

# COSC 2671 Social Media and Network Analytics

## Lab Week 9

### Social Network Analysis

Learning outcomes:

- Learn how to interact with Reedit via its API, and saving and loading of graphs
- Compute centrality and other SNA measures of a graph in networkx

Requirements:

- A PC with Internet connection and Python installed (preference is version 3.X, but 2.7 is also okay, but unfortunately we don't have resources to provide support if there are version incompatibility issues)

Resources:

- This tute-lab worksheet (available on Canvas)
- Associated code in tuteLab6Code.zip (available on Canvas)

Python Packages Required:

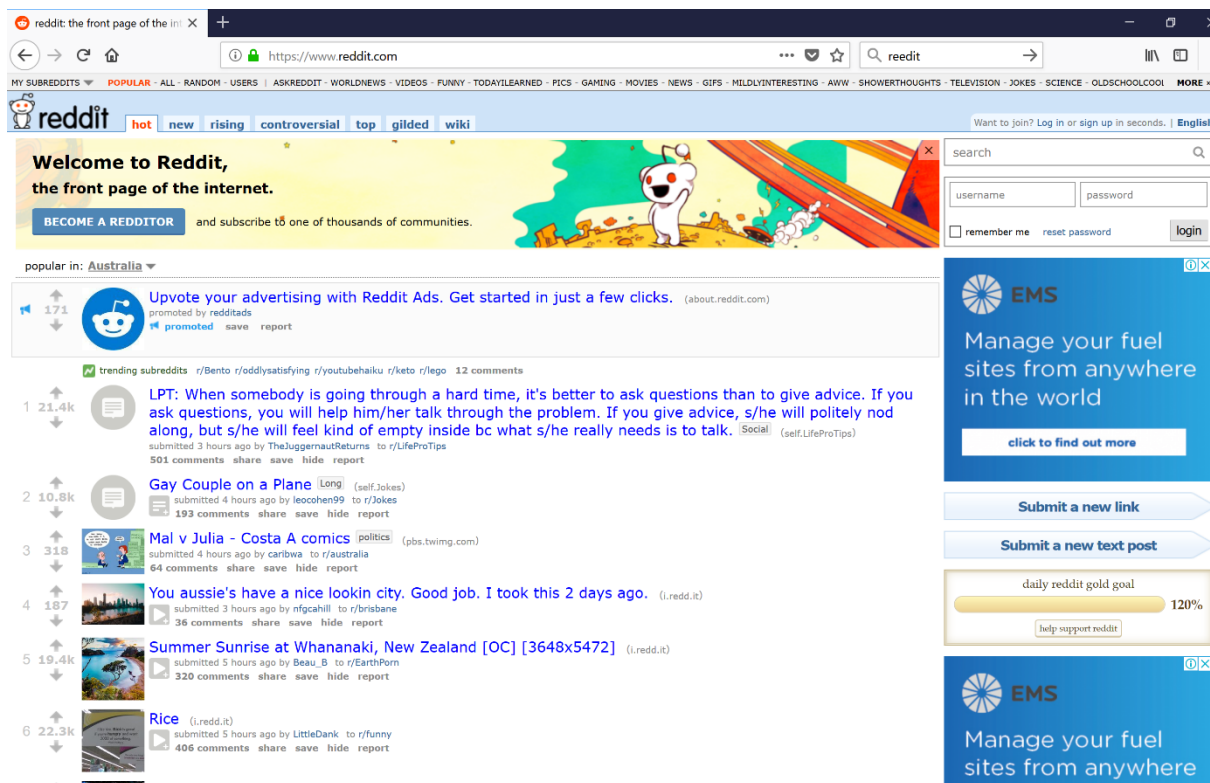
- networkx
- praw (package to access reedit API)

## Lab Introduction

In this lab, we learn how to construct a graph of Reddit discussions and study how to implement and interpret some SNA measures. We will learn about Reddit, learn how to extract information via its API, construct a graph, save it to file and compute some SNA measures on this graph to study its properties.

## Reddit

Reedit claims to be the “front page of the Internet”, and essentially is a crowd-sourced news aggregator, commenting and discussion website, and to date has a few hundred million users. Similar to Twitter, it is a great source of information for the latest topics of interest, as well as the discussion of different special interest groups. Check out the website <https://www.reddit.com/> (below is a screen shot of Reddit page from Aus afternoon of 16/04/18).



