
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
****
Université de Sousse
****
Institut Supérieur d'Informatique et des Technologies de Communication de Hammam Sousse
Réalisé par :
Nidhal Hazbri
3DNI G2
project Data Mining
Text Classification
Tokenization
Word2Vec
K-nearest Neighbor
RMDL
A Survey
HDLTex
Deep Learning
Random Forest

In [ ]:

In [ ]:

Objectifs :

- Maitriser l'API de twitter pour l'extraction des tweets
- Maitriser la partie NLP (natural language processing) avec NLTK en Python
- Appliquer les principes de nettoyage des données
- Classer les tweets : regrouper ensemble les tweets qui sont similaires. C'est une étape qui peutêtre considérée comme une étape

Specifications

Imaginons que vous avez un compte Twitter, et que vous lez suivre les tweets sur ce reseau social. Vu le nombre colossal de Tweets, et faute de temps, vous n'avez pas la possibilite de les lire tous. Pour cela, vous avez besoin d'une application qui va jouer le role d'assistantet qui va vous effectuer un resume de toutes ces informations. Une des approches qu'on peut utiliser estde le classer sous former de groupes de sorte a ce qu'on presente a l'utilisateur un seul Tweet de chaque groupe. Pour cela, on doit proceder en trois grandes etapes :

Travail faire

On a Telecharger les tweets a partir de Twitter en utilisant l'API de twitter. Pour cela, vous devriez un compte « Twitter Developper ». Pour cela, vous devriez telecharger au moins 10 mille twwets. Pour la documentation de l'API de twitter, vous pouvez consulter les liens suivants :

In [1]:
```
import pandas as pd
import tweepy
consumer_key="LHZVzcEN30hfmN2cPBqkoB3wq"
consumer_secret="DGZ7gQFDlqXoPfmAUWHOsY2eMTA0qhgKVb3rbExcx8Vhav3x3a"
access_token="1325046107437752325-a2zNm36NnzJqTFBFkIagjzpkdCadjs"
access_token_secret="7ohQJ7WTf2DuHsr9NNwPkOPXq5zUkaycrzo2nPhPUoGLL"
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)
```

In [2]:
```
twitter_data_analysis = pd.DataFrame(columns = ['text'])
i=0
```

In [3]:
```
tweets = tweepy.Cursor(api. user_timeline , id="twitter").items( 15000)
# Iterate and print tweets
for tweet in tweets:
    twitter_data_analysis.loc[i,"text"] = tweet.text
    i+=1
```

In [4]:
```
print(twitter_data_analysis.shape)
print(twitter_data_analysis)
```

```
(3227, 1)
                                                   text
0     We’re switching back. You can now choose to Re...
1     RT @angnickelodeon: twitter users with 150-350...
2     @yaryoush_ @BeshoyMaximus1 the way you both won 🤍
3     There’s more! We’ll also be testing sharing Tw...
4     Oh snap! 🤭\n\nSharing Tweets directly to your ...
...                                                 ...
3222  @TheMegaBoi2004 Keeping your brain thinking ar...
3223  @GuillaumeTC @HamillHimself @ChrisEvans For th...
3224             @KSJIZE Hi @dog_rates you've got a fan
3225              @insomniacookies cc: @MeCookieMonster
3226  @MNoir1211 &amp;amp; you're guaranteed a good morn...

[3227 rows x 1 columns]
```

In [5]:
```
tweets = tweepy.Cursor(api. user_timeline , id="twitter").items( 15000)
 #Iterate and print tweets
for tweet in tweets:
    twitter_data_analysis.loc[i,"text"] = tweet.text
    i+=1
```

In [6]:
```
print(twitter_data_analysis.shape)
```

```
(6454, 1)
```

In [7]:
```
tweets = tweepy.Cursor(api. user_timeline , id="twitter").items( 15000)
 #Iterate and print tweets
for tweet in tweets:
    twitter_data_analysis.loc[i,"text"] = tweet.text
    i+=1
```

