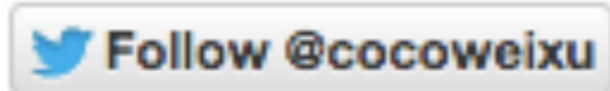


# Social Media & Text Analysis

## A Beginner's Guide



**Ohio State University**

**Instructor: Wei Xu**


**Website: [socialmedia-class.org](http://socialmedia-class.org)**

# About Me

<http://web.cse.ohio-state.edu/~weixu/>

**Wei Xu**

 Follow @cocoweixu

[phonetic pronunciation: way  shoo 

Assistant Professor

Department of Computer Science and Engineering

The Ohio State University

 weixu@cse.ohio-state.edu

 495 Dreese Lab (2015 Neil Ave, Columbus, OH 43210)

My research lies at the intersections of **machine learning**, **natural language processing**, and **social media**. I focus on designing algorithms for learning semantics from large data for natural language understanding, and generation in particular with stylistic variations. I recently received the NSF CRII Award, Criteo Faculty Research Award, CrowdFlower AI for Everyone Award, Best Paper Award at COLING'18, as well as research funds from DARPA. Previously, I was a postdoctoral researcher at the University of Pennsylvania. I received my PhD in Computer Science from New York University where I was a MacCracken Fellow, MS and BS from Tsinghua University.

I am an area chair for EMNLP 2018 (social media area), COLING 2018 (semantics area), EMNLP 2016 (generation area), a workshop chair for ACL 2017, and the publicity chair for NAACL 2016 and 2018. I also created the **Twitter API tutorial** and a new course on **Social Media and Text Analytics**.



# Course Website

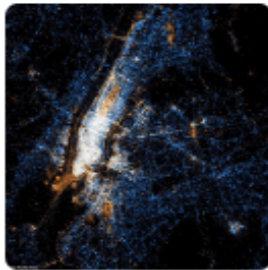
[socialmedia-class.org](http://socialmedia-class.org)

Social Media & Text Analytics

Syllabus

Twitter API Tutorial

Homework ▾



*A visualization showing the location of Twitter messages (blue) and Flickr photos (orange) in New York City by Eric Fischer*

Social media provides a massive amount of valuable information and shows us how language is actually used by lots of people. This course will give an overview of prominent research findings on language use in social media. The course will also cover several machine learning algorithms and the core natural language processing techniques for obtaining and processing Twitter data.

## Instructor

[Wei Xu](#) is an assistant professor in the Department of Computer Science and Engineering at the Ohio State University. Her research interests lie at the intersection of machine learning, natural language processing, and social media. She holds a PhD in Computer Science from New York University. Prior to joining OSU, she was a postdoc at the University of Pennsylvania. She is organizing the ACL/COLING [Workshop on Noisy User-generated Text](#), serving as a workshop co-chair for ACL 2017, an area chair for EMNLP 2016 and the publicity chair for NAACL 2016.

## Time/Place new

**Fall 2016, CSE 5539-0010 The Ohio State University**  
**Cockins Hall Room 218 | Wednesday 2:20PM – 4:10PM**  
dual-listed undergraduate and graduate course

## Prerequisites

In order to succeed in this course, you should know basic probability and statistics, such as the chain rule of probability and Bayes' rule. On the programming side, all projects will be in Python. You should understand basic computer science concepts (like recursion), basic data structures (trees, graphs), and basic algorithms (search, sorting, etc).

## Course Readings

[Various academic papers](#)

## Previous Offerings

Summer 2016, [The North American Summer School on Logic, Language, and Information \(NASSLLI\)](#)  
Teaching evaluation was 5.72 out of 6 at NASSLLI; average across all instructors was 5.23.  
Summer 2015, University of Pennsylvania (where this course was first designed and taught)

# Social Media & Text Analysis

part 1 - Intro to Python



**Ohio State University**

**Instructor: Wei Xu**

**Website: [socialmedia-class.org](http://socialmedia-class.org)**



python



# Why Python?

- Python is an object-oriented and high level programming language (first released in 1991).
- Very beginner-friendly!
  - shorter code needed for the same task.
- Very powerful!
  - many well-maintained libraries (e.g. numpy, Matplotlib, Scikit-learn, PyTorch, TensorFlow,)
  - a popular programming language in AI and machine learning research

# Simplicity of Python

- Create a list of integers:

- Python

```
nums = [45, 23, 51, 32, 5]
```

- Java, in contrast:

```
List<Integer> nums =  
    Arrays.asList(new Integer[] {45, 23, 51, 32, 5});
```

# Simplicity of Python

- Create a list of integers, and print them out:

- Python

```
nums = [45, 23, 51, 32, 5]
for idx, num in enumerate(nums):
    print idx, num
```

- Java, in contrast:

```
List<Integer> nums =
    Arrays.asList(new Integer[] {45, 23, 51, 32, 5});

for (int i = 0; i < nums.size(); i++) {
    String number = nums.get(i);
    System.out.println(i + " " + number);
}
```



# Simplicity of Python

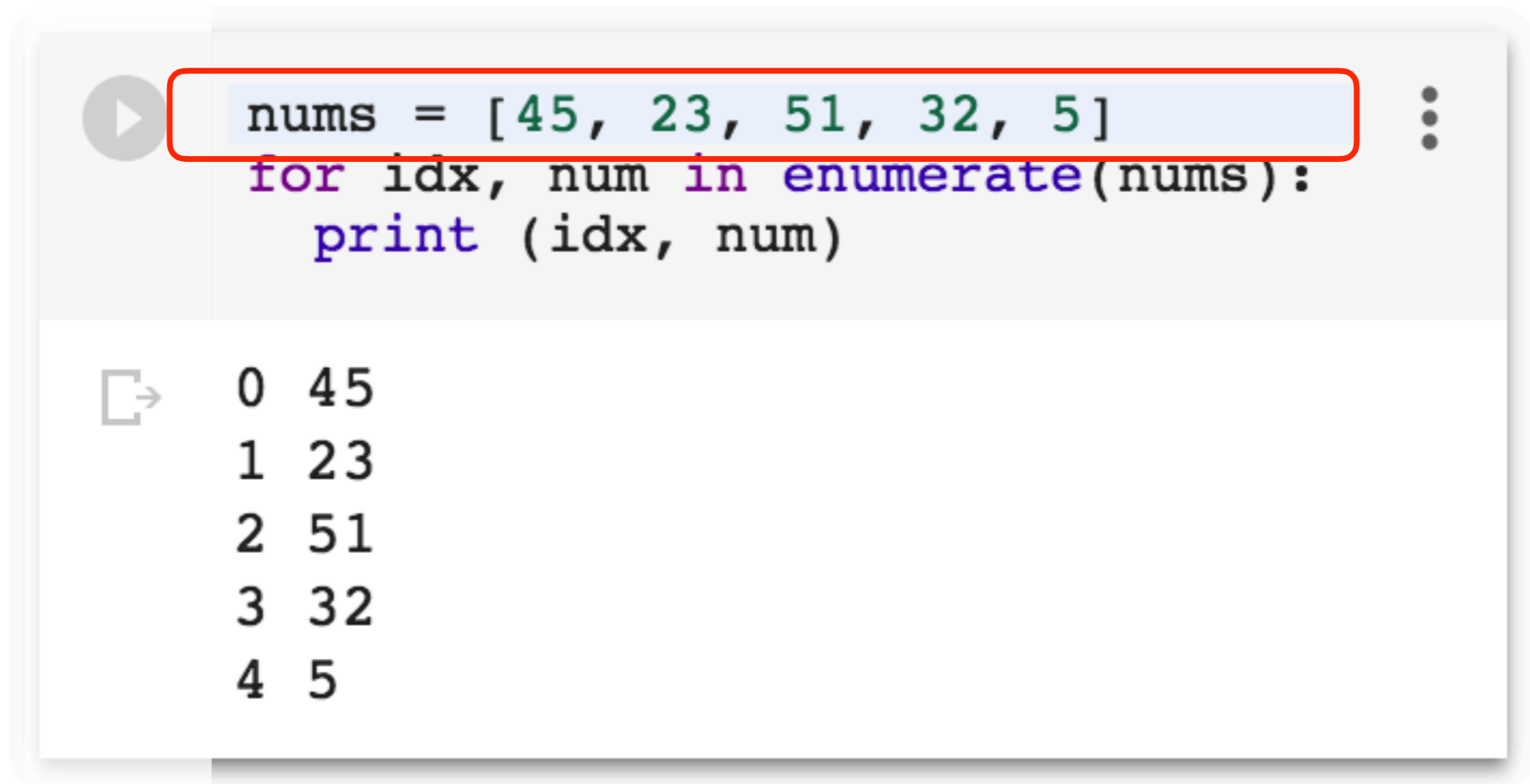
- Create a list of integers, and print them out:

```
▶ nums = [45, 23, 51, 32, 5]
  for idx, num in enumerate(nums):
    print (idx, num)
```

↳ 0 45  
1 23  
2 51 ← output  
3 32  
4 5

# Simplicity of Python

- Create a list of integers, and print them out:



