**PROJECT REPORT**

**Introduction:**

The BDM Assignment required us to setup a infrastructure that reads streaming data using an API known as tweepy which helps us to read tweets from a Social Media Site such as Twitter. It required us to parse and process the streaming data as JSON. and is stored simultaneously in a NoSQL database such as MongoDB. We’re also able to perform some simple analytics on this data and provide some results.

**Requirements: -** The following software or packages are required to be installed or should be pre-installed before the running the twitter streaming and analysis program.

**Cloudera:** Cloudera delivers the modern platform for data management & analytics & can be used for gathering twitter streaming data into a text file and for analysing it using text mining analysis which constitutes 80% of our program code. However, the real problem lies in working up a noSQL database environment like MongoDB in a virtual environment such as cloudera, which was the problem we faced and hence we decided windows was preferable.

**Windows:** The OS that we will be performing our twitter program on.

**CMD:** Windows’command prompt will be used to setup our mongoDB environment in the background to establish connection with our twitter streaming program - TweetRead.py

**MongoDB:** MongoDB is our noSQL database environment which is flexible enough to store the recently streamed tweets that we’ll be working on.

**Anaconda Package:** Anaconda is a python and R distribution which aims to provide everything you need (python wise) for data science applications. It includes:  
  
>The core python language (In our case python3)  
>100+ python "packages" (libraries like pandas, matplotlib, json, re, etc)  
Spyder (IDE/editor)  
Anaconda Prompt (Anaconda’s own command prompt)

**Anaconda Prompt:** As mentioned above it is Anaconda’s own CMD which can be used as a direct python3 compiler to type and execute python scripts.

**Spyder:** Spyder is a IDE used for scientific programming in Python. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source softwares.

For carrying out our twitter python program, we must have python2x or python3x installed in our cloudera or windows os platforms, then we need the following python libraries installed:

**Tweepy:** open-sourced, hosted on GitHub and enables Python to communicate with Twitter platform and use its API. This is an easy-to-use Python library for accessing the Twitter API.

**PyMongo:** Python package containing tools for working with MongoDB, & is recommended to work with MongoDB from Python.

**Pandas:** This is an open source library providing high-performance, easy-to-use data structures and data analysis tools.  
  
**Matplotlib:** This is a Python 2D plotting library which produces publication quality figures in a variety of hard copy formats and interactive environments across platforms.

**Json:** An API similar to pickle for converting in-memory Python objects to a serialized representation known as JavaScript Object Notation.

**Re:** A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. The module re provides full support for Perl-like regular expressions in Python.

These are the steps which we are going to carry out :

1. TweetRead.py - Extract twitter data using tweepy and learn how to handle it using pandas storing it in MongoDb using pymongo.
2. TweetAnalysis.py - Do some basic statistics and visualizations with re, matplotlib and display visualizations with iPython.

**1.Extracting twitter data:**

In order to extract tweets for a posterior analysis, we need to access to our Twitter account and create an app. The website to do this is <https://apps.twitter.com/>. After creating an app in twitter it will provide us with the following codes :

* Consumer Key (API Key)
* Consumer Secret (API Secret)
* Access Token
* Access Token Secret

from pymongo import MongoClient  
import json  
#Import the necessary methods from tweepy library  
from tweepy.streaming import StreamListener  
from tweepy import OAuthHandler  
from tweepy import Stream  
  
#Variables that contains the user credentials to access Twitter API   
access\_token = "993861826910785536-SRwlwm6u1QnlMB8efcPYWURlniZi3o6"  
access\_token\_secret = "YaxmO5FV7cVaLtb5xxxBMGtRr3HECFQSE7rAPrSwY3DSO"  
consumer\_key = "YGxg6Pv3b5DNIB4O9vIxhO8it"  
consumer\_secret = "dzqEyOncIN3TQ2ViOooPpEdKMxUz90Ub368RJw15vUXiIQamdi"  
  
  
#This is a basic listener that just prints received tweets to stdout.  
class StdOutListener(StreamListener):  
  
 def on\_data(self, data):  
 client = MongoClient('localhost', 27017)  
 db = client['twitter\_db']  
 collection = db['twitter\_collection']  
 tweet = json.loads(data)  
   
 collection.insert(tweet)  
   
 print(data)  
 return True  
  
 def on\_error(self, status):  
 print(status)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':

The Below code establishes a connection with the twitter and authenticates the twitter account with which we have created the twitter app using the consumer\_key, consumer\_secret, access\_token, access\_token\_secret.  
  
