda x: -x[1])[:5]
```

```
top_5_edge
```

```
[(('photocentrum', 'fpresources'), 11),
 (('', 'fpresources'), 4),
 (('polaroidteam', 'polaroidgirl'), 4),
 (('catherinegrison', 'hopfoot'), 4),
 (('catherinegrison', 'fpresources'), 4)]
```

```
pgGraph['photocentrum']
```

```
AtlasView({'': {'numberMentions': 3}, 'fpresources': {'numberMentions': 11}, 'andrewbuxto
```

Visualize Mention Graph

Double-click (or enter) to edit

```
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
from plotly.graph_objs import *
import plotly.graph_objects as go
init_notebook_mode(connected=True)
```

```
def configure_plotly_browser_state():
  import IPython
  display(IPython.core.display.HTML('''
        <script src="/static/components/requirejs/require.js"></script>
        <script>
          requirejs.config({
            paths: {
              base: '/static/base',
              plotly: 'https://cdn.plot.ly/plotly-latest.min.js?noext',
            },
          });
        </script>
        '''))
```

```
import random
def addRandomPositions(graph):
    posDict = dict((node,(random.gauss(0,10),random.gauss(0,10))) for node in graph.nodes())
    nx.set_node_attributes(graph, name="pos", values=posDict)
```

```
addRandomPositions(pgGraph)
```

```
nx.get_node_attributes(pgGraph, 'pos')['pwcarey']
```

```
(-9.98258135813841, -28.409247751793888)
```

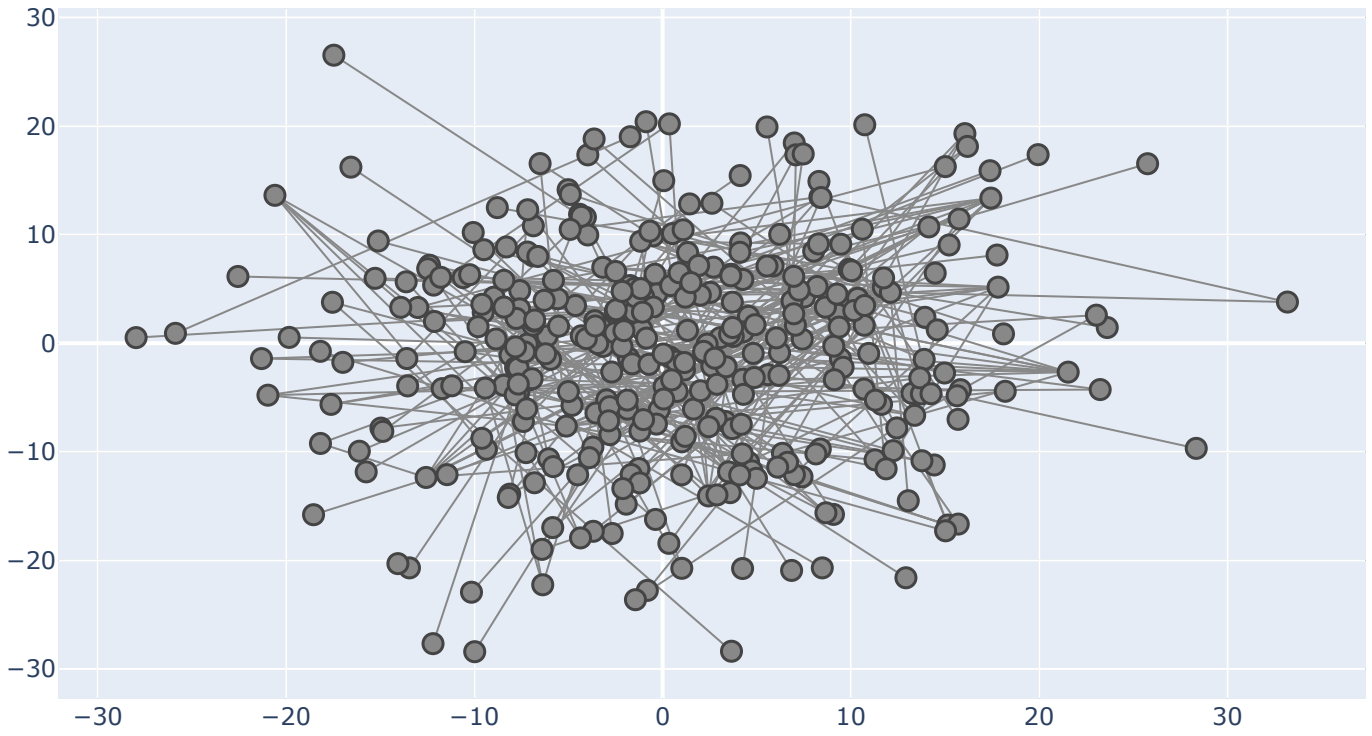
Visualize using Plot.ly scatter plots

```
def plotNetwork(graph):
    scatters=[]

    for (node1, node2) in graph.edges():
        x0, y0 = graph.nodes[node1]['pos']
        x1, y1 = graph.nodes[node2]['pos']
        edgeWidth = graph[node1][node2]['numberMentions']
```