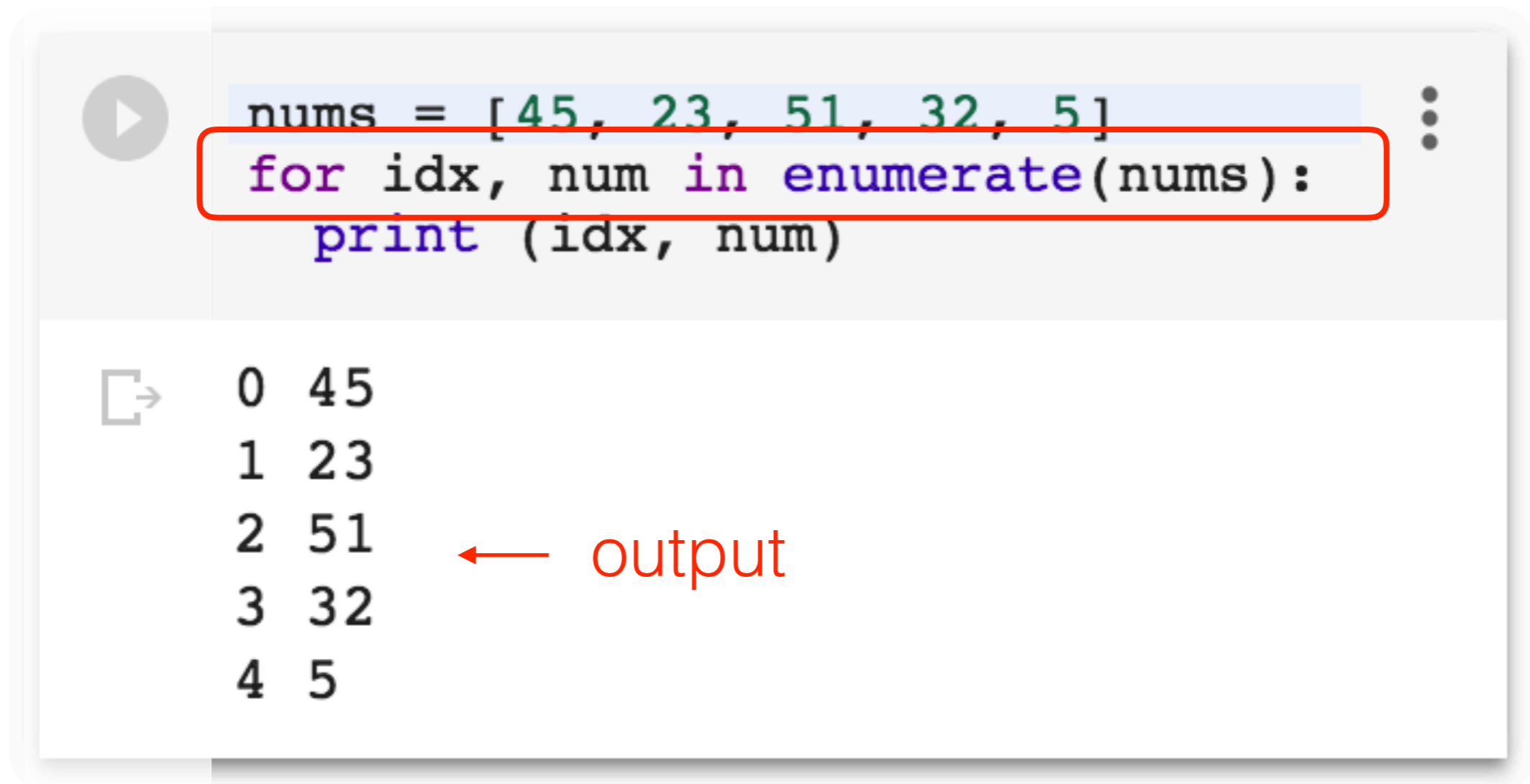
```
▶ nums = [45, 23, 51, 32, 5]
  for idx, num in enumerate(nums):
    print (idx, num)
```

☞ 0 45  
1 23  
2 51  
3 32  
4 5

Lists are declared using [item1, item2, item3, ... ]

# Simplicity of Python

- Create a list of integers, and print them out:



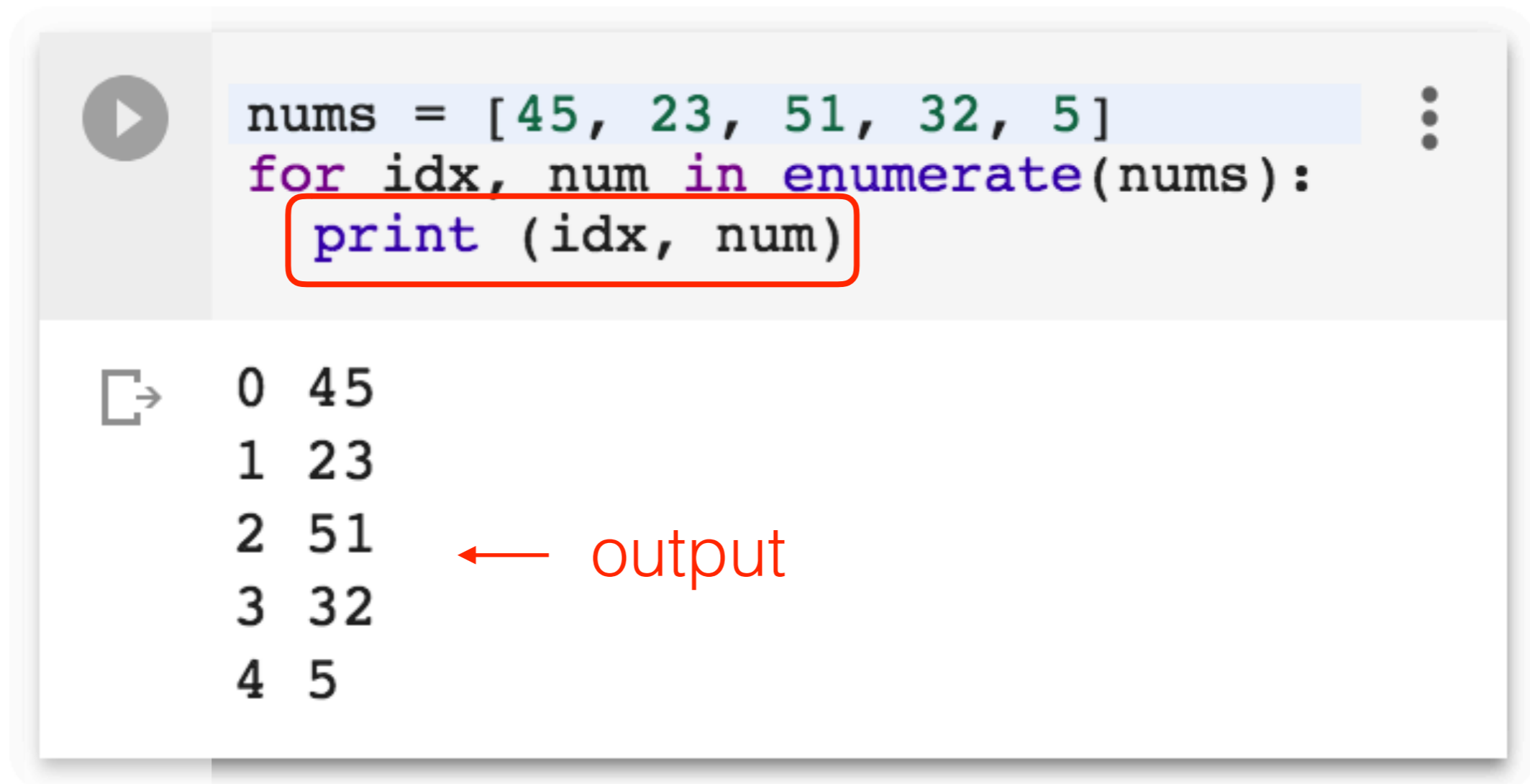
```
nums = [45, 23, 51, 32, 5]
for idx, num in enumerate(nums):
    print (idx, num)
```

0 45  
1 23  
2 51 ← output  
3 32  
4 5

Iterates over each item in the list, yielding (index, value) tuples

# Simplicity of Python

- Create a list of integers, and print them out:



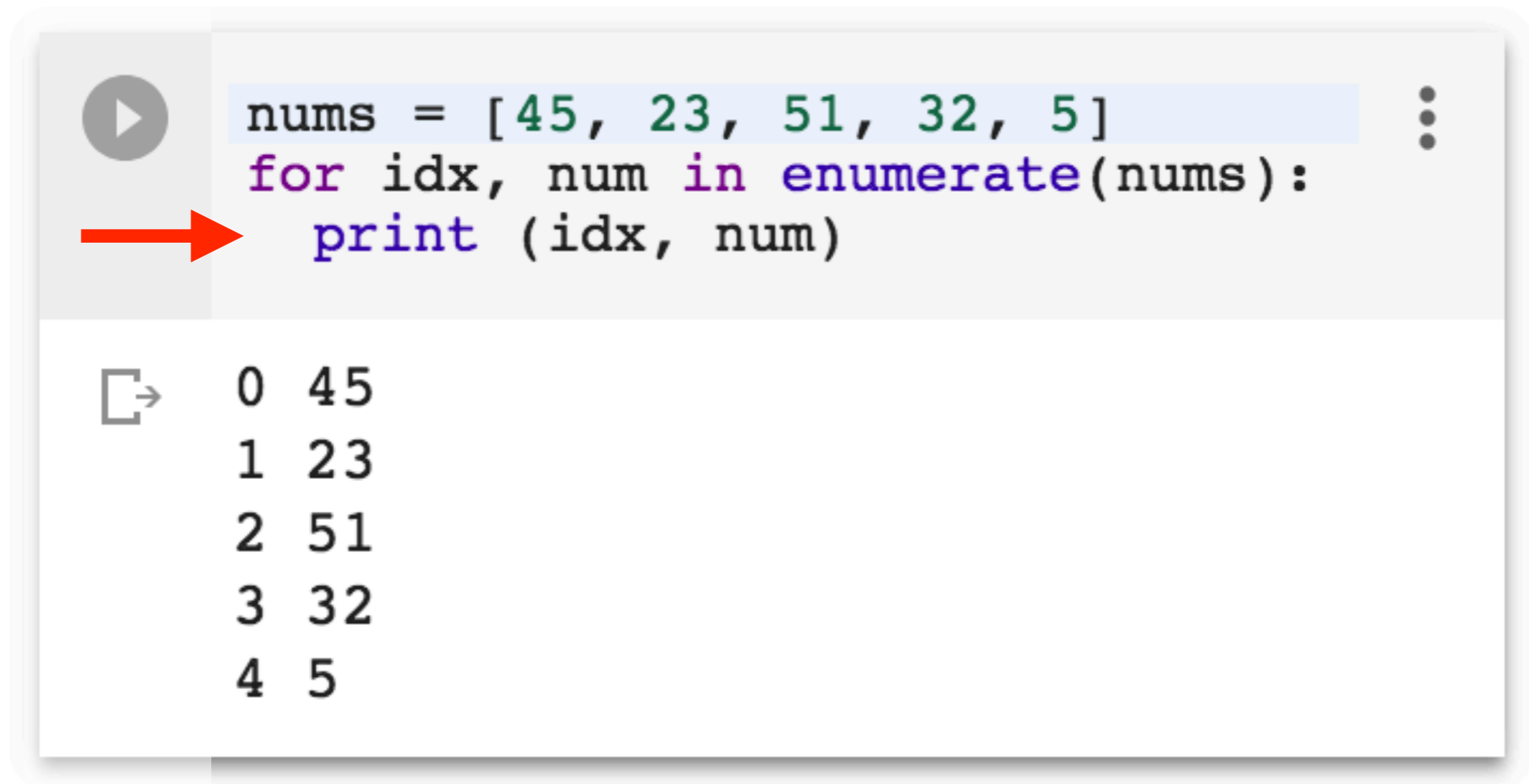
```
nums = [45, 23, 51, 32, 5]
for idx, num in enumerate(nums):
    print (idx, num)
```

