



Università degli Studi di Cagliari
Corso di Laurea DSBAI

Web Analytics e Analisi Testuale

<http://agilegroup.eu>

A.A. 2017/2018

Ing. Marco Ortu

Via Porcell 4, primo piano

mail: marco.ortu@diee.unica.it

Social Web Mining

Mining Social Web

Il cosiddetto **Social Web** rappresenta la più grande miniera di dati sugli utenti del Web liberamente disponibile nonché un'industria da miliardi di dollari. Vedremo come analizzare le caratteristiche di tre importanti social media.

→ Twitter

- ◆ analisi dei topic trend
- ◆ analisi di “di cosa le persone stanno parlando?”

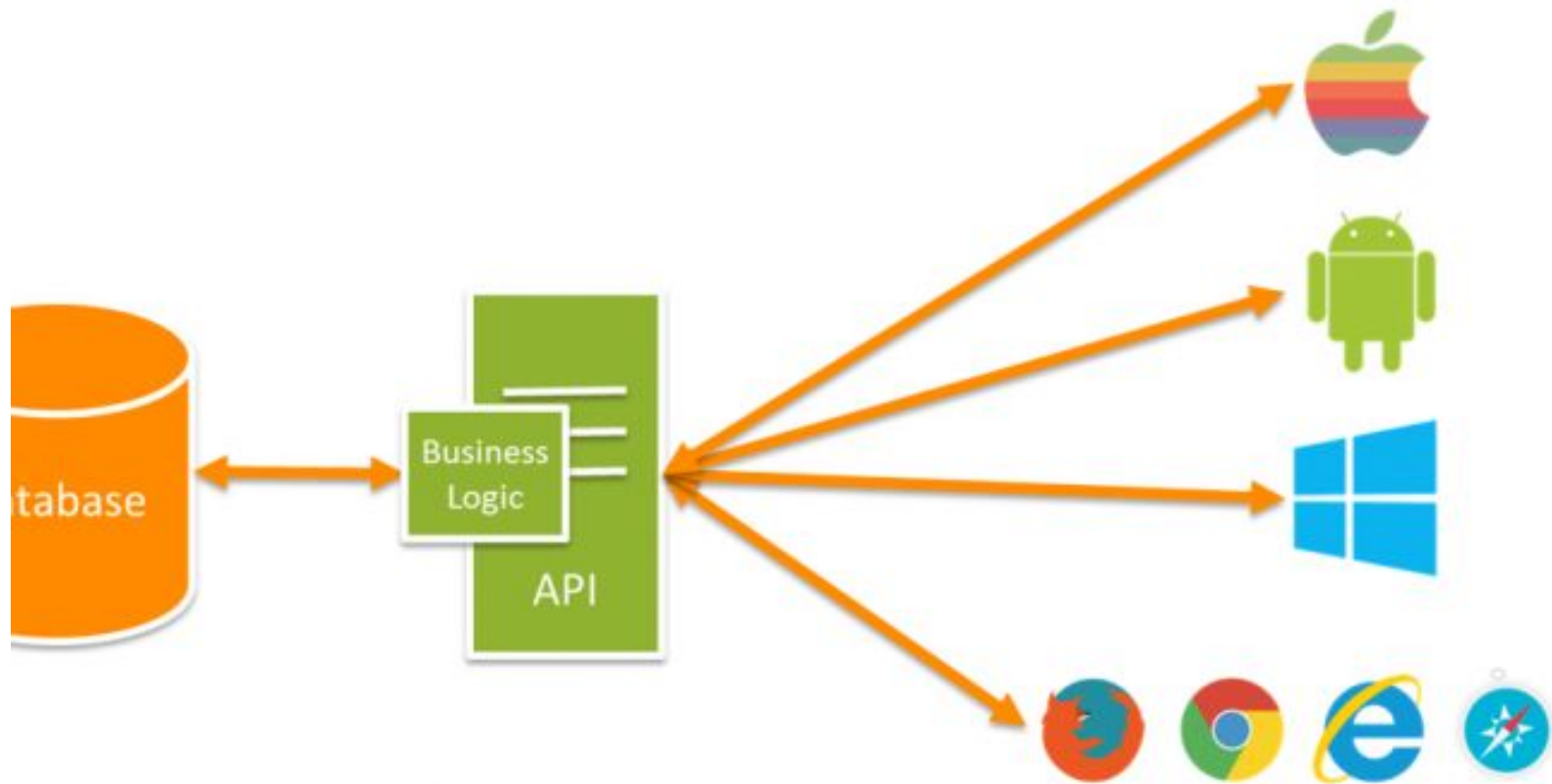
→ Facebook

- ◆ analisi delle Fan Page
- ◆ esplorazione delle Friendship

→ LinkedIn

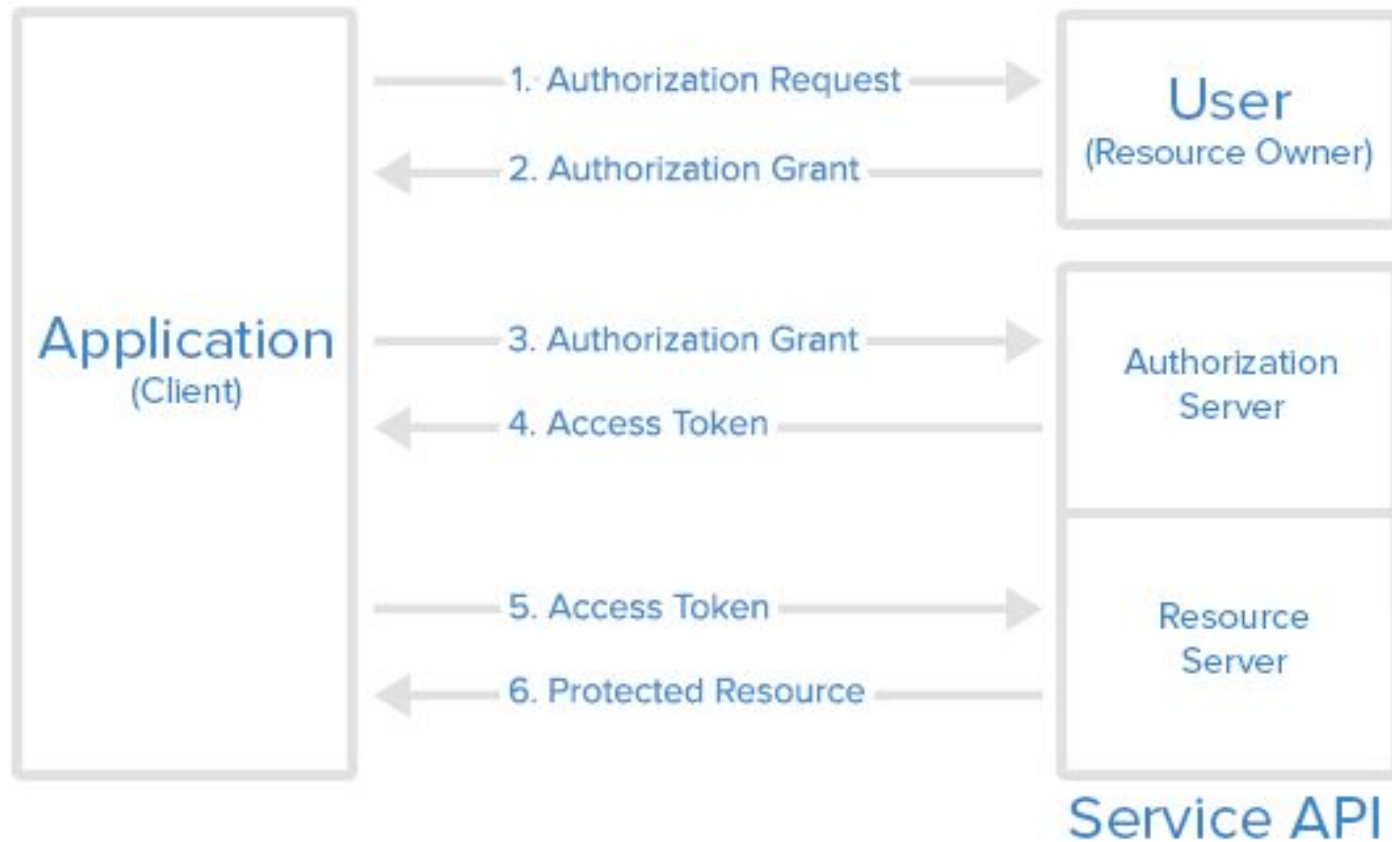
- ◆ Clustering dei colleghi
- ◆ Mining job's titles

