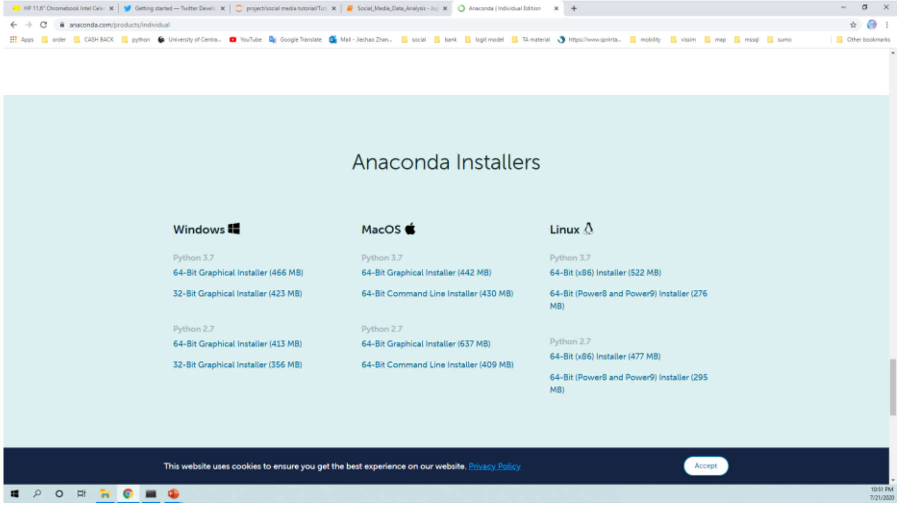
A Tutorial of Social Media Data Analysis

This tutorial is an instruction for running the code of social media data analysis. It mainly contains four parts and the codes can be seen as the attached Jupyter Notebook – 'Social Media Data Analysis.ipynb'. It has four sections: i) data collection, ii) sentiment analysis, iii) visualization of the sentiment analysis results, iv) visualization of the topic analysis results, and iv) topic analysis.

1. Software Installation and Data Downloading

1.1 Software Installation

In this project, we use Jupyter notebook as the integrated development environment for Python. To install the Jupyter notebook, we can download the anaconda from the official website, shown in **Figure 1.1**. The link of the website is: https://www.anaconda.com/products/individual. We can choose the suitable version of anaconda from the website (e.g. 64-Bit Graphical Installer).



Anaconda Installers

Windows

Python 3.7
64-Bit Graphical Installer (466 MB)
32-Bit Graphical Installer (423 MB)

Python 2.7
64-Bit Graphical Installer (413 MB)
32-Bit Graphical Installer (356 MB)

MacOS

Python 3.7
64-Bit Graphical Installer (442 MB)
64-Bit Command Line Installer (430 MB)

Python 2.7
64-Bit Graphical Installer (637 MB)
64-Bit Command Line Installer (409 MB)

Linux

Python 3.7
64-Bit (x86) Installer (522 MB)
64-Bit (Power8 and Power9) Installer (276 MB)

Python 2.7
64-Bit (x86) Installer (477 MB)
64-Bit (Power8 and Power9) Installer (295 MB)

This website uses cookies to ensure you get the best experience on our website. Privacy Policy

Figure 1.1 The official website of Anaconda

After installing the anaconda, we can open the Jupyter notebook from anaconda navigator, shown in **Figure 1.2**.

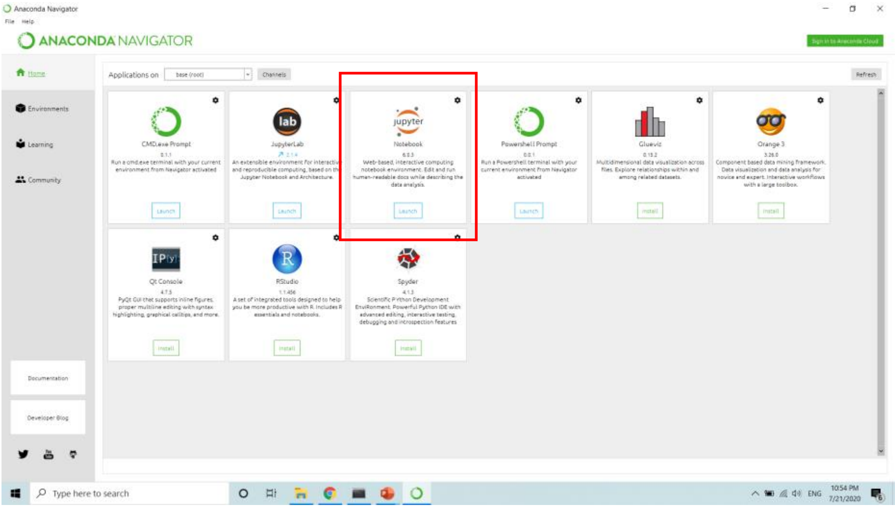


Figure 1.2 The platform of anaconda navigator

In the Jupyter notebook, we can update the code file - Social Media Data Analysis.ipynb, shown in **Figure 1.3**.

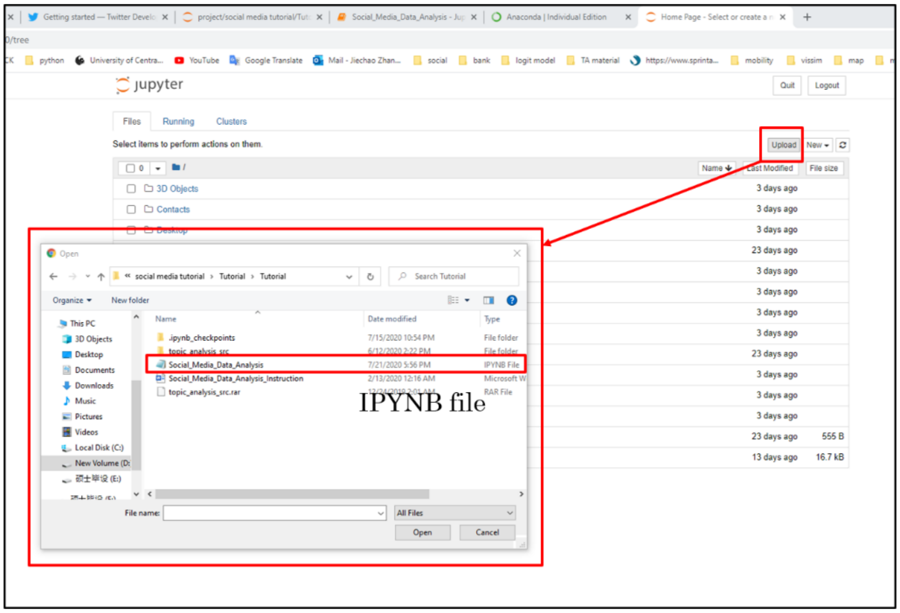
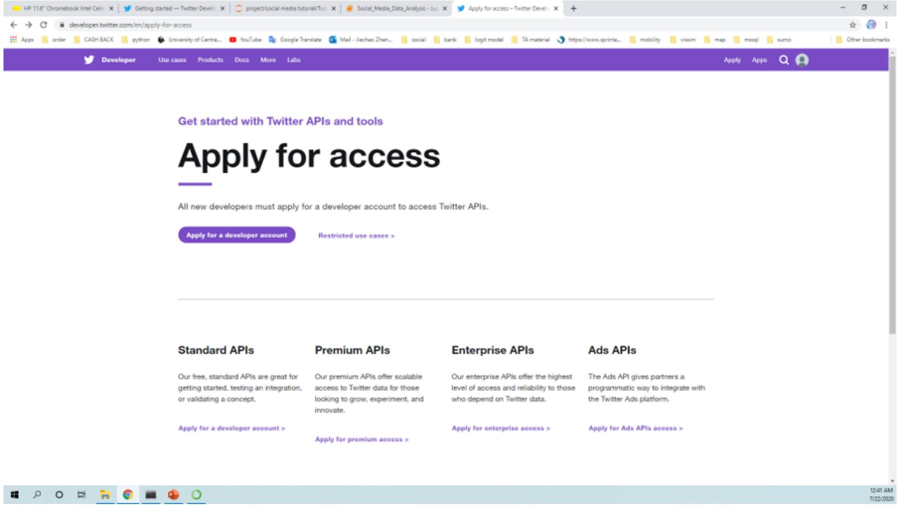


Figure 1.3 The platform of Jupyter notebook

1.2 Data Collection

1.2.1 Apply for a Twitter Developer Account

To collect the Twitter data, one of the prerequisites is to have a Twitter developer account. Thus, we need to apply for a Twitter developer account online through the following link: https://developer.Twitter.com/en/apply-for-access. **Figure 1.4** shows the website.



Get started with Twitter APIs and tools

Apply for access

All new developers must apply for a developer account to access Twitter APIs.

Apply for a developer account

Restricted use cases >

Standard APIs

Our free, standard APIs are great for getting started, testing an integration, or validating a concept.

Apply for a developer account >

Premium APIs

Our premium APIs offer scalable access to Twitter data for those looking to grow, experiment, and innovate.

Apply for premium access >

Enterprise APIs

Our enterprise APIs offer the highest level of access and reliability to those who depend on Twitter data.

Apply for enterprise access >

Ads APIs

The Ads API gives partners a programmatic way to integrate with the Twitter Ads platform.