0 45  
1 23  
2 51 ← output  
3 32  
4 5

simple and very useful print() function

# Simplicity of Python

- Create a list of integers, and print them out:



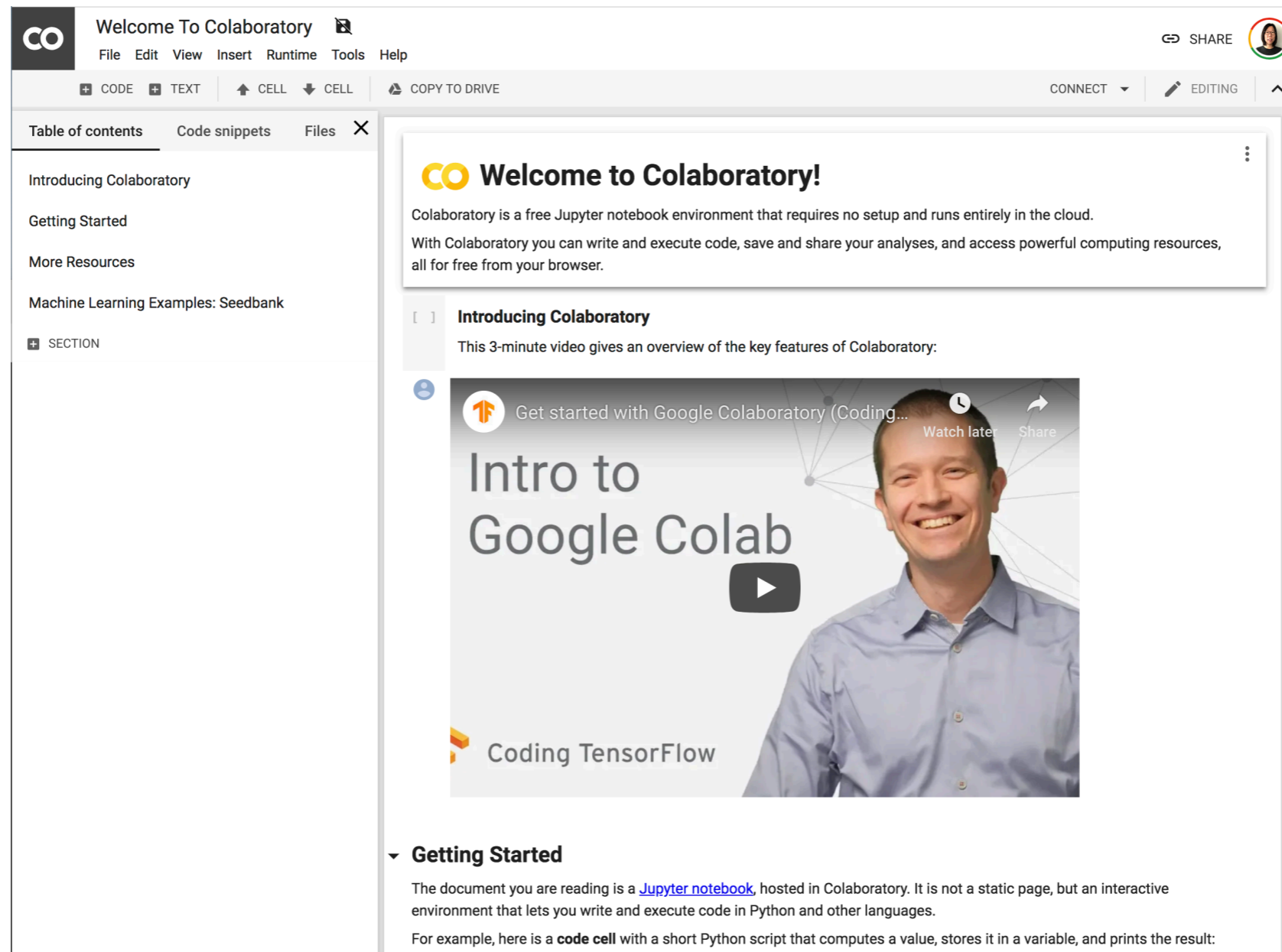
```
▶ nums = [45, 23, 51, 32, 5]
  for idx, num in enumerate(nums):
    print (idx, num)
```

☞ 0 45  
1 23  
2 51  
3 32  
4 5

Indentation is very important in Python!

# Try it Out!

- We will use **Google's Colab** programming environment:



The screenshot displays the Google Colaboratory web interface. At the top, there is a navigation bar with the Colab logo, the text "Welcome To Colaboratory", and a menu with options: File, Edit, View, Insert, Runtime, Tools, and Help. On the right side of the top bar, there are "SHARE" and "CONNECT" buttons, along with a user profile icon. Below the top bar is a secondary navigation bar with buttons for "+ CODE", "+ TEXT", "↑ CELL", "↓ CELL", and "COPY TO DRIVE". On the far right of this bar are "CONNECT", "EDITING", and an upward arrow icon.

On the left side, there is a "Table of contents" sidebar with the following items: "Introducing Colaboratory", "Getting Started", "More Resources", and "Machine Learning Examples: Seedbank". Below these items is a "SECTION" header with a plus icon.

The main content area features a large "Welcome to Colaboratory!" message with the following text: "Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. With Colaboratory you can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser." Below this message is a section titled "Introducing Colaboratory" with the text: "This 3-minute video gives an overview of the key features of Colaboratory:". A video player is embedded, showing a video titled "Intro to Google Colab" by Coding TensorFlow. The video player includes a play button, a "Watch later" button, and a "Share" button. Below the video player is a section titled "Getting Started" with the following text: "The document you are reading is a [Jupyter notebook](#), hosted in Colaboratory. It is not a static page, but an interactive environment that lets you write and execute code in Python and other languages. For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:"

# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])         # The 5th char (index starts from 0)

l = len(sent1)           # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()   # Split a string by space
print (tokens)

print (len(tokens))     # The length (in number of tokens/words)
```

```
↳ Hello world!
○
There are 12 charasters.
['Hello', 'world!']
2
```

# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])         # The 5th char (index starts from 0)

l = len(sent1)           # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()   # Split a string by space
print (tokens)

print (len(tokens))     # The length (in number of tokens/words)
```

↳ Hello world!  
o  
There are 12 charasters. ← output  
['Hello', 'world!']  
2

Characters in a string can be assessed using the [] syntax.



# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])         # The 5th char (index starts from 0)

l = len(sent1)           # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()   # Split a string by space
print (tokens)

print (len(tokens))     # The length (in number of tokens/words)
```

```
↳ Hello world!
o
There are 12 charasters.
['Hello', 'world!']
2
```

The `len(string)` function returns the length of a string.

# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])         # The 5th char (index starts from 0)

l = len(sent1)           # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()   # Split a string by space
print (tokens)

print (len(tokens))     # The length (in number of tokens/words)
```

```
↳ Hello world!
○
There are 12 charasters. ← output
['Hello', 'world!']
2
```

The `str()` function converts values to a string data type.

# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])         # The 5th char (index starts from 0)

l = len(sent1)           # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()   # Split a string by space
print (tokens)

print (len(tokens))     # The length (in number of tokens/words)
```

```
↳ Hello world!
○
There are 12 charasters.
['Hello', 'world!']
2
```

← output

The + operator can concatenate two strings.

# Basic String Operations

- A Code Sample:

```
▶ sent1 = "Hello world!" | # A comment.
print (sent1)             # Another one.

print (sent1[4])          # The 5th char (index starts from 0)

l = len(sent1)            # The length (in number of characters)
print ("There are " + str(l) + " charasters.")

tokens = sent1.split()    # Split a string by space
print (tokens)

print (len(tokens))      # The length (in number of tokens/words)
```

```
↳ Hello world!
o
There are 12 charasters.
['Hello', 'world!'] ← output
2
```

The `split()` function returns a list of substrings.

# More Resources

- Python :
  - Google's class: <https://developers.google.com/edu/python/>
  - Christophe Morisset's notebook: [https://github.com/Morisset/Python-lectures-Notebooks/blob/master/Notebooks/intro\\_Python.pdf](https://github.com/Morisset/Python-lectures-Notebooks/blob/master/Notebooks/intro_Python.pdf)
  - and many others ...
- Got a Error Messages or questions?
  - Search on Google
  - StackOverflow

# Social Media & Text Analysis

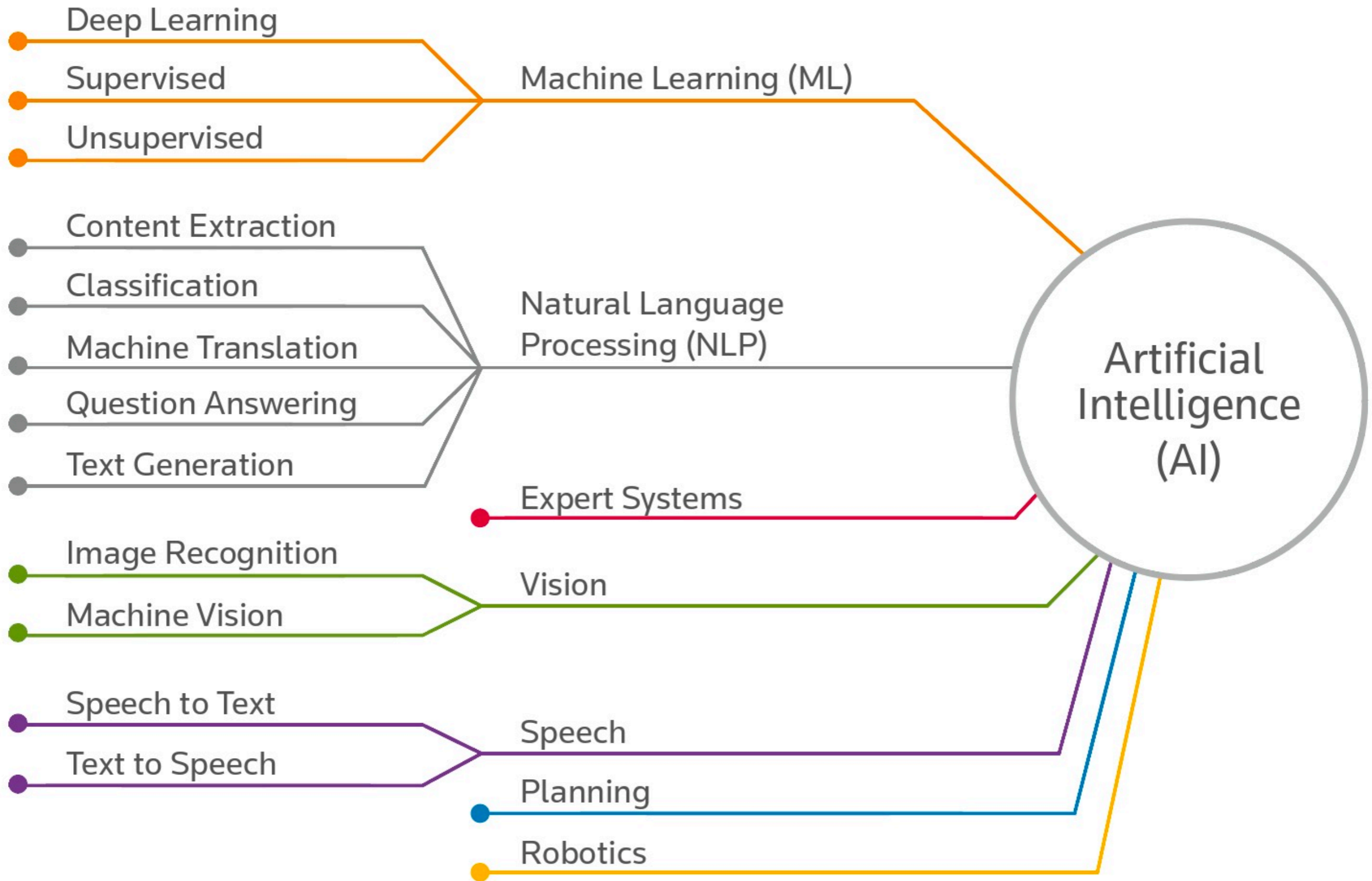
part 2 - Intro to NLP



**Ohio State University**

**Instructor: Wei Xu**

**Website: [socialmedia-class.org](http://socialmedia-class.org)**



# Basic Text Processing

- Tokenization:

```
import nltk
nltk.download('punkt')
sentence = "At eight o'clock in the morning, Arthur didn't feel well."
tokens = nltk.word_tokenize(sentence)
print (tokens)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
['At', 'eight', "o'clock", 'in', 'the', 'morning', ',', 'Arthur', 'did', "n't", 'feel', 'well', '.']
```

breaking text up into words, phrases, symbols, or other meaningful elements called tokens.



# Basic Text Processing

- Tokenization:

```
▶ import nltk

nltk.download('punkt')

sentence = "At eight o'clock in the morning, Arthur didn't feel well."

tokens = nltk.word_tokenize(sentence)
print (tokens)
```

```
↳ [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
['At', 'eight', "o'clock", 'in', 'the', 'morning', ',', 'Arthur', '']
```

To start using Python NLTK (Natural Language Toolkit) library.

# Basic Text Processing

- Tokenization:

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

sentence = "At eight o'clock in the morning, Arthur didn't feel well."

tokens = nltk.word_tokenize(sentence)
print (tokens)
```

```
↳ [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
['At', 'eight', "o'clock", 'in', 'the', 'morning', ',', 'Arthur', ',']
```

Load in a pre-trained tokenizer for English named “Punkt”.

# Basic Text Processing

- Tokenization:

```
▶ import nltk
nltk.download('punkt')
sentence = "At eight o'clock in the morning, Arthur didn't feel well."
tokens = nltk.word_tokenize(sentence)
print(tokens)
```

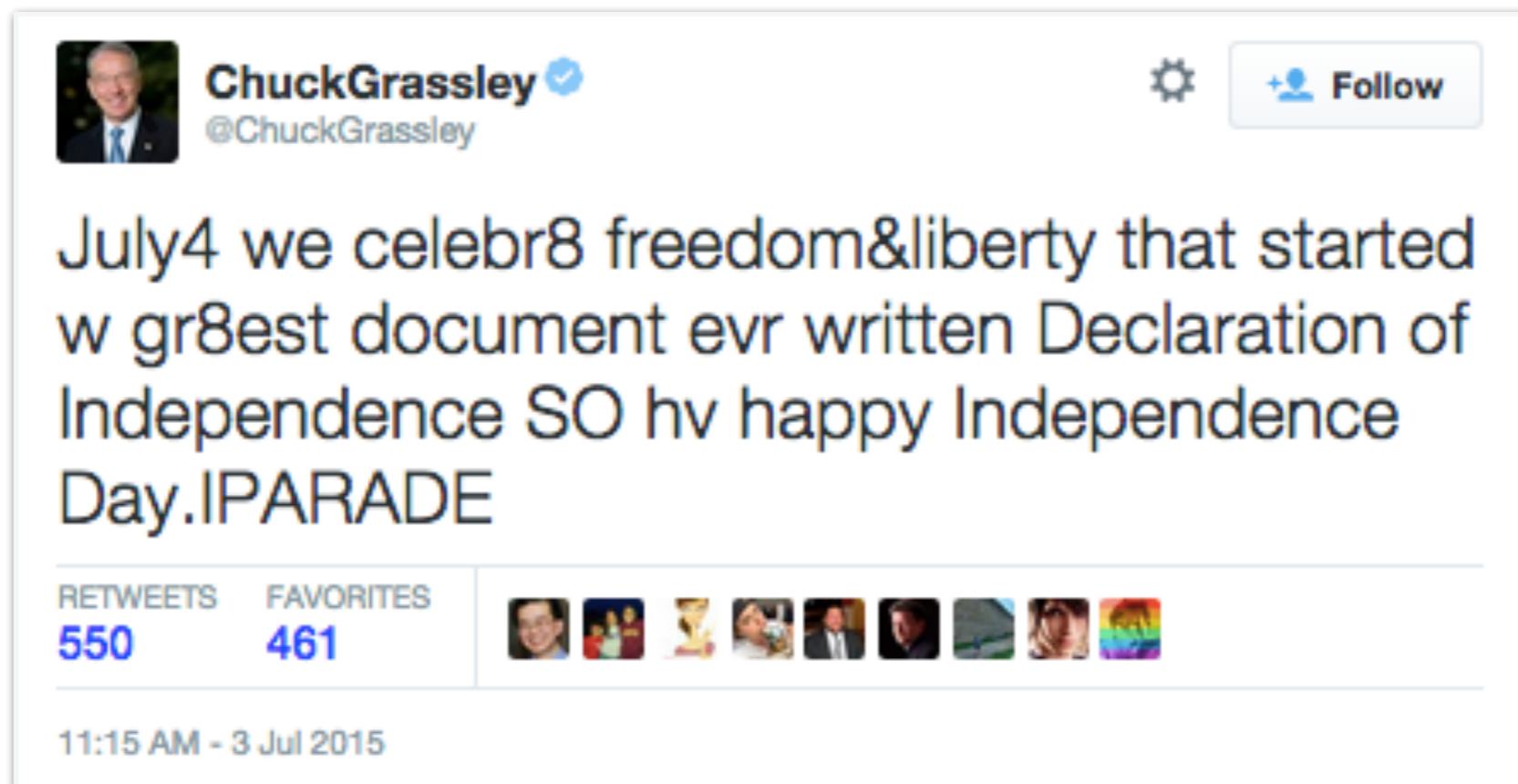
```
↳ [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
['At', 'eight', "o'clock", 'in', 'the', 'morning', ',', 'Arthur', '']
```

Calling the `word_tokenize()` function in the `nltk` module.

# Try it Out!

- We will use **Google's Colab** programming environment:

What if we try to tokenize some tweets?



# Try it Out!

- We will use **Google's Colab** programming environment:

What if we try to tokenize some tweets?

```
[20] tweet1 = "@someone did you check out this #superawesome!! <3"  
      print (nltk.word_tokenize(tweet1))
```

```
↳ ['@', 'someone', 'did', 'you', 'check', 'out', 'this', '#', 'superawesome', '!', '!', '<', '3']
```

# Twitter-specific Tokenizer

**Ttokenize** is another tokenizer specifically designed for processing Twitter data. Google Colab doesn't have it built-in, so we will first use `pip` installer to install the `Ttokenize` package. ⋮

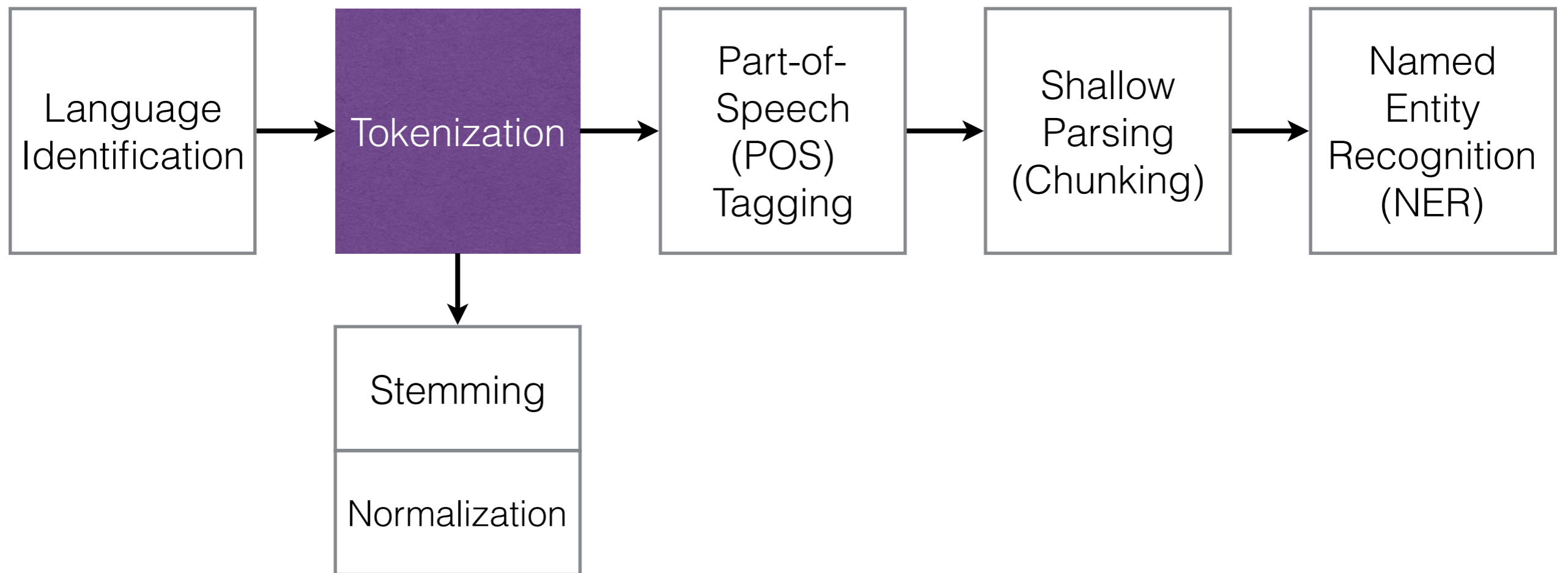
```
[16] !pip install -q twokenize
```

For the same example tweet as well, `Ttokenize` appears to work even better:

```
[25] import twokenize  
  
      tweet = "my heart.. broken T____T</3"  
  
      print (twokenize.tokenizeRawTweetText(tweet))
```