```
        s = Scatter(
                x=[x0, x1],
                y=[y0, y1],
                hoverinfo='none',
                mode='lines',
                line=scatter.Line(width=1 ,color='#888'))
        scatters.append(s)



    for node in graph.nodes():
        xPos, yPos = graph.nodes[node]['pos']
        s = Scatter(
                x=[xPos],
                y=[yPos],
                hoverinfo='none',
                mode='markers',
                marker=dict(
                    color="#888",
                    size=10,
                    line=dict(width=1.5)))
        scatters.append(s)

    layout = Layout(showlegend=False)
    fig = Figure(data=scatters, layout=layout)
    iplot(fig, show_link=False)


pgGraph.nodes['pwcarey']['pos']
```

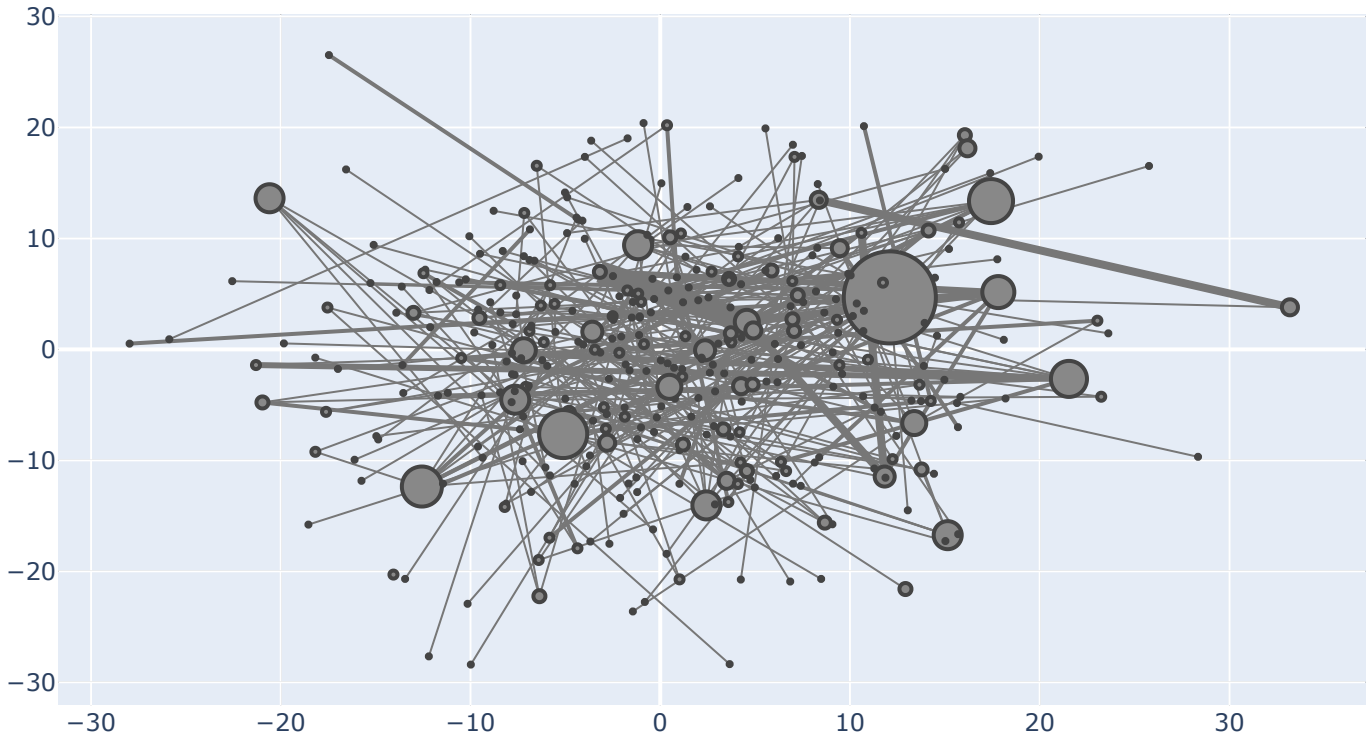
```
(-9.98258135813841, -28.409247751793888)
```

```
configure_plotly_browser_state()
plotNetwork(pgGraph)
```




```
def plotNetworkSize(graph):
    scatters=[]

    for (node1, node2) in graph.edges():
        x0, y0 = graph.nodes[node1]['pos']
        x1, y1 = graph.nodes[node2]['pos']
        edgeWidth = graph[node1][node2]['numberMentions']
        s = Scatter(
                x=[x0, x1],
                y=[y0, y1],
                hoverinfo='text',
                mode='lines',
                line=scatter.Line(width=edgeWidth ,color='#777'))
        scatters.append(s)