Apply for Ads APIs access >

Figure 1.4 The website for applying the Twitter developer account

1.2.2 Data Collection (User Accounts)

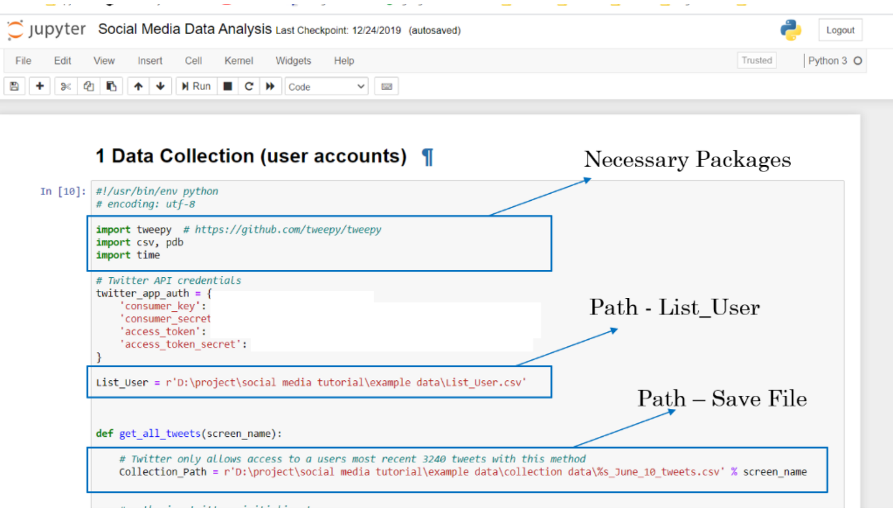
In the Jupyter Notebook, the code for data collection is shown under the heading ‘**1 Data Collection (User Accounts)**’, **seen as Figure 1.5**.

The **Twitter_app_auth** is the Twitter API credentials for which one needs to apply to the Twitter official website.

To collect Twitter data by user accounts, we need a ‘csv’ file called ‘List_User.csv’ (this file name should be fixed), seen in **Figure 1.6**. The ‘List_User.csv’ file contains all the user accounts for which data need to be collected, and the format can be seen in **Figure 1.6**. Put the ‘List_User.csv’ file and the code in the same folder.

For different collection time, we can create different folders to save the Twitter data. Each folder must contain both the ‘List_User.csv’ file and the data collection code. From the code, we can change the ‘June_10_tweets’ (shown in **Figure 1.5**) to the expected date to save the Twitter data with a different file name.

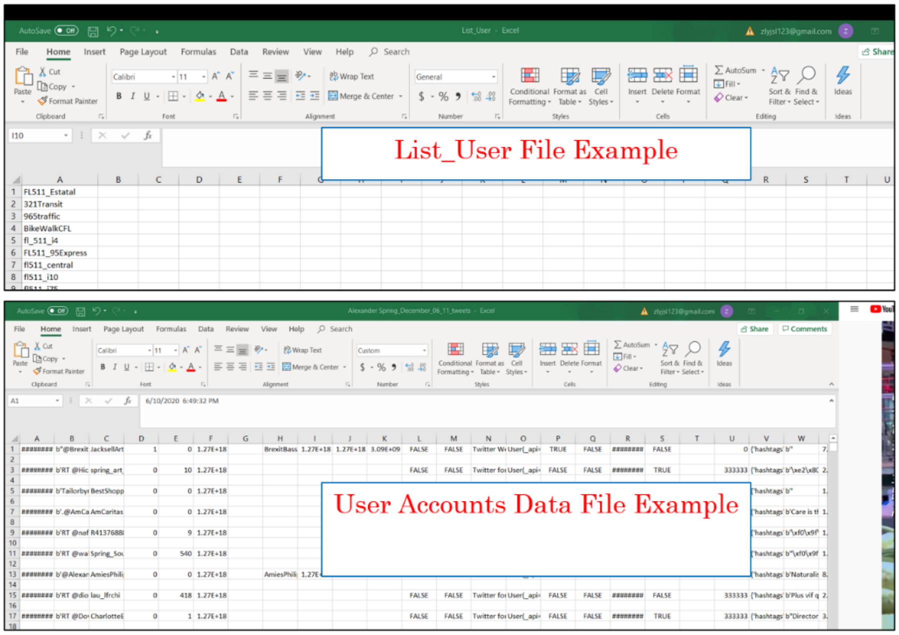
After all the files are prepared, run the code and the Twitter data will be collected in the same folder where the code for data collection is kept.



1 Data Collection (user accounts) ¶

```
In [10]: #!/usr/bin/env python
         # encoding: utf-8

         import tweepy  # https://github.com/tweepy/tweepy
         import csv, pdb
         import time

         # Twitter API credentials
         twitter_app_auth = {
             'consumer_key': [redacted]
             'consumer_secret
             'access_token':
             'access_token_secret':
         }

         List_User = r'D:\project\social media tutorial\example data\List_User.csv'


         def get_all_tweets(screen_name):

             # Twitter only allows access to a users most recent 3240 tweets with this method
             Collection_Path = r'D:\project\social media tutorial\example data\collection data\%s_June_10_tweets.csv' % screen_name
```

Figure 1.5 Example of User Account Data Collection Code



	A	B	C	D	E	F
1	FL511_Estatal					
2	321Transit					
3	965traffic					
4	BikeWalkCFL					
5	fl_511_i4					
6	FL511_95Express					
7	fl511_central					
8	fl511_i10					
9	fl511_i75					

A1 6/10/2020 6:49:32 PM

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	
1	########	b"@Brexit	JacksellArt	1	0	1.27E+18		BrexitBass	1.27E+18	1.27E+18	3.09E+09	FALSE	FALSE	Twitter Wi	User(_api=	TRUE	FALSE	########	FALSE		0	{'hashtags'	b''	7.
2																								
3	########	b'RT @Hic	spring_art_	0	10	1.27E+18						FALSE	FALSE	Twitter foi	User(_api=	FALSE	FALSE	########	TRUE		333333	{'hashtags'	b'\xe2\x8C	2.
4																								
5	########	b'Tailorbyı	BestShopp	0	0	1.27E+18																{'hashtags'	b''	1.
6																								
7	########	b'.@AmCa	AmCaritas	0	0	1.27E+18																{'hashtags'	b'Care is tl	1.
8																								
9	########	b'RT @naf	R41376888	0	9	1.27E+18																{'hashtags'	b'\xf0\x9f'	1.
10																								
11	########	b'RT @wa	Spring_Sou	0	540	1.27E+18																{'hashtags'	b"\xf0\x9f	1.
12																								
13	########	b'@Alexan	AmiesPhili	0	0	1.27E+18		AmiesPhili	1.27E													{'hashtags'	b'Naturalis	8.
14																								
15	########	b'RT @dio	lau_lfrchi	0	418	1.27E+18						FALSE	FALSE	Twitter foi	User(_api=	FALSE	FALSE	########	FALSE		333333	{'hashtags'	b'Plus vif q	2.
16																								
17	########	b'RT @Do	CharlotteE	0	1	1.27E+18						FALSE	FALSE	Twitter foi	User(_api=	FALSE	FALSE	########	TRUE		333333	{'hashtags'	b"Director	3.
18																								

Figure 1.6 The example of List_User.csv and output file

1.2.3 Data Collection (Keywords)

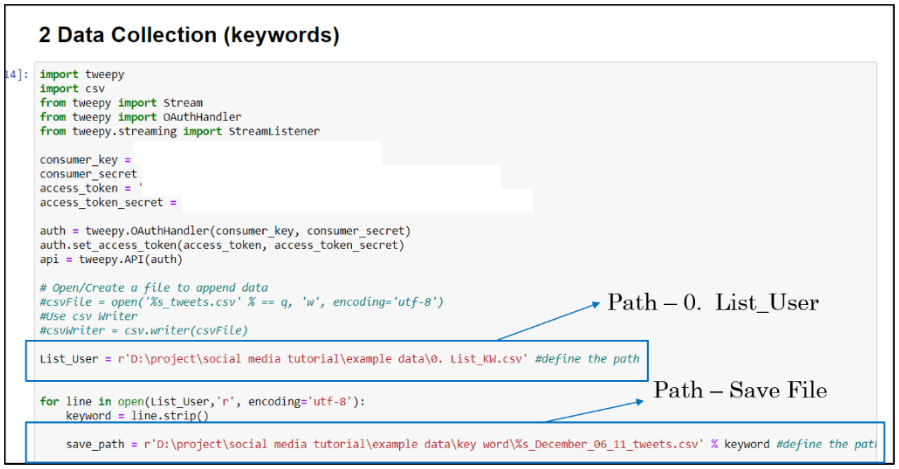
The code for data collection is shown under the heading '**2 Data Collection (keywords)**', **seen as Figure 1.7 (a)**.

The **Twitter_app_auth** is the Twitter API credentials for which one needs to apply to the Twitter official website.

To collect Twitter data by user accounts, we need a 'csv' file called '0.List_KW.csv' (this file name should be fixed), seen as **Figure 1.8**. The '0.List_KW.csv' file contains all the keywords which need to be collected, and the format can be seen as **Figure 1.8**. Put the ‘0.List_KW.csv’ file and the code in the same folder.

For different collection time, we can create different folders to save the Twitter data. Each folder must contain both the ‘0.List_KW.csv’ file and the data collection code. From the code, we can change the time periods (shown in **Figure 1.7 (b)**) to the expected date to save the Twitter data with different file name.

After all the files are prepared, run the code and the Twitter data will be collected in the same folder where code for data collection is kept seen as **Figure 1.8**.