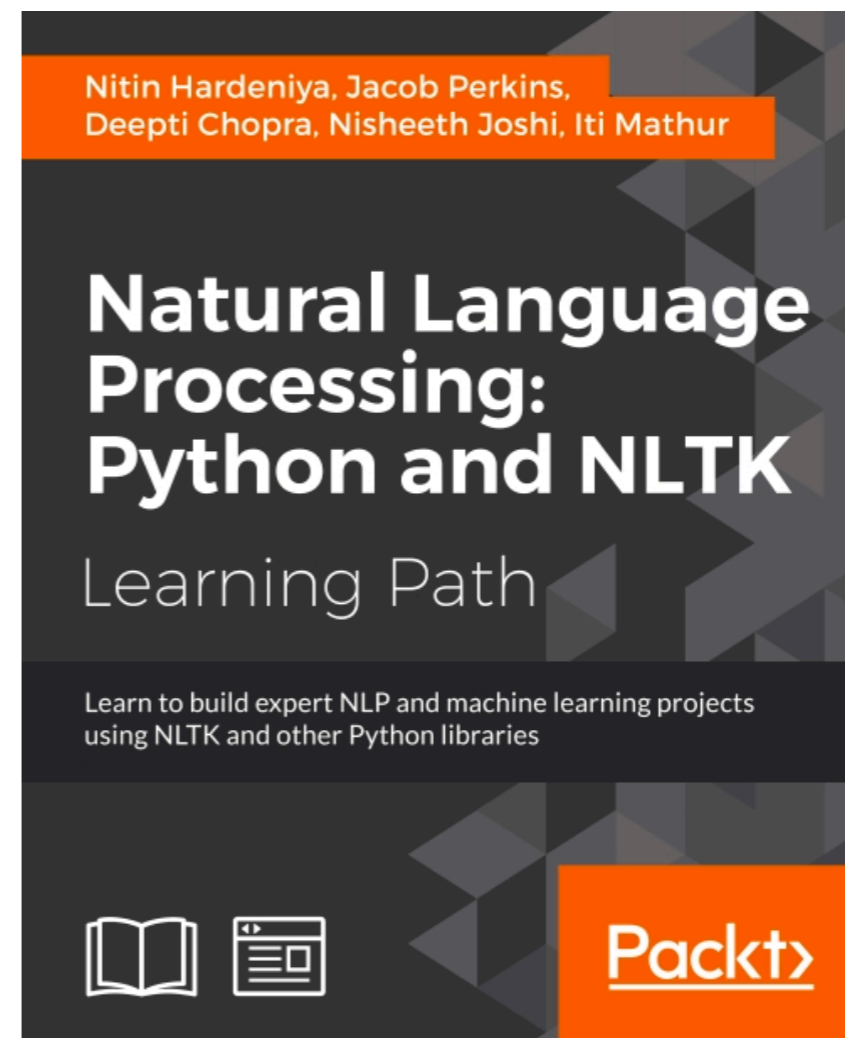
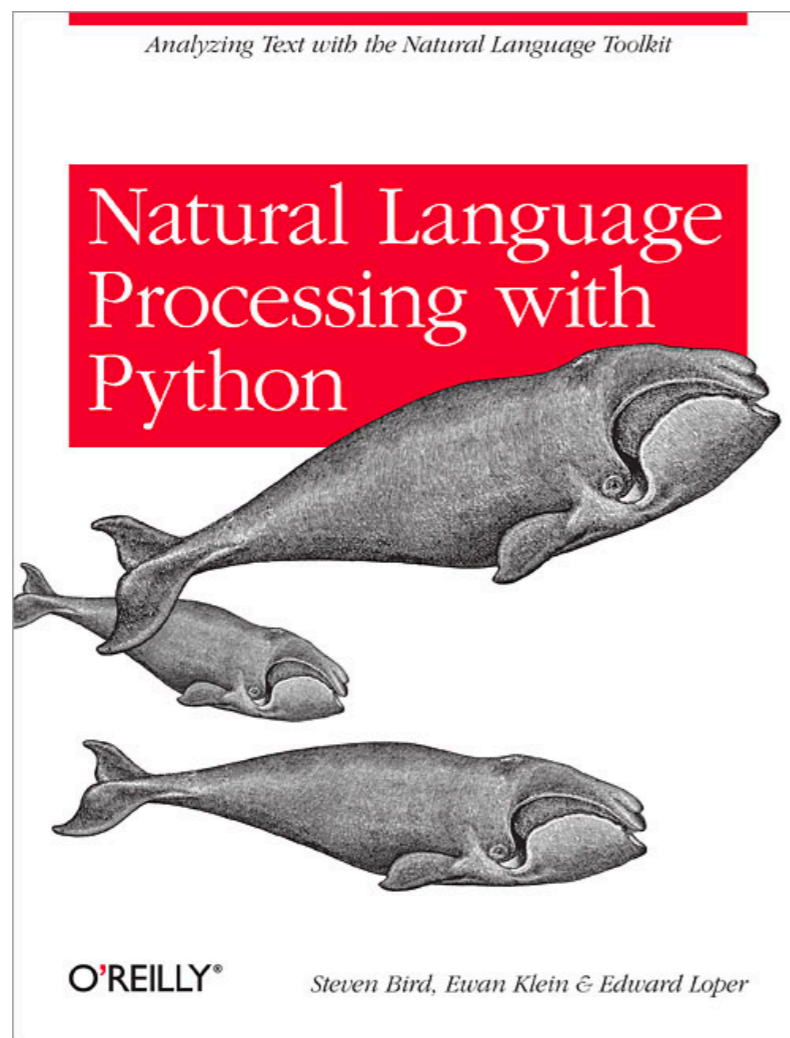
```
☞ ['my', 'heart', '..', 'broken', 'T____T', '</3']
```

# NLP Pipeline



# More Resources

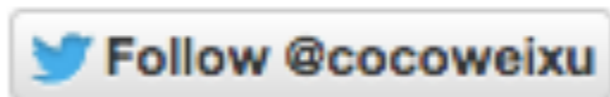
- NLTK (Natural Language Toolkit):
  - NLTK Book: <http://www.nltk.org/book> — free online!





# Social Media & Text Analysis

part 3 - Intro to Twitter API



**Ohio State University**

**Instructor: Wei Xu**

**Website: [socialmedia-class.org](http://socialmedia-class.org)**

# Twitter API Tutorial: [socialmedia-class.org](https://socialmedia-class.org)

Social Media & Text Analytics

Syllabus

Twitter API Tutorial

Homework ▾

High School Ou



Twitter's 404 error page -- the Fail Whale

## Twitter API tutorial

by **Wei Xu** [Follow @cocoweixu](#) and **Jeniya Tabassum** [Follow @JeniyaTabassum](#)


Last updated March 20, 2018 (added a script for obtaining all followers of a Twitter user)  
[\[download the Jupyter notebook for this tutorial\]](#)

### 1. Getting Twitter API keys

To start with, you will need to have a Twitter developer account and obtain credentials (API key and Access token secret) on the to access the Twitter API, following these steps:

- Create a Twitter developer account if you do not already have one from : <https://developer.twitter.com/en/apps>
- Go to <https://developer.twitter.com/en/apps> and log in with your Twitter user account
- Click “Create an app”
- Fill out the form, and click “Create”

# Tweets




**ChuckGrassley** ✓  
@ChuckGrassley

⚙️ [+ Follow](#)

July4 we celebr8 freedom&liberty that started w gr8est document evr written Declaration of Independence SO hv happy Independence Day.IPARADE

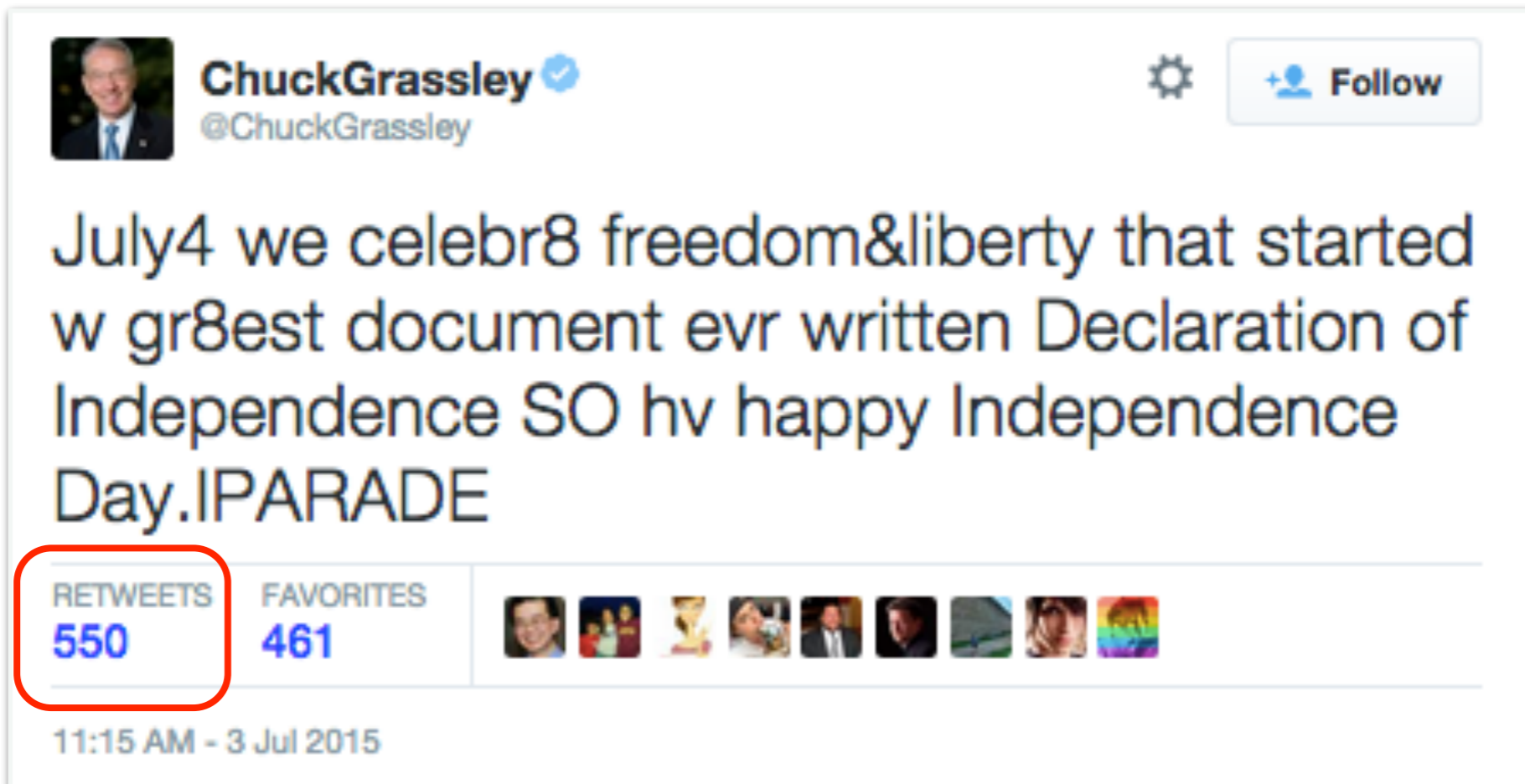
---

RETWEETS **550**   FAVORITES **461**




11:15 AM - 3 Jul 2015

# ReTweets



A screenshot of a tweet from Chuck Grassley (@ChuckGrassley) posted on July 3, 2015, at 11:15 AM. The tweet text reads: "July4 we celebr8 freedom&liberty that started w gr8est document evr written Declaration of Independence SO hv happy Independence Day.IPARADE". The tweet has 550 retweets and 461 favorites. The "RETWEETS 550" section is highlighted with a red box. Below the tweet, there are several small profile pictures of users who interacted with the tweet, including a rainbow flag icon.

**ChuckGrassley**   
@ChuckGrassley

July4 we celebr8 freedom&liberty that started w gr8est document evr written Declaration of Independence SO hv happy Independence Day.IPARADE

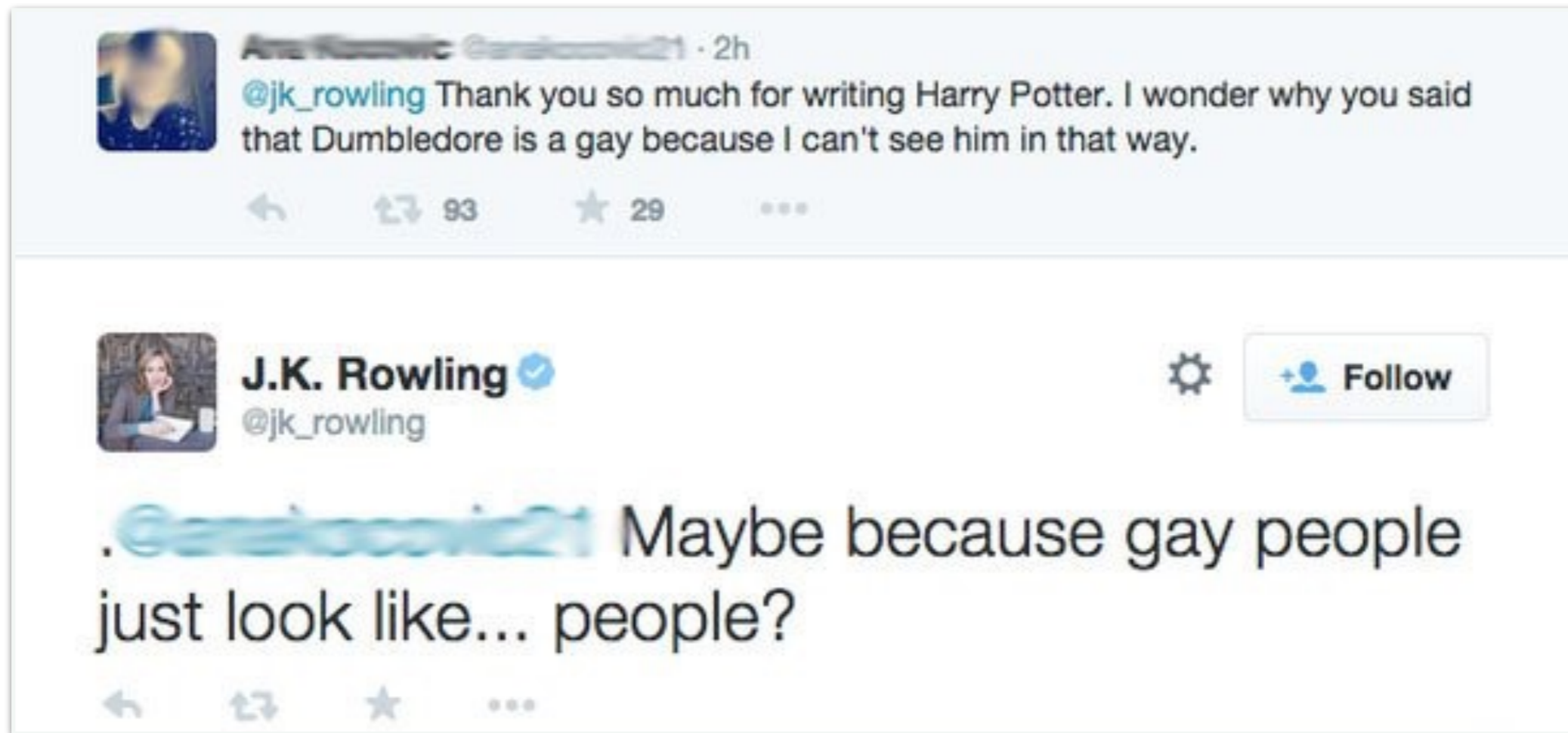
**RETWEETS**  
**550**

**FAVORITES**  
**461**

11:15 AM - 3 Jul 2015

a re-posting of someone else's Tweet

# Replies/Conversations



- Tweet starts with a @username

# Twitter API

# What is an API?

**A**pplication **P**rogramming **I**nterface

API is a set of protocols that specify how software programs communicate with each other.

# What is an API?

## **Without API:**

An app finds the current weather in London by opening <http://www.weather.com/> and reading the webpage like a human does, interpreting the content.

## **With API:**

An app finds the current weather in London by sending a message to the [weather.com](#) API (in a structured format like XML). The [weather.com](#) API then replies with a structured response.



# Twitter API

- Twitter is recognized for having one of the most open and powerful developer APIs of any major technology company.
- The first version of its public API was released in September 2006.

# Twitter API

Streaming API	REST API
a sample of public tweets and events as they published on Twitter (can specify search terms or users)	<ul style="list-style-type: none"><li>- search</li><li>- trends</li><li>- read author profile and follower data</li><li>- post / modify</li></ul>
<b>only</b> real-time data	historical data up to a week
continuous net connection	one-time request
no limit	rate limit (varies for different requests)

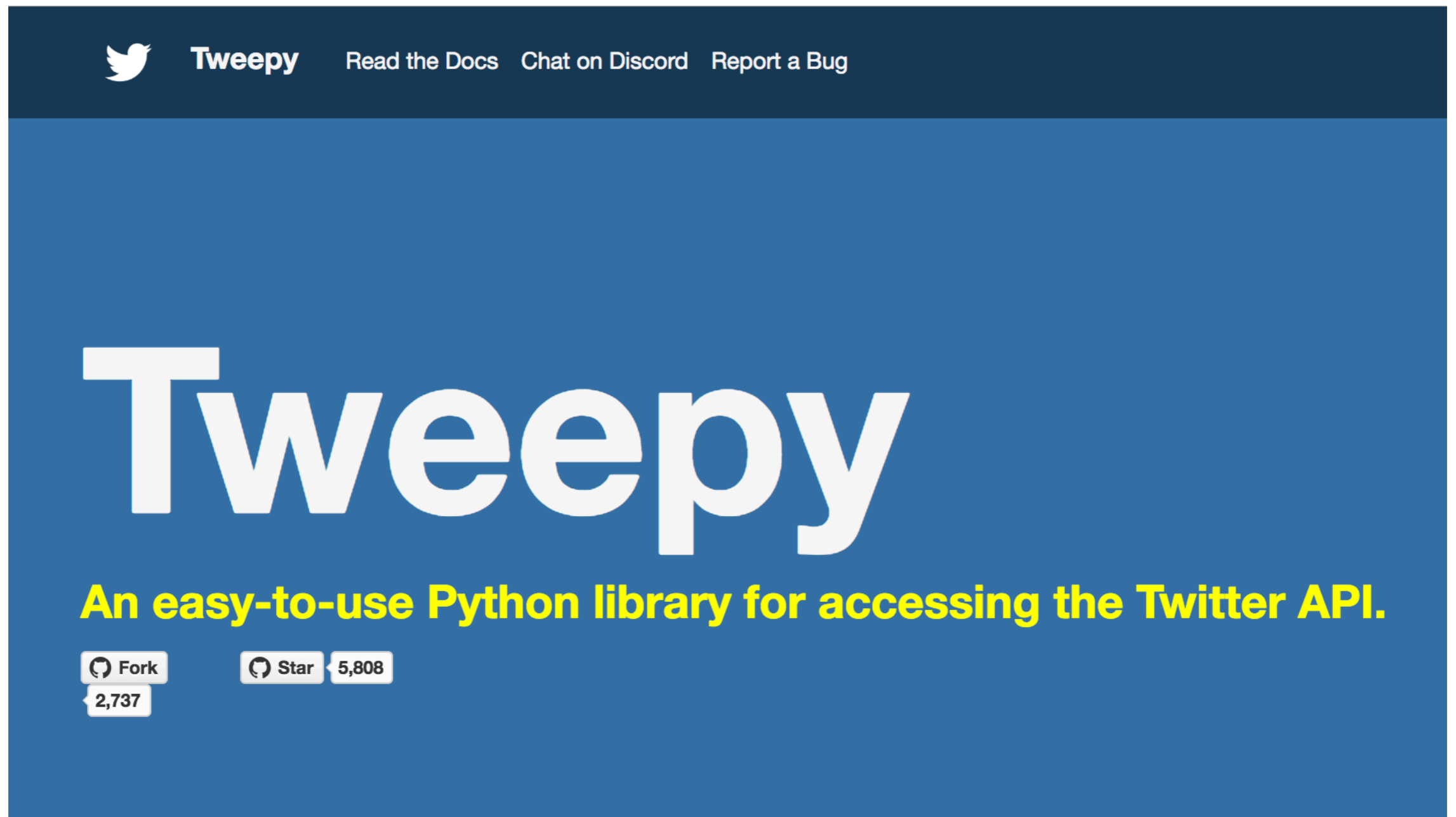
# OAuth Authentication

- Twitter uses OAuth to provide authorized access to its API.
- which means, to start with needs:
  - a Twitter account
  - OAuth access tokens from [apps.twitter.com](https://apps.twitter.com)

OAuth settings	
Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human	
Access level	Read-only <a href="#">About the application permission model</a>
Consumer key	1234567890
Consumer secret	NZsJqxVPe4IP1XebbXtAXpLYrQZcg4RIfCjuXbzjAk4

# Python Twitter Tools

[www.tweepy.org](http://www.tweepy.org)



The screenshot shows the top portion of the Tweepy website. At the top, there is a dark blue navigation bar containing the Tweepy logo (a white bird icon), the word "Tweepy" in white, and three links: "Read the Docs", "Chat on Discord", and "Report a Bug". Below this is a large blue area with the word "Tweepy" in a very large, white, sans-serif font. Underneath the name, the text "An easy-to-use Python library for accessing the Twitter API." is written in a smaller, yellow font. At the bottom left of this blue area, there are three GitHub-style buttons: "Fork" with a fork icon and the number "2,737", "Star" with a star icon and the number "5,808".

# OAuth Authentication

Twitter uses OAuth to provide authorized access to the API.

```
[ ] # Import the tweepy library
import tweepy
from tweepy.streaming import StreamListener

# Variables that contains the user credentials to access Twitter API
ACCESS_TOKEN = 'YOUR ACCESS TOKEN'
ACCESS_SECRET = 'YOUR ACCESS TOKEN SECRET'
CONSUMER_KEY = 'YOUR API KEY'
CONSUMER_SECRET = 'ENTER YOUR API SECRET'

# Setup tweepy to authenticate with Twitter credentials:
auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)
```

# Streaming API

```
[ ] # This is a basic listener that just prints received tweets to stdout.
class StdOutListener(StreamListener):

    def on_data(self, data):
        print(json.loads(data))
        return True

    def on_error(self, status):
        if status_code == 420:
            return False

# tweepy.Stream.sample() will give a live stream (~1% sample) of all public tweets
# Warning: it will continue to run indefinitely until you stop it.

listener = StdOutListener()
twitterStream = tweepy.Stream(auth, listener)
twitterStream.sample()
```

# JSON

## **JavaScript Object Notation**

JSON is a minimal, readable format for structuring data.

# A Tweet in JSON



#CFP Workshop on Noisy User-generated Text at ACL - Beijing 31 July 2015. Papers due: 11 May 2015. [noisy-text.github.io](http://noisy-text.github.io)  
#NLProc #WNUT15

```
{
  "favorited": false,
  "contributors": null,
  "truncated": false,
  "text": "#CFP Workshop on Noisy User-generated Text at ACL - Beijing 31 July 2015. Papers due: 11 May 2015. http://t.co/rcygyEowqH #NLProc #WNUT15",
  "possibly_sensitive": false,
  "in_reply_to_status_id": null,
  "user": {
    "follow_request_sent": null,
    "profile_use_background_image": true,
    "default_profile_image": false,
    "id": 237918251,
    "verified": false,
    "profile_image_url_https": "https://pbs.twimg.com/profile_images/527088456967544832/Dn
```



# Search

Home

Moments

Notifications

Messages



#nlproc



Tweet

ACL2019 and 4 others liked



**Manaal Faruqi** @manaalfar · 6h

As a junior AC for @ACL2019\_Italy I have to write meta-reviews for only 10 papers in the morphology/phonology track. So much better than ACL 2017 when I had to do it for 25 papers in Semantics track. Refreshing to be an AC of a smaller area! :D #NLProc

1



13



**Stanford NLP Group** @stanfordnlp · 6h

Today, Mar 20—@Stanford CS224N NLP with Deep Learning Poster Session 5–9pm Arrillaga Alumni Center. Free parking after 4pm in A/C spots on Galvez, lots, garages. Come talk with 500 amazing Stanford students about question answering, dialog, MT etc #NLProc [facebook.com/events/1218481...](https://facebook.com/events/1218481...)



2

13



**Machine Learning and NLP** @ML\_NLP · 6h

GluonNLP 0.6: Closing the Gap in Reproducible Research with BERT [medium.com/apache-mxnet/g...](https://medium.com/apache-mxnet/g...) #NLProc

# Search API

```
[ ] # Search for latest tweets about "#nlproc"
    tweets = tweepy.Cursor(api.search, q='#nlproc')

    # Print out the latest 10 tweets that contain "#nlproc" hashtag
    for item in tweets.items(10):
        print(item._json)
```

```
{'name': 'Dwayne Haskins', 'url': 'http://twitter.com/search?q=%22Dwayne+Haskins%22', 'promoted_content': None, 'query': '%22Dwayne+Haskins%22', 'tweet_volume': 70085}
{'name': 'McCain', 'url': 'http://twitter.com/search?q=McCain', 'promoted_content': None, 'query': 'McCain', 'tweet_volume': 381900}
{'name': 'Lima', 'url': 'http://twitter.com/search?q=Lima', 'promoted_content': None, 'query': 'Lima', 'tweet_volume': 70085}
{'name': '#firstdayofspring', 'url': 'http://twitter.com/search?q=%23firstdayofspring', 'promoted_content': None, 'query': '%23firstdayofspring', 'tweet_volume': 70085}
{'name': 'Daniel Caesar', 'url': 'http://twitter.com/search?q=%22Daniel+Caesar%22', 'promoted_content': None, 'query': '%22Daniel+Caesar%22', 'tweet_volume': 70085}
{'name': '#InternationalDayOfHappiness', 'url': 'http://twitter.com/search?q=%23InternationalDayOfHappiness', 'promoted_content': None, 'query': '%23InternationalDayOfHappiness', 'tweet_volume': 70085}
{'name': 'AirPods', 'url': 'http://twitter.com/search?q=AirPods', 'promoted_content': None, 'query': 'AirPods', 'tweet_volume': 131265}
{'name': 'Pro Day', 'url': 'http://twitter.com/search?q=%22Pro+Day%22', 'promoted_content': None, 'query': '%22Pro+Day%22', 'tweet_volume': 19808}
{'name': '#SpringEquinox', 'url': 'http://twitter.com/search?q=%23SpringEquinox', 'promoted_content': None, 'query': '%23SpringEquinox', 'tweet_volume': 70085}
{'name': 'Flume', 'url': 'http://twitter.com/search?q=Flume', 'promoted_content': None, 'query': 'Flume', 'tweet_volume': 37576}
{'name': '#HappinessInOneWord', 'url': 'http://twitter.com/search?q=%23HappinessInOneWord', 'promoted_content': None, 'query': '%23HappinessInOneWord', 'tweet_volume': 70085}
{'name': '#StrangerThings3', 'url': 'http://twitter.com/search?q=%23StrangerThings3', 'promoted_content': None, 'query': '%23StrangerThings3', 'tweet_volume': 70085}
{'name': 'Eloy', 'url': 'http://twitter.com/search?q=Eloy', 'promoted_content': None, 'query': 'Eloy', 'tweet_volume': None}
{'name': 'Happy Spring', 'url': 'http://twitter.com/search?q=%22Happy+Spring%22', 'promoted_content': None, 'query': '%22Happy+Spring%22', 'tweet_volume': 70085}
{'name': 'Bill & Ted 3', 'url': 'http://twitter.com/search?q=%22Bill+%26+Ted+3%22', 'promoted_content': None, 'query': '%22Bill+%26+Ted+3%22', 'tweet_volume': 70085}
```

# Trends

Home Moments Notifications Messages Messages Search Twitter Tweet


**Wei Xu**  
@cocoweixu  
Tweets 423 Following 510 Followers 2,767

**Columbus trends** · [Change](#)

- #ApexSeason1**  
Apex Legends Season 1 is Here  
Promoted by Apex Legends
- Terry McLaurin**
- Ohio State**  
13.4K Tweets
- Haskins**  
9,893 Tweets
- Pro Day**  
21.9K Tweets
- Johnnie Dixon**
- Nick Bosa**  
1,853 Tweets
- Giants**  
35.9K Tweets
- #firstdayofspring**  
It is officially the first day of spring! 🌸🌻🌻
- Lima**  
63.6K Tweets

What's happening?