In [8]:
```
print(twitter_data_analysis.shape)
```

```
(9681, 1)
```

In [9]:
```
import csv
twitter_data_analysis.to_csv('twitter_data_analysis.csv',index = False)
twitter_data_analysis.head(10)
```

Out[9]:

	text
0	We’re switching back. You can now choose to Re...
1	RT @angnickelodeon: twitter users with 150-350...
2	@yaryoush_ @BeshoyMaximus1 the way you both won 🤍
3	There’s more! We’ll also be testing sharing Tw...
4	Oh snap! 🤩\n\nSharing Tweets directly to your ...
5	@levantinepali a stamp of approval https://t.c...
6	2020 in one word
7	@Astro_AJC this is what cuffing season means t...
8	@un3asyy 2 is also cute
9	@DeePeeArts you're all amazing

In [10]:
```
twitter_data_analysis.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9681 entries, 0 to 9680
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    9681 non-null   object
dtypes: object(1)
memory usage: 471.3+ KB
```

Pretraitement des tweets

Dans cette etape, l'objectif est d'eliminer le texte inutile des tweets tels que les #, les noms des utilisateurs, les url,emoji ...

In [11]:
```
import re
numbers = r'(?:(?:\d+,?)+(?:\.?\d+)?)'
URL = r'http[s]?://(?:[a-z]|[0-9]|[$-_@.&+]|[!*\(\),]|(?:%[0-9a-f][0-9a-f]))+'
html_tag = r'<[^>]+>'
hash_tag = r"(?:\#+[\w_]+[\w\'_\-]*[\w_]+)"
at_sign = r'(?:@[\w_]+)'
start_quest_pound = r"(?:^|\s)([#?])(\w+)" # Start with ? or with #
cont_number = r'(\w*\d\w*)' # Words containing numbers

short_words = r'\W*\b\w{1,3}\b' # Short words of 3 character or less

emoji_pattern = re.compile("["
                           u"\U0001F600-\U0001F64F"  # emoticons
                           u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                           u"\U0001F680-\U0001F6FF"  # transport & map symbol
                           u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                           u"\U00002500-\U00002BEF"  # chinese char
                           u"\U00002702-\U000027B0"
                           u"\U00002702-\U000027B0"
                           u"\U000024C2-\U0001F251"
                           u"\U0001f926-\U0001f937"
                           u"\U00010000-\U0010ffff"
                           u"\u2640-\u2642"
                           u"\u2600-\u2B55"
                           u"\u200d"
                           u"\u23cf"
                           u"\u23e9"
                           u"\u231a"
                           u"\ufe0f"  # dingbats
                           u"\u3030"
                           "]+", flags=re.UNICODE)



word_garb = ['here', 'there', 'where', 'when', 'would', 'should', 'could','thats'
             'thank', 'https', 'since', 'wanna', 'gonna', 'aint', 'http', 'unto',
             'dont', 'done', 'cant', 'werent', 'https', 'u', 'isnt', 'go', 'theyr
            'weve', 'theyve','that','this','thery','That','then','yor','what']

bad_chars = {';', ':', '!', "*",".","?",",",")","(","'","_"}
```

In [76]:
```
for index, row in twitter_data_analysis.iterrows():
    err = row['text']
    new0 = re.sub(URL, "", err)
    new1 = re.sub(hash_tag, "", new0)
    new2 = re.sub(at_sign, "", new1)
    new3 = re.sub(r"\n+", "", new2)
    new4 = re.sub(r"RT+", "", new3)
    new5 = re.sub("hhh+", '', new4)
    new6 = re.sub(numbers, "", new5)
    new7 = re.sub(html_tag, "", new6)
    new8 = re.sub(r'\W*\b\w{1,3}\b', "", new7)
    new9 = re.sub(emoji_pattern, "", new8)
    new10 = re.sub(short_words, "", new9)
    twitter_data_analysis.loc[index,'text'] =new10



twitter_data_analysis['text'] = twitter_data_analysis['text'].str.replace('|'.joi
twitter_data_analysis['text'] = twitter_data_analysis['text'].str.replace('|'.joi
```