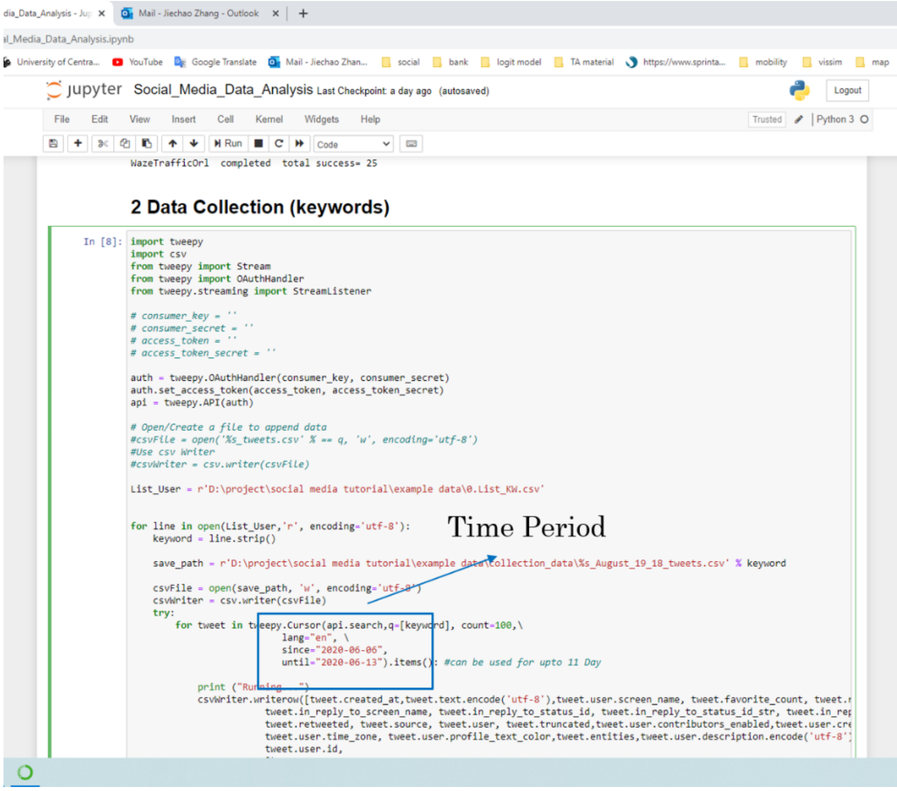
2 Data Collection (keywords)

```
14]: import tweepy
     import csv
     from tweepy import Stream
     from tweepy import OAuthHandler
     from tweepy.streaming import StreamListener

     consumer_key = [redacted]
     consumer_secret
     access_token = '
     access_token_secret =

     auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
     auth.set_access_token(access_token, access_token_secret)
     api = tweepy.API(auth)

     # Open/Create a file to append data
     #csvFile = open('%s_tweets.csv' % == q, 'w', encoding='utf-8')
     #Use csv Writer
     #csvWriter = csv.writer(csvFile)

     List_User = r'D:\project\social media tutorial\example data\0. List_KW.csv' #define the path

     for line in open(List_User,'r', encoding='utf-8'):
         keyword = line.strip()

         save_path = r'D:\project\social media tutorial\example data\key word\%s_December_06_11_tweets.csv' % keyword #define the pat
```

Path – 0. List_User

Path – Save File

(a)



```
WazeTrafficOrl  completed  total success= 25
```

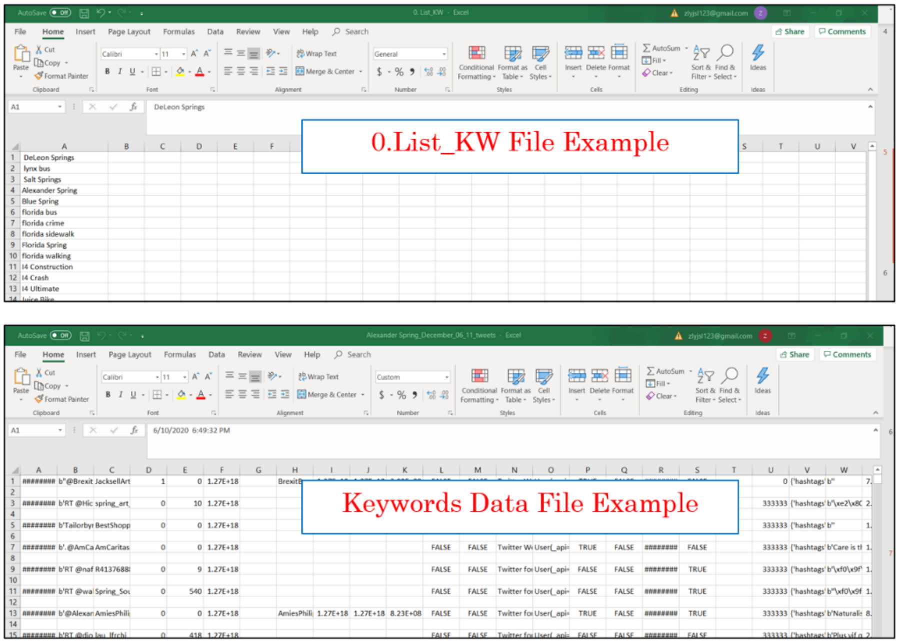
2 Data Collection (keywords)

```
In [8]: import tweepy
        import csv
        from tweepy import Stream
        from tweepy import OAuthHandler
        from tweepy.streaming import StreamListener

        # consumer_key = ''
        # consumer_secret = ''
        # access_token = ''
        # access_token_secret = ''

        auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
        auth.set_access_token(access_token, access_token_secret)
        api = tweepy.API(auth)

        # Open/Create a file to append data
        #csvFile = open('%s_tweets.csv' % == q, 'w', encoding='utf-8')
        #Use csv Writer
        #csvWriter = csv.writer(csvFile)

        List_User = r'D:\project\social media tutorial\example data\0.List_KW.csv'


        for line in open(List_User,'r', encoding='utf-8'):
            keyword = line.strip()

            save_path = r'D:\project\social media tutorial\example data\collection_data\%s_August_19_18_tweets.csv' % keyword

            csvFile = open(save_path, 'w', encoding='utf-8')
            csvWriter = csv.writer(csvFile)
            try:
                for tweet in tweepy.Cursor(api.search,q=[keyword], count=100,\
                                   lang="en", \
                                   since="2020-06-06",
                                   until="2020-06-13").items(): #can be used for upto 11 Day

                    print ("Running...")
                    csvWriter.writerow([tweet.created_at,tweet.text.encode('utf-8'),tweet.user.screen_name, tweet.favorite_count, tweet.r
                                  tweet.in_reply_to_screen_name, tweet.in_reply_to_status_id, tweet.in_reply_to_status_id_str, tweet.in_rep
                                  tweet.retweeted, tweet.source, tweet.user, tweet.truncated,tweet.user.contributors_enabled,tweet.user.cre
                                  tweet.user.time_zone, tweet.user.profile_text_color,tweet.entities,tweet.user.description.encode('utf-8')
                                  tweet.user.id,
```

Time Period

(b)

Figure 1.7 Example of Keyword Data Collection Code: (a) the code for input and output files; (b) the code for changing the time period of collection


0.List_KW - Excel
0.List_KW File Example
Alexander Spring_December_06_11_tweets - Excel
Keywords Data File Example

A1: DeLeon Springs

	A
1	DeLeon Springs
2	lynx bus
3	Salt Springs
4	Alexander Spring
5	Blue Spring
6	florida bus
7	florida crime
8	florida sidewalk
9	Florida Spring
10	florida walking
11	I4 Construction
12	I4 Crash
13	I4 Ultimate
14	Juice Bike

A1: 6/10/2020 6:49:32 PM

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	
1	########	b"@Brexit	JacksellArt	1	0	1.27E+18		BrexitB													0	{'hashtags'	b''	7.
2																								
3	########	b'RT @Hic	spring_art_	0	10	1.27E+18															333333	{'hashtags'	b"\xe2\x8C	2.
4																								
5	########	b'Tailorbyr	BestShopp	0	0	1.27E+18															333333	{'hashtags'	b''	1.
6																								
7	########	b'.@AmCa	AmCaritas	0	0	1.27E+18						FALSE	FALSE	Twitter W	User(_api=	TRUE	FALSE	########	FALSE		333333	{'hashtags'	b'Care is th	1.
8																								
9	########	b'RT @naf	R41376888	0	9	1.27E+18						FALSE	FALSE	Twitter for	User(_api=	FALSE	FALSE	########	TRUE		333333	{'hashtags'	b"\xf0\x9f"	1.
10																								
11	########	b'RT @wal	Spring_Sou	0	540	1.27E+18						FALSE	FALSE	Twitter for	User(_api=	FALSE	FALSE	########	TRUE		333333	{'hashtags'	b"\xf0\x9f	1.
12																								
13	########	b'@Alexan	AmiesPhili	0	0	1.27E+18		AmiesPhili	1.27E+18	1.27E+18	8.23E+08	FALSE	FALSE	Twitter for	User(_api=	TRUE	FALSE	########	TRUE		333333	{'hashtags'	b'Naturalis	8.
14																								
15	########	b'RT @dio	Jau_lfrchi	0	418	1.27E+18						FALSE	FALSE	Twitter for	User(_api=	FALSE	FALSE	########	FALSE		333333	{'hashtags'	b'Plus vif q	2.

