CS5540: Principles of Big Data Management

A system to store, analyze, and visualize Twitter's tweets

Project Phase 1 Document

Team # 7

Avni Mehta

Sneha Mishra

Arvind Tota

Step 1: Collect tweets using Twitter's Streaming API

Python Code for Tweets Collection

Collect_Tweets.py

```
# ------------------------------------------------------------------------
#  * CS5540: Principles of Big Data Management
#  * Project Phase 1
#  * Team #7: Avni Mehta, Sneha Mishra, Arvind Tota
# ------------------------------------------------------------------------

# Import necessary methods from tweepy library
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

# Import other libraries
import datetime


# This is a basic listener for received tweets to stdout.
class TwitterListener(StreamListener):

    tweet_number = 0                    # class variable

    def __init__(self,max_tweets):
        super().__init__()
        self.max_tweets=max_tweets          # max number of tweets
        self.last = datetime.datetime.now()  # last time for status

    def on_data(self, data):

        # Write to file
        with open('twitter_data.txt', 'a') as my_file:
            my_file.write(data)

        # Count no. of tweets collected
        self.tweet_number += 1

```

Collect_Tweets.py

```
        # Stop when the limit is reached
        if self.tweet_number >= self.max_tweets:
            print('\tStopping data collection : Limit of '+str(self.max_tweets)+' tweets reached.')
            # Return False to the listener's on_data() of streaming.py API
            return False

        self.status(self.tweet_number)
        return True

    # Status method prints out tweet counts every ten minutes
    def status(self, tweetCount):
        now = datetime.datetime.now()
        # Print status every 10 mins
        if (now - self.last).total_seconds() > 600:
            print('\t' + str(tweetCount) + ' tweets collected..')
            self.last = now

    # On error, print error status
    def on_error(self, status):
        """ Handles the response error status. """
        print('Error status code : ' + str(status))

```

Collect_Tweets.py

```
56
57
58  # Main Activity
59  if __name__ == '__main__':
60      print('\n---------------------------------------------------------------------------------------')
61      print('Big Data Project :')
62      print('Collect 100K+ tweets, extract hashtags and urls and run wordcount using Hadoop & Spark')
63      print('---------------------------------------------------------------------------------------')
64
65      # Variables that contains the user credentials to access Twitter API
66      access_token = "712180562-wnFa9ahIaiR7mFZrHyodaOmYepgl0cL2Rsr2bGfs"
67      access_token_secret = "GHYZH6CpQouvUS3EeRSbrGMOtxHPkBfLM9r6dvX9MYHAW"
68      consumer_key = "bR6Laeo9MOxlvqCVNvBlwPuvn"
69      consumer_secret = "fmjeR54JsgF5yQouBBcmubWZCtGn602zNy8G2KEJFZk9G3Ahli"
70
71      n = 10000  # no of tweets to be collected
72
73      # Twitter authentication and the connection to Twitter Streaming API
74      listener = TwitterListener(n)
75      auth = OAuthHandler(consumer_key, consumer_secret)
76      auth.set_access_token(access_token, access_token_secret)
77      stream = Stream(auth, listener)
78
79      print('\nStep 1. Collecting tweets..')
80
81      # Filter twitter streams to capture data by the specified keywords and languages
82      stream.filter(track=['MachineLearning', 'BigData'], languages=["en"])
83
84
```

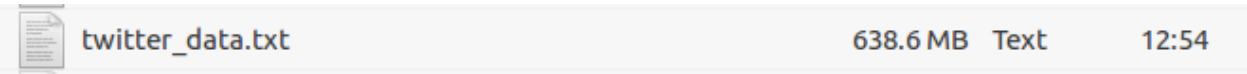
The above program collects tweets into a text file by using Twitter's streaming API Tweepy.