In [77]:
```
twitter_data_analysis.head(40)
```

Out[77]:

	text
0	switching back choose Retweet Qote Tweet before
1	twitter sers with - followers backbone society
2	both
3	more also testing sharing Tweets Stories smal...
4	snap Sharing Tweets directly Snapchat Stories...
5	stamp approval
6	word
7	cffing season means
8	also
9	amazing
10	twitter before
11	
12	drink water replaced morning
13	taking oomf Fleets
14	remember dedicate Tweet"
15	they torists
16	proof doing right
17	some hating Fleeting
18	thing didn Tweet wanted didn close were like...
19	
20	aren feet
21	Tweet gradated with honors
22	love wait wedding pics
23	
24	breathe
25	apology accepted
26	
27	THIRSTY
28	looking hydrated
29	moon will share
30	bark among stars
31	rbber dcky knew along
32	moon hydrate
33	Reading article before Retweeting growthBefor...

	text
34	made temporary change Retweet fnction When Re...
35	seeing Twitter friends never person scceed
36	
37	dedication
38	single person
39	Tweet

In [78]:
```
twitter_data_analysis.to_csv('cleaning_twitter_data_analysis.csv',index = False)
```

In [79]:
```
import nltk
nltk.download('stopwords' )
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\nidhal\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[79]:
```
True
```

In [80]:
```
import matplotlib.pyplot as plt
import nltk
import matplotlib
from wordcloud import WordCloud

txt = twitter_data_analysis.text.str.lower().str.replace(r'\|', ' ').str.cat(sep=
words = nltk.tokenize.word_tokenize(txt)
word_dist = nltk.FreqDist(words)
wc = WordCloud(width=800, height=400, max_words=100,background_color = 'lightcyar
plt.style.use('fivethirtyeight')
plt.figure(figsize=(12,30))
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
#plt.savefig('d:/temp/result.png')
```

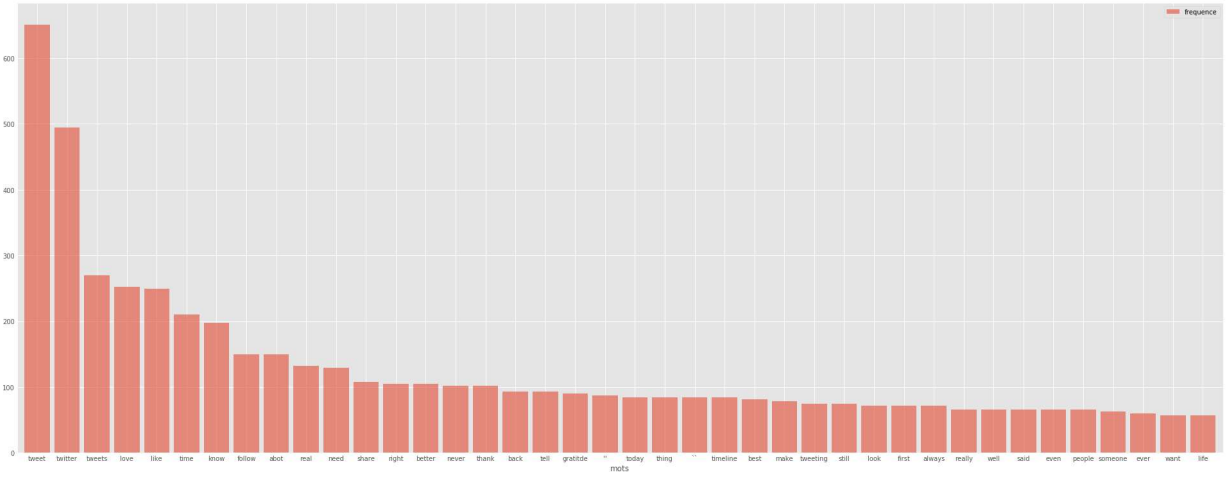
Out[80]: (-0.5, 799.5, 399.5, -0.5)

In [81]:
```
top_N = 40