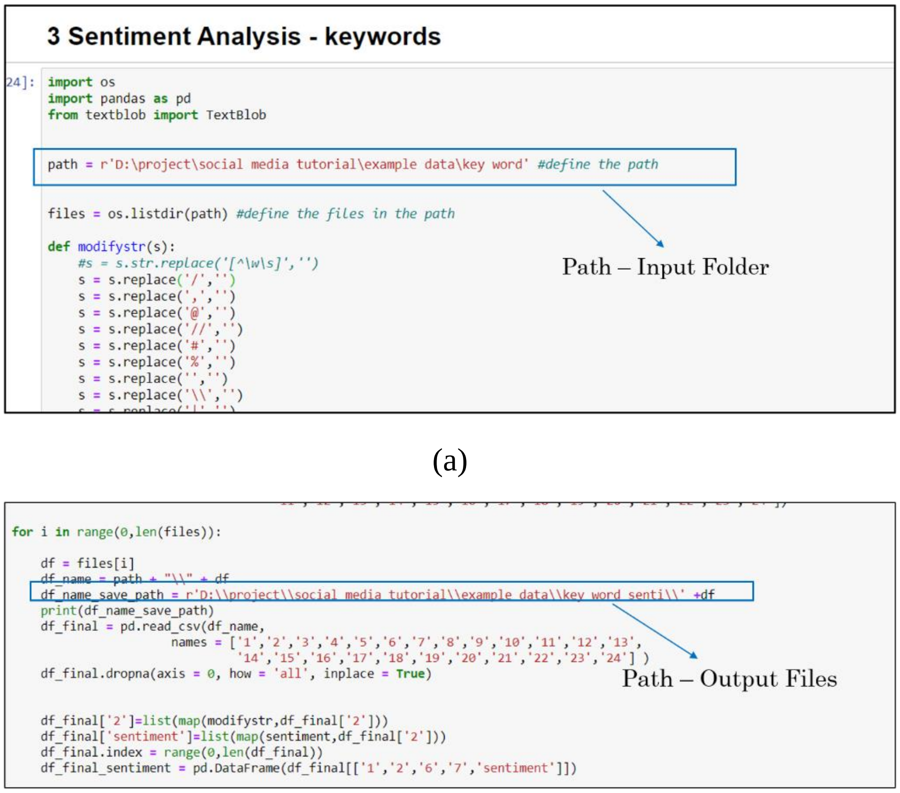
Figure 1.8 The example of 0.List_KW.csv and output file

2. Sentiment Analysis and Visualization

2.1 Sentiment Analysis

The second part in the Jupyter Notebook is the sentiment analysis which can be seen under the heading '**3 Sentiment Analysis**'. The input of the sentiment analysis is the data collected from the part 1. From the codes, the '**path**' defines the path of the input file folder and the 'files' is a list of the filename. Given the specific folder path, the output of this program is the sentiment analysis results. The sentiment analysis results contain the 'user_id', 'polarity', and 'subjectivity'. The examples can be seen as **Figure 2.1**.

After setting the input path, run the codes and the '**df_final_sentiment**' is the output data frame. The example of input file path in the codes can be seen as **Figure 2.1 (a)** and the output file path can be seen as **Figure 2.1 (b)**.



3 Sentiment Analysis - keywords

```
24]: import os
     import pandas as pd
     from textblob import TextBlob


     path = r'D:\project\social media tutorial\example data\key word' #define the path

     files = os.listdir(path) #define the files in the path

     def modifystr(s):
         #s = s.str.replace('[^\w\s]','')
         s = s.replace('/','')
         s = s.replace(',','')
         s = s.replace('@','')
         s = s.replace('//','')
         s = s.replace('#','')
         s = s.replace('%','')
         s = s.replace('','')
         s = s.replace('\\','')
         s = s.replace('|','')
```

(a)

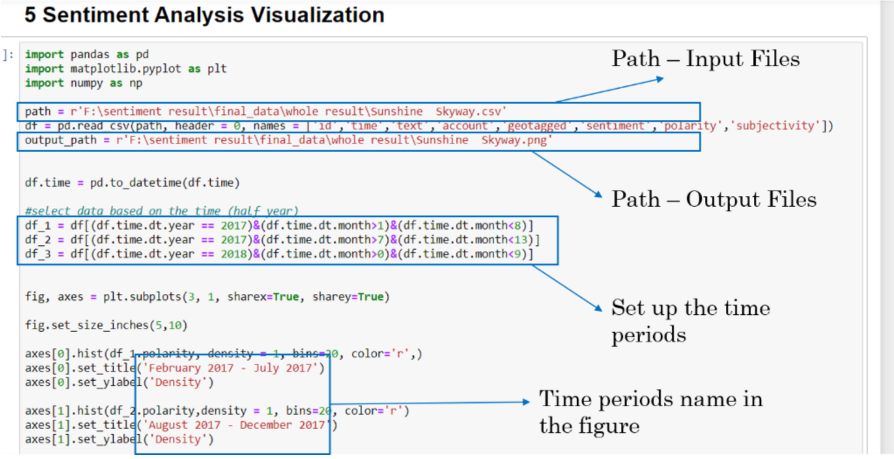
```
for i in range(0,len(files)):

    df = files[i]
    df_name = path + "\\" + df
    df_name_save_path = r'D:\\project\\social media tutorial\\example data\\key word senti\\' +df
    print(df_name_save_path)
    df_final = pd.read_csv(df_name,
                      names = ['1','2','3','4','5','6','7','8','9','10','11','12','13',
                               '14','15','16','17','18','19','20','21','22','23','24'] )
    df_final.dropna(axis = 0, how = 'all', inplace = True)


    df_final['2']=list(map(modifystr,df_final['2']))
    df_final['sentiment']=list(map(sentiment,df_final['2']))
    df_final.index = range(0,len(df_final))
    df_final_sentiment = pd.DataFrame(df_final[['1','2','6','7','sentiment']])
```

(b)

Figure 2.1 Example of Sentiment Analysis Code: (a) the code for input files; (b) the code for output files

2.2 Visualization of Sentiment Analysis Results

The visualization of sentiment analysis results can be seen as '**5 Sentiment Analysis Visualization**' in the code file. The example of the code for visualizing the sentiment analysis is shown as **Figure 2.2**. From the code, we can change the path of input and output files as well as the time periods seen from **Figure 2.2**.



5 Sentiment Analysis Visualization

```
]: import pandas as pd
   import matplotlib.pyplot as plt
   import numpy as np

   path = r'F:\sentiment result\final_data\whole result\Sunshine  Skyway.csv'
   df = pd.read_csv(path, header = 0, names = ['id','time','text','account','geotagged','sentiment','polarity','subjectivity'])
   output_path = r'F:\sentiment result\final_data\whole result\Sunshine  Skyway.png'


   df.time = pd.to_datetime(df.time)

   #select data based on the time (half year)
   df_1 = df[(df.time.dt.year == 2017)&(df.time.dt.month>1)&(df.time.dt.month<8)]
   df_2 = df[(df.time.dt.year == 2017)&(df.time.dt.month>7)&(df.time.dt.month<13)]
   df_3 = df[(df.time.dt.year == 2018)&(df.time.dt.month>0)&(df.time.dt.month<9)]


   fig, axes = plt.subplots(3, 1, sharex=True, sharey=True)

   fig.set_size_inches(5,10)

   axes[0].hist(df_1.polarity, density = 1, bins=20, color='r',)
   axes[0].set_title('February 2017 - July 2017')
   axes[0].set_ylabel('Density')

   axes[1].hist(df_2.polarity,density = 1, bins=20, color='r')
   axes[1].set_title('August 2017 - December 2017')
   axes[1].set_ylabel('Density')
```

Figure 2.2 Example of Sentiment Analysis Code: (a) the code for input files; (b) the code for output files

Run this code with the input file, the figure will be generated. **Figure 2.3** shows an example figure.

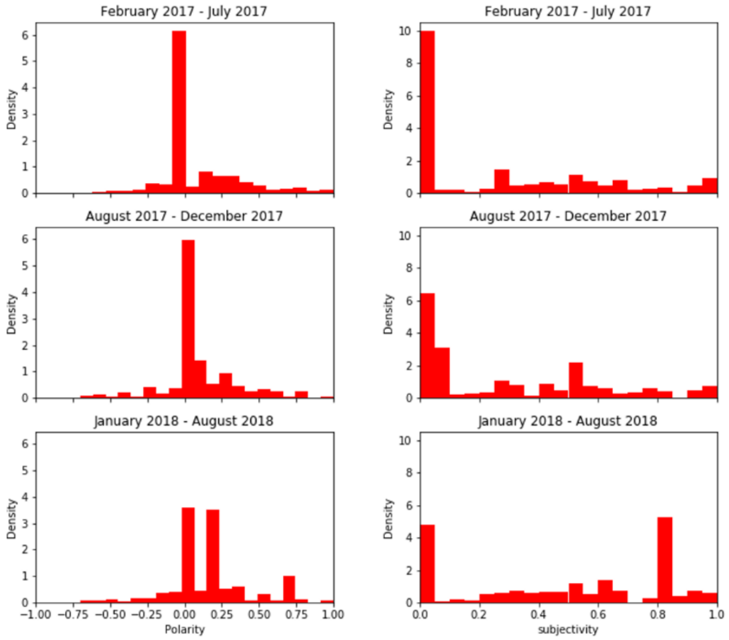
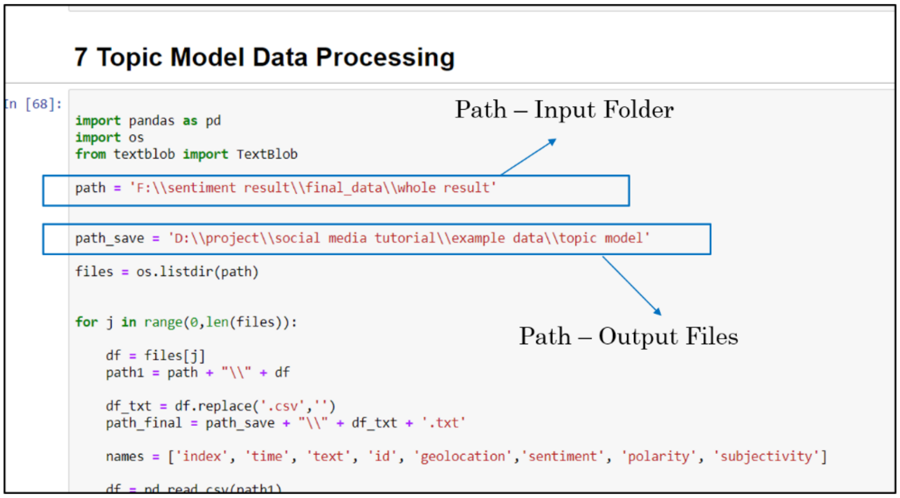