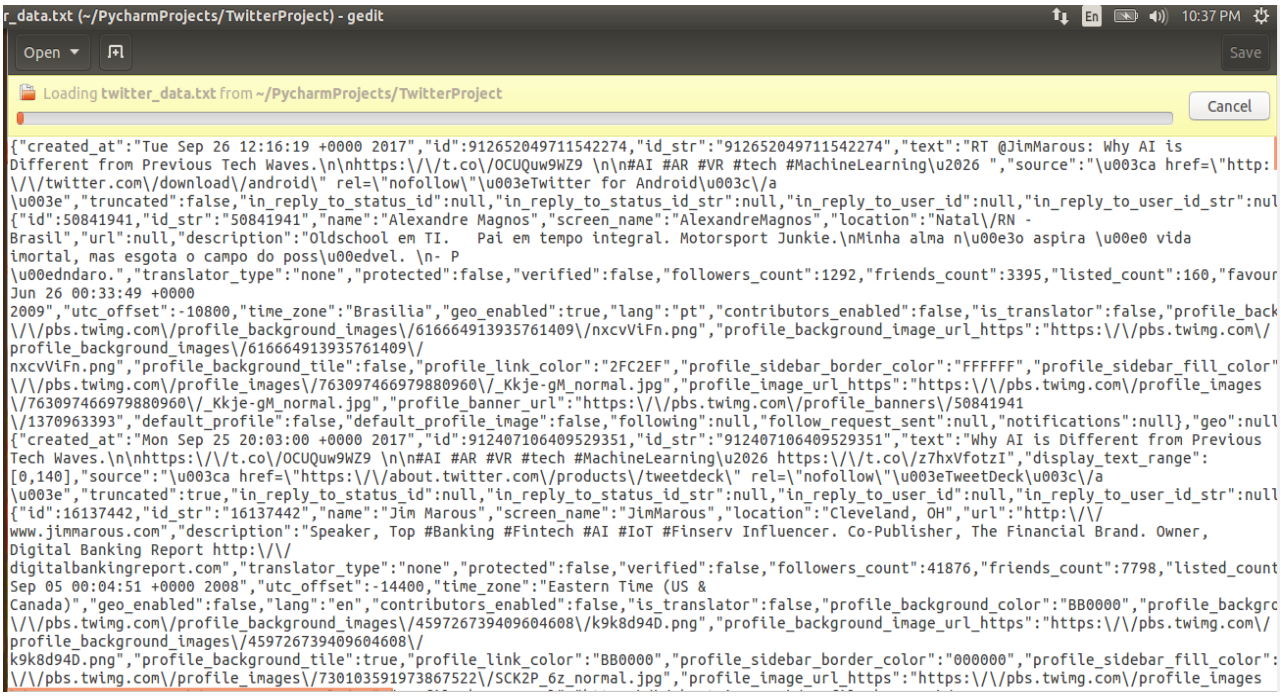
i. In order to connect to the Twitter's Streaming API, we need to authenticate using credentials from our twitter developer account.
ii. The program filters data by keywords ('MachineLearning', 'BigData'), and language ('en').
iii. The data is appended to file 'twitter_data.txt'.
iv. The program prints the total number of tweets collected every 10 mins on the console.

We have collected 100k+ tweets in batches.

Output File

twitter_data.txt (Not included in the output folder as size > 600MB)





