Twitter API and Python Package Tweepy

School of Information Studies
Syracuse University

Twitter API

Gathering Tweets:

- See what is trending
- See who is responding
- Discover frequencies

Two widely used Python packages

- Twitter
- Tweepy

Using Tweepy

Gaining in user base

May be installed through conda

May require Python 3.5

Try installing using instructions in Toolbox

Using Tweepy

Do not have to have Twitter account

You may use tweets already downloaded

Twitter and Twitter API

Microblogging service

280-character messages

Sign up for an account at Twitter.com

Twitter page can show you trends

- Tweets of users
- Tweets addressed to you
- Allow you to compose a tweet

Twitter Statistics

Over 600 million signed up

Over 328 million active users each month

Social networking site

“Interest” network

User accounts

- People
- Companies
- Other entities

Tweets Received

Has metadata fields

- User who sent it
- Whether it was retweeted
- How many times retweeted
- Entities and places

Twitter is changing what is in the 280 characters

- Moving @mentions and URLs to metadata

Timeline used to organize

Tweets as Streams

Streams

- Collections of tweets
- Firehose of all tweets—high volumes accessed by paid permission
- Public APIs (us)—sample from 1% of tweets
- Collect tweets on particular topic or user—may get many of them

Search API

- Search for keywords over a time frame

Streaming API

- Real-time access to tweets—no history

Open Authorization

Allows third-party developer

Allows end user to access server resources

Don’t have to give their passwords

Facebook API

School of Information Studies
Syracuse University

Facebook

Online social network

- Started as service for college students
- Now largest networking service worldwide

Privacy controls

- Allows user to keep activities private
- Decide with whom they want to share

Self-proclaimed social graph

- Based on friends relationship
- Relationships are reciprocal
- “Likes” feature

Graph API

REST API

- Requests encoded as http URLs

Uses Facebook Query Language (FQL)

- SQL-like syntax

Page where you submit requests

- Resulting JSON results

Supports Open Authorization

Graph API Results

Results—JSON objects

Similar notation to JSON in Python

- Using ‘{ }’ for dictionaries
- Using ‘[ ]’ for lists

Use of edge names

- Fields associated with your account
- Connection fields

Graph API Terms

Feed—things user sees on their wall

Posts—any content a user posts

Statuses—status updates

Installing Facebook

Most widely used package

- Facebook-sdk

To install in python:

- Open an Anaconda prompt
- pip install facebook-sdk

Get Access Token (good for about an hour)

https://developers.facebook.com/tools/explorer

Facebook SDK

Facebook API

Basically 4 functions:

- get_object(self, id, **args)
- get_objects(self, id, **args)
- get_connections(self, id, connection_name, **args)
- request(self, path, args=None, post_args=None)
- Example usage: request("search",{'q':'social web', 'type':'page'})

Facebook in Python

```
>>> import facebook
>>> import json
>>>ACCESS_TOKEN = '<put access token here>'
>>>fb = facebook.GraphAPI(ACCESS_TOKEN)
```

My Facebook information

>>>myFB = fb.get_object(‘me’)

(then print it out)

>>>print(json.dumps(myFB, indent =2)

(my Friends)

>>>myFriends = fb.get_connections(‘me’, ‘friends’)

>>>print(json.dumps(myFriends, indent =2)

Using Search Function

```
>>> DJ = fb.request("search", {'q' : 'jurafsky', 'type':'page'})
>>>type(DJ)
<class 'dict'>
>>>DJ.keys()
dict_keys(['data', 'paging'])
```

Printing Data

```
>>>Djdata = DJ[‘data’]
>>>type(Djdata)
<class ‘list’>
>>>print(json.dumps(Djdata, indent=2))
```

Now we could dive deeper into Dan Jurafsky