Figure 2.3 Example of the visualization of sentiment analysis

3. Topic Analysis and Visualization

3.1 Data Processing for Topic Analysis

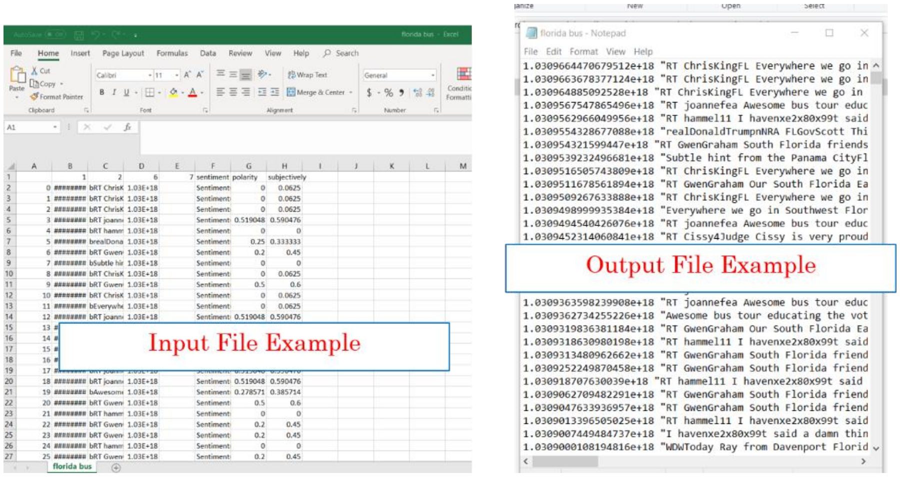
The third part in the Jupyter Notebook is the topic analysis. The first step of the topic analysis is to process the data which can be seen under the heading **'7 Topic Model Data Processing'**, shown in Figure 3.1. From the code, we can change the path of input and output files to save the processed data for topic analysis. Some examples of input and output files are shown in **Figure 3.2**.



7 Topic Model Data Processing

```
In [68]:
import pandas as pd
import os
from textblob import TextBlob

path = 'F:\\sentiment result\\final_data\\whole result'

path_save = 'D:\\project\\social media tutorial\\example data\\topic model'

files = os.listdir(path)


for j in range(0,len(files)):

    df = files[j]
    path1 = path + "\\" + df

    df_txt = df.replace('.csv','')
    path_final = path_save + "\\" + df_txt + '.txt'

    names = ['index', 'time', 'text', 'id', 'geolocation','sentiment', 'polarity', 'subjectivity']

    df = pd.read_csv(path1)
```

Figure 3.1 Example of the topic analysis



	A	B	C	D	E	F	G	H
1		1	2	6	7	sentiment	polarity	subjectively
2	0	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
3	1	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
4	2	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
5	3	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
6	4	########	bRT hamm	1.03E+18		Sentiment	0	0
7	5	########	brealDona	1.03E+18		Sentiment	0.25	0.333333
8	6	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
9	7	########	bSubtle hir	1.03E+18		Sentiment	0	0
10	8	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
11	9	########	bRT Gwen	1.03E+18		Sentiment	0.5	0.6
12	10	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
13	11	########	bEverywhe	1.03E+18		Sentiment	0	0.0625
14	12	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
15	13	#						
16	14	#						
17	15	#						
18	16	#						
19	17	#						
20	18	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
21	19	########	bAwesom	1.03E+18		Sentiment	0.278571	0.385714
22	20	########	bRT Gwen	1.03E+18		Sentiment	0.5	0.6
23	21	########	bRT hamm	1.03E+18		Sentiment	0	0
24	22	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
25	23	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
26	24	########	bRT hamm	1.03E+18		Sentiment	0	0
27	25	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45

```
1.0309664470679512e+18 "RT ChrisKingFL Everywhere we go in
1.0309663678377124e+18 "RT ChrisKingFL Everywhere we go in
1.030964885092528e+18 "RT ChrisKingFL Everywhere we go in
1.0309567547865496e+18 "RT joannefea Awesome bus tour educ
1.0309562966049956e+18 "RT hammel11 I havenxe2x80x99t said
1.0309554328677088e+18 "realDonaldTrumpnNRA FLGovScott Thi
1.030954321599447e+18 "RT GwenGraham South Florida friends
1.0309539232496681e+18 "Subtle hint from the Panama CityFl
1.0309516505743809e+18 "RT ChrisKingFL Everywhere we go in
1.0309511678561894e+18 "RT GwenGraham Our South Florida Ea
1.0309509267633888e+18 "RT ChrisKingFL Everywhere we go in
1.0309498999935384e+18 "Everywhere we go in Southwest Flor
1.0309494540426076e+18 "RT joannefea Awesome bus tour educ
1.0309452314060841e+18 "RT Cissy4Judge Cissy is very proud

1.0309363598239908e+18 "RT joannefea Awesome bus tour educ
1.0309362734255226e+18 "Awesome bus tour educating the vot
1.0309319836381184e+18 "RT GwenGraham Our South Florida Ea
1.0309318630980198e+18 "RT hammel11 I havenxe2x80x99t said
1.0309313480962662e+18 "RT GwenGraham South Florida friend
1.0309252249870458e+18 "RT GwenGraham South Florida friend
1.030918707630039e+18 "RT hammel11 I havenxe2x80x99t said
1.0309062709482291e+18 "RT GwenGraham South Florida friend
1.0309047633936957e+18 "RT GwenGraham South Florida friend
1.0309013396505025e+18 "RT hammel11 I havenxe2x80x99t said
1.0309007449484737e+18 "I havenxe2x80x99t said a damn thin
1.0309000108194816e+18 "WDWToday Ray from Davenport Florid
```

Figure 3.2 Example of the input and output files of topic analysis

3.2 Prerequisite for Topic Analysis

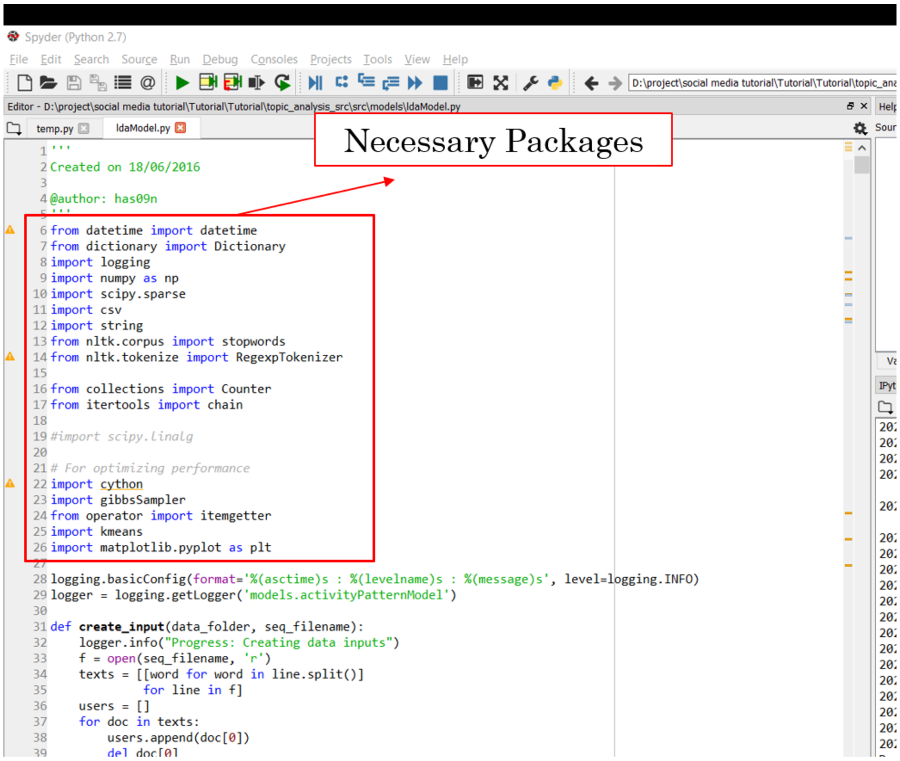
The code for topic analysis is written in Python 2 version which means that we need to use python 2 version to run the topic analysis. We use the Spyder IDE (in anaconda navigator) for python 2.7 version to apply the topic analysis. We can follow the instructions to install the required software and run the model.