{"created_at":"Tue Sep 26 12:16:19 +0000 2017","id":912652049711542274,"id_str":"912652049711542274","text":"RT @JimMarous: Why AI is
Different from Previous Tech Waves.\n\nhttps:\/\/t.co\/OCUQuw9WZ9 \n\n#AI #AR #VR #tech #MachineLearning\u2026 ","source":"\u003ca href=\"http:
\/\/twitter.com\/download\/android\" rel=\"nofollow\"\u003eTwitter for Android\u003c\/a
\u003e","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in_reply_to_user_id":null,"in_reply_to_user_id_str":nul
{"id":50841941,"id_str":"50841941","name":"Alexandre Magnos","screen_name":"AlexandreMagnos","location":"Natal\/RN -
Brasil","url":null,"description":"Oldschool em TI.   Pai em tempo integral. Motorsport Junkie.\nMinha alma n\u00e3o aspira \u00e0 vida
imortal, mas esgota o campo do poss\u00edvel. \n- P
\u00edndaro.","translator_type":"none","protected":false,"verified":false,"followers_count":1292,"friends_count":3395,"listed_count":160,"favour
Jun 26 00:33:49 +0000
2009","utc_offset":-10800,"time_zone":"Brasilia","geo_enabled":true,"lang":"pt","contributors_enabled":false,"is_translator":false,"profile_back
\/\/pbs.twimg.com\/profile_background_images\/616664913935761409\/nxcvViFn.png","profile_background_image_url_https":"https:\/\/pbs.twimg.com\/
profile_background_images\/616664913935761409\/
nxcvViFn.png","profile_background_tile":false,"profile_link_color":"2FC2EF","profile_sidebar_border_color":"FFFFFF","profile_sidebar_fill_color"
\/\/pbs.twimg.com\/profile_images\/763097466979880960\/_Kkje-gM_normal.jpg","profile_image_url_https":"https:\/\/pbs.twimg.com\/profile_images
\/763097466979880960\/_Kkje-gM_normal.jpg","profile_banner_url":"https:\/\/pbs.twimg.com\/profile_banners\/50841941
\/1370963393","default_profile":false,"default_profile_image":false,"following":null,"follow_request_sent":null,"notifications":null},"geo":null
{"created_at":"Mon Sep 25 20:03:00 +0000 2017","id":912407106409529351,"id_str":"912407106409529351","text":"Why AI is Different from Previous
Tech Waves.\n\nhttps:\/\/t.co\/OCUQuw9WZ9 \n\n#AI #AR #VR #tech #MachineLearning\u2026 https:\/\/t.co\/z7hxVfotzI","display_text_range":
[0,140],"source":"\u003ca href=\"https:\/\/about.twitter.com\/products\/tweetdeck\" rel=\"nofollow\"\u003eTweetDeck\u003c\/a
\u003e","truncated":true,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in_reply_to_user_id":null,"in_reply_to_user_id_str":null
{"id":16137442,"id_str":"16137442","name":"Jim Marous","screen_name":"JimMarous","location":"Cleveland, OH","url":"http:\/\/
www.jimmarous.com","description":"Speaker, Top #Banking #Fintech #AI #IoT #Finserv Influencer. Co-Publisher, The Financial Brand. Owner,
Digital Banking Report http:\/\/
digitalbankingreport.com","translator_type":"none","protected":false,"verified":false,"followers_count":41876,"friends_count":7798,"listed_count
Sep 05 00:04:51 +0000 2008","utc_offset":-14400,"time_zone":"Eastern Time (US &
Canada)","geo_enabled":false,"lang":"en","contributors_enabled":false,"is_translator":false,"profile_background_color":"BB0000","profile_backgrc
\/\/pbs.twimg.com\/profile_background_images\/459726739409604608\/k9k8d94D.png","profile_background_image_url_https":"https:\/\/pbs.twimg.com\/
profile_background_images\/459726739409604608\/
k9k8d94D.png","profile_background_tile":true,"profile_link_color":"BB0000","profile_sidebar_border_color":"000000","profile_sidebar_fill_color":
\/\/pbs.twimg.com\/profile_images\/730103591973867522\/SCK2P_6z_normal.jpg","profile_image_url_https":"https:\/\/pbs.twimg.com\/profile_images

Step 2: Extracting Hashtags and URLs

Python Code for Extraction

Extract_Fields.py

```
# ------------------------------------------------------------------------
#  * CS5540: Principles of Big Data Management
#  * Project Phase 1
#  * Team #7: Avni Mehta, Sneha Mishra, Arvind Tota
# ------------------------------------------------------------------------

# Import libraries
import json


# This functions takes .txt file as input
# and returns tweets as a list of json
def parse_tweets():

    tweets_data_path = 'twitter_data.txt'
    twt_data = []
    tweets_file = open(tweets_data_path, "r")
    for line in tweets_file:
        try:
            tweet = json.loads(line)
            twt_data.append(tweet)
        except:
            continue

    print('\tSuccessfully parsed data into json format')
    return twt_data

```

Extract_Fields.py

```

# This function extracts hashtags and urls from tweets data
def extract_hashtags_urls(twt_data):

    # Extract hashtags and urls into separate files
    htfile = 'hashtags.txt'
    urlfile = 'urls.txt'

    htoutfile = open(htfile, 'w')
    urloutfile = open(urlfile, 'w')

    for i in range(len(twt_data)):

        # Extracting hashtags
        ht = twt_data[i].get('entities').get('hashtags')
        for j in range(len(ht)):
            htoutfile.write(ht[j].get('text'))
            htoutfile.write(' ')

        # Extracting urls (expanded urls only)
        url = twt_data[i].get('entities').get('urls')
        for k in range(len(url)):
            urloutfile.write(url[k].get('expanded_url'))
            urloutfile.write(' ')

    # Print once extraction is complete
    print('\tExtracted hashtags in file - ' + htfile)
    print('\tExtracted urls in file - ' + urlfile)
    print('\nExiting the application.')


