Python for Data Science

Twitter Data Analysis: Access the Twitter API

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Understand what are the Twitter API
- Create a Twitter application
- Download authentication credentials
- Save them to disk for later usage

The Twitter API

Twitter offers a Web Application Programming Interface (API) to access all tweets on their website.

Send queries via web to Twitter's servers for tweets based on:

users, locations, trends, search terms, hashtags.

It requires authentication.

Create a Twitter account

Skip this step if you have already a personal Twitter account

- Open your browser and navigate to: https://twitter.com/signup
- Follow the instructions to create a new account
- No need to add interests or follow any account
- Make sure to click on the email verification link

Create a Twitter App

- Login on https://twitter.com with your credentials
- Navigate to: https://apps.twitter.com/
- Click on the Create New App button
- Choose python4ds_yourname as name, write a description, set http://google.com as Website, leave Callback empty
- Confirm the creation of the app
- Depending on your country, you might need to verify your phone first

Save the credentials to disk

- Click on the "Keys and Access Tokens" tab
- Open the Jupyter Notebook about Twitter

Python for Data Science

twitter: Authenticate with the Twitter API in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Install the twitter Python package
- Authenticate successfully with the Twitter API

Python for Data Science

twitter:
Explore Twitter trends

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Request Twitter trends using Python
- Explore local trends

Twitter Trends

Trending topics are hashtags (#example) or words that are currently popular.

Categorized by country or city i.e.:

- Worldwide
- USA
- San Diego

Worldwide Trends

Current worldwide trends:

```
trends = twitter_api.trends.place(_id=1)
```

Extract only names:

```
[t["name"] for t in trends[0]["trends"]]
```

Check on https://twitter.com this is the same list displayed in the left column

Local trends

Twitter identifies locations with an integer number named "Where On Earth ID", 1 for "Worldwide", lookup your country or town at:

http://woeid.rosselliot.co.nz/

Execute the trends API call in the previous slide replacing 1 with this number.

Retrieving Trends

Execute: **Example 2. Retrieving trends**

Display raw response

The API response is in JSON format, see **Example 3. Displaying API responses as pretty-printed JSON**

JSON is a data format used to transfer data on the web.

It is roughly equivalent to nested Python dictionaries and lists.

Compute the intersection of trends

Check: **Example 4. Computing the intersection of two sets of trends**

The Python set data structure provides a intersection() method to find common trends in different locations

Python for Data Science

twitter:
Explore Twitter search

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Request Twitter search results using Python
- Use a Python for loop and list lookup to filter duplicates
- Inspect a JSON data structure

Twitter search

Access tweets about a topic with:

```
twitter_api.search.tweets(q="Topic")
```

q can be any string, for example one of the trending topics from the previous video

Each Tweet Has Many Metadata

```
['id_str', 'possibly_sensitive',
'created_at', 'source',
'is_quote_status',
'in_reply_to_status_id_str', 'entities',
'in_reply_to_user_id',
'in_reply_to_user_id_str', 'text',
'coordinates', 'retweeted_status',
'extended_entities',
'in_reply_to_screen_name',
'in_reply_to_status_id', 'retweeted',
'favorite_count', 'retweet_count',
'favorited', 'contributors', 'user',
'place', 'lang', 'id', 'geo',
'truncated', 'metadata']
```

Extract Text, Screen name and Hashtags

We can extract the most interesting fields, see **Example 6. Extracting text, screen names, and hashtags from tweets**

Python for Data Science

twitter:
Create Frequency Distributions

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Create frequency distributions with collections.Counter
- Space padding in string formatting
- Sort a list of tuples

Create Frequency Distributions

```
from collections import Counter
```

It gets lists and counts how many times each item is repeated.

Its most_common() method returns the sorted counts.

See **Example 7. Creating a basic frequency distribution from the words in tweets**

But: **HARD TO READ!**

Advanced String Formatting in Python

```
print("{:20} | {:>6}".format(k,v))
```

"{:20}" format(s) pad the string to 20 spaces

"{:^20}" format(s) centered

"{:>20}" format(s) right-aligned

More at https://pyformat.info

Print tables for Frequency Counts

Create nicely formatted tables with print and string formatting.

Check the prettyprint_counts function in

Example 8. Create a prettyprint function to display tuples in a nice tabular format

Find the most popular retweets

We can extract:

- number of retweets
- screen name
- text

use the sorted function to sort by the first value in the tuple, using reverse=True for descending order.

Print Tables of Tweets

We can create a more advanced version of the prettyprint_counts function that automatically adapts the output to varying tweet length.

see **Example 9. Finding the most popular retweets**