## Praw and Constructing the Reddit Reply Graph

In this lab, we will use the Python package *praw* to access Reddit and collect data from it. We will also practice saving the graph, so we can avoid continually calling the Reddit API and can load it for reuse. There is also a very large data dump of all Reddit posts in 2015, which is a great source of data if we don't need the most up to date and researching properties in Reddit and online communities. If interested, go to [https://www.reddit.com/r/datasets/comments/6lqbsd/downloading\\_reddit/](https://www.reddit.com/r/datasets/comments/6lqbsd/downloading_reddit/), which might be useful for your assignment 2.

We first access the Reddit API, and construct a reply graph, where the nodes are Redditors (registered Reddit users) and directed edges represent a Redditor replying to/commenting another Redditor's submission or comments. The reply graph is a good way to understand users, their communities and their behaviours via social network analysis. We will save the constructed graph to a file. Note an alternative is to save the data from Reddit directly as a json, csv or alternative file structure and build graph from that data (similar to what we previously did with Twitter data) – since its similar to previous labs, please explore this option if you are interested (might come handy for your assignment 2).

Going back to the lab, please install *praw* first;

```
$pip install praw
```

Have a read of the official documentation for *praw*: <https://praw.readthedocs.io/en/latest/>

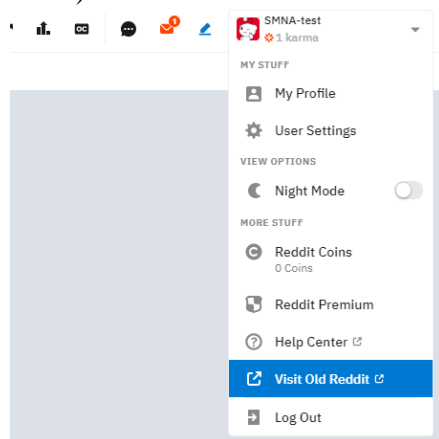
The terminology in Reddit can be confusing. A *subreddit* is essentially a channel or forum, usually about a specific topic, e.g., politics, NBA and funny. Within each subreddit are a number of *submissions*, which can be considered as either discussion threads, interesting articles and links. For each submission, there may be more one or more *comments* (replies essentially).

Downloading all subreddits will take a while and likely to violate Reddit's community etiquette. Hence we focus on one we are quite familiar with and everyone in this class should know about – Python. In Reddit this is written as r/Python (<https://www.reddit.com/r/Python/>).

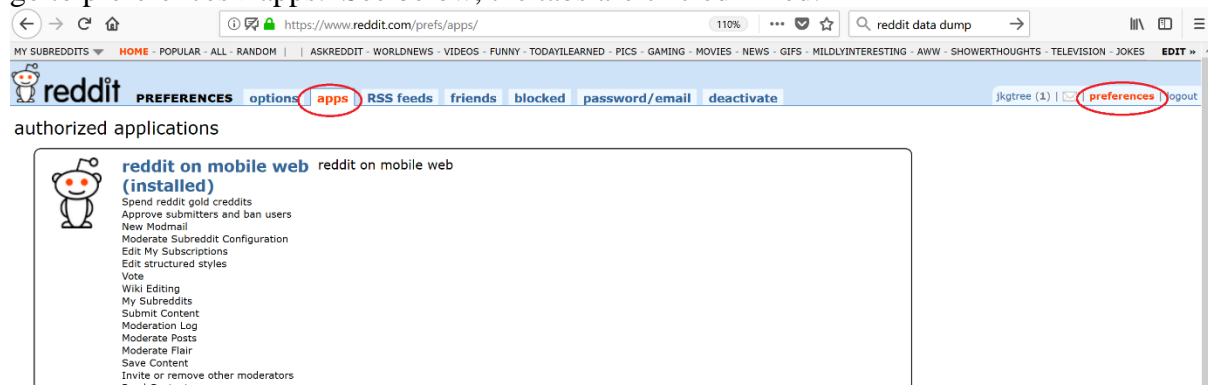
## Registering Reddit Account

Before we can download some submissions of r/Python, we first need to setup the Reddit accounts. We don't need an account to access Reddit, but it allows greater functionality, so I suggest you to do so. First, register a Reddit account (even if you have one, you can use this newly registered one as a sandbox account). <https://www.reddit.com/register/>

Once registered, go to (you could explore the new site if interested, but the functionality is same)



go to preferences->apps. See below, the tabs are circled in red:



Register an app (see below for what to fill in, and type in the name of the app (in name field) of your own choosing, e.g., ABC app.