- Download Anaconda (python 2.7) 32-bit Graphical Installer
- Open Spyder from the anaconda navigator (python 2.7) version
- Open the ldaModel.py
- Install all the necessary python packages
- Change the input path and file name
- Run the model

3.3 Topic Analysis

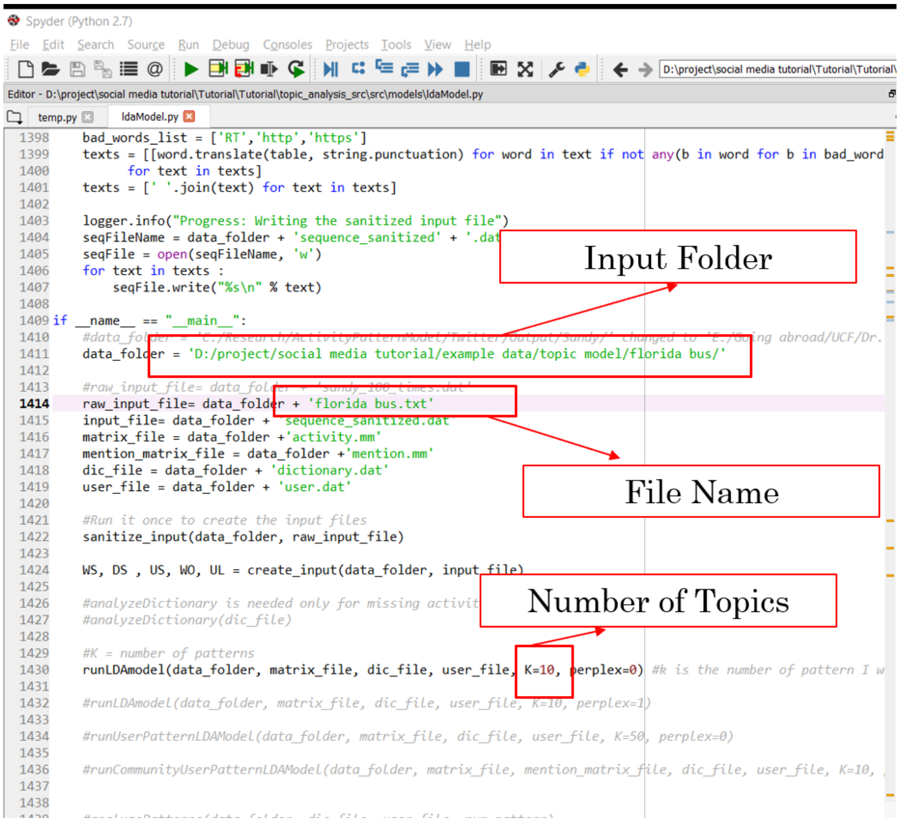
For the topic analysis, we use a tool based on Python 2.7 version environment. Thus, Python 2.7 is required in the topic analysis program. The procedure of topic analysis can be seen as follows:

- Download the ‘topic_analysis_src.rar’ archive and unzip the archive.
- Find the ldaModel.py file, which contains the code for topic analysis.
- Based on the ldaModel.py, install all the required python packages, seen as **Figure 3.3 (a)**.
- Process the raw Twitter data into the input file of topic analysis. The input file of topic analysis contains two column – ‘user_id’ and ‘tweets’, which can be found in Figure 6.
- At the end of the codes (ldaModel.py), the ‘**data_folder**’ (seen as **Figure 3.3 (b))** should be changed into the path where the ‘topic_analysis_src/model’ is and the ‘raw_input_file’ is the path of the input file. The example of input file can be seen as **Figure 3.4.**
- In the **runLDAmodel** function, ***k*** represents the number of topics. Set the number of topics by changing the value of ‘**k**’.
- After running all the above process, run the ldaModel.py and the results will be saved in the same path of the ‘topic_analysis_src’. The output file name is ‘RT_LDA_patterns’.


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\project\social media tutorial\Tutorial\Tutorial\topic_ana
Editor - D:\project\social media tutorial\Tutorial\Tutorial\topic_analysis_src\src\models\ldaModel.py
temp.py
ldaModel.py
Help
Sour
Necessary Packages

```
'''
Created on 18/06/2016

@author: has09n
'''
from datetime import datetime
from dictionary import Dictionary
import logging
import numpy as np
import scipy.sparse
import csv
import string
from nltk.corpus import stopwords
from nltk.tokenize import RegexpTokenizer

from collections import Counter
from itertools import chain

#import scipy.linalg

# For optimizing performance
import cython
import gibbsSampler
from operator import itemgetter
import kmeans
import matplotlib.pyplot as plt

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)
logger = logging.getLogger('models.activityPatternModel')

def create_input(data_folder, seq_filename):
    logger.info("Progress: Creating data inputs")
    f = open(seq_filename, 'r')
    texts = [[word for word in line.split()]
              for line in f]
    users = []
    for doc in texts:
        users.append(doc[0])
        del doc[0]
```

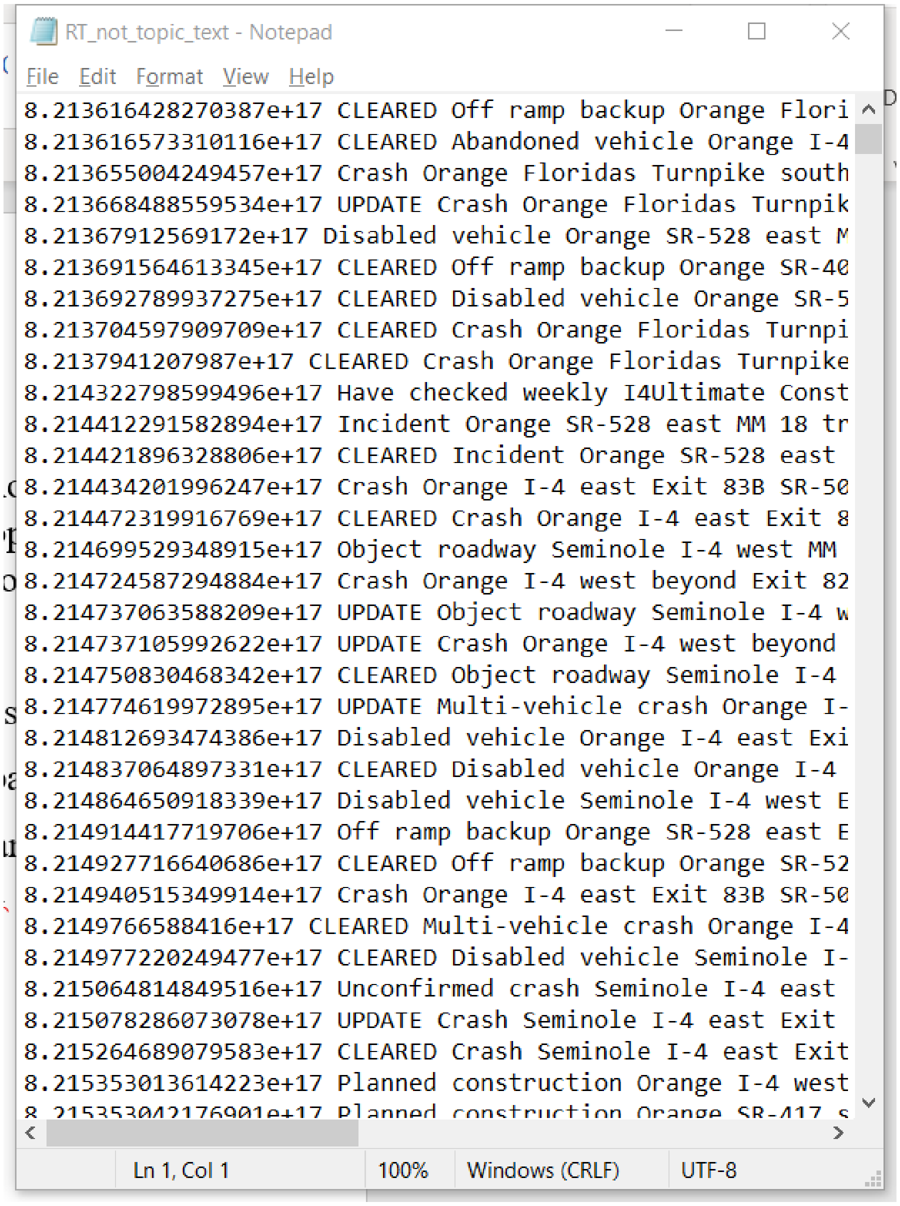
(a)


Spyder (Python 2.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\project\social media tutorial\Tutorial\Tutorial\
Editor - D:\project\social media tutorial\Tutorial\Tutorial\topic_analysis_src\src\models\ldaModel.py
temp.py
ldaModel.py
Input Folder
File Name
Number of Topics

```python
    bad_words_list = ['RT','http','https']
    texts = [[word.translate(table, string.punctuation) for word in text if not any(b in word for b in bad_word
           for text in texts]
    texts = [' '.join(text) for text in texts]

    logger.info("Progress: Writing the sanitized input file")
    seqFileName = data_folder + 'sequence_sanitized' + '.dat
    seqFile = open(seqFileName, 'w')
    for text in texts :
        seqFile.write("%s\n" % text)

if __name__ == "__main__":
    #data_folder = 'C:/Research/ActivityPatternModel/Twitter/Output/Sandy/' changed to 'E:/Going abroad/UCF/Dr.
    data_folder = 'D:/project/social media tutorial/example data/topic model/florida bus/'

    #raw_input_file= data_folder + 'sandy_100_times.dat'
    raw_input_file= data_folder + 'florida bus.txt'
    input_file= data_folder + 'sequence_sanitized.dat'
    matrix_file = data_folder +'activity.mm'
    mention_matrix_file = data_folder +'mention.mm'
    dic_file = data_folder + 'dictionary.dat'
    user_file = data_folder + 'user.dat'

    #Run it once to create the input files
    sanitize_input(data_folder, raw_input_file)

    WS, DS , US, WO, UL = create_input(data_folder, input_file)

    #analyzeDictionary is needed only for missing activit
    #analyzeDictionary(dic_file)

    #K = number of patterns
    runLDAmodel(data_folder, matrix_file, dic_file, user_file, K=10, perplex=0) #k is the number of pattern I w

    #runLDAmodel(data_folder, matrix_file, dic_file, user_file, K=10, perplex=1)

    #runUserPatternLDAModel(data_folder, matrix_file, dic_file, user_file, K=50, perplex=0)

    #runCommunityUserPatternLDAModel(data_folder, matrix_file, mention_matrix_file, dic_file, user_file, K=10,


    #analyzePatterns(data_folder, dic_file, user_file, num_pattern)
```