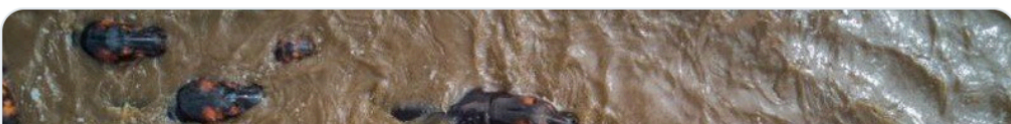
**Women in Analytics Conference (WIA)** @wia\_conference · 4s  
We are so excited to see everyone tomorrow at the 2019 Women in Analytics Conference! Keep us in the loop by using hashtag #WIA2019 when sharing your favorite conference moments on @LinkedIn, @Twitter, @Facebook, and @Instagram.  
[womeninanalytics.org](http://womeninanalytics.org)



2019  
Ethics in  
Algorithms

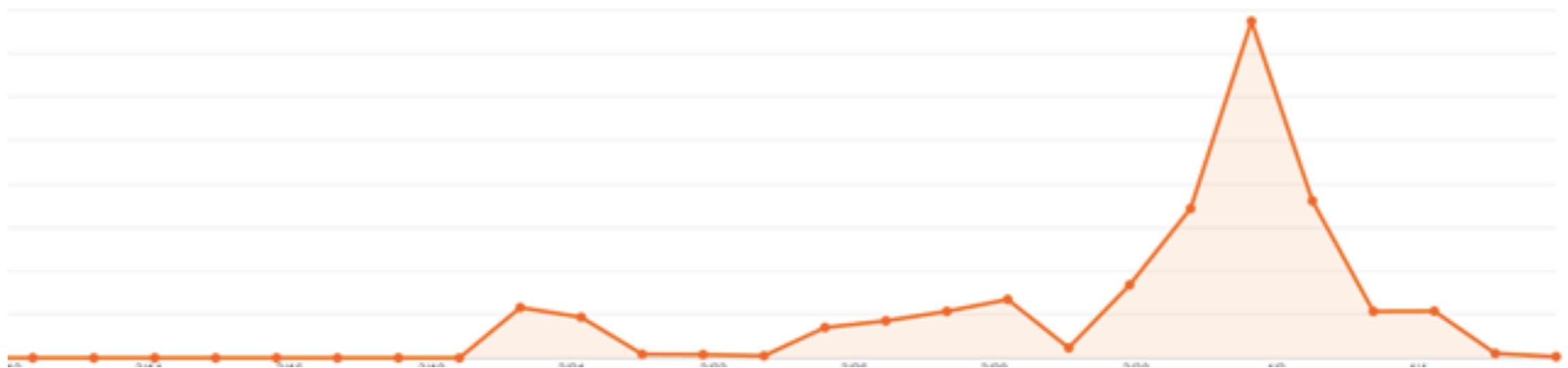
2019 Women in Analytics Conference | Ethics in Algorithms  
[womeninanalytics.org](http://womeninanalytics.org)

**Travelfuntu** @travel\_funtu · Mar 10  
This Photo Has Not Been Edited, Look Closer



# Trends

trending topics are determined by an unpublished algorithm, which finds words, phrases and hashtags that have had a sharp increase in popularity, as opposed to overall volume.



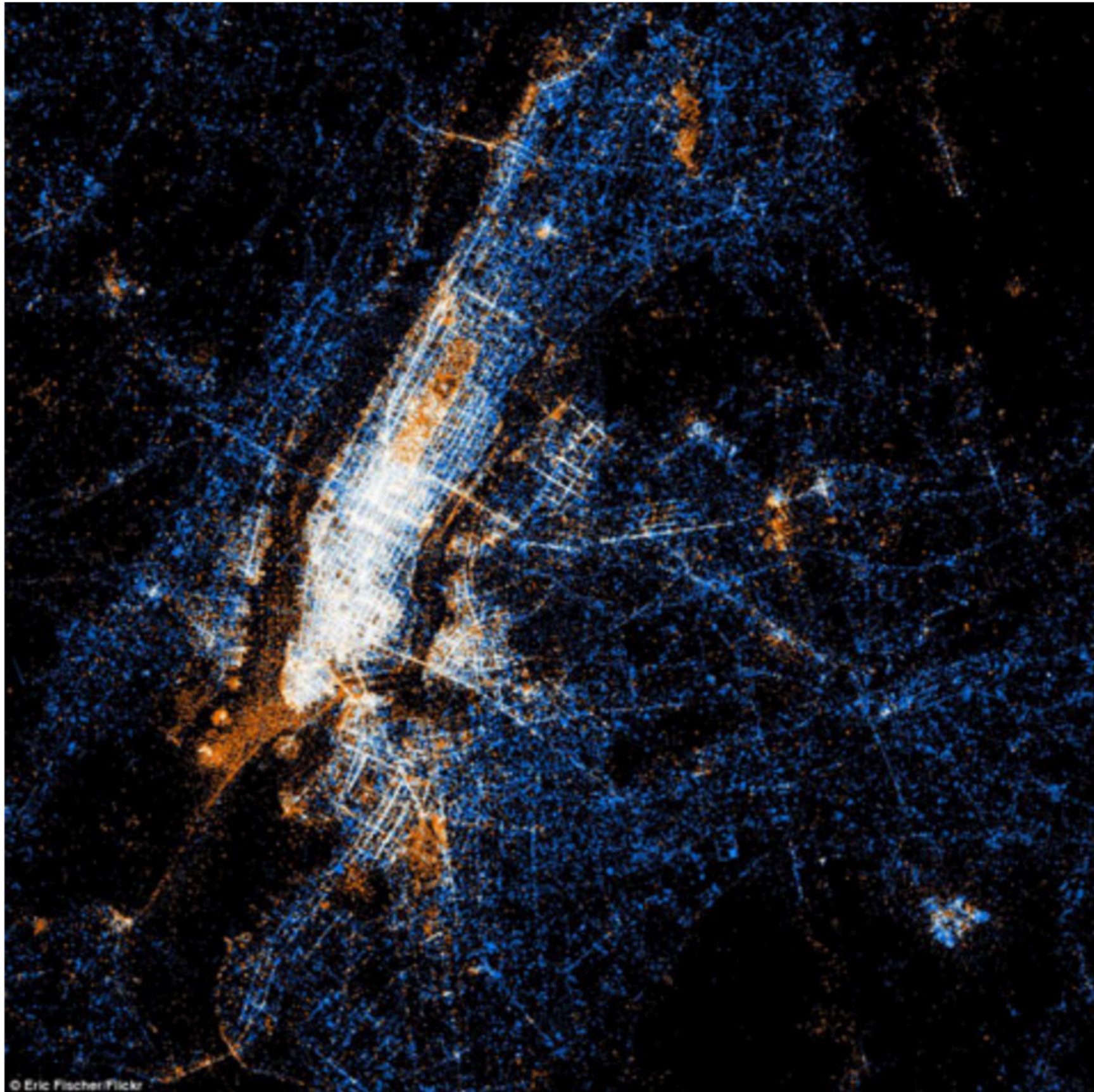
# Trends API

Where On Earth ID



```
# Where On Earth ID for Columbus, Ohio is 2383660.  
COLUMBUS_WOE_ID = 2383660  
  
columbus_trends = api.trends_place(COLUMBUS_WOE_ID)  
  
trends = json.loads(json.dumps(columbus_trends, indent=1))  
  
for trend in trends[0]["trends"]:  
    print (trend)
```





visualizations by Eric Fischer A visualization showing the location of Twitter messages and Flickr photos in New York City.

# Social Science

wonderfully delightfully beautifully fine well good nicely superbly



**she says**



**he says**

(also age & income)

# Detecting Similar Tweets

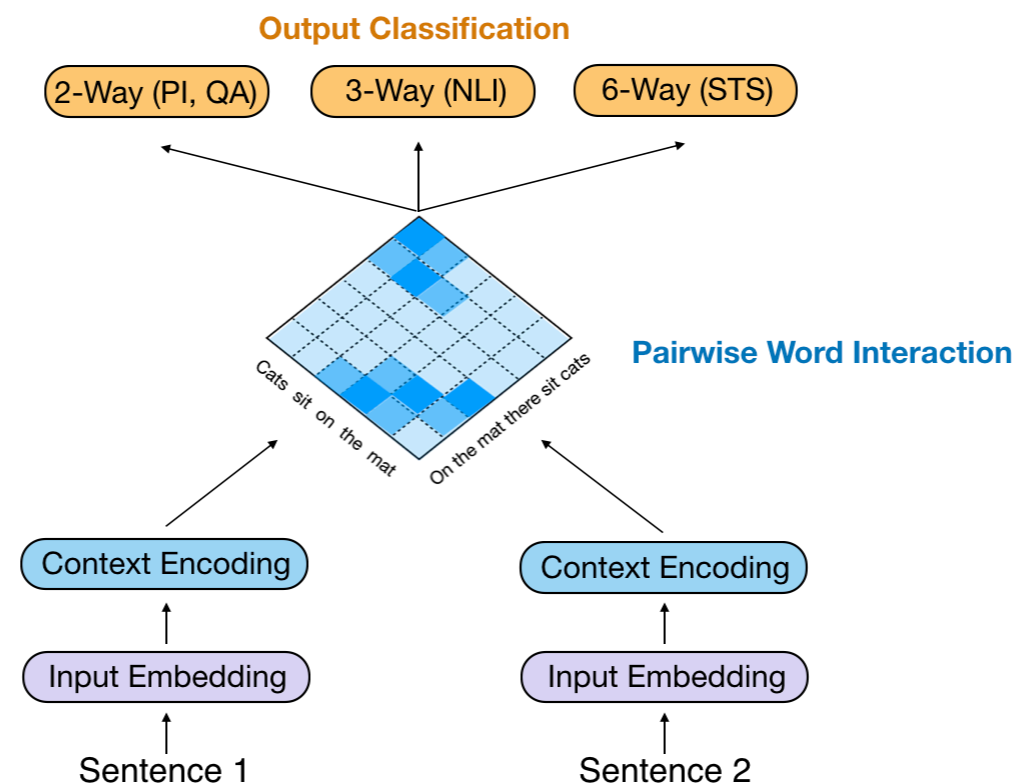
**paraphrase**

*Ezekiel Ansah is wearing real3D glasses with the lenses punched out*

*Ezekiel Ansah wearing 3D glasses wout the lens*

**non-paraphrase**

*I wore the 3D glasses wout lenses before Ezekiel Ansah*



Wei Xu, Alan Ritter, Chris Callison-Burch, Bill Dolan, and Yangfeng Ji. "Extracting Lexically Divergent Paraphrases from Twitter" in TACL (2014)

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. "A Continuously Growing Dataset of Sentential Paraphrases" in EMNLP (2017)

Wuwei Lan and Wei Xu. "Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering" in COLING (2018) Best Paper Award