remote_jobs_opportunity

June 23, 2020

Topic Modelling of tweets on remote job opportunity available. Due to the present situation of things in the land (Covid 19), one of the measures put in place to curb the spread of the disease is to maintain `social distancing`. in lue of that, most employers has resolved to working remotely. The analysis below is an exploreatory analysis of tweets gathered from January 1st 2020, uptil June 20, 2020.

[6]:
```
# Importing modules
import os
import re
import pandas as pd
import numpy as np
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
import plotly as py
import plotly.graph_objs as go
import gensim
from gensim import corpora, models, similarities
import logging
import tempfile
from nltk.corpus import stopwords
from string import punctuation
from collections import OrderedDict
import seaborn as sns
import pyLDAvis.gensim
import matplotlib.pyplot as plt
%matplotlib inline
import string
import nltk
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.corpus import stopwords
from string import punctuation
from nltk.tokenize import TweetTokenizer
from nltk import tokenize
from wordcloud import WordCloud
from PIL import Image
import warnings
warnings.filterwarnings("ignore")
```

[7]:
```
# reading the data.
# we searched for tweets on the text `remote jobs available in any sector` and␣
 ↪'working remotely as'

#os.chdir('..')
tweet1 = pd.read_csv('remote_jobs_inanysector.csv')
tweet2 = pd.read_csv('covid_19.csv')   #'working remotely as'
tweets.head(3)
```

[7]:

```
   Unnamed: 0  screen_name          username              user_id  \
0           0  basedremote  Find Remote Jobs  1182341109525159936
1           1  basedremote  Find Remote Jobs  1182341109525159936
2           2      247work             Tiana             21278799

              tweet_id                                tweet_url  \
0  1221219985462300672  /basedremote/status/1221219985462300672
1  1221219977371508737  /basedremote/status/1221219977371508737
2  1221217439427571714      /247work/status/1221217439427571714

             timestamp  timestamp_epochs  \
0  2020-01-25 23:55:08        1579996508
1  2020-01-25 23:55:06        1579996506
2  2020-01-25 23:45:01        1579995901

                                                text  \
0  New remote job: Senior Software Engineer at In…
1  REMOTE JOB as Senior Machine Learning - Series…
2  #jobs Remote Position: Budget and Finance Asso…

                                           text_html  … has_media  \
0  <p class="TweetTextSize js-tweet-text tweet-te…  …       True
1  <p class="TweetTextSize js-tweet-text tweet-te…  …       True
2  <p class="TweetTextSize js-tweet-text tweet-te…  …      False

                                            img_urls  video_url likes  \
0  ['https://pbs.twimg.com/media/EPKlGvBX4AAGnKf…        NaN     0
1  ['https://pbs.twimg.com/media/EPKlGRNUUAEatE1…        NaN     0
2                                                 []        NaN     0

   retweets  replies  is_replied  is_reply_to  parent_tweet_id  reply_to_users
0         0        0       False        False              NaN              []
1         1        0       False        False              NaN              []
2         0        0       False        False              NaN              []

[3 rows x 22 columns]
```

[8]:
```
# concatenating the two tweets
frames = [tweet1,tweet2]
tweets = pd.concat(frames)
```

[9]:
```
# we need to know the lenght of the data set we are working with
len(tweets)
```

[9]: 25951

[10]:
```
# since we have 2 different tweets, there might be a possibility of duplicates,␣
 ↪hence we would have to drop all duplicates.

# sorting by text
tweets.sort_values("text", inplace = True)

# dropping ALL duplicte values
tweets.drop_duplicates(subset ="text",
                       keep = False, inplace = True)

# displaying new len
len(tweets)
```

[10]: 23583

[ ]:

[11]:
```
# Separating the time variable by hour, day, month and year for further␣
 ↪analysis using datetime

tweets['timestamp'] = pd.to_datetime(tweets['timestamp'])
tweets['hour'] = tweets['timestamp'].apply(lambda x: x.hour)
tweets['month'] = tweets['timestamp'].apply(lambda x: x.month)
tweets['day'] = tweets['timestamp'].apply(lambda x: x.day)
tweets['year'] = tweets['timestamp'].apply(lambda x: x.year)
tweets['length'] = tweets["text"].apply(len)
tweets['num_of_words'] = tweets["text"].str.split().apply(len)
# addding 1 column for counting
# (dodanie 1 kolumny do zliczania)
tweets['dummy_count'] = 1
```

[12]:
```
#!pip3 install plotly
#!pip3 install pyLDAvis
```

[13]:
```
# Who twitted most about romote jobs in the last 6 months
grouped = pd.DataFrame(tweets.groupby('username').size().rename('counts')).
 ↪sort_values('counts', ascending=False)
grouped.head(10)
```

[13]:

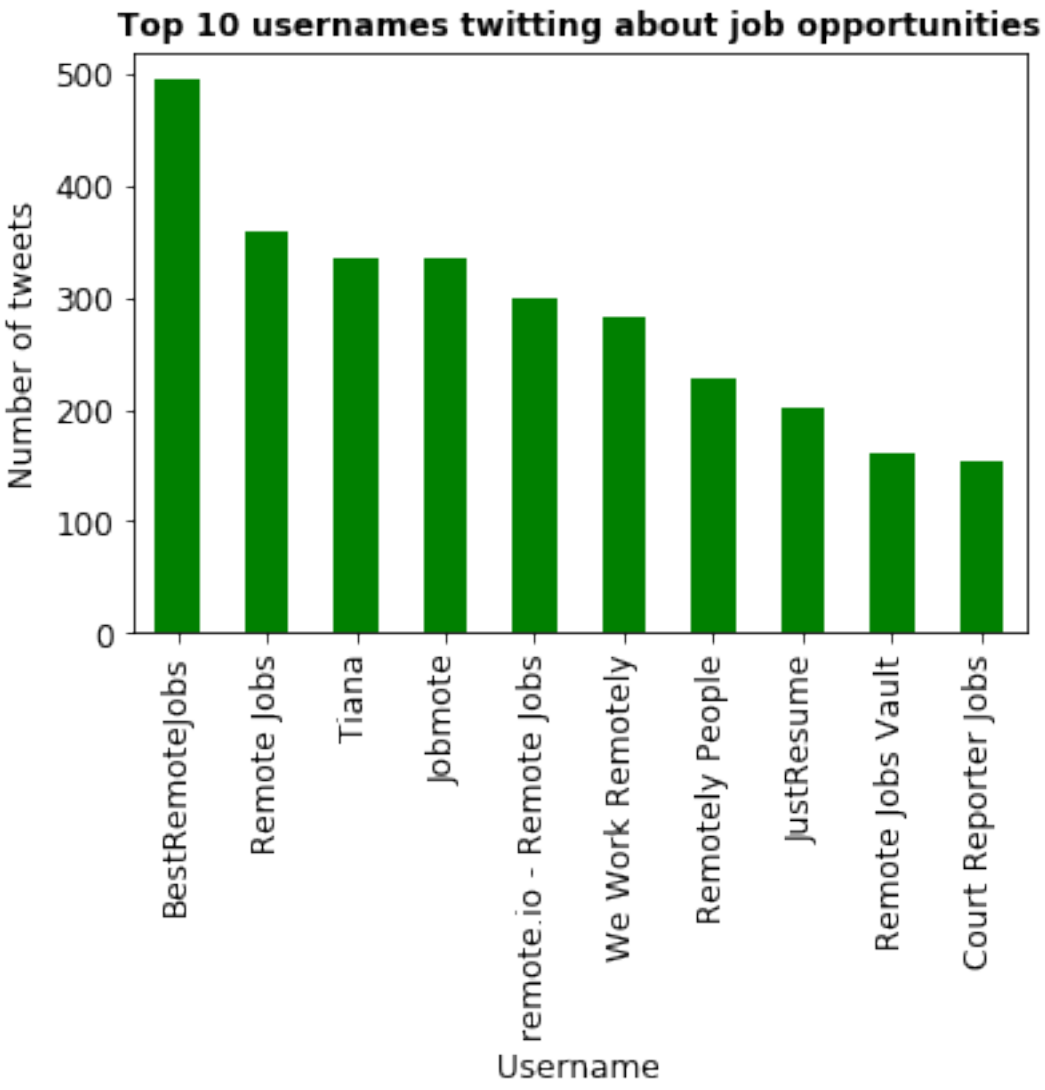
username	counts
BestRemoteJobs	495
Remote Jobs	360
Jobmote	336
Tiana	336
remote.io - Remote Jobs	299
We Work Remotely	282
Remotely People	228
JustResume	202
Remote Jobs Vault	160
Court Reporter Jobs	154