Social Web: Web API



Social Web: Protocollo OAuth-2

Abstract Protocol Flow

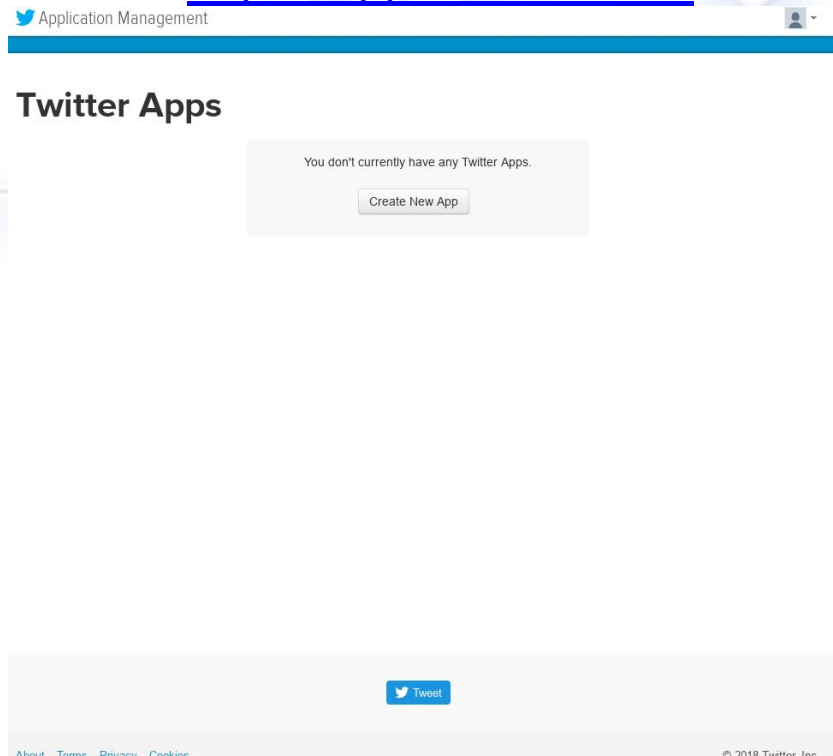


Social Web: Twitter

Il primo step per poter lavorare con Twitter è quello di ottenere delle credenziali per poter accedere alle API fornite.

Per ottenere delle credenziali è necessario creare una “**app**” al seguente indirizzo:

<https://apps.twitter.com/>



Social Web: Twitter

Create an application

Application Details

Name *

Your application name. This is used to identify the source of authorized tweets in user-facing authorization screens. 32 characters max.

Description *

Maximum length: 255 characters. This is used to identify the source of authorized tweets in user-facing authorization screens. 255 characters max.

Website *

Your application's website. This is used to identify the source of authorized tweets in user-facing authorization screens. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URLs

Where should we return after successfully authenticating? OAuth 1.0a applications must explicitly specify their oauth_callback URL(s) here, as well as include the one of the URLs below in the request token step. To restrict your application from using callbacks, leave this field blank.

Nome della vostra applicazione

Http://example-url.com

Accettare i termini di utilizzo

Developer Agreement

Yes, I have read and agree to the [Twitter Developer Agreement](#).

Social Web: Twitter

Application Management [User Profile]

WAAT_2018 Test OAuth

[Details](#) [Settings](#) **Keys and Access Tokens** [Permissions](#)

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	[Redacted]
Consumer Secret (API Secret)	[Redacted]
Access Level	Read and write (modify app permissions)
Owner	[Redacted]
Owner ID	[Redacted]

Otteniamo la CONSUMER_KEY e il CONSUMER_SECRET

Application Actions

[Regenerate Consumer Key and Secret](#) [Change App Permissions](#)

Your Access Token

You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to make API calls right away. The access token generated will be assigned your application's current permission level.

Social Web: Twitter

Generiamo adesso gli access tokens per la nostra applicazione



Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	[REDACTED]
Access Token Secret	[REDACTED]
Access Level	Read-only
Owner	[REDACTED]
Owner ID	[REDACTED]

Otteniamo l'ACCESS_TOKEN e l'ACCESS_TOKEN_SECRET

Token Actions

Regenerate My Access Token and Token Secret

Revoke Token Access

Social Web: Twitter

1. Dal repository <https://github.com/marcoortu/WAAT-2019> fare il checkout del branch **social**.
2. Creare il file **credentials.py** che contiene le credenziali per i diversi social media che analizzeremo, sostituire le credenziali appena ottenute nelle seguenti variabili.

Mettere qui i dati corretti per Twitter

```
TWITTER_CONSUMER_KEY = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'  
TWITTER_CONSUMER_SECRET = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'  
TWITTER_ACCESS_TOKEN = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'  
TWITTER_ACCESS_TOKEN_SECRET = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
```

Social Web: Twitter

Utilizziamo la libreria di appoggio *tweepy* (<http://www.tweepy.org/>) che permette di fare richieste alle Web API di Twitter in maniera molto semplice.

Creiamo un client di autenticazione con le credenziali ottenute nei passaggi precedenti.

```
import tweepy
from tweepy import OAuthHandler
auth = OAuthHandler(
    TWITTER_CONSUMER_KEY,
    TWITTER_CONSUMER_SECRET
)
auth.set_access_token(TWITTER_ACCESS_TOKEN,
    TWITTER_ACCESS_TOKEN_SECRET)
twitterAPI = tweepy.API(auth)
```

Otteniamo l'oggetto *twitterAPI* che ci permette di interagire con le Web API fornite da Twitter.

Social Web: Twitter

Testiamo le nostre credenziali facendo una ricerca di tweets dato un testo di ricerca.

```
fetchdTweets = twitterAPI.search(q="data scientist", count=100)
for tweet in fetchdTweets:
    pprint(tweet)
```

Il metodo **search** richiede la query di ricerca e un numero massimo di tweet. Il nostro oggetto **twitterAPI** è semplicemente un wrapper che si occupa di fare le chiamate alle Web API di Twitter per noi. Nella documentazione troviamo tutte le informazioni per utilizzare la chiamata.

<https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets>

Social Web: Twitter

Standard search API

Returns a collection of relevant [Tweets](#) matching a specified query.

Please note that Twitter's search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface.

To learn how to use [Twitter Search](#) effectively, please see the [Standard search operators](#) page for a list of available filter operators. Also, see the [Working with Timelines](#) page to learn best practices for navigating results by `since_id` and `max_id`.

Resource URL

`https://api.twitter.com/1.1/search/tweets.json`

URL da chiamare tramite protocollo HTTP

Resource Information

Response formats	JSON
Requires authentication?	Yes
Rate limited?	Yes
Requests / 15-min window (user auth)	180
Requests / 15-min window (app auth)	450

Limiti delle chiamate

Parameters

Name	Required	Description	Default Value	Example
q	required	A UTF-8, URL-encoded search query of 500 characters maximum, including operators. Queries may additionally be limited by complexity.		<code>@noradio</code>

Parametri della richiesta

Social Web: Twitter

```
fetchdTweets = twitterAPI.search(q="data scientist", count=100)
```

```
for tweet in fetchdTweets:
```

```
    print tweet
```

Il parametro count specifica quanti risultati mettere per pagina, con un massimo di 100 e un valore predefinito di 15.

count	optional	The number of tweets to return per page, up to a maximum of 100. Defaults to 15. This was formerly the "rpp" parameter in the old Search API.	100
-------	----------	-----------------------------------------------------------------------------------------------------------------------------------------------	-----

Per una lista completa delle funzionalità fornite dalle Web API di Twitter si può consultare la pagine ufficiale della documentazione.

<https://developer.twitter.com/en/docs>

Social Web: Twitter Limits

Le richieste che si possono fare alle API sono limitate.

<https://developer.twitter.com/en/docs/basics/rate-limiting>

Una app con permessi standard (default) può effettuare un massimo di 15 richieste ogni 15 minuti. Superato questo limite per ogni richiesta successiva verrà restituito il seguente errore:

[HTTP 429 “Too Many Requests”](#)

Social Web: Twitter

Creiamo ora una piccola applicazione per la sentiment analysis. Per semplicità utilizziamo una libreria di appoggio che si occupa di valutare il sentiment di un dato testo: ***TextBlob***.

TextBlob è una libreria di alto livello costruita utilizzando la libreria ***NLTK***. Quando passiamo del testo per creare un oggetto ***TextBlob***, le seguenti elaborazioni vengono effettuate:

1. Tokenize del tweet, cioè le parole divise dal corpo del testo.
2. Rimuove le stopwords dai token (le stopwords sono le parole comunemente usate che sono irrilevanti nell'analisi del testo come I, am, you, are, ecc.)
3. Fai tagging POS (part of speech) dei token e seleziona solo features/token significative come aggettivi, avverbi, ecc.
4. Passa i token a un classificatore di sentimenti che classifica il sentimento del tweet come positivo, negativo o neutro assegnandogli una polarità tra -1.0 e 1.0.