```

```
58
59      # Main Activity
60      if __name__ == '__main__':
61
62          print('\n---------------------------------------------------------------------------------------')
63          print('Big Data Project :')
64          print('Collect 100K+ tweets, extract hashtags and urls and run wordcount using Hadoop & Spark')
65          print('---------------------------------------------------------------------------------------')
66
67          # Step 2: Parsing the data
68          print('\nStep 2. Parsing data..')
69          tweets_data = parse_tweets()
70          print(str(len(tweets_data)))
71
72          # Step 3: Extracting hashtags and urls
73          print('\nStep 3. Extracting hashtags and urls..')
74          #extract_hashtags_urls(tweets_data)
```

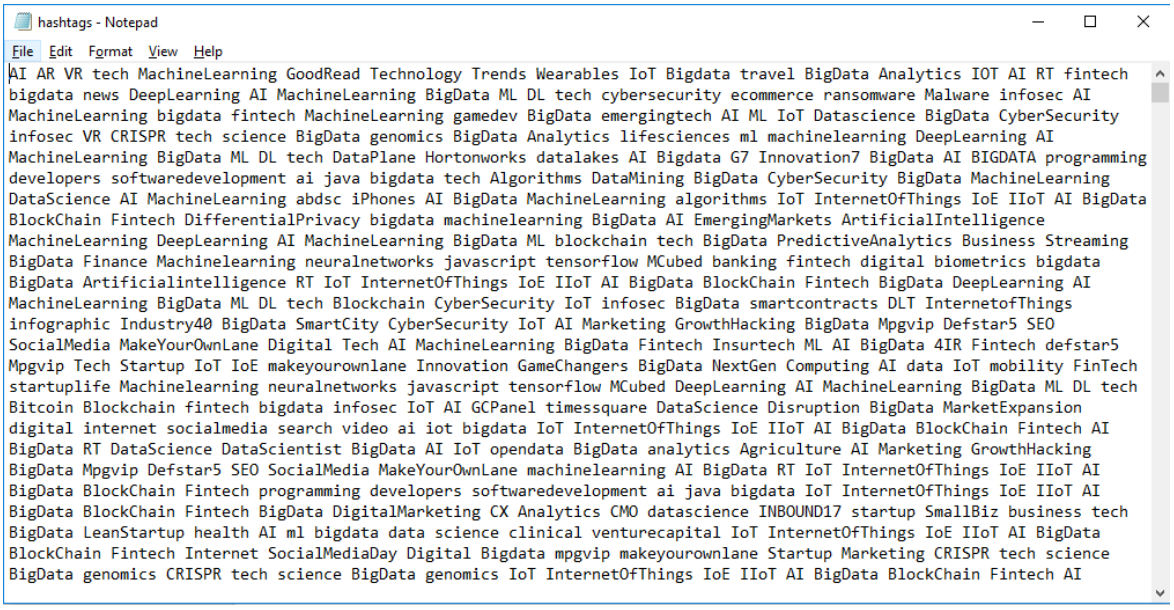
There are two methods in this program.

i. The *parse_tweets* method takes the output from previous program as input, and parses the same as a list of json.

ii. The *extract_hashtags_urls* method extracts hashtags and urls into two separate files.

Output files

hashtags.txt



urls.txt

Step 3: Loading file in HDFS

```
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$ $HADOOP_HOME/bin/hdfs dfs -copyFromLocal hashtags.txt /twitterproject
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$ $HADOOP_HOME/bin/hdfs dfs -copyFromLocal urls.txt /twitterproject
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$ $HADOOP_HOME/bin/hdfs dfs -ls /twitterproject
Found 2 items
-rw-r--r--   1 avni supergroup    4119450 2017-09-29 21:01 /twitterproject/hashtags.txt
-rw-r--r--   1 avni supergroup    3104002 2017-09-29 21:01 /twitterproject/urls.txt
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$
```

Both output files viz. hashtags.txt and urls.txt are loaded to HDFS

Step 4: Running word count on hashtags and URLs

Word count for Hashtags

```
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.2.0
      /_/

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_131)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val htFile = sc.textFile("hdfs://localhost:9000/twitterproject/hashtags.txt")
htFile: org.apache.spark.rdd.RDD[String] = hdfs://localhost:9000/twitterproject/hashtags.txt MapPartitionsRDD[1] at textFile at <console>:24

scala> val htWords = htFile.flatMap(line => line.split(" "))
htWords: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at <console>:26

scala> val htCounts = htWords.map(word => (word.toLowerCase, 1)).reduceByKey(_ + _)
htCounts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:28

scala> val htSortedCount = htCounts.sortBy(-_._2)
htSortedCount: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[9] at sortBy at <console>:30

scala> htSortedCount.saveAsTextFile("hdfs://localhost:9000/twitterproject/htoutput")
```