## create application

Please [read the API usage guidelines](#) before creating your application. After creating, you will be required to [register](#) for production API use.

**name**

☐ web app A web based application

☐ installed app An app intended for installation, such as on a mobile phone

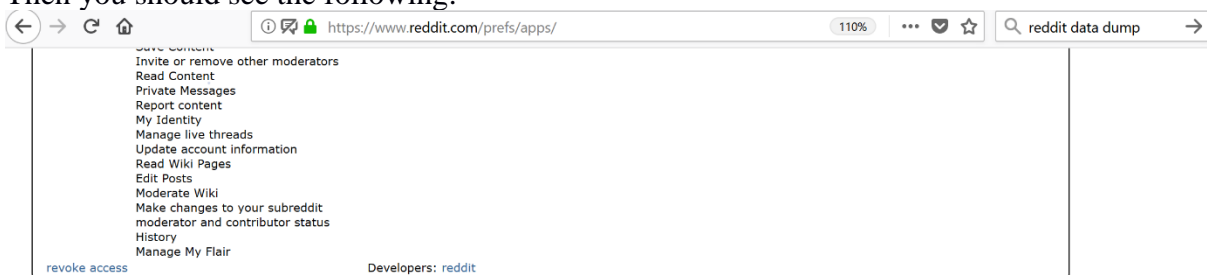
☒ script Script for personal use. Will only have access to the developers accounts

**description**


**about url**

**redirect uri**

Then you should see the following:



## developed applications

 **Social Media** For university course.

personal use script  
GEAECzM58rcwm0 ← **user id**

secret [blacked out] ← **client secret**

name Social Media

description For university course.

about url

redirect uri

update app delete app

developers jkgtree (that's you!) remove

add developer:

← **user name**

From this page, obtain the client id, client secret (blacked out in my screenshot) and user name. Password is the one you registered for your account/user name.

Write down the following information:

- client ID
- client secret
- password
- user name

Open up **redditClient.py**. This is similar to the Twitter client we have been using, but for Reddit. Enter the above information and assign the appropriate variables. Once we done so, then we can reuse this Python script to connect to the Reddit API.

## Constructing Reply Graph

Open `redditGraph.ipynb`, which is in the code provided to you. We have written the initial code to connect to Reddit and obtain some submissions from `r/Python`. As usual, please study it.

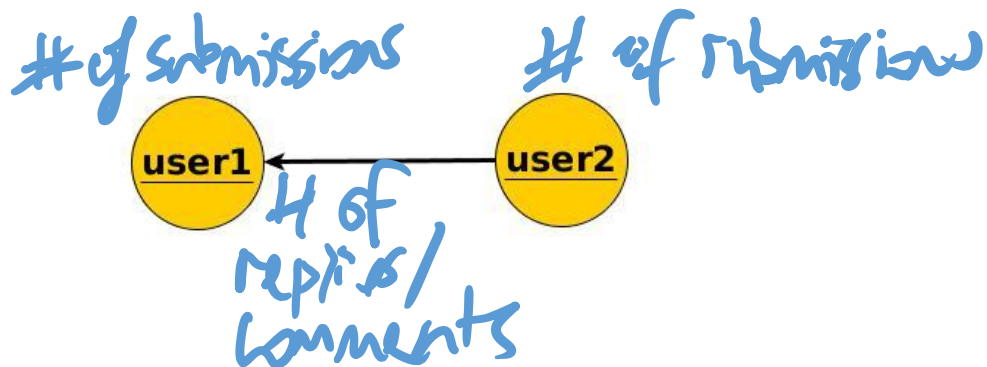
There are several ways to collect submissions to a subreddit, including `hot()`, `new()` and `stream()`. We will demonstrate with `hot()`, which retrieves the hottest submissions, but `stream()` is particularly interesting, as it opens up a stream to continually collect submissions.