stopwords = nltk.corpus.stopwords.words('english')
words_except_stop_dist = nltk.FreqDist(w for w in words if w not in stopwords)

print('All frequencies, including STOPWORDS:')
print('=' * 60)
rslt = pd.DataFrame(word_dist.most_common(top_N),
                    columns=['mots', 'frequence'])
print(rslt)
print('=' * 60)

rslt = pd.DataFrame(words_except_stop_dist.most_common(top_N),
                    columns=['mots', 'frequence']).set_index('mots')

matplotlib.style.use('ggplot')

rslt.plot.bar(rot=0,width=0.85, alpha=0.6, figsize=(30,12))
```

```
All frequencies, including STOPWORDS:
============================================================
        mots  frequence
0      tweet        651
1    twitter        495
2     tweets        270
3       love        252
4       like        249
5       have        216
6       time        210
7       know        198
8       they        195
9       with        192
10    follow        150
11      abot        150
12      real        132
13      need        129
14      more        126
15      from        120
16      this        114
17     share        108
18     right        105
19    better        105
20     never        102
21     thank        102
22      back         93
23      tell         93
24  gratitde         90
25        ''         87
26     today         84
27     thing         84
28        ``         84
29  timeline         84
30      what         84
31      will         81
```

```
32       only         81
33       best         81
34       make         78
35       were         75
36  tweeting          75
37      still         75
38       look         72
39      first         72
============================================================
```

Out[81]: <matplotlib.axes._subplots.AxesSubplot at 0x25f5ccd7648>



Traitement des tweets: NLP (Natural LanguageProcessing)

On doit proceder a !'analyse du tweet en respectant les differentes etapes du NLP (Natural LanguageProcessing). La bibliotheque a utiliser est NLTK en Python.

In [64]:
```
from nltk.stem.porter import PorterStemmer
from nltk.corpus import stopwords
ps = PorterStemmer()
stemed_dataset=[]
for i in range(0,twitter_data_analysis.shape[0]):
    stemmed_array=twitter_data_analysis['text'][i].split()
    stemmed=[ps.stem(word) for word in stemmed_array if not word in set(stopwords
    stemmed=' '.join(stemmed)
    stemed_dataset.append(stemmed)

print(stopwords.words('english'))
print("--"*60)
print(stemed_dataset[0:10])
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
"you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "i
t's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what',
'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'i
s', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'havin
g', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'a
gainst', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'b
elow', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under',
'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'h
ow', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'suc
h', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very',
's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now',
'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "could
n't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't",
'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "must
n't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wa
sn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
--------------------------------------------------------------------------------
----------------------------------------
['switch back choos retweet qote tweet', 'twitter ser - follow backbon societ
i', '', 'T also test share tweet stori small keep', 'snap share tweet directli
yor snapchat stori easier ever roll today', 'stamp approv', 'word', 'cffing sea
son mean', 'also cte', 'amaz']
```

In [55]:
```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
X=cv.fit_transform(stemed_dataset)
#print(cv.get_feature_names())
print(X)
pd.DataFrame(X.toarray(), columns=cv.get_feature_names())
```

```
  (0, 1996)     1
  (0, 120)      1
  (0, 338)      1
  (0, 1663)     1
  (0, 1581)     1
  (0, 2163)     1
  (1, 2171)     1
  (1, 1772)     1
  (1, 763)      1
  (1, 121)      1
  (1, 1871)     1
  (3, 2163)     1
  (3, 60)       1
  (3, 2038)     1
  (3, 1787)     1
  (3, 1970)     1
  (3, 1849)     1
  (3, 1056)     1
  (4, 2163)     1
  (4, 1787)     1
  (4, 1970)     1
  (4, 1860)     1
  (4, 543)      1
  (4, 2333)     1
  (4, 1861)     1
  :     :
  (9670, 2163)  1
  (9670, 541)   1
  (9671, 1129)  1
  (9671, 2072)  1
  (9671, 1086)  1
  (9672, 2063)  1
  (9672, 1554)  1
  (9672, 1159)  1
  (9673, 2171)  1
  (9673, 1347)  1
  (9675, 120)   1
  (9675, 386)   1
  (9676, 1056)  1
  (9676, 2333)  1
  (9676, 238)   1
  (9676, 2056)  1
  (9676, 95)    1
  (9677, 51)    1
  (9677, 2272)  1
  (9677, 385)   1
  (9680, 2163)  1
  (9680, 1370)  2
  (9680, 1280)  1
```