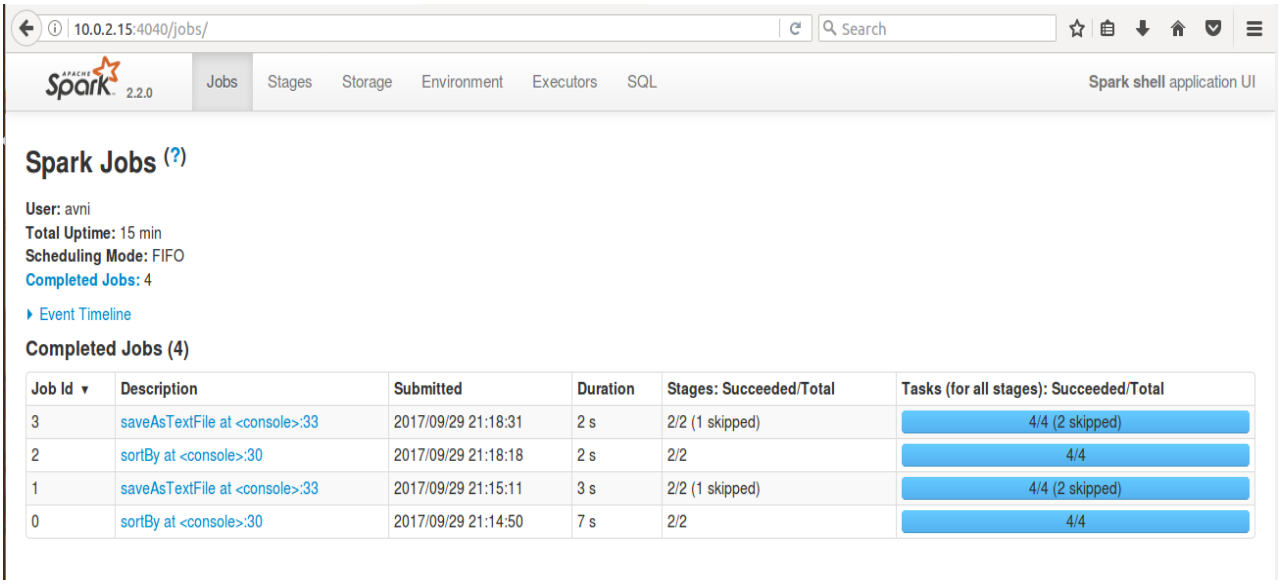
i. A RDD is created from the input file hashtags.txt.
ii. Next, each line in the RDD is split into words and stored in htWords.
iii. Then a Key-Value pair RDD is created with key as lowered-case word and value 1. By using the reduce operation, all pairs with key are reduced and values are added. This is stored in htCounts.
iv. htCounts is sorted by values in descending order and stored in htSortedCount.
v. It is then saved as a text file at the given hdfs location

Word count for URLs

```
scala> val urlFile = sc.textFile("hdfs://localhost:9000/twitterproject/urls.txt")
urlFile: org.apache.spark.rdd.RDD[String] = hdfs://localhost:9000/twitterproject/urls.txt MapPartitionsRDD[12] at textFile at <console>:24

scala> val urlWords = urlFile.flatMap(line => line.split(" "))
urlWords: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[13] at flatMap at <console>:26

scala> val urlCounts = urlWords.map(word => (word, 1)).reduceByKey(_ + _)
urlCounts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[15] at reduceByKey at <console>:28

scala> val urlSortedCount = urlCounts.sortBy(-_._2)
urlSortedCount: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[20] at sortBy at <console>:30

scala> urlSortedCount.saveAsTextFile("hdfs://localhost:9000/twitterproject/urloutput")
```

The word count for URLs is gathered in similar manner as hashtags with the exception of lower-casing of word.