Data Visualization

Observation: These are the first 10 people/organization that tweeted most about remote jobs in the past 6 months. `BestRemoteJobs` has 495 count followed by `Remote Jobs`. The visualization is as follows:

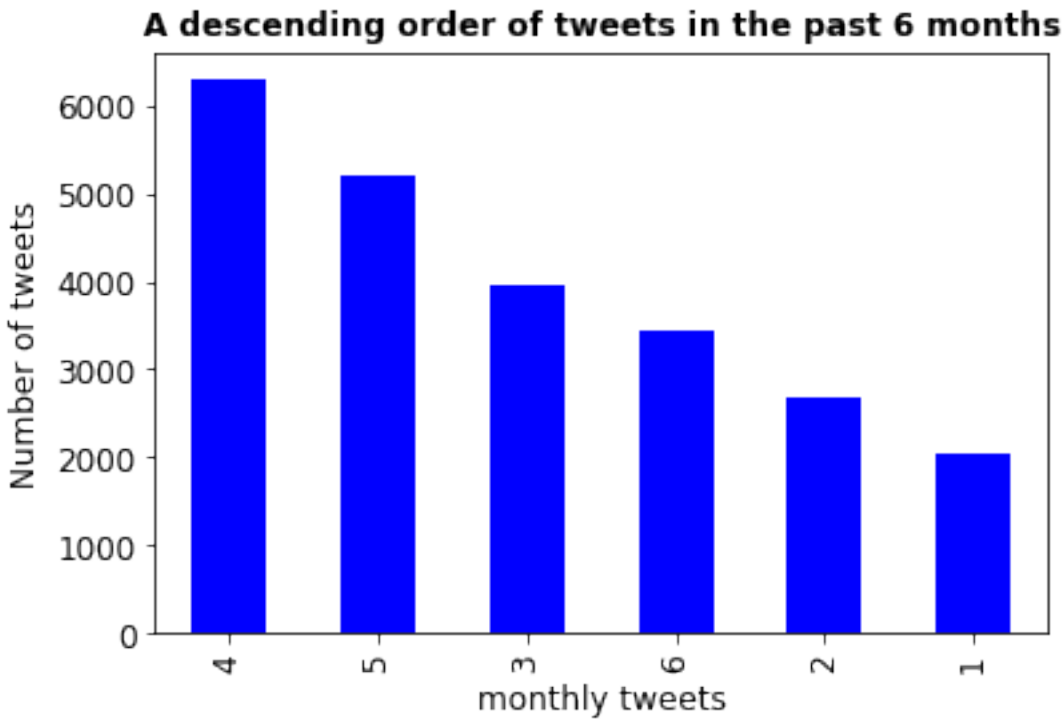
[14]:
```
get_ipython().magic('matplotlib inline')
tweets_by_username = tweets['username'].value_counts()

fig, ax = plt.subplots()
ax.tick_params(axis='x', labelsize=12)
ax.tick_params(axis='y', labelsize=12)
ax.set_xlabel('Username', fontsize=12)
ax.set_ylabel('Number of tweets' , fontsize=12)
ax.set_title('Top 10 usernames twitting about job opportunities', fontsize=12,␣
 ↪fontweight='bold')
tweets_by_username[:10].plot(ax=ax, kind='bar', color='green')
```