```
(9680, 1335)  1
(9680, 804)   1
```

Out[55]:

	abbot	abl	abort	abot	absolt	accent	accept	access	accidentally	accomplish	...	york
0	0	0	0	0	0	0	0	0	0	0	...	0
1	0	0	0	0	0	0	0	0	0	0	...	0
2	0	0	0	0	0	0	0	0	0	0	...	0
3	0	0	0	0	0	0	0	0	0	0	...	0
4	0	0	0	0	0	0	0	0	0	0	...	0
...	...	...	...	...	...	...	...	...	...	...	...	...
9676	0	0	0	0	0	0	0	0	0	0	...	0
9677	0	0	0	0	0	0	0	0	0	0	...	0
9678	0	0	0	0	0	0	0	0	0	0	...	0
9679	0	0	0	0	0	0	0	0	0	0	...	0
9680	0	0	0	0	0	0	0	0	0	0	...	0

9681 rows × 2344 columns

Classification des tweets

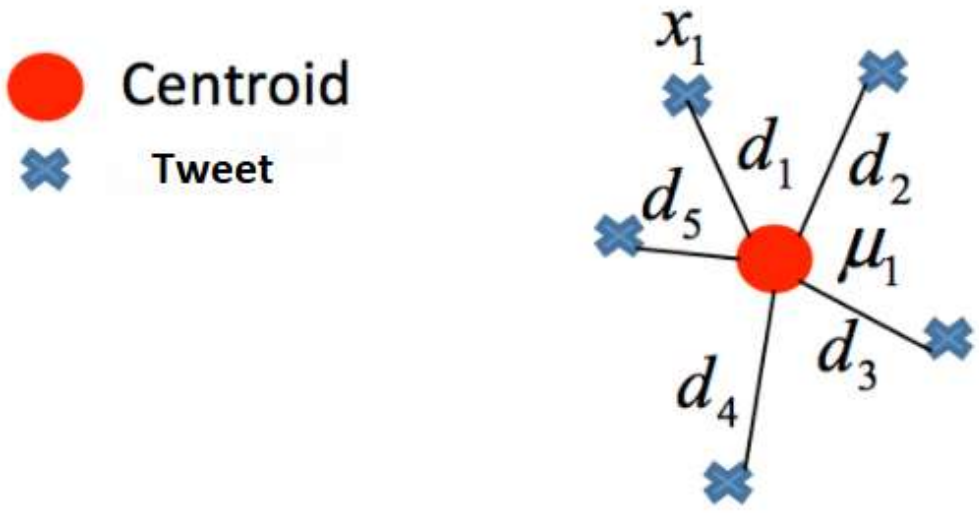
Etant donne un ensemble de tweets, l'objectif est de les resumer sous formes de groupes de sorte a ce que les Tweets qui sont dans le meme groupe soient similaires. Ainsi, l'utilisateur pourra par la suite lire juste un Tweet de chaque groupe (le Tweet qui est le centro"ide de groupes). on a Utiliser l'algorithme K-Means pour classer les Tweets en k classes ,valeurs de k allant de 1 a30 par exemple).

In [20]:
```
from sklearn.cluster import KMeans
wcss=[]
```

In [21]:
```
for i in range(1,30):
    Kmeans=KMeans(n_clusters=i,init='k-means++',max_iter=300,n_init=10,random_sta
    Kmeans.fit(X)
    wcss.append(Kmeans.inertia_)
```

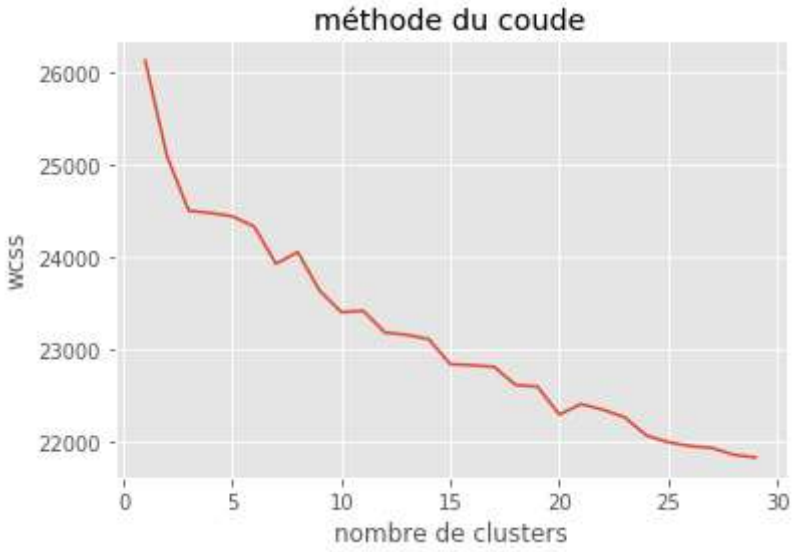
```
