Data Mining - Foursquare II

Bruno Gonçalves
www.bgoncalves.com

Tips

- Users can leave tips in venues at any time (without checking in)
- (Reduced) Tips for a venue can be accessed using .venues.tips(venue_id)
- Limited to a maximum of 500 per call, defined with the "count" parameter. Get further tips with "offset" parameter (same as for friends).
- Full Tip objets can be obtained with .tips(tip_id)
- Contain (Reduced) User object and are public, providing an easy way to connect users with venues.

Challenge

- Obtain the complete list of user_id who left a tip at

43695300f964a5208c291fe3

Challenge

- Obtain the complete list of user_id who left a tip at

```
import foursquare
from foursquare_accounts import accounts

app = accounts["tutorial"]

client = foursquare.Foursquare(client_id = app["client_id"],
                               client_secret = app["client_secret"])

client.set_access_token(app["access_token"])

venue_id = "43695300f964a5208c291fe3"

tips = client.venues.tips(venue_id)
tips_list = tips["tips"]["items"]
tip_count = tips["tips"]["count"]

while len(tips_list) < tip_count:
    tips = client.venues.tips(venue_id, {"offset": len(tips_list)})
    tips_list += tips["tips"]["items"]

print len(tips_list), tip_count

for tip in tips_list:
    print tip["user"]["id"]
```

Checkins

- Checkins are the *Raison d'être* of Foursquare.

- They connect Users with Venues providing valuable temporal and demographic information.

- .checkins(checkin_id) Returns the Checkin object

- .users.checkins(user_id) Returns the list of ***Public*** checkins for User user_id or all checkins if user_id is friends of the user using the application.

Challenge

- Obtain the Checkin object for checkin_id

5089b44319a9974111a6c882

- and get the name of the user who performed it.

Challenge

- Obtain the Checkin object for checkin_id

5089b44319a9974111a6c882

- and get

```
import foursquare
from foursquare_accounts import accounts

app = accounts["tutorial"]

client = foursquare.Foursquare(client_id = app["client_id"],
                               client_secret = app["client_secret"])

client.set_access_token(app["access_token"])

checkin_id = "5089b44319a9974111a6c882"

checkin = client.checkins(checkin_id)
user_name = checkin["checkin"]["user"]["firstName"]

print checkin_id, "was made by", user_name
```

Checkins

- Users have the option of sharing their checkins through Twitter and Facebook, making them publicly accessible
- The status text is shared along with the URL of the web version of the checkin.
- To allow Twitter and Facebook friends to access the checkin, a special "access_token", called a signature, is added to the checkin URL.
- Each signature is valid for just a single checkin and it allows anyone to access the respective checkin

Checkins

- Signed checkin urls are of the form:

  https://foursquare.com/<user>/checkin/<checkin_id>?s=<signature>&ref=tw

- For example:

  https://foursquare.com/tyayayayaa/checkin/5304b652498e734439d8711f?s=ScMqmpSLg1buhGXQicDJS4A_FVY&ref=tw

- corresponds to user tyayayayaa performing checkin 5304b652498e734439d8711f and has signature ScMqmpSLg1buhGXQicDJS4A_FVY

- .checkins(checkin_id, {“signature”: signature}) can be used to query the API using the signature key to access a private checkin

Challenge

- Use urllib2 and posixpath to parse the url

https://foursquare.com/tyayayayaa/checkin/5304b652498e734439d8711f?s=ScMqmpSLg1buhGXQicDJS4A_FVY&ref=tw

- and extract the users screen_name, checkin_id and signature.

- With this information use the API to get the users User object and the respective Checkin object

Challenge

```
import urllib2
import posixpath
import foursquare
from foursquare_accounts import accounts

app = accounts["tutorial"]

client = foursquare.Foursquare(client_id = app["client_id"],
                               client_secret = app["client_secret"])

client.set_access_token(app["access_token"])

url = "https://foursquare.com/tyayayayaa/checkin/5304b652498e734439d8711f?
s=ScMqmpSLg1buhGXQicDJS4A_FVY&ref=tw"

parsed_url = urllib2.urlparse.urlparse(url)
checkin_id = posixpath.basename(parsed_url.path)
query = urllib2.urlparse.parse_qs(parsed_url.query)
screen_name = parsed_url.path.split('/')[1]

signature = query["s"][0]
source = query["ref"]

user_search = client.users.search({"twitter": screen_name})
user_id = user_search["results"][0]["id"]

user = client.users(user_id)
checkin = client.checkins(checkin_id, {"signature": signature})
```

Challenge

- Using this approach we can easily obtain a lot of information on checkins, users and venues. However, it can also lead to many requests to the API, causing problems with the API rate limits.

- Another alternative is to scrape the webpage directly using BeautifulSoup to extract the information directly from the HTML code.

- Write a script that uses requests to obtain the HTML code for url

  http://4sq.com/18aGENW

- and extract the checkin_id and venue_id. Don't forget to change your User-agent header.

```
from BeautifulSoup import BeautifulSoup
import urllib2
import requests
import posixpath

url = "http://4sq.com/18aGENW"

headers = {"User-agent" : "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:25.0) Gecko/20100101
Firefox/25.0"}

request = requests.get(url, headers=headers)

final_url = request.url
parsed = urllib2.urlparse.urlparse(final_url)
query = parsed.query
signature = urllib2.urlparse.parse_qs(query)["s"][0]

checkin_id = posixpath.basename(parsed.path)

soup = BeautifulSoup(request.text)

checkInHeader = soup.div(attrs={"class":"leftCheckInHeader"})[0]
username = checkInHeader.find("span", attrs={"class":"userName"})
user = username.a["href"][1:]
screen_name = username.text

venue = [a["href"] for a in checkInHeader.findAll("a")
         if a["href"][:3] == '/v/'][0]

print "Checkin %s is for User \"%s\" with Name \"%s\" checking in at %s"\
         % (checkin_id, user, screen_name, posixpath.basename(venue))
```