One particular part of the code (in cell 3) that does need some explanation is the following:

```
submission.comments.replace_more(limit=None)
for comment in submission.comments.list():
    ...
```

The way that praw and Reddit works is that the comments are returned, up to 32 at a time. The `submission.comments.list()` will keep retrieving these until there are none left to retrieve.

The code constructs the following type of graph:



### Exercise:

Examine the code. Can you figure out how it does so? Break it up into parts. First figure out which part constructs the nodes and node attribute. Then figure out which part extracts the edges, representing replies, and their attributes. Please ensure you work this out, as you can reuse similar approach to construct a graph from any social media and network.

Now the final part of the code is to save it. Networkx has many ways to output a graph, see <https://networkx.github.io/documentation/networkx-1.10/reference/readwrite.html>. We'll output as graphml, which allows us both to visualise and store it. Note, you can use any of the other ways, or even generate your own csv file.

Examine [https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.readwrite.graphml.write\\_graphml.html](https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.readwrite.graphml.write_graphml.html) to figure out how to output file. The name of the filename is written in `replyFilename` variable.

## Compute SNA measures

Final part of the lab is to compute some SNA measures and visualise them in gephi. Open **redditAnalysis.ipynb**. See the code that reloads the file you written out earlier? Now, we want to compute the centrality measures we learnt in lectures, namely degree, eigenvector and katz centrality. Conveniently they are available in networkx, see <https://networkx.github.io/documentation/networkx-1.10/reference/algorithms.centrality.html>

### Exercise:

To get started, we have computed the degree centrality of all the nodes in the reply graph. Compute the eigenvector and katz centrality, using `eigenvector_centrality_numpy()` and `katz_centrality_numpy()`. Note use the numpy versions of these centrality calculations, as the other version uses an eigensolver that can sometimes causes errors.

Consider the following code. What does it do?

```
plt.subplot(1,3,1)
plt.hist(list(lDegCentrality.values()))
plt.title('Degree')
plt.xlabel('centrality')
```

Recall subplot which is used to display plots side by side? There is an example to plot the degree centrality. Repeat for the other two centralities. If you don't remember how subplot works, examine [https://matplotlib.org/api/as\\_gen/matplotlib.pyplot.subplot.html](https://matplotlib.org/api/as_gen/matplotlib.pyplot.subplot.html) and [https://matplotlib.org/examples/pylab\\_examples/subplot\\_demo.html](https://matplotlib.org/examples/pylab_examples/subplot_demo.html) (the latter is an example).

What does the histograms indicate? Why do you think they have slightly different distributions?

Histograms are good for summarising descriptive statistics, but we may also want to see which individual nodes causes the centrality to be different. What we are going to do is to modify our input graphml file to add in extra information to the nodes – the centrality values, then load the new graph in gephi.

First, consider the following code. What does it do? Hint: `replyGraph.node` accesses the nodes, and `replyGraph.node[nodeId]` accesses the dictionary of attributes, values associated with node 'nodeId'.

```
for nodeId, cent in lEigenvectorCentrality.items():
    replyGraph.node[nodeId]['eigen'] = float(cent)
```

Repeat for katz centrality and add both to the code.

Afterwards, we write out the new graph (with extra attribute information). Use `write_graphml()` again, but output file to 'mod-' + `args.replyFilename`.

After this, open gephi and load this file in. Use the degree, eigenvector and katz centrality to colour and resize the nodes. What do you see? Does it help you answer question about why the distributions of the three centralities differ?

**Extra analysis** (if you don't complete in lab, please finish it afterwards):

Finally, we want to compute the following SNA measures:

- global clustering coefficient or transitivity of a graph
- number of strongly connected components
- number of weakly connected components
- number of bridges (for the last one, you'll need to convert the directed graph to an undirected one. This can be done by `replyGraph.to_undirected()`)

Hint: read the networkx reference and find the functions that does the above. Then call these functions with our graph to obtain the values of these measures. What do they say about the reply graph? What type of analysis do you think you could do using this?