Spark Jobs UI



Step 5: Copying Output from HDFS to local

Output folders created in HDFS

```
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$ $HADOOP_HOME/bin/hdfs dfs -ls /twitterproject
Found 4 items
-rw-r--r--   1 avni supergroup    4119450 2017-09-29 21:01 /twitterproject/hashtags.txt
drwxr-xr-x   - avni supergroup          0 2017-09-29 21:15 /twitterproject/htoutput
drwxr-xr-x   - avni supergroup          0 2017-09-29 21:18 /twitterproject/urloutput
-rw-r--r--   1 avni supergroup    3104002 2017-09-29 21:01 /twitterproject/urls.txt
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject$ $HADOOP_HOME/bin/hdfs dfs -copyToLocal /twitterproject
```

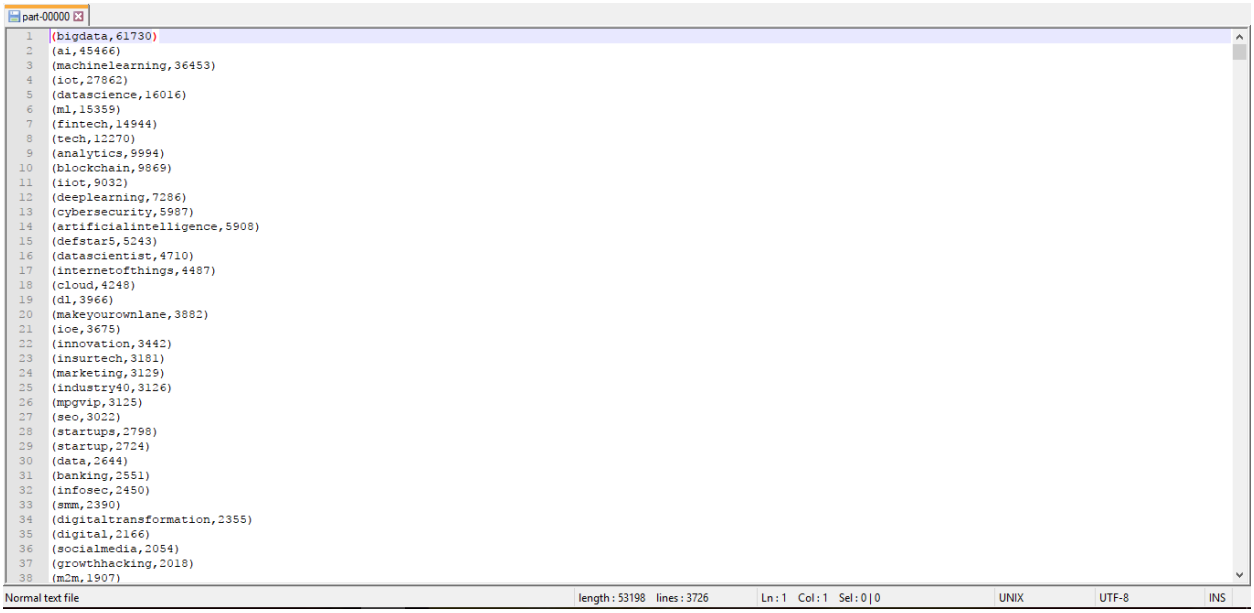
Output folders copied to local

```
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject/twitterproject$ ls -l
total 7064
-rw-r--r-- 1 avni avni 4119450 Sep 29 21:20 hashtags.txt
drwxrwxr-x 2 avni avni    4096 Sep 29 21:20 htoutput
drwxrwxr-x 2 avni avni    4096 Sep 29 21:20 urloutput
-rw-r--r-- 1 avni avni 3104002 Sep 29 21:20 urls.txt
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject/twitterproject$
```

```
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject/twitterproject/htoutput$ ls -l
total 84
-rw-r--r-- 1 avni avni 53198 Sep 29 21:20 part-00000
-rw-r--r-- 1 avni avni 32005 Sep 29 21:20 part-00001
-rw-r--r-- 1 avni avni     0 Sep 29 21:20 _SUCCESS
```

```
avni@avni-VirtualBox:~/PycharmProjects/TwitterProject/twitterproject/urloutput$ ls -l
total 1112
-rw-r--r-- 1 avni avni 1137832 Sep 29 21:20 part-00000
-rw-r--r-- 1 avni avni       0 Sep 29 21:20 part-00001
-rw-r--r-- 1 avni avni       0 Sep 29 21:20 _SUCCESS
```

Hashtags Wordcount Output



URLs Wordcount Output