(b)

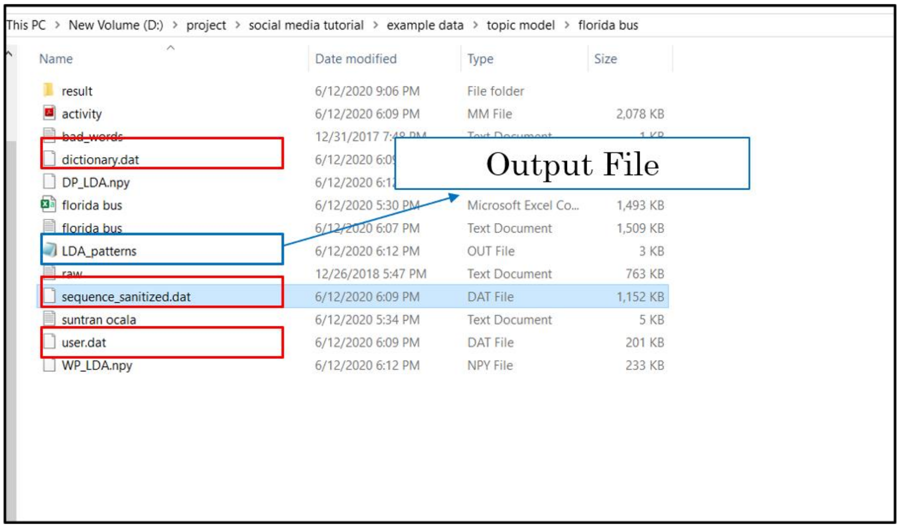
Figure 3.3 Example of the topic analysis: (a) necessary packages; (b) input path



```
8.213616428270387e+17 CLEARED Off ramp backup Orange Flori
8.213616573310116e+17 CLEARED Abandoned vehicle Orange I-4
8.213655004249457e+17 Crash Orange Floridas Turnpike south
8.213668488559534e+17 UPDATE Crash Orange Floridas Turnpik
8.21367912569172e+17 Disabled vehicle Orange SR-528 east M
8.213691564613345e+17 CLEARED Off ramp backup Orange SR-40
8.213692789937275e+17 CLEARED Disabled vehicle Orange SR-5
8.213704597909709e+17 CLEARED Crash Orange Floridas Turnpi
8.2137941207987e+17 CLEARED Crash Orange Floridas Turnpike
8.214322798599496e+17 Have checked weekly I4Ultimate Const
8.214412291582894e+17 Incident Orange SR-528 east MM 18 tr
8.214421896328806e+17 CLEARED Incident Orange SR-528 east
8.214434201996247e+17 Crash Orange I-4 east Exit 83B SR-50
8.214472319916769e+17 CLEARED Crash Orange I-4 east Exit 8
8.214699529348915e+17 Object roadway Seminole I-4 west MM
8.214724587294884e+17 Crash Orange I-4 west beyond Exit 82
8.214737063588209e+17 UPDATE Object roadway Seminole I-4 w
8.214737105992622e+17 UPDATE Crash Orange I-4 west beyond
8.214750830468342e+17 CLEARED Object roadway Seminole I-4
8.214774619972895e+17 UPDATE Multi-vehicle crash Orange I-
8.214812693474386e+17 Disabled vehicle Orange I-4 east Exi
8.214837064897331e+17 CLEARED Disabled vehicle Orange I-4
8.214864650918339e+17 Disabled vehicle Seminole I-4 west E
8.214914417719706e+17 Off ramp backup Orange SR-528 east E
8.214927716640686e+17 CLEARED Off ramp backup Orange SR-52
8.214940515349914e+17 Crash Orange I-4 east Exit 83B SR-50
8.2149766588416e+17 CLEARED Multi-vehicle crash Orange I-4
8.214977220249477e+17 CLEARED Disabled vehicle Seminole I-
8.215064814849516e+17 Unconfirmed crash Seminole I-4 east
8.215078286073078e+17 UPDATE Crash Seminole I-4 east Exit
8.215264689079583e+17 CLEARED Crash Seminole I-4 east Exit
8.215353013614223e+17 Planned construction Orange I-4 west
8.215353042176901e+17 Planned construction Orange SR-417 s
```

Figure 3.4 Example of the input file

There are also three required files – dictionary.dat, sequence_sanitized.dat and user.dat, seen as **Figure 3.5.**



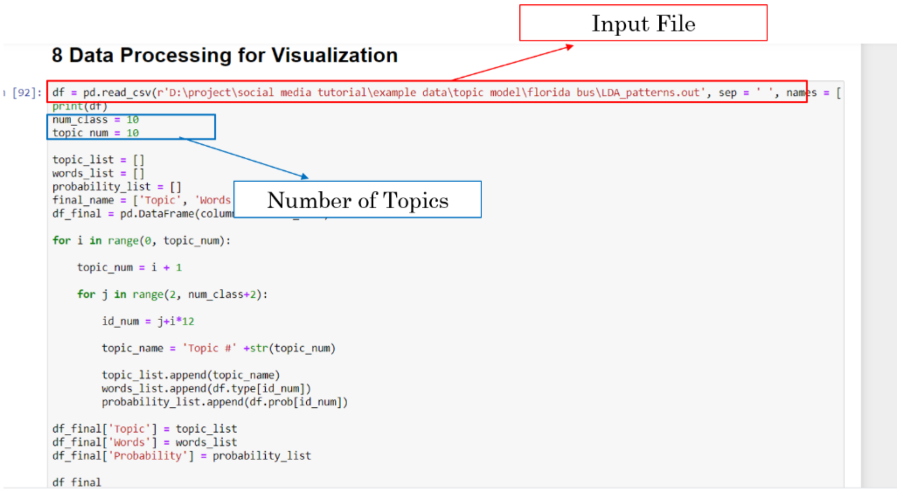
This PC > New Volume (D:) > project > social media tutorial > example data > topic model > florida bus

Name	Date modified	Type	Size
result	6/12/2020 9:06 PM	File folder	
activity	6/12/2020 6:09 PM	MM File	2,078 KB
bad_words	12/31/2017 7:48 PM	Text Document	1 KB
dictionary.dat	6/12/2020 6:09		
DP_LDA.npy	6/12/2020 6:12		
florida bus	6/12/2020 5:30 PM	Microsoft Excel Co...	1,493 KB
florida bus	6/12/2020 6:07 PM	Text Document	1,509 KB
LDA_patterns	6/12/2020 6:12 PM	OUT File	3 KB
raw	12/26/2018 5:47 PM	Text Document	763 KB
sequence_sanitized.dat	6/12/2020 6:09 PM	DAT File	1,152 KB
suntran ocala	6/12/2020 5:34 PM	Text Document	5 KB
user.dat	6/12/2020 6:09 PM	DAT File	201 KB
WP_LDA.npy	6/12/2020 6:12 PM	NPY File	233 KB

Figure 3.5 Example of required files and output file for topic analysis

3.4 Topic Analysis Results Visualization

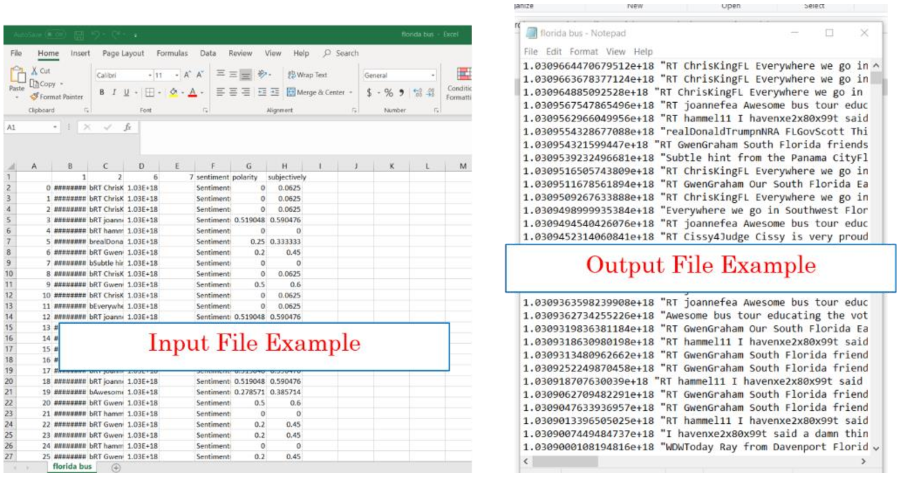
The first step for visualization of the topic analysis results is to process the data which can be seen under the heading '**8 Data Processing for Visualization**'. In the codes, the '**path_input**' is the path of the input file (format of the input file can be seen as **Figure 3.7**). The code for topic analysis visualization can be seen as **Figure 3.8.** In the code, we should change the input file path to read the input file and output figure path to save the output figures.



8 Data Processing for Visualization

```
[92]: df = pd.read_csv(r'D:\project\social media tutorial\example data\topic model\florida bus\LDA_patterns.out', sep = ' ', names = [
print(df)
num_class = 10
topic_num = 10

topic_list = []
words_list = []
probability_list = []
final_name = ['Topic', 'Words
df_final = pd.DataFrame(colum

for i in range(0, topic_num):

    topic_num = i + 1

    for j in range(2, num_class+2):

        id_num = j+i*12

        topic_name = 'Topic #' +str(topic_num)

        topic_list.append(topic_name)
        words_list.append(df.type[id_num])
        probability_list.append(df.prob[id_num])

df_final['Topic'] = topic_list
df_final['Words'] = words_list
df_final['Probability'] = probability_list

df_final
```

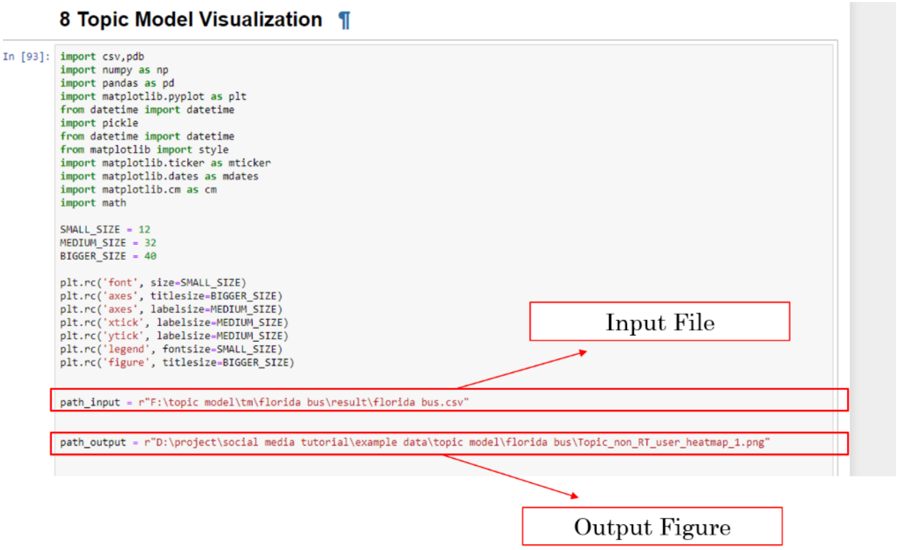
Figure 3.6 Data Processing for Visualization of Topic Analysis



	A	B	C	D	E	F	G	H
1		1	2	6	7	sentiment	polarity	subjectively
2	0	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
3	1	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
4	2	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
5	3	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
6	4	########	bRT hamm	1.03E+18		Sentiment	0	0
7	5	########	brealDona	1.03E+18		Sentiment	0.25	0.333333
8	6	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
9	7	########	bSubtle hir	1.03E+18		Sentiment	0	0
10	8	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
11	9	########	bRT Gwen	1.03E+18		Sentiment	0.5	0.6
12	10	########	bRT ChrisK	1.03E+18		Sentiment	0	0.0625
13	11	########	bEverywhe	1.03E+18		Sentiment	0	0.0625
14	12	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
15	13	#						
16	14	#						
17	15	#						
18	16	#						
19	17	[illegible]	[illegible]	[illegible]		[illegible]	[illegible]	[illegible]
20	18	########	bRT joann	1.03E+18		Sentiment	0.519048	0.590476
21	19	########	bAwesom	1.03E+18		Sentiment	0.278571	0.385714
22	20	########	bRT Gwen	1.03E+18		Sentiment	0.5	0.6
23	21	########	bRT hamm	1.03E+18		Sentiment	0	0
24	22	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
25	23	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45
26	24	########	bRT hamm	1.03E+18		Sentiment	0	0
27	25	########	bRT Gwen	1.03E+18		Sentiment	0.2	0.45

```
1.0309664470679512e+18 "RT ChrisKingFL Everywhere we go in
1.0309663678377124e+18 "RT ChrisKingFL Everywhere we go in
1.030964885092528e+18 "RT ChrisKingFL Everywhere we go in
1.0309567547865496e+18 "RT joannefea Awesome bus tour educ
1.0309562966049956e+18 "RT hammel11 I havenxe2x80x99t said
1.0309554328677088e+18 "realDonaldTrumpnNRA FLGovScott Thi
1.030954321599447e+18 "RT GwenGraham South Florida friends
1.0309539232496681e+18 "Subtle hint from the Panama CityFl
1.0309516505743809e+18 "RT ChrisKingFL Everywhere we go in
1.0309511678561894e+18 "RT GwenGraham Our South Florida Ea
1.0309509267633888e+18 "RT ChrisKingFL Everywhere we go in
1.0309498999935384e+18 "Everywhere we go in Southwest Flor
1.0309494540426076e+18 "RT joannefea Awesome bus tour educ
1.0309452314060841e+18 "RT Cissy4Judge Cissy is very proud
1.0309363598239908e+18 "RT joannefea Awesome bus tour educ
1.0309362734255226e+18 "Awesome bus tour educating the vot
1.0309319836381184e+18 "RT GwenGraham Our South Florida Ea
1.0309318630980198e+18 "RT hammel11 I havenxe2x80x99t said
1.0309313480962662e+18 "RT GwenGraham South Florida friend
1.0309252249870458e+18 "RT GwenGraham South Florida friend
1.030918707630039e+18 "RT hammel11 I havenxe2x80x99t said
1.0309062709482291e+18 "RT GwenGraham South Florida friend
1.0309047633936957e+18 "RT GwenGraham South Florida friend
1.0309013396505025e+18 "RT hammel11 I havenxe2x80x99t said
1.0309007449484737e+18 "I havenxe2x80x99t said a damn thin
1.0309000108194816e+18 "WDWToday Ray from Davenport Florid
```

Figure 3.7 Data Samples for Visualization of Topic Analysis



```
import csv,pdb
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
import pickle
from datetime import datetime
from matplotlib import style
import matplotlib.ticker as mticker
import matplotlib.dates as mdates
import matplotlib.cm as cm
import math

SMALL_SIZE = 12
MEDIUM_SIZE = 32
BIGGER_SIZE = 40

plt.rc('font', size=SMALL_SIZE)
plt.rc('axes', titlesize=BIGGER_SIZE)
plt.rc('axes', labelsize=MEDIUM_SIZE)
plt.rc('xtick', labelsize=MEDIUM_SIZE)
plt.rc('ytick', labelsize=MEDIUM_SIZE)
plt.rc('legend', fontsize=SMALL_SIZE)
plt.rc('figure', titlesize=BIGGER_SIZE)

path_input = r"F:\topic model\tm\florida bus\result\florida bus.csv"

path_output = r"D:\project\social media tutorial\example data\topic model\florida bus\Topic_non_RT_user_heatmap_1.png"
```

Figure 3.8 Example of Visualization of Topic Analysis

Run this code with the input file, the figure can be shown. **Figure 3.9** shows one of the examples.

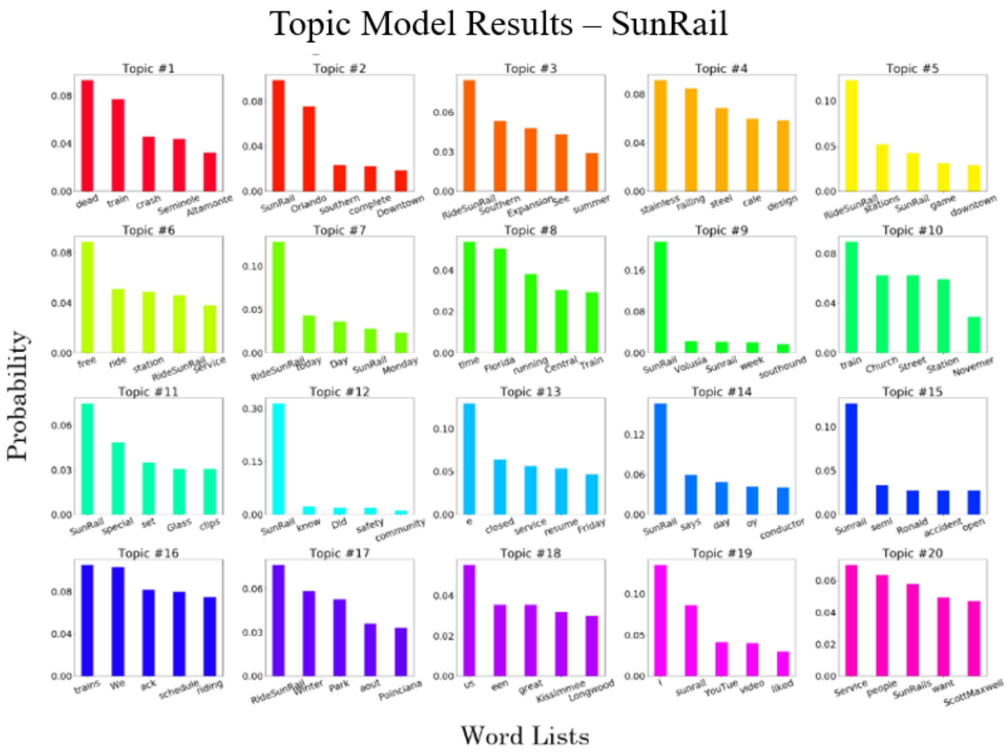


Figure 3.9 Example of the visualization of topic analysis