Initialization complete
Iteration  0, inertia 64968.000
Iteration  1, inertia 26127.983
Converged at iteration 1: center shift 0.000000e+00 within tolerance 1.151405
e-07
Initialization complete
Iteration  0, inertia 36105.000
Iteration  1, inertia 26127.983
Converged at iteration 1: center shift 0.000000e+00 within tolerance 1.151405
e-07
Initialization complete
Iteration  0, inertia 36105.000
Iteration  1, inertia 26127.983
Converged at iteration 1: center shift 0.000000e+00 within tolerance 1.151405
e-07
Initialization complete
Iteration  0, inertia 45780.000
Iteration  1, inertia 26127.983
Converged at iteration 1: center shift 0.000000e+00 within tolerance 1.151405
```

Within-Cluster-Sum-of-Squares (WCSS)



$$\text{wcss} = \sum_{x_j \in S_1} d_j^2 = d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2$$

In [22]:
```
import matplotlib.pyplot as plt
plt.plot(range(1,30),wcss)
plt.title('méthode du coude')
plt.xlabel('nombre de clusters')
plt.ylabel('wcss')
plt.show()
```



In [23]:
```
import numpy as np
true_k=30
Kmeans=KMeans(n_clusters=true_k,init='k-means++',n_init=1)
Kmeans.fit(X)
centroids = Kmeans.cluster_centers_
kmeans_labels = Kmeans.labels_
print ('\nCluster labels')
print(kmeans_labels)
print ('\n Cluster Centroids')
print (centroids)
```