Social Web: Twitter

```
from textblob import TextBlob
```

```
import re
```

```
def cleanTweet(tweet):
```

```
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) | (\w+:\ / \ / \S+)", "",  
tweet).split())
```

```
def getTweetSentiment(tweet):
```

```
    analysis = TextBlob(cleanTweet(tweet))
```

```
    if analysis.sentiment.polarity > 0:
```

```
        return 'positive'
```

```
    elif analysis.sentiment.polarity == 0:
```

```
        return 'neutral'
```

```
    else:
```

```
        return 'negative'
```

Social Web: Twitter

```
def cleanTweet(tweet):  
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) | (\w+:\ / \ / \S+)", "", tweet).split())
```

La funzione di appoggio ***cleanTweet*** serve a rimuovere dal testo links e caratteri speciali che creano problemi nell'analisi del sentiment, utilizzando le regular expressions.

```
def getTweetSentiment(tweet):  
    analysis = TextBlob(cleanTweet(tweet))  
    if analysis.sentiment.polarity > 0:  
        return 'positive'  
    elif analysis.sentiment.polarity == 0:  
        return 'neutral'  
    else:  
        return 'negative'
```

La funzione ***getTweetSentiment*** crea un oggetto ***TextBlob***, passando il testo ripulito del tweet, che contiene i valori del sentiment calcolati.

Social Web: Twitter

Ecco come viene creato il classificatore dei sentimenti:

1. **TextBlob** utilizza il corpus delle Movie Reviews di **NLTK** in cui le recensioni sono già state contrassegnate come positive o negative.
2. Le feature positive e negative sono estratte rispettivamente da ciascuna recensione positiva e negativa.
3. I dati di addestramento comprendono features positive e negative con etichetta. Questi dati sono addestrati su un classificatore **Naive Bayes**.

Usiamo il metodo **sentiment.polarity** della classe **TextBlob** per ottenere la polarità del tweet tra -1 e 1. Quindi, classifichiamo la polarità come:

```
if analysis.sentiment.polarity > 0:  
    return 'positive'  
elif analysis.sentiment.polarity == 0:  
    return 'neutral'  
else:  
    return 'negative'
```

Social Web: Twitter

Un aspetto molto importante quando si estraggono dati dai social media è la mole dei dati da estrarre. Sarebbe infatti impensabile andare a scaricare centinaia di migliaia di tweet con una sola richiesta alle Web API.

Per questi motivi la maggior parte dei social media fornisce le proprie API con un sistema di paginazione basato sui **Cursori**. Consideriamo il seguente esempio.

```
page = 1
while True:
    statuses = twitterAPI.user_timeline(page=page)
    if statuses:
        for status in statuses:
            # process status here
            pass
    else:
        # All done
        break
    page += 1 # next page
```

Social Web: Twitter

In questo caso bisogna considerare manualmente la paginazione, continuando ad iterare finchè ci sono nuove pagine. Ad ogni iterazione il nostro client farà una nuova richiesta per ottenere i nuovi tweet. Utilizzando un oggetto **Cursor** invece possiamo ottenere un oggetto direttamente iterabili:

```
for status in tweepy.Cursor(twitterAPI.user_timeline, id="twitter").items():  
    print status
```

In questo caso il cursore continuerà ad ottenere dati finchè questi saranno disponibili, gestendo automaticamente la paginazione.

Social Web: Twitter

```
for status in tweepy.Cursor(twitterAPI.user_timeline, id="twitter").items():  
    print status
```

In questo caso il cursore continuerà ad ottenere dati finchè questi saranno disponibili, gestendo automaticamente la paginazione.

Itera solo i sui primi 200 elementi

```
for status in tweepy.Cursor(twitterAPI.user_timeline).items(200):  
    print status
```

itera solo sulle prime 3 pagine

```
for page in tweepy.Cursor(twitterAPI.user_timeline).pages(3):  
    print page
```

Social Web: Twitter

Analizziamo nel dettaglio le informazioni contenute negli oggetti restituiti dalla ricerca.

```
tweets = tweepy.Cursor(twitterAPI.search, q=query).items(count)
status_texts, screen_names, hashtags= [],[],[]
for tweet in tweets:
    status_texts.append(tweet.text)
    screen_names += [userMention['screen_name'] for userMention in tweet.entities['user_mentions']]
    hashtags += [hashtag['text'] for hashtag in tweet.entities['hashtags']]
words = [word for text in status_texts for word in word_tokenize(text)]
for label, data in [('Word', words),
                  ('Screen Names', screen_names),
                  ('Hashtag', hashtags)
                  ]:
    prettyTable = PrettyTable(field_names=[label, 'Count'])
    counter = Counter(data)
    [prettyTable.add_row(kv) for kv in counter.most_common()[:10]]
    prettyTable.align[label] = 'l'
    prettyTable.align['Count'] = 'r'
    print prettyTable
```

Social Web: Twitter

PrettyTable è una libreria Python per la generazione di semplici tabelle ASCII. È ispirata dalle tabelle ASCII utilizzate nella shell di PostgreSQL `psql`.

Possiamo controllare molti aspetti di una tabella, come la larghezza del riempimento della colonna, l'allineamento del testo o il bordo della tabella. Possiamo ordinare i dati.

Dato un insieme di tweet estraiamo dai risultati, il testo, il nome utente, le **user mentions** (@username), le parole contenute nel testo e gli hashtag presenti.

```
tweets = tweepy.Cursor(twitterAPI.search, q=query).items(count)
status_texts, screen_names, hashtags= [],[], []
for tweet in tweets:
    status_texts.append(tweet.text)
    screen_names += [userMention['screen_name'] for userMention in tweet.entities['user_mentions']]
    hashtags += [hashtag['text'] for hashtag in tweet.entities['hashtags']]
words = [word for text in status_texts for word in word_tokenize(text)]
```

Social Web: Twitter

Utilizzando PrettyTable costruiamo poi quattro tabelle per visualizzare le quattro colonne metriche raccolte.

```
for label, data in [('Word', words),
                   ('Screen Names', screen_names),
                   ('Hashtag', hashtags)
                  ]:
    prettyTable = PrettyTable(field_names=[label, 'Count'])
    counter = Counter(data)
    [prettyTable.add_row(kv) for kv in counter.most_common()[:10]]
    prettyTable.align[label] = 'l'
    prettyTable.align['Count'] = 'r'
    print prettyTable
```

Social Web: Twitter

Word	Count
#	410
:	123
Trump	105
RT	78
@	69
.	64
https	63
you	54
believe	52
Donald	49

Hashtag	Count
Trump	66
Donald	46
donald	40
DonaldJTrump	26
USA	9
Kim	8
alt	6
Alte	6
Treffen	6
lobt	6

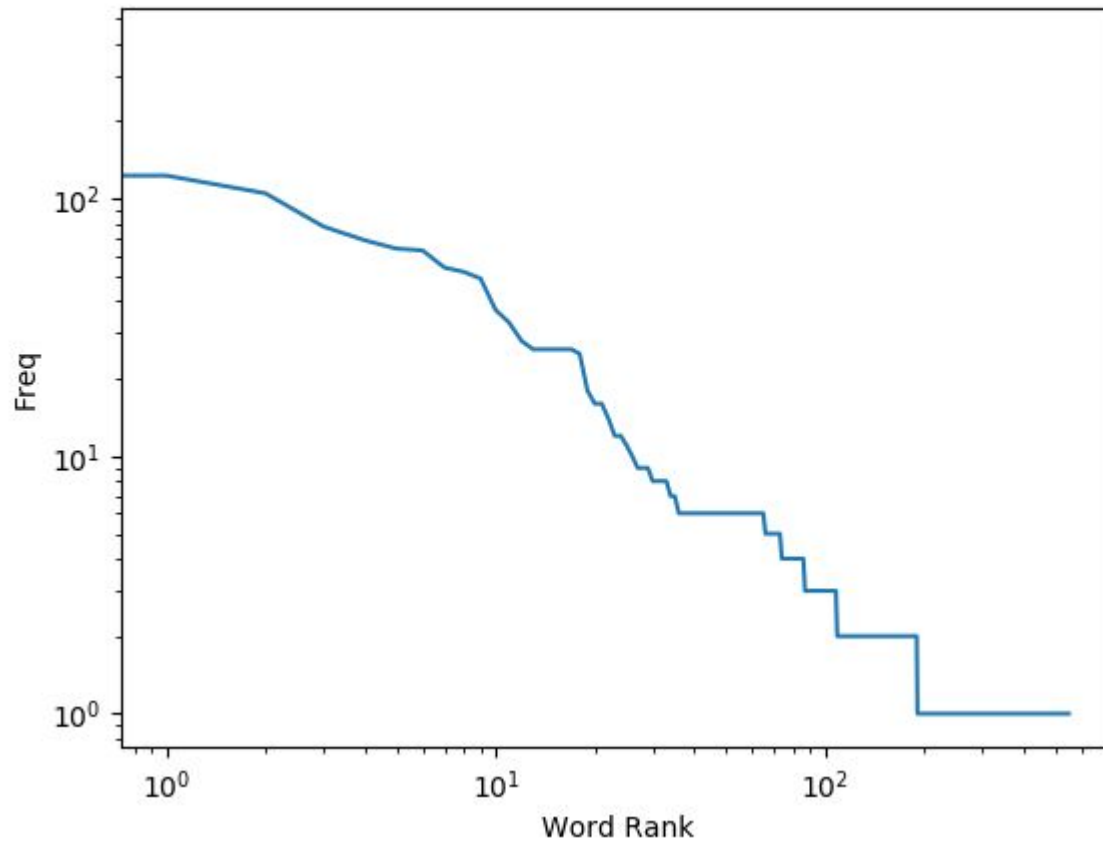
Social Web: Twitter

Possiamo visualizzare la frequenza delle parole.

```
def tweetsWordFrequency(query, count=10):
    tweets = tweepy.Cursor(twitterAPI.search, q=query).items(count)
    status_texts = []
    screen_names = []
    hashtags = []
    for tweet in tweets:
        status_texts.append(tweet.text)
        screen_names += [userMention['screen_name'] for userMention in tweet.entities['user_mentions']]
        hashtags += [hashtag['text'] for hashtag in tweet.entities['hashtags']]
    words = [word for text in status_texts for word in word_tokenize(text)]

    wordCounts = sorted(Counter(words).values(), reverse=True)
    plt.loglog(wordCounts)
    plt.ylabel("Freq")
    plt.xlabel("Word Rank")
    plt.show()
```

Social Web: Twitter



Social Web: Twitter

Possiamo visualizzare i retweet più popolari.

```
def retweetFrequency(query, count=10):
    tweets = tweepy.Cursor(twitterAPI.search, q=query).items(count)
    retweets = [
        (tweet.retweet_count,
         tweet.retweeted_status.user.screen_name,
         tweet.text
        )
        for tweet in tweets
        if hasattr(tweet, 'retweeted_status')
    ]
    prettyPrint = PrettyTable(field_names=['Count', 'Screen', 'Text'])
    [prettyPrint.add_row(row) for row in sorted(retweets, reverse=True)[:5]]
    prettyPrint.max_width['Text'] = 50
    prettyPrint.align = 'l'
    print prettyPrint
```

Social Web: Twitter

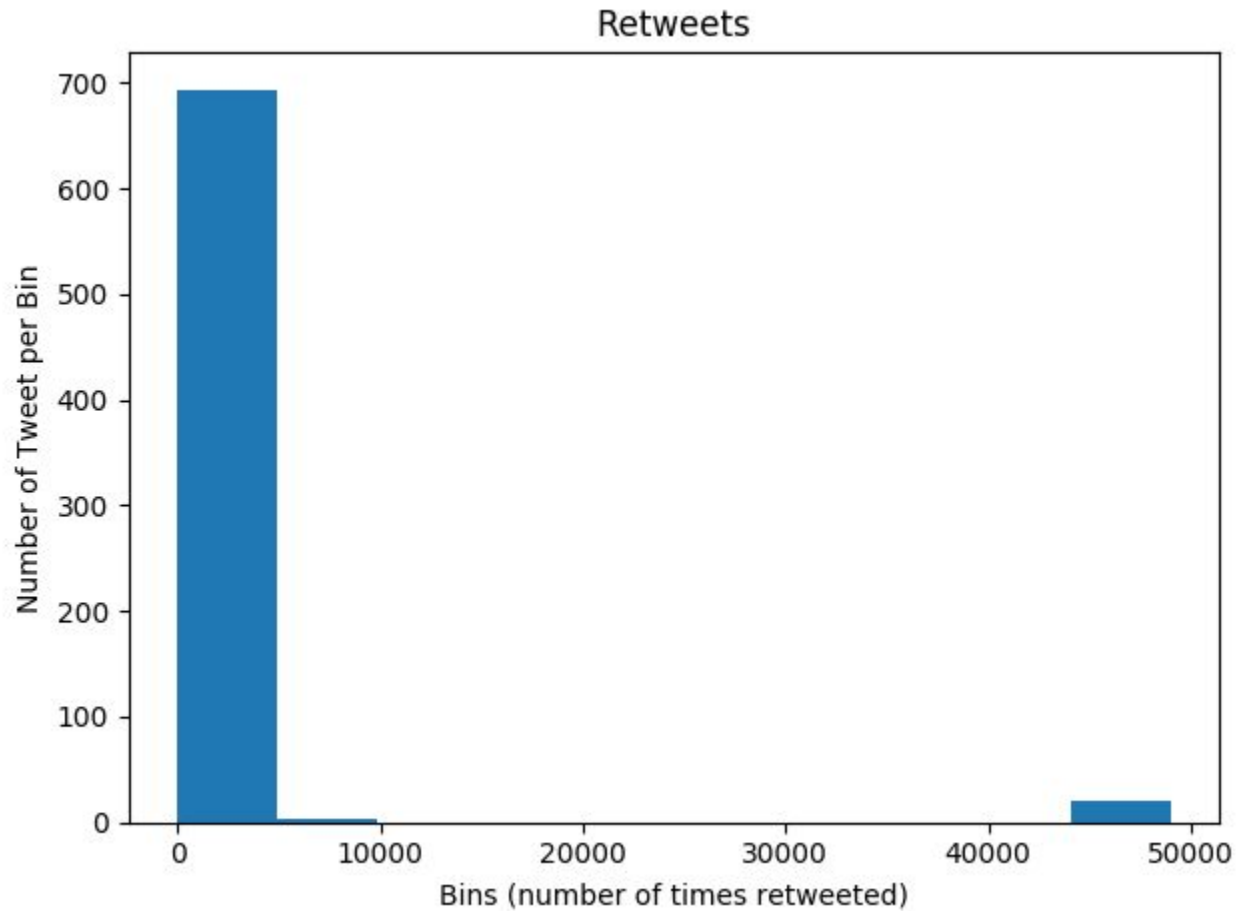
Count	Screen	Text
49062	acnewsitics	RT @acnewsitics: Donald Trump is: 1) "a billionaire" but you can't see his taxes 2) "a genius" but you can't see his grades 3) "exonerat...
4137	ProudResister	RT @ProudResister: We are all being gaslighted. Donald Trump and his campaign colluded with the Russian government. We watched it all unfol...
2805	DonaldJTrumpJr	RT @DonaldJTrumpJr: Donald Trump Jr.: It Became A Business Model For Media Organizations To Push Hoax Trump-Russia Narrative Video Real...
1661	EdKrassen	RT @EdKrassen: The Democratic party is more split right now than the Republican party, and that's a sad thing.
1482	kinguilfoyle	Bernie Sanders is a good pe... RT @kinguilfoyle: Donald Trump Jr.: It Became A Business Model For Media Organizations To Push Trump-Russia Narrative https://t.co/YNxv8cW...

Social Web: Twitter

Possiamo visualizzare l'istogramma dei retweet.

```
def retweetHistogram(query, count=10):
    tweets = tweepy.Cursor(twitterAPI.search, q=query).items(count)
    retweets = [
        (tweet.retweet_count,
         tweet.retweeted_status.user.screen_name,
         tweet.text
        )
        for tweet in tweets
        if hasattr(tweet, 'retweeted_status')
    ]
    counts = [count for coun, _, _ in retweets]
    plt.hist(counts)
    plt.title("Retweets")
    plt.xlabel("Bins (number of times retweeted)")
    plt.ylabel("Number of Tweet per Bin")
    plt.show()
```