[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f526c46f050>
```



[15]:
```
get_ipython().magic('matplotlib inline')
tweets_by_username = tweets['month'].value_counts()

fig, ax = plt.subplots()
ax.tick_params(axis='x', labelsize=12)
ax.tick_params(axis='y', labelsize=12)
ax.set_xlabel('monthly tweets', fontsize=12)
ax.set_ylabel('Number of tweets' , fontsize=12)
ax.set_title('A descending order of tweets in the past 6 months', fontsize=12,␣
↪fontweight='bold')
tweets_by_username[:15].plot(ax=ax, kind='bar', color='blue')
```

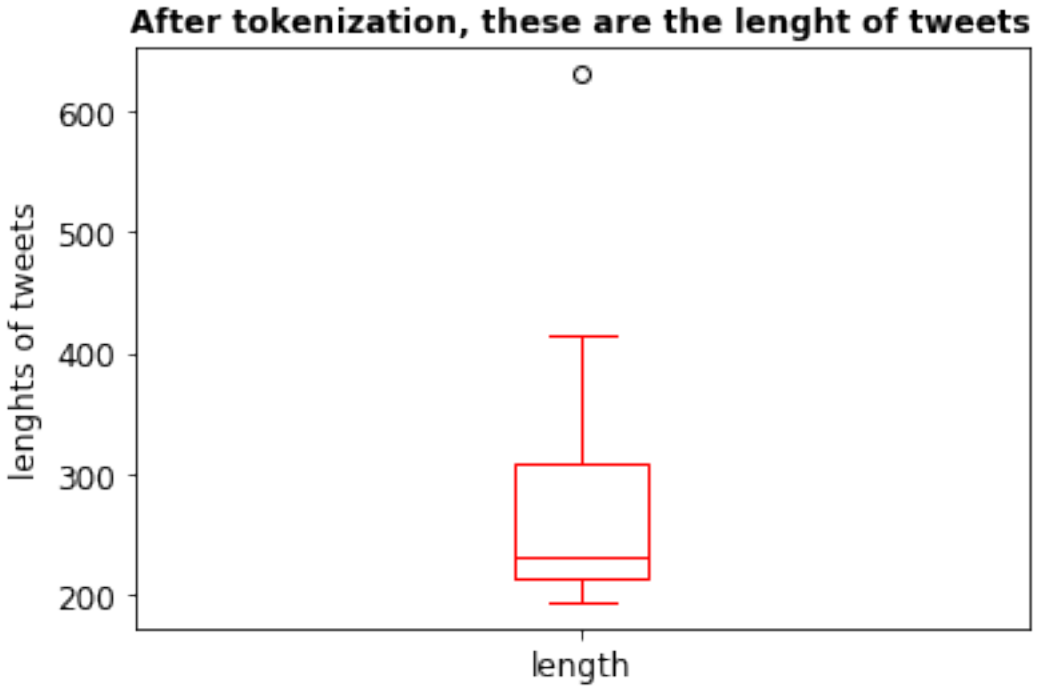
[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5269363c90>
```



Observation: Above is a bra plot showing the number of tweets as regards remote jobs in the past 6 months. From the graph, it is observed that `April` being the 4th month had the highest tweet on remote job opportunities, this might be due to the fact that, as at then almost all companies and organization were still on lock down, and the only way they could keep up with daily activities was to work remotely. Other copany also saw this as an opportunity to advertise and make awareness on the importance of working remotely. In as at `January` which happend to be the first month most people were still trying to grasp the shock of the pandemic, and so, never paid much attention to working remotely, their major concern where perhaps trying to prevent them elves from the deadly disease. However, in the month of `May and June`, the flare for remote job started reducing gradually as most people had started going back to their work place, things are almost becoming normal again.

[16]:
```
get_ipython().magic('matplotlib inline')
tweets_by_username = tweets['length'].value_counts()

fig, ax = plt.subplots()
ax.tick_params(axis='x', labelsize=12)
ax.tick_params(axis='y', labelsize=12)
#ax.set_xlabel('tweets', fontsize=12)
ax.set_ylabel('lenghts of tweets' , fontsize=12)
ax.set_title('After tokenization, these are the lenght of tweets', fontsize=12,␣
↪fontweight='bold')
tweets_by_username[:15].plot(ax=ax, kind='box', color='red')
```

[16]: <matplotlib.axes._subplots.AxesSubplot at 0x7f526b489a90>



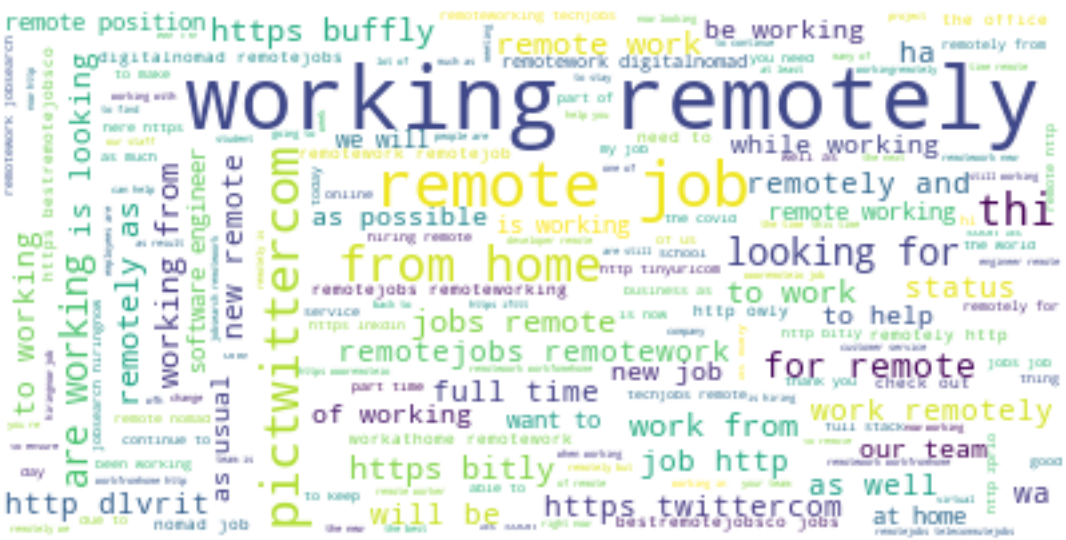
Observation: The above box plot shows the lenght of tweets as after tokenization(spliting the sentence into token or worrds) has been done. From the above plot, it is visible that the minimum lenght per tweet was approximately 180 words, while the maximum was approximately 420 words. Looking closely, we see there is a particular tweet with more than 600 words, this can might be classified as an outlier.

[17]:
```
# Remove punctuation
tweets['tweets_text_processed'] = tweets['text'].map(lambda x: re.sub('[,\.!?
 ↪]', '', x))
# Convert the titles to lowercase
tweets['tweets_text_processed'] = tweets['tweets_text_processed'].map(lambda x:␣
 ↪x.lower())
# Print out the first rows of tweets
tweets['tweets_text_processed'].head()
```

[17]:
```
8261     \ncan i do the job remote\n"well we actually d…
5914     \nmy only regret in that regard is how slow th…
13015     \n\nthank you to some big companies like @nik…
2838       are you a customer support agent (8am-4pm ms…
7100       are you a licensed #electrician in ny nj ct …
Name: tweets_text_processed, dtype: object
```

[18]:
```
# Import the wordcloud library
from wordcloud import WordCloud

# Join the different processed titles together.
long_string = ','.join(list(tweets['tweets_text_processed'].values))
# Create a WordCloud object
wordcloud = WordCloud(background_color="white", max_words=500, contour_width=3,␣
↪contour_color='steelblue')
# Generate a word cloud
wordcloud.generate(long_string)
# Visualize the word cloud
wordcloud.to_image()
```

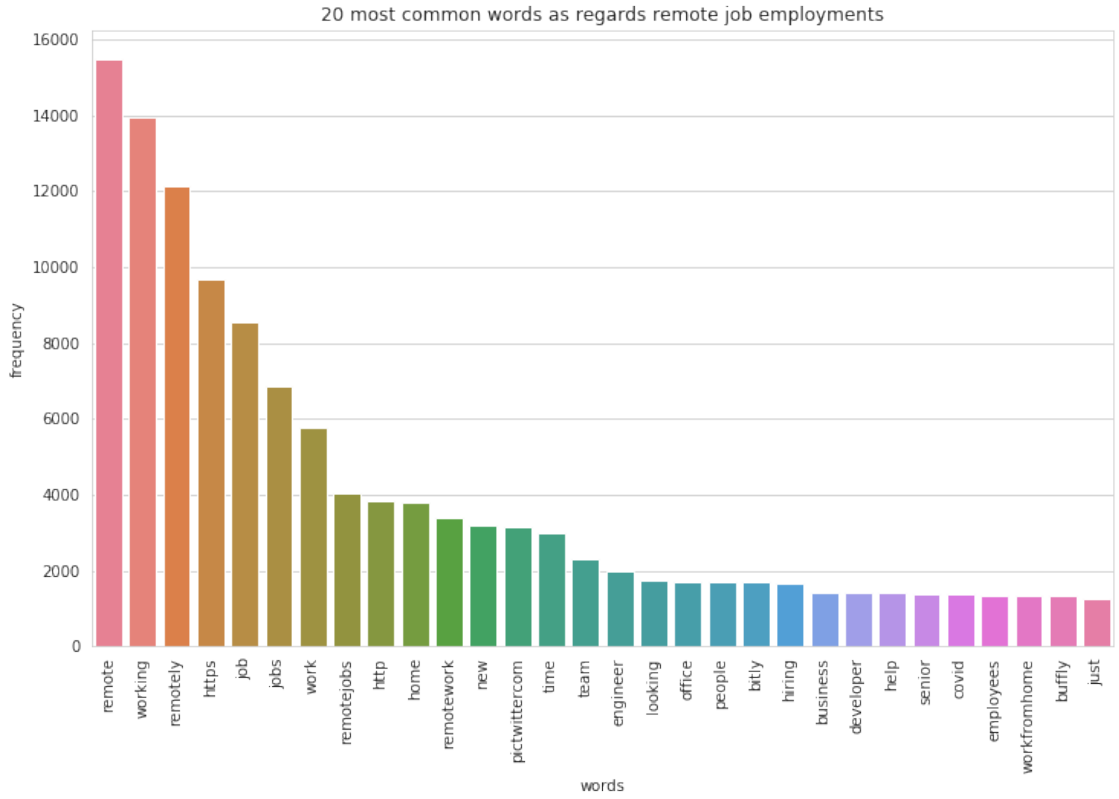
[18]:



Observations: The above diagram is called a wordcloud, we try to see the 500 most words in the tweet. as we can see the words like working remotely, remote jobs, from home, e.t.c are quit obvious.

[19]:
```
# Load the library with the CountVectorizer method
from sklearn.feature_extraction.text import CountVectorizer
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
%matplotlib inline

# Helper function
def plot_10_most_common_words(count_data, count_vectorizer):
    words = count_vectorizer.get_feature_names()
    total_counts = np.zeros(len(words))
```

```
    for t in count_data:
        total_counts+=t.toarray()[0]

    count_dict = (zip(words, total_counts))
    count_dict = sorted(count_dict, key=lambda x:x[1], reverse=True)[0:30]
    words = [w[0] for w in count_dict]
    counts = [w[1] for w in count_dict]
    x_pos = np.arange(len(words))

    plt.figure(2, figsize=(12, 12/1.6180))
    plt.subplot(title='20 most common words as regards remote job employments')
    sns.set_context("notebook", font_scale=1, rc={"lines.linewidth": 1.5})
    sns.barplot(x_pos, counts, palette='husl')
    plt.xticks(x_pos, words, rotation=90)
    plt.xlabel('words')
    plt.ylabel('frequency')
    plt.show()

# Initialise the count vectorizer with the English stop words
count_vectorizer = CountVectorizer(stop_words='english')

# Fit and transform the processed titles
count_data = count_vectorizer.fit_transform(tweets['tweets_text_processed'] )

# Visualise the 10 most common words
plot_10_most_common_words(count_data, count_vectorizer)
```



Observation: The bar chart above is similar to the wordcloud we had before, in the sence that, we are still visualization the most common words in the tweet. however, unlike the word cloud, the bar chart helps us to see both the most common words, as well as it's frequency(i.e the number of times they appear in the tweet). from the graph, we can see the the word remote, working, remmotely e.t.c appears so much in the tweets. now let's consider the tail end of the gragh, we can also notice that the word team, engineer, business, developer, also appears almost 2000 times in the tweet, meaning that there are some professions or job titles that really benefits more from the remote work. logically, it is obvious that a software engineer can work from home and still be productive, likewise a developer and a business owner whose job is not limited to phisical contact.

[ ]:

Topic Modelling using the LDA

[20]:
```
import warnings
warnings.simplefilter("ignore", DeprecationWarning)
# Load the LDA model from sk-learn
from sklearn.decomposition import LatentDirichletAllocation as LDA
```

```
# Helper function
def print_topics(model, count_vectorizer, n_top_words):
    words = count_vectorizer.get_feature_names()
    for topic_idx, topic in enumerate(model.components_):
        print("\nTopic #%d:" % topic_idx)
        print(" ".join([words[i]
                        for i in topic.argsort()[:-n_top_words - 1:-1]]))

# Tweak the two parameters below
number_topics = 10
number_words = 15

# Create and fit the LDA model
lda = LDA(n_components=number_topics, n_jobs=-1)
lda.fit(count_data)
# Print the topics found by the LDA model
print("Topics found via LDA:")
print_topics(lda, count_vectorizer, number_words)
```

```
Topics found via LDA:

Topic #0:
job remote https new twitter utm_medium remotejobs 2020 jobs social posted
utm_campaign buffly listing covid19

Topic #1:
remote job jobs https remotejobs remotework http new engineer looking developer
hiring senior manager workathome

Topic #2:
https remote working remotely bitly jobs continue employees pictwittercom join
want looking job apply hq

Topic #3:
remote https engineer job remotework senior jobs software remotejobs developer
looking workfromhome bitly hiring remotejob

Topic #4:
remote jobs work https job home buffly time jobsearch http position hiringnow
companies pictwittercom working

Topic #5:
working remotely work home people time job remote like just ve office day don
know

Topic #6:
remote https job remotely pictwittercom working time just work facebook new
```

employees 2020 bitly teams

Topic #7:
working remotely app home customer covid possible safety healthy stay understand safe ensure 19 digital

Topic #8:
working remotely https pictwittercom team home covid work business 19 tips staff http bitly new

Topic #9:
jobs work remote http remotejobs usa workfromhome telecommutejobs dlvrit https 50 amazon help giveaway card

Observation: From the above LDA analysis, we generated 10 number_topics,with 15 number_words each, these two parameters can be tweak according to out needs, but from what we have displayed, it is obvious that the last topic #9 is about how to tech business persons on remote working tips. topic #6 is about a remote jobs for a senior developer engineer.

Conclusion: We have been able to scrap tweets from the first day of January up til the 20 day of Juneseen the various tweets from people around the world as regards remote jobs opportinities and how the trend is moving. Aside the topic modelling, one can dive deep into this data to explore more, as there are still some interesting findings to be made from it.