    for node in graph.nodes():
        xPos, yPos = graph.nodes[node]['pos']
        s = Scatter(
                x=[xPos],
                y=[yPos],
                hoverinfo='none',
                mode='markers',
                marker=dict(
```

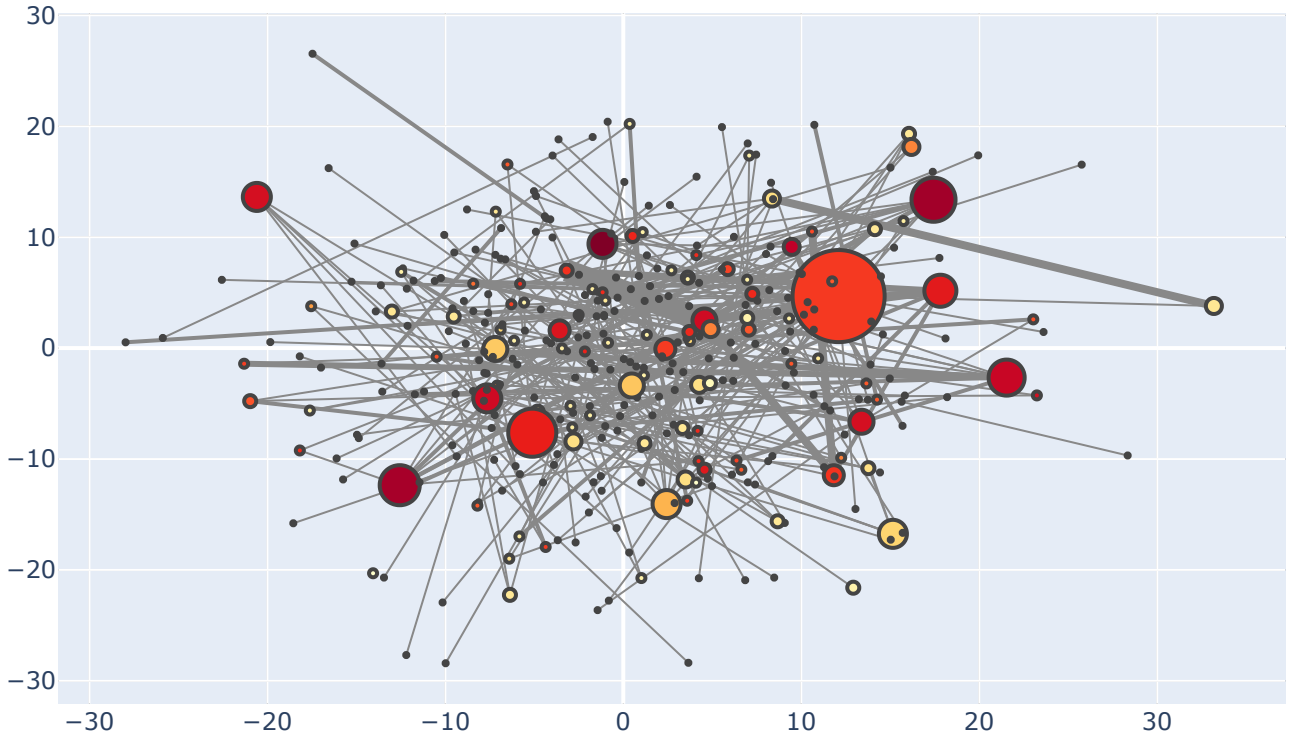
```
                    color="#888",
                    size=nx.degree(graph,node)*2,
                    line=dict(width=2)))
        scatters.append(s)

    layout = Layout(showlegend=False)
    fig = Figure(data=scatters, layout=layout)
    iplot(fig, show_link=False)


configure_plotly_browser_state()
plotNetworkSize(pgGraph)
```




```
import colorlover as cl
from IPython.display import HTML


# map purd color scale to 300 cells
purd = cl.scales['9']['seq']['YlOrRd']
purd300 = cl.interp(purd, 300)
HTML(cl.to_html(purd300))
```

```
def plotNetworkSizeColor(graph):
    closenessCentr = nx.closeness_centrality(pgGraph)
    maxCentr = max(closenessCentr.values())
    minCentr = min(closenessCentr.values())

    scatters=[]

    for (node1, node2) in graph.edges():
        x0, y0 = graph.nodes[node1]['pos']
        x1, y1 = graph.nodes[node2]['pos']
        edgeWidth = graph[node1][node2]['numberMentions']
        s = Scatter(
                x=[x0, x1],
                y=[y0, y1],
                hoverinfo='none',
                mode='lines',
                line=scatter.Line(width=edgeWidth ,color='#888'))
        scatters.append(s)



    for node in graph.nodes():
        nodeCentr = closenessCentr[node]
        nodeColor = int(299*(nodeCentr-minCentr)/(maxCentr-minCentr))
        xPos, yPos = graph.nodes[node]['pos']
        s = Scatter(
                x=[xPos],
                y=[yPos],
                text="User: %s\nCloseness: %.3f" % (node, nodeCentr),
                hoverinfo='text',
                mode='markers',
                marker=dict(
                    color=purd300[nodeColor],
                    size=nx.degree(graph,node)*2,
                    line=dict(width=2)))
        scatters.append(s)

    layout = Layout(showlegend=False)
    fig = Figure(data=scatters, layout=layout)
    iplot(fig, show_link=False)


configure_plotly_browser_state()
```

```
plotNetworkSizeColor(pgGraph)
```



Q3. Content Analysis

a) the most common words in tweets

```
pgTag[['tweet']].head()
```

	tweet
1147	Took a walk at lunch. My lunch-time architectu...
12882	Just added myself to the http://wefollow.com t...
14482	Just added myself to the http://wefollow.com t...
15171	All I can find at the moment about the Noche d...
23958	Finally got around to setting up my #photograp...

```
twts = pgTag['tweet']
```

```
! pip install tweet-preprocessor
import preprocessor as p
```

```
# this is the package that helps you clean tweets text
```

```
Collecting tweet-preprocessor
  Downloading https://files.pythonhosted.org/packages/2a/f8/810ec35c31cca89bc4f1a02c14b04
Building wheels for collected packages: tweet-preprocessor
  Building wheel for tweet-preprocessor (setup.py) ... done
  Created wheel for tweet-preprocessor: filename=tweet_preprocessor-0.5.0-cp36-none-any.w
  Stored in directory: /root/.cache/pip/wheels/1b/27/cc/49938e98a2470802ebdefae9d2b3f5247
Successfully built tweet-preprocessor
Installing collected packages: tweet-preprocessor
Successfully installed tweet-preprocessor-0.5.0
```

```
clean_tweets = twts.apply(lambda x: p.clean(x))
```

```
pgTag['clean_tweets'] = clean_tweets
```

```
pgTag.head()
```

	date	user	tweet	mentioned	clean_tweets
1147	2009-06-11 16:58:53	base10	Took a walk at lunch. My lunch-time architectu...	[]	Took a walk at lunch. My lunch-time architectu...
12882	2009-06-11 17:15:44	replayphotos	Just added myself to the http://wefollow.com t...	[]	Just added myself to the twitter directory under:
14482	2009-06-11 17:17:56	cottontw	Just added myself to the http://wefollow.com t...	[]	Just added myself to the twitter directory under:
					All I can find at

```
# clean txt and tokenize
# reference https://towardsdatascience.com/detecting-bad-customer-reviews-with-nlp-d8b36134dc7
#! pip install nltk
#nltk.download("stopwords")
import re
import nltk
import string
from nltk.corpus import wordnet
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer
nltk.download('stopwords')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')

def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
```

```
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN

def clean_text(text):
    # lower text
    text = text.lower()
    # tokenize text and remove puncutation
    text = [word.strip(string.punctuation) for word in text.split(" ")]
    # remove words that contain numbers
    text = [word for word in text if not any(c.isdigit() for c in word)]
    # remove stop words
    stop_words = set(stopwords.words('english'))
    text = [x for x in text if x not in stop_words]
    # remove empty tokens
    text = [t for t in text if len(t) > 0]
    # pos tag text
    pos_tags = pos_tag(text)
    # lemmatize text
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]
    # remove words with only one letter
    text = [t for t in text if len(t) > 1]
    # join all
    text = " ".join(text)
    return(text)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
```

```
# applying function and clean data
pgTag['clean_tweets'] = pgTag['clean_tweets'].apply(lambda x: clean_text(x))


pgTag.head(10)
```

	date	user	tweet	mentioned	clean_tweets
1147	2009-06-11 16:58:53	base10	Took a walk at lunch. My lunch-time architectu...	[]	take walk lunch lunch-time architectural photo...
12882	2009-06-11 17:15:44	replayphotos	Just added myself to the http://wefollow.com t...	[]	add twitter directory
14482	2009-06-11 17:17:56	cottontw	Just added myself to the http://wefollow.com t...	[]	add twitter directory
15171	2009-06-11 17:18:34	gospain	All I can find at the moment about the Noche d...	[]	find moment noche de fotografia
23958	2009-06-11 17:32:48	hiway	Finally got around to setting up my #photograp...	[]	finally get around set studio basic camera lig...
31519	2009-06-11 17:46:37	pwcarey	@murrayed Thanks again for the book recommend....	[murrayed]	thanks book recommend use form today wedding c...
34693	2009-06-11	karensperling	It's official! New writeup in Lexjet Focus #Ba...	[]	official new writeup lexjet focus

```
import collections
itemAnalysisDf = pgTag[['clean_tweets']]
def getTopK(df, k, value_column='clean_tweets'):
    stop = stopwords.words('english')
    #Add possible Stop Words for Hotel Reviews
    stop.append('twitter')

    counter = Counter()
    for review in df[value_column]:
            counter.update([word.lower()
                            for word
                            in re.findall(r'\w+', review)
                            if word.lower() not in stop and len(word) > 2])
    topk = counter.most_common(k)
    return topk


topk = getTopK(df=itemAnalysisDf, k=50)


topk
```

```
[('photo', 107),
 ('add', 80),
 ('directory', 74),
 ('photography', 68),
 ('day', 51),
 ('via', 47),
 ('shot', 41),
 ('new', 41),
 ('review', 33),
 ('blog', 32),
 ('digital', 30),
 ('get', 29),
 ('great', 28),
 ('check', 26),
 ('canon', 25),
 ('post', 24),
 ('image', 22),
 ('nice', 20),
 ('camera', 19),
 ('photographer', 19),
 ('world', 19),
 ('today', 18),
 ('powershot', 18),
 ('flickr', 18),
 ('like', 17),
 ('take', 16),
 ('use', 16),
 ('update', 16),
 ('free', 16),
 ('look', 15),
 ('beautiful', 15),
 ('nature', 15),
 ('set', 14),
 ('pic', 14),
 ('exceptional', 14),
 ('love', 13),
 ('photograph', 13),
 ('see', 13),
 ('tip', 13),
 ('job', 12),
 ('essay', 12),
 ('high', 12),
 ('shoot', 12),
 ('light', 11),
 ('man', 11),
 ('wonder', 11),
 ('last', 10),
 ('know', 10),
 ('video', 10),
 ('speed', 10)]
```

we can clearly see that the most common words are really close to photo and behavior related to taking photos

c) add hover info to show more details

most common words for each user

```
nltk.download('punkt')
```

```
from nltk.tokenize import sent_tokenize, word_tokenize
tk = pgTag[['user', 'clean_tweets']]
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
tk['tk_words'] = tk['clean_tweets'].apply(word_tokenize)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/user_gu
```

```
tk = tk[['user', 'tk_words']]
tk.head()
```

	user	tk_words
1147	base10	[take, walk, lunch, lunch-time, architectural,...
12882	replayphotos	[add, twitter, directory]
14482	cottontw	[add, twitter, directory]
15171	gospain	[find, moment, noche, de, fotografia]
23958	hiway	[finally, get, around, set, studio, basic, cam...

```
tk['tk_words'].apply(lambda x: [k for k, v in Counter(x).most_common(3)])
```

```
1147              [take, walk, lunch]
12882       [add, twitter, directory]
14482       [add, twitter, directory]
15171           [find, moment, noche]
23958          [finally, get, around]
                      ...
3416456       [useful, action, photo]
3418053     [bruce, percy, portfolio]
3431662          [awesome, set, long]
3431663          [awesome, set, long]
3435787      [gate, interest, series]
Name: tk_words, Length: 658, dtype: object
```

```
tk_new = tk.groupby(by = tk['user'])['tk_words'].apply(list).reset_index(name = 'common_words'
```

```
tk_new.head()
```

	user	common_words
0	02138now	[[fresh, love, farmer, market, harvard, square]]
1	0lli	[[portrait, landscape, dolce, pic, tip]]
2	1001noisycamera	[[add, blog, twitter, directory], [congrats, r...
3	1kc	[[canon, powershot, review, via], [photo, tip,...
4	2live4him	[[free, brush]]

```
import itertools
def merge(text):
    merged = list(itertools.chain.from_iterable(text))
    return merged
tk_new['common_words'] = tk_new['common_words'].apply(merge)

tk_new.head()
```

	user	common_words
0	02138now	[fresh, love, farmer, market, harvard, square]
1	0lli	[portrait, landscape, dolce, pic, tip]
2	1001noisycamera	[add, blog, twitter, directory, congrats, rt, ...
3	1kc	[canon, powershot, review, via, photo, tip, we...
4	2live4him	[free, brush]

```
k_new['top3_words'] = tk_new['common_words'].apply(lambda x: [k for k, v in Counter(x).most_co
k_new.head()
```

	user	common_words	top3_words
0	02138now	[fresh, love, farmer, market, harvard, square]	[fresh, love, farmer]
1	0lli	[portrait, landscape, dolce, pic, tip]	[portrait, landscape, dolce]
2	1001noisycamera	[add, blog, twitter, directory, congrats, rt, ...	[pool, rt, flickr]
3	1kc	[canon, powershot, review, via, photo, tip, we...	[via, canon, powershot]

```
#tk_new['user'][2]
tk_new.set_index('user', inplace=True)

tk_new.head()
```

user	common_words	top3_words
02138now	[fresh, love, farmer, market, harvard, square]	[fresh, love, farmer]
0lli	[portrait, landscape, dolce, pic, tip]	[portrait, landscape, dolce]
1001noisycamera	[add, blog, twitter, directory, congrats, rt, ...	[pool, rt, flickr]
1kc	[canon, powershot, review, via, photo, tip, we...	[via, canon, powershot]

```
def plotNetworkWidthColor_Top3(graph):
    closenessCentr = nx.closeness_centrality(pgGraph)
    maxCentr = max(closenessCentr.values())
    minCentr = min(closenessCentr.values())

    scatters=[]

    for (node1, node2) in graph.edges():
        x0, y0 = graph.nodes[node1]['pos']
        x1, y1 = graph.nodes[node2]['pos']
        edgeWidth = graph[node1][node2]['numberMentions']
        s = Scatter(
                x=[x0, x1],
                y=[y0, y1],
                hoverinfo='none',
                mode='lines',
                line=scatter.Line(width=edgeWidth ,color='#888'))
        scatters.append(s)



    for node in graph.nodes():
        nodeCentr = closenessCentr[node]
        top3 = getTopK(df=itemAnalysisDf, k=3)
        nodeColor = int(299*(nodeCentr-minCentr)/(maxCentr-minCentr))
        xPos, yPos = graph.nodes[node]['pos']
        for index, row in tk_new.iterrows():
            if node == index:
                s = Scatter(
                        x=[xPos],
                        y=[yPos],
                        #text="User: %s\nCloseness: %.3f" % (node, nodeCentr),
                        hoverinfo='text',
                        text = 'Node: %s,Top words: %s' % (node, tk_new.loc[index]['top3_words
                        mode='markers',
                        marker=dict(
                            color=purd300[nodeColor],
                            size=nx.degree(graph,node)*2,
                            line=dict(width=2)))
                scatters.append(s)
```

```
    layout = Layout(showlegend=False)
    fig = Figure(data=scatters, layout=layout)
    iplot(fig, show_link=False)

def applyLayout(graph, layoutFunc):
    posDict = layoutFunc(graph)
    nx.set_node_attributes(graph, name="pos", values=posDict)

pgGraph_top3 = pgGraph.copy()
applyLayout(pgGraph_top3, nx.spring_layout)
configure_plotly_browser_state()
plotNetworkWidthColor_Top3(pgGraph_top3)
```

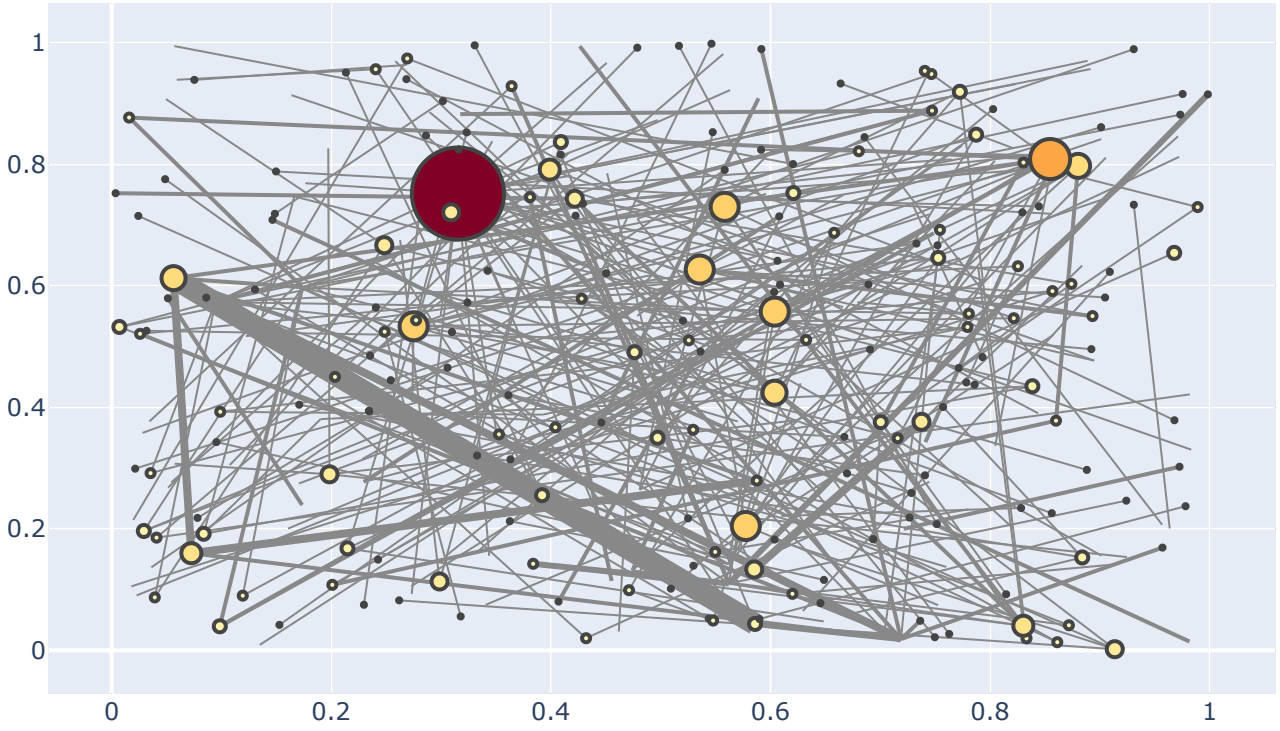
Q4. Centrality Analysis

a) choose two centrality measures

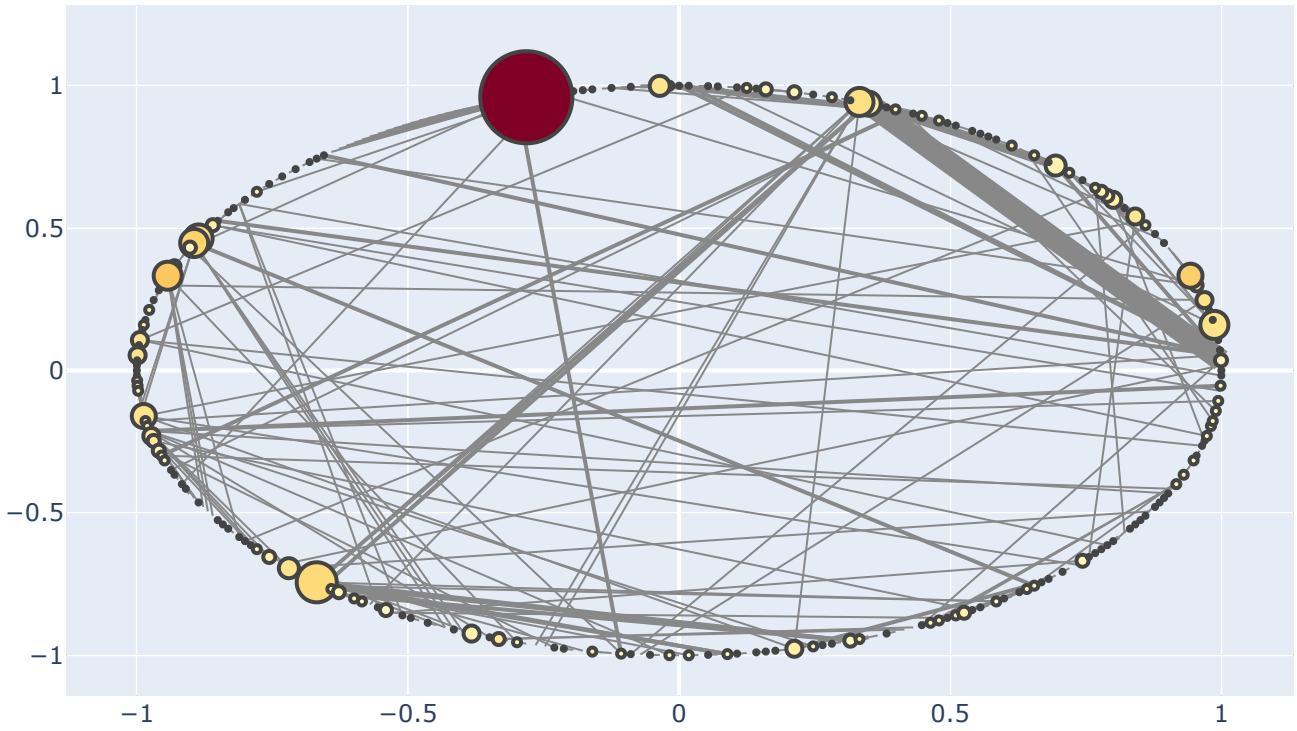
```
degree_centr = nx.degree_centrality(pgGraph)

page_centr = nx.pagerank(pgGraph)
```

b) visualization

```
def plotNetworkWidthColor_Top3(graph, centrality):

    closenessCentr = nx.closeness_centrality(pgGraph)
    maxCentr = max(centrality.values())
    minCentr = min(centrality.values())

    scatters=[]

    for (node1, node2) in graph.edges():
        x0, y0 = graph.nodes[node1]['pos']
        x1, y1 = graph.nodes[node2]['pos']
        edgeWidth = graph[node1][node2]['numberMentions']
        s = Scatter(
                x=[x0, x1],
                y=[y0, y1],
                hoverinfo='none',
                mode='lines',
                line=scatter.Line(width=edgeWidth ,color='#888'))
        scatters.append(s)



    for node in graph.nodes():
        nodeCentr = centrality[node]
        top3 = getTopK(df=itemAnalysisDf, k=3)
        nodeColor = int(299*(nodeCentr-minCentr)/(maxCentr-minCentr))
        xPos, yPos = graph.nodes[node]['pos']
        for index, row in tk_new.iterrows():
            if node == index:
                s = Scatter(
                        x=[xPos],
                        y=[yPos],
                        #text="User: %s\nCloseness: %.3f" % (node, nodeCentr),
                        hoverinfo='text',
                        text = 'Node: %s,Top words: %s, Centr: %s' % (node, tk_new.loc[index][
                        mode='markers',
                        marker=dict(
                            color=purd300[nodeColor],
                            size=nx.degree(graph,node)*2,
                            line=dict(width=2)))
                scatters.append(s)

    layout = Layout(showlegend=False)
    fig = Figure(data=scatters, layout=layout)
    iplot(fig, show_link=False)


pgGraph_top3 = pgGraph.copy()
applyLayout(pgGraph_top3, nx.random_layout)
configure_plotly_browser_state()
plotNetworkWidthColor_Top3(pgGraph_top3, degree_centr)
```




```
pgGraph_top3 = pgGraph.copy()
applyLayout(pgGraph_top3, nx.circular_layout)
configure_plotly_browser_state()
plotNetworkWidthColor_Top3(pgGraph_top3, page_centr)
```



c)

1. They are really similar, especially for top results
2. if we want to just have a general picture of the centrality. Degree might work. However, betweeness and page rank both have other prespectives of viewing centrality. betweeness shows how important one node between groups. page rank explicitly shows authority of this node.

for this hashtag #photography i chose, probably page rank makes more sense.

Q5 Cliques analysis

a) 1. number of maximal cliques

```
num_maxCliques = nx.graph_number_of_cliques(pgGraph)
num_maxCliques
```

```
264
```

```
fc = nx.find_cliques(pgGraph)
```

```
fc
```

```
<generator object find_cliques at 0x7efc04a11048>
```

2. size of the largest maximal clique containing each given node

```
cliques_number = nx.node_clique_number(pgGraph)
cliques_number
sorted(cliques_number.items(), key=lambda x: -x[1])[:10]
```

```
[('photocentrum', 3),
 ('', 3),
 ('havepack', 3),
 ('joanna_haugen', 3),
 ('lindseygirl', 3),
 ('gullivergo', 3),
 ('catherinegrison', 3),
 ('2live4him', 3),
 ('str8photography', 3),
 ('rweiher', 3)]
```

```
# some other findings
# find cliques
gclique = list(nx.find_cliques(pgGraph))
print (gclique)
# find connected components
comps = nx.connected_components(pgGraph)
#print (len(comps))
#print (comps[0])
#print (comps[1])
print (comps)
```

```
ging'], ['sonofgroucho', 'robinmwood'], ['sonofgroucho', 'grunberghausvt'], ['sonofgroucho
```

```
testGraph = nx.pagerank(pgGraph)
print(testGraph[max(testGraph, key=testGraph.get)])
print(max(testGraph.values()))
```

```
0.030435536044451005
0.030435536044451005
```

b) insights on connectivity

so through my hashtag, there are a lot of cliques but most of them are really small, which means they are not a large group or several large groups. as below:

```
# Size of the largest maximal clique containing each given node
pd.DataFrame(list(nx.node_clique_number(pgGraph).items()),
             columns=['user', 'Maximal clique']).sort_values('Maximal clique',ascending=0)
```

	user	Maximal clique
303	yipeedoodah	3
282	rhysb123	3
79	redmanandy	3
37	joanna_haugen	3
38	lindseygirl	3
...	...	...
127	thecruiseguy	2
126	say_my_name	2
125	njray	2
334	designerm	1
341	darladeleon	1

352 rows × 2 columns

```
# Maximal cliques for each node
pd.DataFrame(list(nx.number_of_cliques(pgGraph).items()),
             columns=['user', 'Maximal clique']).sort_values('Maximal clique',ascending=0)
```

	user	Maximal clique
104	jacksoncj1	23
188	hashphoto	12
224	wearephotogs)	10
223	amazingpics	9
64	rweiher	8
...	...	...
130	chacal_lachaise	1
129	sherri_meyer	1
128	thagoodson	1
127	thecruiseguy	1
351	angelsamudre	1

352 rows × 2 columns

according to wiki, **The clique problem arises in the following real-world setting. Consider a social network, wh the graph's vertices represent people, and the graph's edges represent mutual acquaintance. Then a clique**

represents a subset of people who all know each other, and algorithms for finding cliques can be used to

therefore, from the analysis, we know the user "jacksoncj1" might be the center of several cliques and we can s other potential friends inside cliques as well.

```
pd.options.display.max_colwidth = 1000

# figure our who th is jacksoncj1
pgTag[pgTag['user'] == 'jacksoncj1']
```

	date	user	tweet	mentioned	clean_tweets
979278	2009-06-12 14:55:18	jacksoncj1	We are now left with no alternative other than to proceed with the wage reduction" said Sulzberger #BostonGlobe http://bit.ly/BoMVi\n2009-06-12 14:55:18\tjimcim\tOriginal composition by Jim Cim ♫ Jim Cim – Everything I Do - http://blip.fm/~7y3ue via @addthis\n2009-06-12 14:55:18\tjmal18\tawesome ad for Louis Vutton using US astronauts http://bit.ly/K6StK\n2009-06-12 14:55:18\tkamytran\thttp://bit.ly/19twoL , http://bit.ly/14Cmjc awww! these videos are so cuteee (L)\n2009-06-12 14:55:18\tkhopjackpaper\tCool now we can tell if the 'Celeb' we're following is actually that celebrity. Thank you Twitter. http://twitter.com/help/verified\n2009-06-12 14:55:18\tlasersmom\tRT @ronjons Developing our women's swimwear for next season. Can't decide, gold or silver metallic accent? You pick, most votes wins.\n2009-06-12 14:55:18\tmarykenah\thttp://twitpic.com/773xc	[addthis, ronjons, djednice, cspanwj, midnightpr, funkybrownchick, millz1, msdrama, teeteebee, dj2tone, collegiate84, alisha14209, teebiscuit, rhys_isterix, urbanfly, willfrancis, joe, andrew_davis, terziev, ten_thirteen, njray, say_my_name, thecruiseguy]	leave alternative proceed wage reduction say sulzberger jimcim original composition jim cim jim cim everything via awesome ad louis vutton use astronauts kamytran awww video cuteee khopjackpaper cool tell celeb we're follow actually celebrity thank twitter lasersmom rt develop women's swimwear next season can't decide gold silver metallic accent pick vote win marykenah real housewife nj irl mmasjer hot free video update daily nice cute girl asian european msdrama rt mspbjnews medtronic boston sci torax med get acid reflux-fighting tech onenewsnowcom prejean homosexual comment cost crown former miss california usa carrie

I literally check her tweeter account, from her intro:

"I'm all about sewing and crafts when I'm not doing marcom for a Big 12 b-school in TX. Sewing, NPR, PBS, saltwater fishing."

C.J. Jackson (@jacksoncj1) is a social media influencer. but not a photograher as i thought before