Social Web: Twitter

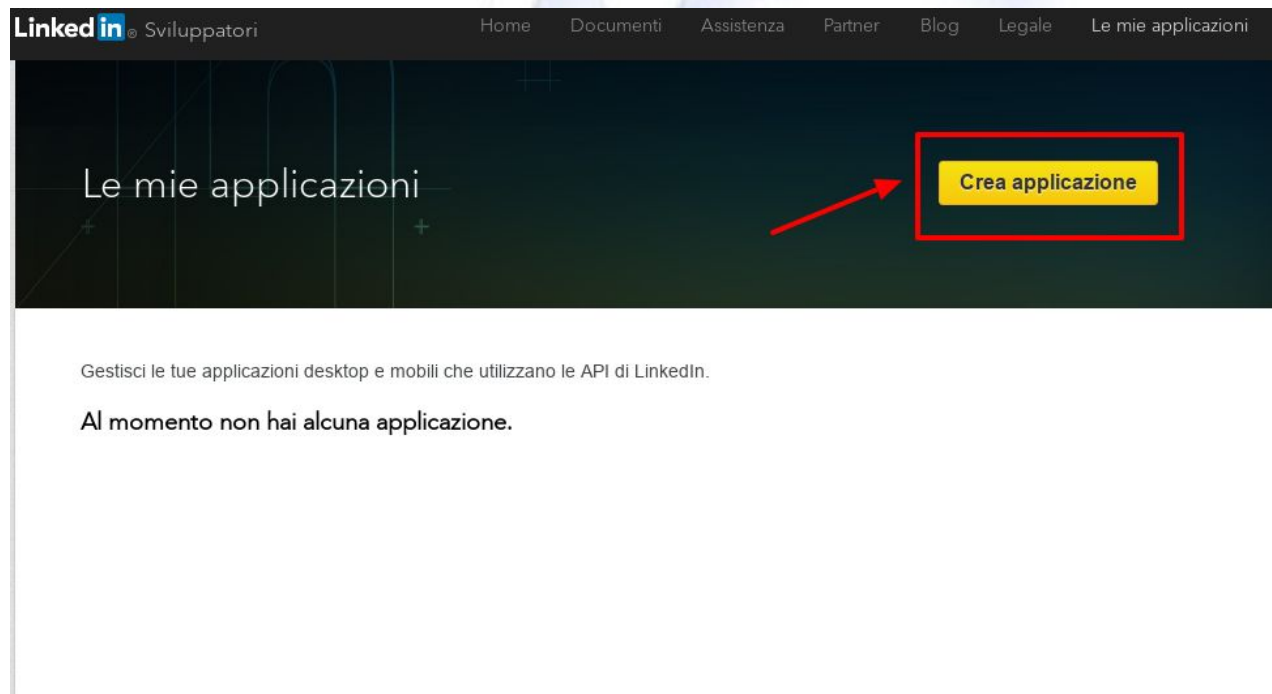


Social Web: LinkedIn


Il primo step per poter lavorare con LinkedIn è quello di ottenere delle credenziali per poter accedere alle API fornite.

Per ottenere delle credenziali è necessario registrarsi come sviluppatori e creare una “**app**” al seguente indirizzo:

<https://www.linkedin.com/developer/apps>



Social Web: LinkedIn

LinkedIn  Sviluppatori

Home Documenti Assistenza Partner Blog Legale Le mie applicazioni

Le mie applicazioni

[Crea applicazione](#)

Crea una nuova applicazione


Nome azienda:*

marco

Nome:*

Descrizione:*

Logo dell'applicazione:*

 [Seleziona file da caricare](#)

Social Web: LinkedIn

Utilizzo dell'applicazione:*

Indirizzo del sito Web:*

Privacy Policy URL:

Email aziendale:*

Telefono aziendale:*

Ho letto e accetto le[condizioni d'uso delle interfacce API di LinkedIn.](#)

*Inserire una URL fittizia:
<http://example.com>*

Social Web: LinkedIn

1. Dal repository <https://github.com/marcoortu/WAAT-2019> fare il checkout del branch **social**.
2. Il file **credential.py** contiene le credenziali per i diversi social media che analizzeremo, sostituire le credenziali appena ottenute nelle seguenti variabili.

Mettere qui i dati corretti per LinkedIn

LINKEDIN_CLIENT_KEY = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

LINKEDIN_CLIENT_SECRET = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

LINKEDIN_USERNAME = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

LINKEDIN_PASSWORD = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

Social Web: LinkedIn

Possiamo ora interagire con le API fornite da LinkedIn:

```
credential = getCredential()
auth = linkedin.LinkedInDeveloperAuthentication(LINKEDIN_CLIENT_KEY,
                                               LINKEDIN_CLIENT_SECRET,
                                               credential['oauth_token'],
                                               credential['oauth_token_secret'],
                                               "
                                               ",
                                               permissions=linkedin.PERMISSIONS.enums.values()
                                               )
app = linkedin.LinkedInApplication(auth)

if __name__ == '__main__':
    print app.get_profile()
```

Social Web: LinkedIn

Esportare i propri contatti in csv:

The screenshot shows the LinkedIn user interface. At the top, there is a navigation bar with icons for Home, Rete, Lavoro, Messaggistica, and Notifiche. Below this is a banner for the 'Mobile 360 Series' event. The main content area displays '160 collegamenti' and a list of contacts. A red box highlights the 'Gestisci i contatti importati' button, with a red arrow pointing to it. To the right, a sidebar contains a notification about contact importation and a 'Promosso' section with various advertisements.

160 collegamenti

Ordina per: Aggiunti di recente

Cerca per nome

Esegui la ricerca usando i filtri

Gestisci i contatti importati

L'importazione dei tuoi contatti è pronta

Collegati con i tuoi contatti per non perderti più di vista

Continua

Altre opzioni

Promosso

MIT online AI program

Register Now for MIT's 6 Week Online Program in Artificial Intelligence

Integrate HubSpot

IConduct - The easiest, no coding, integration for HubSpot. Live Demo!

Invito da LinkedIn

Pubblica annunci pubblicitari su LinkedIn. Acquisisci nuovi clienti per la tua azienda. Prova subito.

Informazioni Centro assistenza

Privacy e condizioni Pubblicità

Social Web: LinkedIn

Esportare i propri contatti in csv e salvare il file nella cartella data del progetto.

The screenshot displays the LinkedIn interface for managing imported contacts. The top navigation bar includes the LinkedIn logo, a search bar, and icons for Home, Rete, Lavoro, Messaggistica, Notifiche, and Tu. A 'Prova Premium gratis' button is also visible.

The main content area is titled 'Gestisci i tuoi contatti' and features a section for 'Importati'. Below this, a message states: 'Questi sono i contatti che hai importato tramite l'importazione della rubrica.' A checkbox labeled 'Seleziona tutto (1.075)' is checked. An 'Elimina selezionati' button is present.

The contact list includes the following entries:

Nome	Indirizzo email
Francesco Milla	francesco.milla@francesco.com
Antonio David	antonio.david@gmail.com
Martina Milla	martina.milla@martina.com
Cristian Milla	cristian.milla@cristian.com
Matteo Milla	matteo.milla@matteo.com
Giulia Milla	giulia.milla@giulia.com
Roberto Milla	roberto.milla@roberto.com
Matteo Milla	matteo.milla@matteo.com
Antonio Portugal	antoniojodavid@gmail.com

On the right sidebar, under 'Azioni avanzate', the 'Esporta contatti' option is highlighted with a red box and a red arrow.

Social Web: K-Means

Per ottenere una stima del K migliore da utilizzare come numero di clusters del K-Means si può utilizzare il così detto metodo *Elbow*.

Immaginiamo di ottenere una trasformazione TF-IDF data una lista di testi *textList*.

```
tfidfVectorizer = TfidfVectorizer(max_features=100,  
                                stop_words='english',  
                                use_idf=True,  
                                tokenizer=word_tokenize,  
                                ngram_range=(1, 2))  
tfidfMatrix = tfidfVectorizer.fit_transform(textList)
```

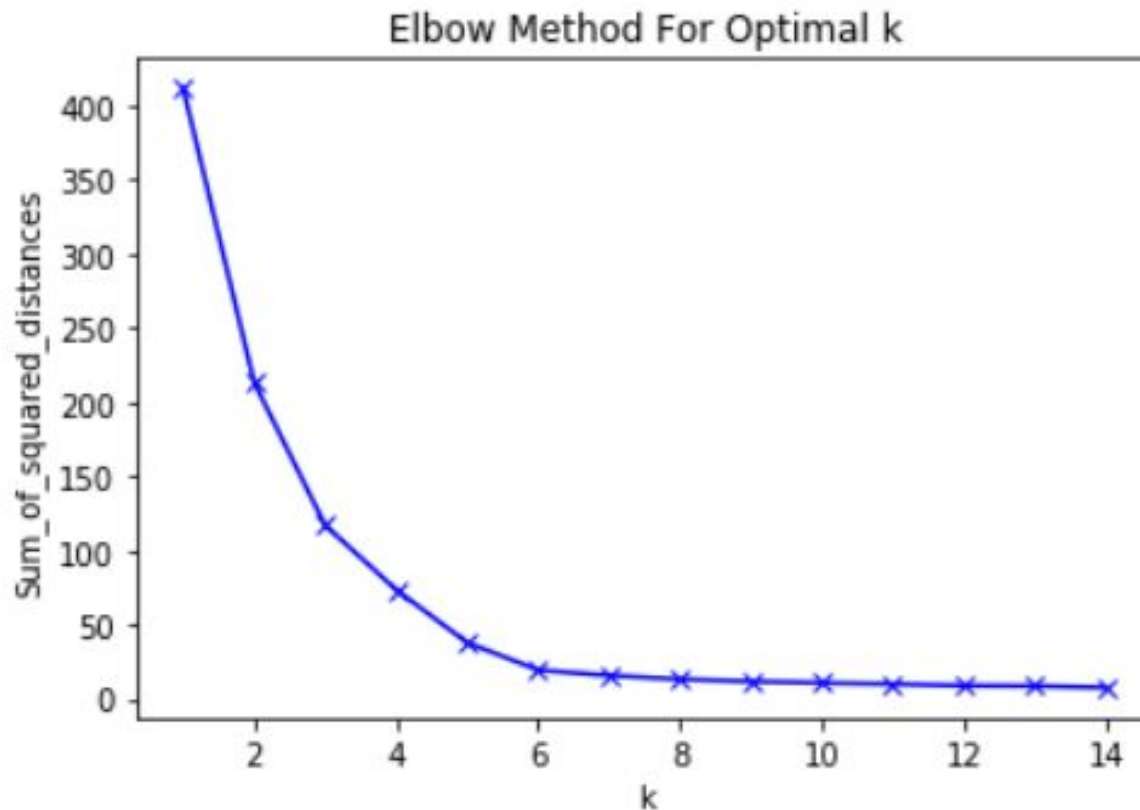
Social Web: K-Means

Definiamo una funzione che calcola la [somma dei quadrati delle distanza più vicine ad ogni cluster](#) ad ogni iterazione (parametro *inertia_* del kmeans).

```
def getBestKForKMeans(tfidfMatrix, maxClusters=15):
    sumOfSquaredDistances = []
    K = range(1, maxClusters)
    for k in K:
        km = KMeans(n_clusters=k)
        km = km.fit(tfidfMatrix)
        # inertia_: Sum of squared distances of samples to their closest cluster
        center.
        sumOfSquaredDistances.append(km.inertia_)
    print sumOfSquaredDistances
    plt.plot(K, sumOfSquaredDistances, 'bx-')
    plt.xlabel('k')
    plt.ylabel('Sum of Squared Distances')
    plt.title('Elbow Method For Optimal k')
    plt.show()
```

Social Web: K-Means

Si ottiene una curva al variare di K. Se la curva ha la forma di un gomito (Elbow appunto) allora il valore di K ottimale si trova in prossimità del gomito.



Social Web: LinkedIn

1. Dal repository <https://github.com/marcoortu/WAAT-2019> fare il checkout del branch **social**.
2. Scaricare l'archivio dei vostri dati su linkedin per svolgere l'esercizio 2.

Social Web: Facebook Lockdown

<https://newsroom.fb.com/news/2018/04/restricting-data-access/>

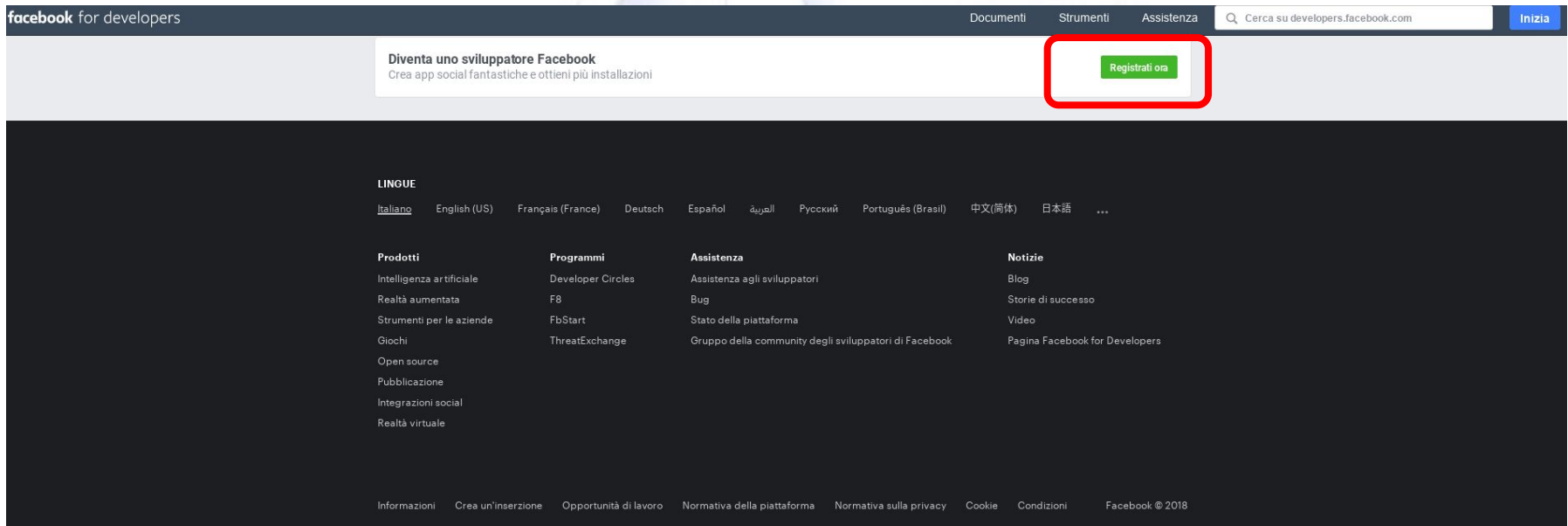


Social Web: Facebook

Il primo step per poter lavorare con Facebook è quello di ottenere delle credenziali per poter accedere alle API fornite.

Per ottenere delle credenziali è necessario registrarsi come sviluppatori e creare una “**app**” al seguente indirizzo:

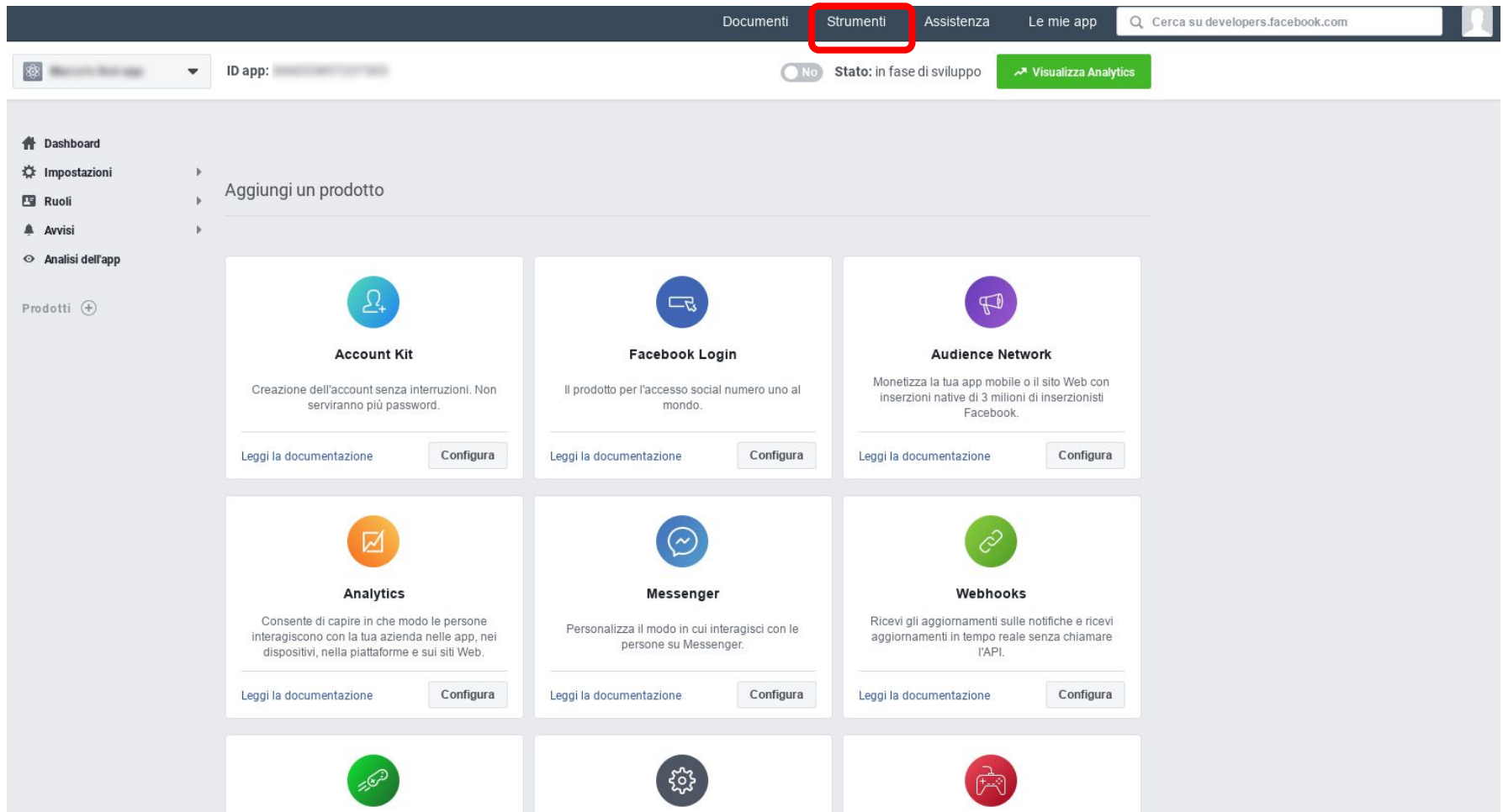
<https://developers.facebook.com/apps>



The screenshot shows the Facebook for developers website. The top navigation bar includes the text "facebook for developers" on the left, and "Documenti", "Strumenti", and "Assistenza" in the center. On the right, there is a search bar with the text "Cerca su developers.facebook.com" and an "Inizia" button. Below the navigation bar, a prominent banner reads "Diventa uno sviluppatore Facebook" with the subtext "Crea app social fantastiche e ottieni più installazioni". A green "Registrati ora" button is highlighted with a red rectangular box. The main content area is dark-themed and contains several sections: "LINGUE" with links for Italiano, English (US), Français (France), Deutsch, Español, العربية, Русский, Português (Brasil), 中文(简体), 日本語, and "..."; "Prodotti" with links for Intelligenza artificiale, Realtà aumentata, Strumenti per le aziende, Giochi, Open source, Pubblicazione, Integrazioni social, and Realtà virtuale; "Programmi" with links for Developer Circles, F8, FbStart, and ThreatExchange; "Assistenza" with links for Assistenza agli sviluppatori, Bug, Stato della piattaforma, and Gruppo della community degli sviluppatori di Facebook; and "Notizie" with links for Blog, Storie di successo, Video, and Pagina Facebook for Developers. At the bottom, there is a footer with links for "Informazioni", "Crea un'inserzione", "Opportunità di lavoro", "Normativa della piattaforma", "Normativa sulla privacy", "Cookie", "Condizioni", and "Facebook © 2018".

Social Web: Facebook

Dopo aver completato il wizard di installazione di arriva alla Dashboard delle applicazioni. Andare su Strumenti -> Tool di esplorazione per la API Graph



The screenshot displays the Facebook Developer Dashboard interface. At the top, a navigation bar includes 'Documenti', 'Strumenti' (highlighted with a red box), 'Assistenza', and 'Le mie app'. A search bar on the right contains the text 'Cerca su developers.facebook.com'. Below the navigation bar, the 'Strumenti' section is active, showing a 'No' toggle and the status 'Stato: in fase di sviluppo', along with a green 'Visualizza Analytics' button. The main content area is titled 'Aggiungi un prodotto' and features a grid of six product cards, each with an icon, title, description, and 'Configura' button. The cards are: Account Kit (blue icon), Facebook Login (blue icon), Audience Network (purple icon), Analytics (orange icon), Messenger (blue icon), and Webhooks (green icon). A left sidebar contains navigation links for Dashboard, Impostazioni, Ruoli, Avvisi, and Analisi dell'app, along with a 'Prodotti +' button.

Documenti **Strumenti** Assistenza Le mie app

ID app: [redacted] No Stato: in fase di sviluppo [Visualizza Analytics](#)

Dashboard
Impostazioni
Ruoli
Avvisi
Analisi dell'app

Prodotti +

Account Kit
Creazione dell'account senza interruzioni. Non serviranno più password.
[Leggi la documentazione](#) [Configura](#)

Facebook Login
Il prodotto per l'accesso social numero uno al mondo.
[Leggi la documentazione](#) [Configura](#)

Audience Network
Monetizza la tua app mobile o il sito Web con inserzioni native di 3 milioni di inserzionisti Facebook.
[Leggi la documentazione](#) [Configura](#)

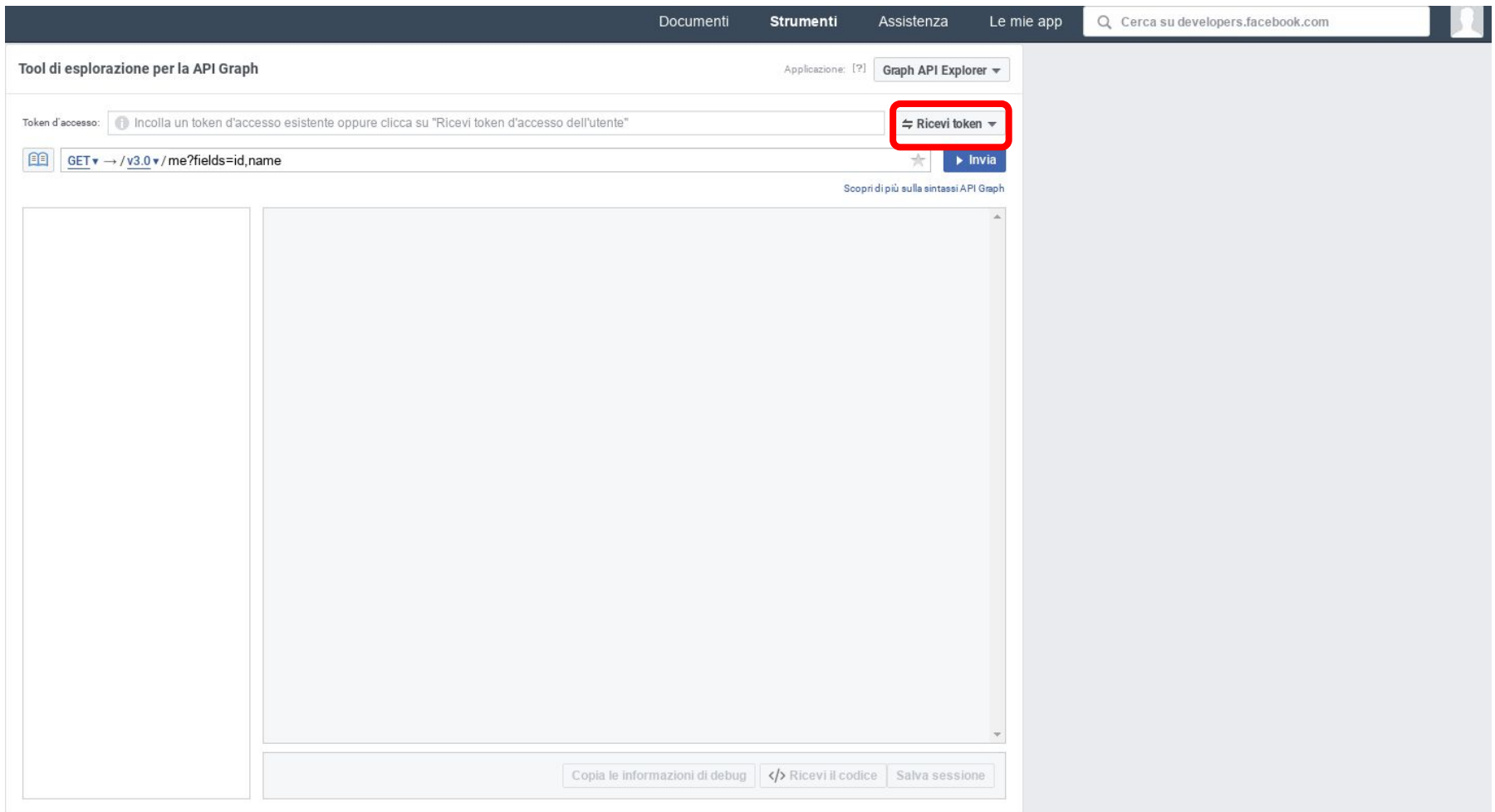
Analytics
Consente di capire in che modo le persone interagiscono con la tua azienda nelle app, nei dispositivi, nella piattaforme e sui siti Web.
[Leggi la documentazione](#) [Configura](#)

Messenger
Personalizza il modo in cui interagisci con le persone su Messenger.
[Leggi la documentazione](#) [Configura](#)

Webhooks
Ricevi gli aggiornamenti sulle notifiche e ricevi aggiornamenti in tempo reale senza chiamare l'API.
[Leggi la documentazione](#) [Configura](#)

Social Web: Facebook

Generare un token di accesso cliccando sul pulsante “Ricevi Token”.



The screenshot shows the Facebook Graph API Explorer interface. At the top, there is a navigation bar with links for "Documenti", "Strumenti", "Assistenza", and "Le mie app", along with a search bar for "Cerca su developers.facebook.com". The main area is titled "Tool di esplorazione per la API Graph" and includes a dropdown menu for "Applicazione: [?] Graph API Explorer". Below this, there is a text input field for "Token d'accesso:" with a placeholder message: "Incolla un token d'accesso esistente oppure clicca su 'Ricevi token d'accesso dell'utente'". A red box highlights the "Ricevi token" button next to this field. Below the input field, there is a URL input field containing "GET → /v3.0 /me?fields=id,name" and an "Invia" button. At the bottom of the interface, there are three buttons: "Copia le informazioni di debug", "Ricevi il codice", and "Salva sessione".

Social Web: Facebook

Selezionare i permessi di accesso della vostra applicazione.

<https://developers.facebook.com/docs/graph-api/using-graph-api/>

The screenshot shows the Facebook Graph API Explorer interface. A modal dialog titled "Seleziona autorizzazioni" (Select permissions) is open, displaying a list of permissions to be selected for the application. The permissions are organized into three categories:

- Autorizzazioni dati utente (User Data Permissions):**
 - email
 - user_age_range
 - user_birthday
 - user_friends
 - user_gender
 - user_hometown
 - user_likes
 - user_link
 - user_location
 - user_photos
 - user_posts
 - user_status
 - user_tagged_places
 - user_videos
- Eventi, gruppi e Pagine (Events, Groups, and Pages):**
 - ads_management
 - ads_read
 - business_management
 - groups_access_member_info
 - manage_pages
 - pages_manage_cta
 - pages_manage_instant_articles
 - pages_messaging
 - pages_messaging_payments
 - pages_messaging_phone_number
 - pages_messaging_subscriptions
 - pages_show_list
 - publish_pages
 - publish_to_groups
 - read_page_mailboxes
 - user_events
 - user_managed_groups
- Altro (Other):**
 - instagram_basic
 - instagram_manage_comments
 - instagram_manage_insights
 - read_audience_network_insights
 - read_insights

At the bottom of the dialog, there is a note: "Profilo pubblico incluso come impostazione predefinita" (Public profile included as default setting). Below this note are three buttons: "Ricevi token d'accesso" (Get access token), "Cancella" (Cancel), and "Annulla" (Dismiss).

Social Web: Facebook

Copiare l'access token ricevuto.

Documenti Strumenti Assistenza Le mie app Cerca su developers.facebook.com

Tool di esplorazione per la API Graph Applicazione: [?] Graph API Explorer

Token d'accesso: [redacted] Ricevi token

GET → /v3.0 /me?fields=id,name Invia

Scopri di più sulla sintassi API Graph

Copia le informazioni di debug </> Ricevi il codice Salva sessione

Social Web: Facebook

1. Dal repository <https://github.com/marcoortu/WAAT-2019> fare il checkout del branch **social**.
2. Il file **credential.py** contiene le credenziali per i diversi social media che analizzeremo, sostituire le credenziali appena ottenute nelle seguenti variabili.

Mettere qui i dati corretti per Facebook

FACEBOOK_ACCESS_TOKEN = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'

Social Web: Facebook

Con l'access token ottenuto precedentemente possiamo creare il nostro client per utilizzare le Graph API di Facebook.

```
import facebook
```

```
from credentials import FACEBOOK_ACCESS_TOKEN
```

```
graph = facebook.GraphAPI(access_token=FACEBOOK_ACCESS_TOKEN)
```

```
if __name__ == '__main__':  
    print graph
```

Social Web: Facebook

Il metodo search permette di ricercare vari tipi di oggetti su facebook.

Search for places near 1 Hacker Way in Menlo Park, California.

```
places = graph.search(type='place',  
                      center='37.4845306,-122.1498183',  
                      fields='name,location')
```

Each given id maps to an object the contains the requested fields.

```
for place in places['data']:  
    print('%s %s' % (place['name'].encode('utf8'), place['location'].get('zip')))
```

Social Web: Facebook

Creiamo una funzione di supporto che permette di serializzare gli oggetti in python e visualizzarli in formato JSON (Javascript Object Notation).

```
def serialize(object):  
    return json.dumps(object, indent=1)
```

Possiamo utilizzare questa funzione per stampare il contenuto dei vari oggetti.

```
print(serialize(graph.get_object('me')))
```

```
{  
  "name": "Marco Ortu",  
  "id": "XXXXXXXXXXXXXXXXXX"  
}
```

Social Web: Facebook

```
print(serialize(graph.get_connections('me', "friends")))
{
  "data": [],
  "summary": {
    "total_count": 0
  }
}
```

Estraiamo i post dalla pagina personale di Bill Gates:

```
user = 'BillGates'
profile = graph.get_object(user)
posts = graph.get_connections(profile['id'], 'posts')
print serialize(posts)
```

Social Web: Facebook

```
{
  "paging": {
    "next":
      "https://graph.facebook.com/v2.6/216311481960/posts?access_token=EAACEdEose0cBAKuLoNGTmFaF9aVAY7ZAEGSMZCfrLfCQ4NVMuyskbGUnPHZBvr8goTCZCkPRxe6AISybaoD4ZBFiCwaHaMfqYXhBC05PEV8rG0xdQ8tVa3OgGZAZBGB9AahaAxEt747fa3KRogH2fpSAmslOdZC5SxSswPolERHPhb7hYuj2GKFbPZAMmxf4QccPtfkKAs5RZAMPycIVzTs2&limit=25&until=1523383499&__paging_token=enc_AdBJi9PnCZA2BSN8ZCwtjocpY7kpCIRkEN4xZCnOF9WjLV3pp5giObH1mP7WtRFZAbOtKAfLdpE0mhr3nhNUquPY3YgpnY1ilCAyu609Om4ySrrvQZDZD",
    "previous":
      "https://graph.facebook.com/v2.6/216311481960/posts?since=1526385603&access_token=EAACEdEose0cBAKuLoNGTmFaF9aVAY7ZAEGSMZCfrLfCQ4NVMuyskbGUnPHZBvr8goTCZCkPRxe6AISybaoD4ZBFiCwaHaMfqYXhBC05PEV8rG0xdQ8tVa3OgGZAZBGB9AahaAxEt747fa3KRogH2fpSAmslOdZC5SxSswPolERHPhb7hYuj2GKFbPZAMmxf4QccPtfkKAs5RZAMPycIVzTs2&limit=25&__paging_token=enc_AdBlJoYhZCoOoDRhIQSxnqrAkFLHzZCdjclF20tgcumhWJZBuvIMowobR7e1IRuL0ZB8Cyo5aaQ9hY3SIncN5gClgriEJkHw7vMEmUMXKQD0E6UQEgZDZD&__previous=1"
  },
  "data": [
    {
      "created_time": "2018-05-15T12:00:03+0000",
      "message": "It\u2019s always thrilling to be back on campus \u2014 2014 and it makes me wish I could be a student all over again. Here are some of the things I learned from my visit to University of Central Florida: https://b-gat.es/2L09Crm",
      "id": "216311481960_10155451558261961"
    }
  ]
}
```

Social Web: Facebook

Utilizziamo ora la paginazione per ottenere tutti i post di una determinata pagina.

```
user = 'DonaldTrump'  
profile = graph.get_object(user)  
posts = graph.get_connections(profile['id'], 'posts')  
while True:  
    try:  
        for post in posts['data']:  
            print serialize(post)  
        posts = requests.get(posts['paging']['next']).json()  
    except KeyError:  
        break
```

Social Web: Facebook

```
{  
  "created_time": "2018-05-15T17:29:07+0000",  
  "message": "Trump Just Had Every Single One Of Them Arrested! The ENTIRE  
Democratic Party Is FURIOUS!",  
  "id": "780469468772262_1142542515898287"  
}  
{  
  "created_time": "2018-05-15T17:01:04+0000",  
  "message": "BAM! Trump isn't playing around!",  
  "story": "President Donald J Trump shared a post.",  
  "id": "780469468772262_1142524029233469"  
}  
{  
  "created_time": "2018-05-15T16:10:34+0000",  
  "message": "SICK! Scumbag Author Stephen King Thinks Melania Lied, Knows  
Why She Was Really Hospitalized",  
  "id": "780469468772262_1142495322569673"  
}
```

Social Web: Facebook

Esercizio 2

Eseguire una analisi comparativa del sentiment associato a due pagine Facebook comparabili (Pepsi e Coca Cola per esempio).

1. Estrarre 100 post dalle pagine Facebook.
2. Eseguire la sentiment analysis utilizzando la libreria ***TextBlob***.
3. Riportare i risultati con ***prettyTable*** stampando:
 - a. sentiment medio
 - b. percentuale negative/neutral/positive
4. Visualizzare il boxplot dei sentiment delle sue pagine.
5. Visualizzare la variazione temporale del sentiment delle due pagine.

Social Web: Facebook

Esercizio 3

Eseguire una analisi comparativa del sentiment associato a due pagine Facebook comparabili.

1. Estrarre tutti i post dalle pagine Facebook di entrambe le pagine a partire da uno stesso intervallo di tempo (esempio un mese).
2. Eseguire la sentiment analysis utilizzando la libreria ***TextBlog***.
3. Visualizzare il grafico della variazione temporale del sentiment delle due pagine.
4. Visualizzare il grafico del sentiment medio giornaliero.

Referenze

- Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More (ISBN-13: 978-1449367619)
- Best K for Kmeans:
<https://blog.cambridgespark.com/how-to-determine-the-optimal-number-of-clusters-for-k-means-clustering-14f27070048f>
- <http://facebook-sdk.readthedocs.io/en/latest/>
 - ◆ capitoli dal 1 al 3