 #This handles Twitter authentication and the connection to Twitter Streaming API  
 l = StdOutListener()  
 auth = OAuthHandler(consumer\_key, consumer\_secret)  
 auth.set\_access\_token(access\_token, access\_token\_secret)  
 stream = Stream(auth, l)  
  
 #This line filter Twitter Streams to capture data by the keywords: 'trump', 'modi', 'putin'  
 stream.filter(track=['trump', 'modi', 'putin'])

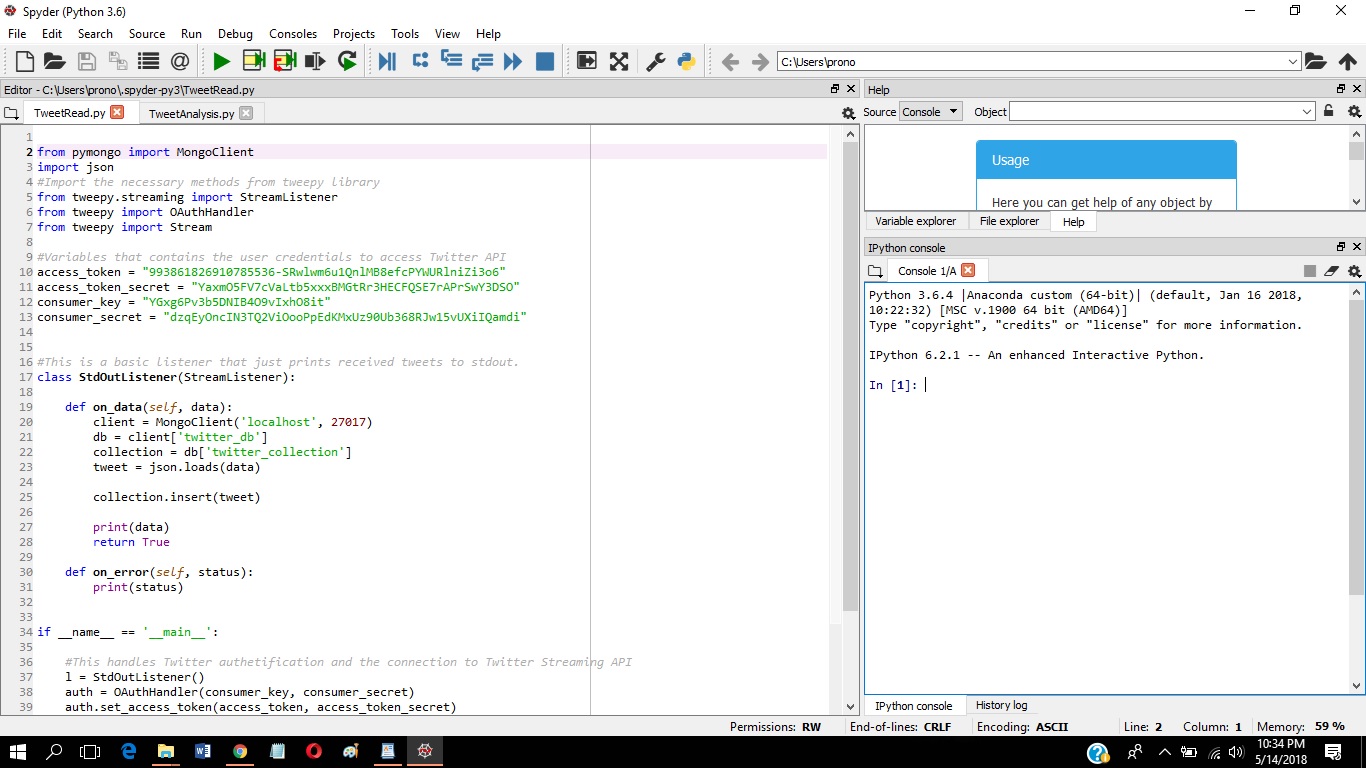
**2. Visualization and basic statistics**

We are using pandas and matplotlib to do some analysis on the collected tweets and display the results in the form of graphs.

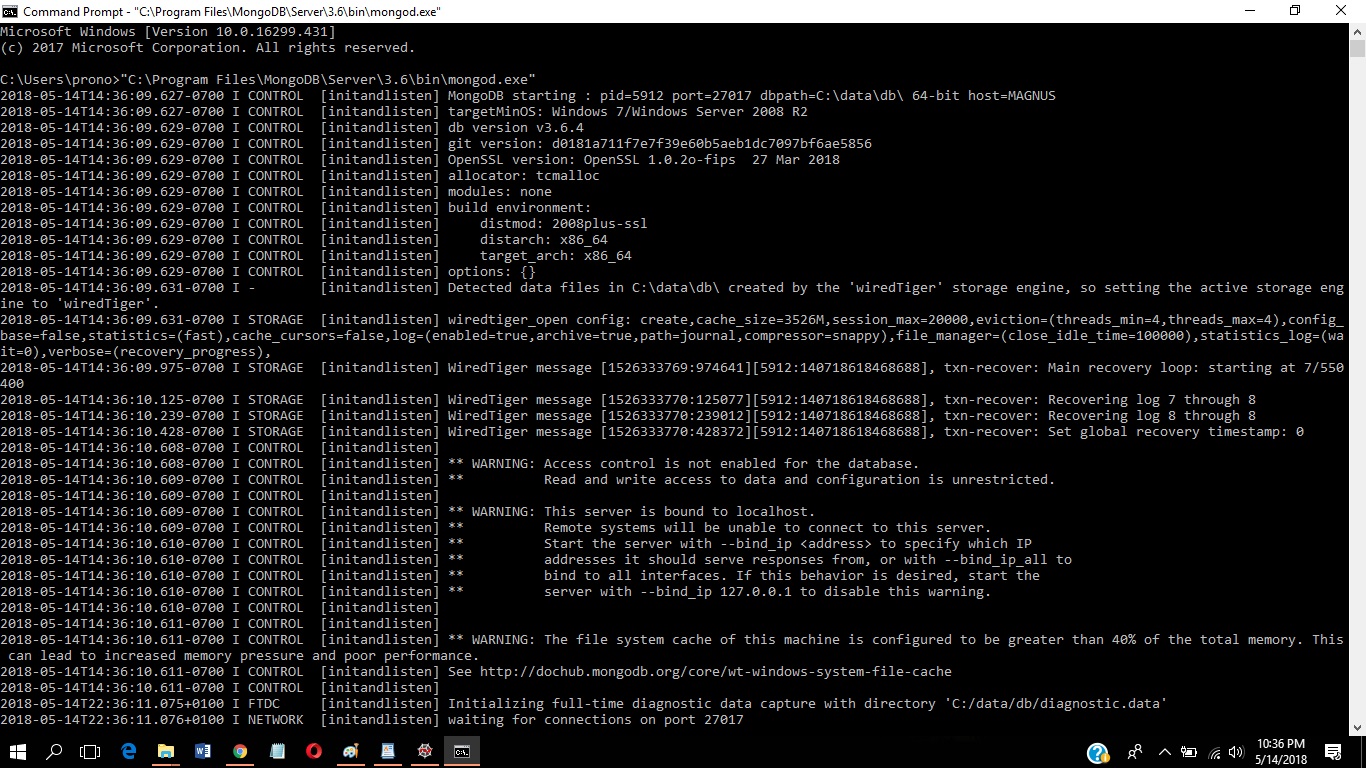
import json  
import pandas as pd  
import matplotlib.pyplot as plt  
import re  
  
#preprocessing and parsing of the text is done here,all the text is converted into lowercase letters  
def word\_in\_text(word, text):  
 word = word.lower()  
 text = text.lower()  
 match = re.search(word, text)  
 if match:  
 return True  
 return False  
  
#this function allows to extract all the links in the given file  
def extract\_link(text):  
 regex = r'https?://[^\s<>"]+|www\.[^\s<>"]+'  
 match = re.search(regex, text)  
 if match:  
 return match.group()  
 return ''  
  
  
def main():  
 # Reading Tweets  
 print('Reading Tweets\n')  
 tweets\_data\_path = "C:/Users/prono/.spyder-py3/twitter\_data.txt"  
  
 tweets\_data = []  
 tweets\_file = open(tweets\_data\_path, "r")  
 for line in tweets\_file:  
 try:  
 tweet = json.loads(line)  
 tweets\_data.append(tweet)  
 except:  
 continue  
  
 print(len(tweets\_data))  
   
 # Structuring Tweets  
 print('Structuring Tweets\n')  
 tweets = pd.DataFrame()  
 tweets['text'] = list([tweet.get('text','') for tweet in tweets\_data])  
 tweets['lang'] = list([tweet.get('lang','') for tweet in tweets\_data])  
 tweets['country'] = list([tweet.get('place','').get('country','') if tweet.get('place') != None else None for tweet in tweets\_data])  
  
  
 # Analyzing Tweets by Language  
 print('Analyzing tweets by language\n')  
 tweets\_by\_lang = tweets['lang'].value\_counts()  
 fig, ax = plt.subplots()  
 ax.tick\_params(axis='x', labelsize=15)  
 ax.tick\_params(axis='y', labelsize=10)  
 ax.set\_xlabel('Languages', fontsize=15)  
 ax.set\_ylabel('Number of tweets', fontsize=15)  
 ax.set\_title('Top 5 languages', fontsize=15, fontweight='bold')  
 tweets\_by\_lang[:5].plot(ax=ax, kind='bar', color='red')  
 plt.savefig('tweet\_by\_lang', format='png')  
  
  
 # Analyzing Tweets by Country  
 print('Analyzing tweets by country\n')  
 tweets\_by\_country = tweets['country'].value\_counts()  
 fig, ax = plt.subplots()  
 ax.tick\_params(axis='x', labelsize=15)  
 ax.tick\_params(axis='y', labelsize=10)  
 ax.set\_xlabel('Countries', fontsize=15)  
 ax.set\_ylabel('Number of tweets', fontsize=15)  
 ax.set\_title('Top 5 countries', fontsize=15, fontweight='bold')  
 tweets\_by\_country[:5].plot(ax=ax, kind='bar', color='blue')  
 plt.savefig('tweet\_by\_country', format='png')  
  
  
 # Adding president columns to the tweets DataFrame  
 print('Adding president tags to the data\n')  
 tweets['trump'] = tweets['text'].apply(lambda tweet: word\_in\_text('trump', tweet))  
 tweets['modi'] = tweets['text'].apply(lambda tweet: word\_in\_text('modi', tweet))  
 tweets['putin'] = tweets['text'].apply(lambda tweet: word\_in\_text('putin', tweet))  
  
  
 # Analyzing Tweets by president: First attempt  
 print('Analyzing tweets by president: First attempt\n')  
 prg\_langs = ['trump', 'modi', 'putin']  
 tweets\_by\_prg\_lang = [tweets['trump'].value\_counts()[True], tweets['modi'].value\_counts()[True],  
 tweets['putin'].value\_counts()[True]]  
 x\_pos = list(range(len(prg\_langs)))  
 width = 0.8  
 fig, ax = plt.subplots()  
 plt.bar(x\_pos, tweets\_by\_prg\_lang, width, alpha=1, color='g')  
 ax.set\_ylabel('Number of tweets', fontsize=15)  
 ax.set\_title('Ranking: trump vs. modi vs. putin (Raw data)', fontsize=10, fontweight='bold')  
 ax.set\_xticks([p + 0.4 \* width for p in x\_pos])  
 ax.set\_xticklabels(prg\_langs)  
 plt.grid()  
 plt.savefig('tweet\_by\_president\_1', format='png')  
  
  
 # Targeting relevant tweets  
 print('Targeting relevant tweets\n')  
 tweets['president'] = tweets['text'].apply(lambda tweet: word\_in\_text('president', tweet))  
 tweets['best'] = tweets['text'].apply(lambda tweet: word\_in\_text('best', tweet))  
 tweets['relevant'] = tweets['text'].apply(  
 lambda tweet: word\_in\_text('president', tweet) or word\_in\_text('best', tweet))  
  
  
 # Analyzing Tweets by president: Second attempt  
 print('Analyzing tweets by president: First attempt\n')  
  
 import IPython  
 IPython.embed()  
  
 def get\_value\_counts(tweets, language):  
 try:  
 return tweets[tweets['relevant'] == True][language].value\_counts()[True]  
 except KeyError:  
 return 0  
  
 tweets\_by\_prg\_lang = [get\_value\_counts(tweets, 'trump'),  
 get\_value\_counts(tweets, 'modi'),  
 get\_value\_counts(tweets, 'putin')]  
 x\_pos = list(range(len(prg\_langs)))  
 width = 0.8  
 fig, ax = plt.subplots()  
 plt.bar(x\_pos, tweets\_by\_prg\_lang, width, alpha=1, color='g')  
 ax.set\_ylabel('Number of tweets', fontsize=15)  
 ax.set\_title('Ranking: trump vs. modi vs. putin (Relevant data)', fontsize=10, fontweight='bold')  
 ax.set\_xticks([p + 0.4 \* width for p in x\_pos])  
 ax.set\_xticklabels(prg\_langs)  
 plt.grid()  
 plt.savefig('tweet\_by\_president\_2', format='png')  
  
  
 # Extracting Links  
 tweets['link'] = tweets['text'].apply(lambda tweet: extract\_link(tweet))  
 tweets\_relevant = tweets[tweets['relevant'] == True]  
 tweets\_relevant\_with\_link = tweets\_relevant[tweets\_relevant['link'] != '']  
  
 print('\n Below are some Python links that we extracted\n')  
 print(tweets\_relevant\_with\_link[tweets\_relevant\_with\_link['trump'] == True]['link'].head())  
  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Procedure:** The steps are outlined as follows:

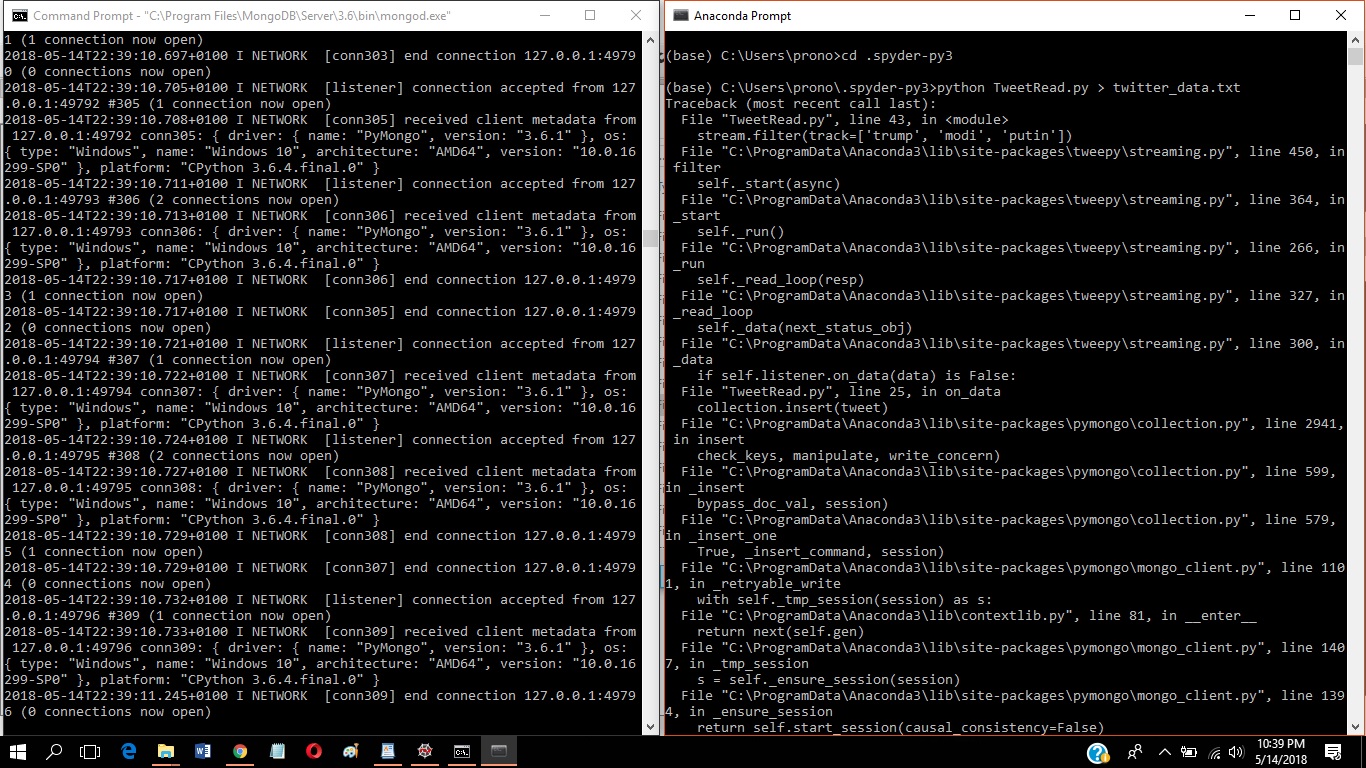
1) Opening TweetRead.py on Spyder IDE.



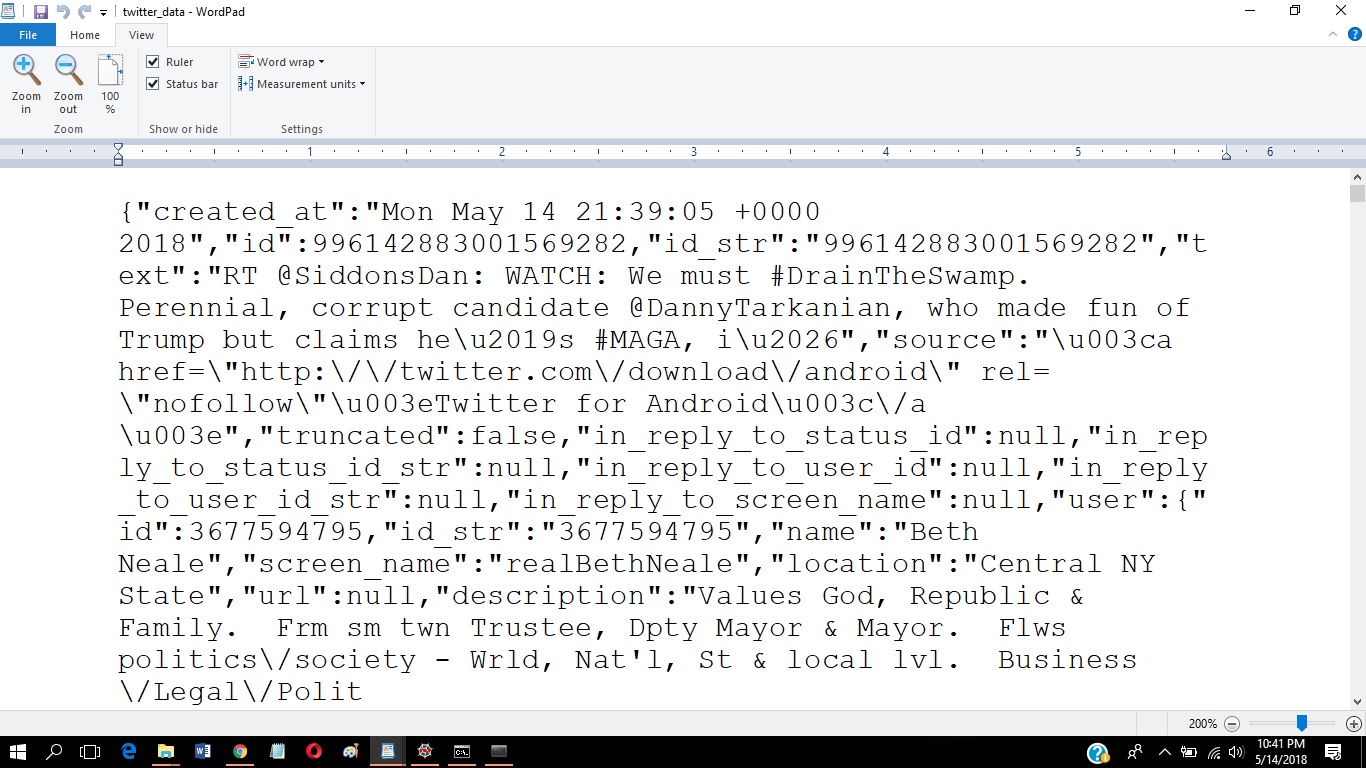
2) Running MongoDB (Mongod.exe) in Background.



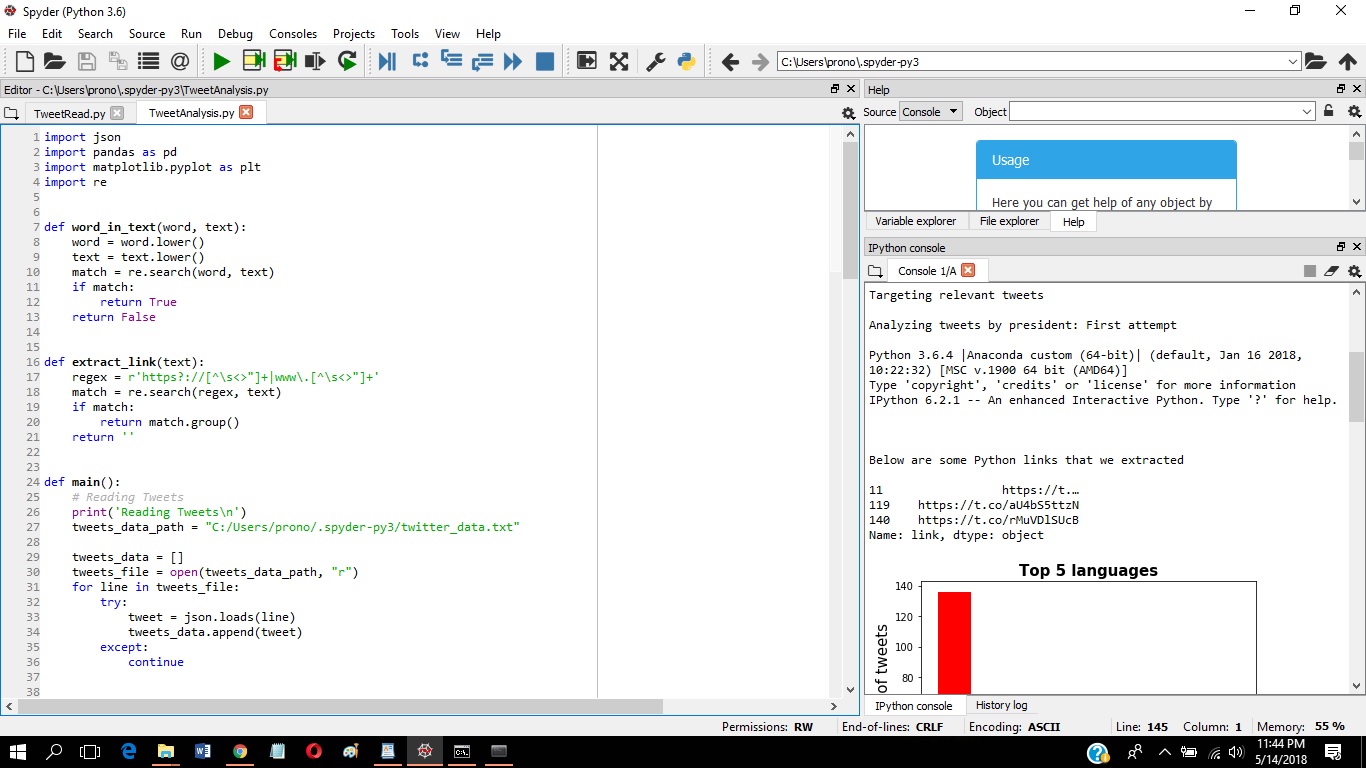
3) In Anaconda Prompt, typing the command - “ python TweetRead.py > twitter\_data.txt “ to save streamed data to text file. Simultaneously, opened connection with MongoDB to store recent tweets.



4) Pressing Ctrl+C stops the streaming process and we can view streamed data in the twitter\_data.txt text file, same data is stored in mongodb.



5) We perform simple text mining analysis on the tweets collected using our TweetAnalysis.py program.



**OUTPUT:** Analysis is done for the recently twitter tweets data that were streamed and stored simultaneously on MongoDB and a text file. The tweets are extracted from twitter\_data.txt file and text mining analysis is performed on the them to gain some information. Note that the output may or may not be the same as the analysis depends on recent tweets on the topics for ‘trump’, ‘modi’ and ‘putin’

Reading Tweets

154

Structuring Tweets

Analyzing tweets by language

Analyzing tweets by country

Adding president tags to the data

Analyzing tweets by president: First attempt

Targeting relevant tweets

Analyzing tweets by president: First attempt

Python 3.6.4 |Anaconda custom (64-bit)| (default, Jan 16 2018, 10:22:32) [MSC v.1900 64 bit (AMD64)]

Type 'copyright', 'credits' or 'license' for more information

IPython 6.2.1 -- An enhanced Interactive Python. Type '?' for help.

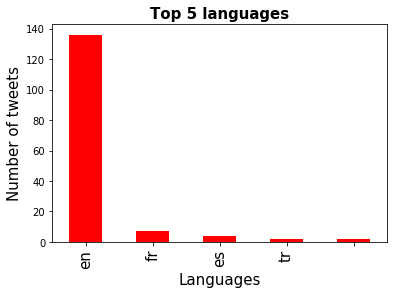
Below are some Python links that we extracted

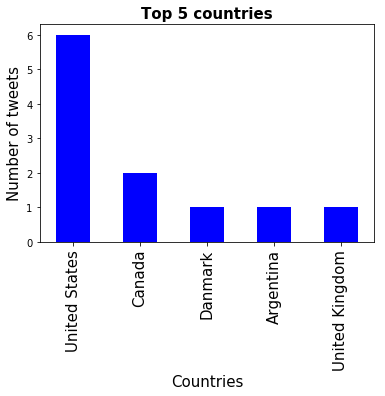
11 https://t.…

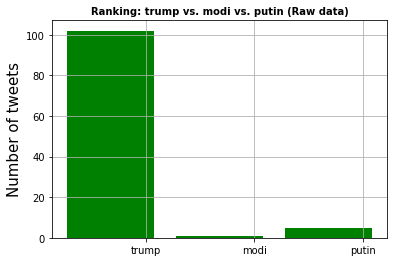
119 https://t.co/aU4bS5ttzN

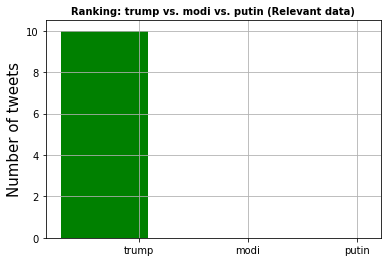
140 https://t.co/rMuVDlSUcB

Name: link, dtype: object

**￼**

****

****

****

**￼**

**￼**

**￼**