```
Cluster labels
[17  0  0 ...  0  0 26]

 Cluster Centroids
[[0.00079777 0.00039888 0.00039888 ... 0.00039888 0.00039888 0.00039888]
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 ...
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]]
```

La cellule suivante contient les mots cles de chaque cluster

In [51]:
```
def print_top_words(model, feature_names, n_top_words):
    for topic_idx, topic in enumerate(order_centroids):
        print("cluster #%d:" % topic_idx)
        print(" ".join([feature_names[i]
                        for i in topic.argsort()[- n_top_words - 1:][::-1]]))
        print("\n")
    print()
```

In [69]:
```
tf_feature_names = cv.get_feature_names()
print_top_words(Kmeans, tf_feature_names, 20)
```

```
cluster #0:
nderstand soft readi reader read thanksgiv reaction react bottl flirt third t
hirsti reac rdolph rdd rbber aditorim astronom razzl takeseveryon rate


cluster #1:
ad lot proof propos prose protect protein prove promis prrfect psh ptting qad
rpl qalifi qaliti qeen pset profil profici professor prevent


cluster #2:
bro sre rearrang realli realiti real readi reader read reaction abot react re
ac rdolph rdd rbber razzl ball raven rate rat


cluster #3:
month thirteen replac repeat renew renesme remov remix ankl remind abbot reme
mb remast remain reliabl reli releas cermak relax reject recip
```

On a choisir un Tweet par classe comme representant. Les tweets choisis seront les resumes de toutes les informations contenues dans les tweets.

In [25]:
```
result_final_twitter_data_analysis = pd.DataFrame(columns = [ 'text'])
i=0
j=0
while i<30:
    while True:
        Y=cv.transform([stemed_dataset[j]])
        prediction=Kmeans.predict(Y)
        if i == prediction:
            print("tweet of cluster "+str(prediction)+twitter_data_analysis.loc[j
            result_final_twitter_data_analysis.loc[i,"text"] =twitter_data_analys
            j=0
            break
        j+=1
    i+=1
result_final_twitter_data_analysis.to_csv('result_final_twitter_data_analysis.csv
```

```
tweet of cluster [0]  twitter sers with - followers backbone society
tweet of cluster [1]  Tweet even load  lcky gess
tweet of cluster [2] thing didn Tweet wanted didn close  were like have place…
tweet of cluster [3] jst make sre yor open
tweet of cluster [4] voices conversations yor Timeline sing Lists make List dis
cover Lists…
tweet of cluster [5] Thank mch  love share gratitde Follow
tweet of cluster [6] ing twitter headqarters will replacing one mst risk…
tweet of cluster [7] mtals short mtal weirdness
tweet of cluster [8] jst said wants help make twitter accont wants follow
tweet of cluster [9] bother bother bother bother bother bother bother
tweet of cluster [10] streaming back door from yor stream back door
tweet of cluster [11] jst tta mte  mte away
tweet of cluster [12] miss Tweets abot yor Tweet Retweets with comments place
tweet of cluster [13] Mte "boyfriend girlfriend partner anniversary proposed en
gaged hsband wife wedding" forever
tweet of cluster [14] thoght abot dogs leads another thoght abot dogs leads ano
ther thoght abot dogs leads another thoght abot dogs
tweet of cluster [15]  anytime meet someone  says follow Twitter heart stops fl
l beat like  diary please…
tweet of cluster [16] Tweet Tweet Tweet yor voiceRolling today record Tweet…
tweet of cluster [17] switching back choose Retweet Qote Tweet before
tweet of cluster [18] testing icons instead labels within replies Check know  t
hink
tweet of cluster [19] people  yor twitter light mode
tweet of cluster [20] cher voice COULD SPEED TIME COULD FIND
tweet of cluster [21] never thoght  happen yassss happened never esperredittt
tweet of cluster [22] second thoght seven months long time kinda jst keep toch
tweet of cluster [23] October thReply need spooky nameLike yor Twitter name alr
eady ghastly respect revere candy corn
tweet of cluster [24]  Black History Month HERE This year also celebrating prid
e present abot connecting…
tweet of cluster [25]Reply  better
tweet of cluster [26] drink water replaced od morning
tweet of cluster [27] Twitter introdcing orselves Twitter
tweet of cluster [28] Someone tta make first move hang" energy
tweet of cluster [29] Poems Tweets making world feel smaller
```

conclusion :

on a charge les tweets d'apres l'api de twitter, on les a mis dans le fichier csv twitter_data_analysis. puis on a fait le data cleaning et on a mis le resultat dans le fichier cleaning_twitter_data_analysis.csv.Et enfin on a mis un tweet de chaque cluster dans le fichier result_final_twitter_data_analysis.

lien github (https://github.com/hazbri/projectDataMining/)

In [ ]:

In [ ]: