

EECS 6893 Big Data Analytics

Homework Assignment 3

UNI: qg2172

Student: Qiaoyu Gu

Due: 11/1/2019

1. Screenshot of code to do all the tasks

1.1 hashtagCount()

```
def hashtagCount(words):  
    """  
    Calculate the accumulated hashtags count sum from the beginning of the stream  
    and sort it by descending order of the count.  
    Ignore case sensitivity when counting the hashtags:  
    "#Ab" and "#ab" is considered to be a same hashtag  
    You have to:  
    1. Filter out the word that is hashtags.  
        Hashtag usually start with "#" and followed by a series of alphanumeric  
    2. map (hashtag) to (hashtag, 1)  
    3. sum the count of current DStream state and previous state  
    4. transform unordered DStream to a ordered Dstream  
    Hints:  
        you may use regular expression to filter the words  
        You can take a look at updateStateByKey and transform transformations  
    Args:  
        dstream(DStream): stream of real time tweets  
    Returns:  
        DStream Object with inner structure (hashtag, count)  
    """  
    htc=words.map(lambda w:w.lower()).filter(lambda x:len(x)>2).filter(lambda x:x[0]=='#')  
    htc=htc.map(lambda x:(x,1))  
    htc_one=htc.reduceByKey(lambda x,y:x+y)  
    htc_total=htc_one.updateStateByKey(lambda x,y:sum(x+(y or 0)))  
    return htc_total
```

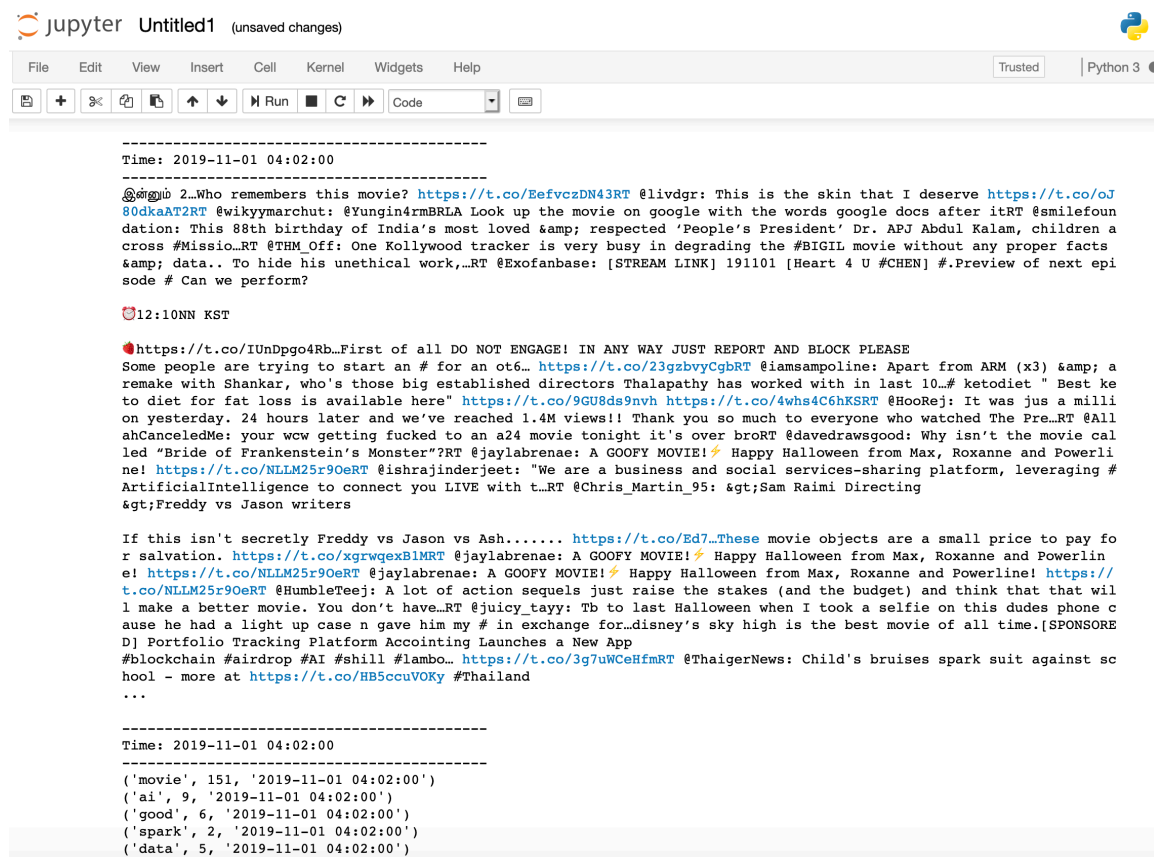
1.2 wordCount()

```
def wordCount(words):  
    """  
    Calculate the count of 5 sepcial words for every 60 seconds (window no overlap)  
    You can choose your own words.  
    Your should:  
    1. filter the words  
    2. count the word during a special window size  
    3. add a time related mark to the output of each window, ex: a datetime type  
    Hints:  
        You can take a look at reduceByKeyAndWindow transformation  
        Dstream is a serious of rdd, each RDD in a DStream contains data from a certain interval  
        You may want to take a look of transform transformation of DStream when trying to add a time  
    Args:  
        dstream(DStream): stream of real time tweets  
    Returns:  
        DStream Object with inner structure (word, count, time)  
    """  
    wordcount=words.map(lambda w:w.lower()).filter(lambda w: w in WORD)  
    wordcount=wordcount.map(lambda w:(w,1))  
    wordcount=wordcount.reduceByKeyAndWindow(lambda x,y:x+y, lambda x,y:x-y,60,60)  
    wordcounttime=wordcount.transform(lambda time, rdd: rdd.map(  
        lambda x: (x[0], x[1], time.strftime("%Y-%m-%d %H:%M:%S"))))  
    return wordcounttime
```

1.3 Save hashtags count and word count to google storage

```
# save hashtags count and word count to google storage.  
topTags.foreachRDD(lambda rdd: saveToStorage(rdd, output_directory_hashtags,columns_name_hashtags,mode="overwrite"))  
wordCount.foreachRDD(lambda rdd: saveToStorage(rdd, output_directory_wordcount,columns_name_wordcount, mode="append"))
```

1.4 Demo of real-time output when executing sparkStreaming.py in Jupyter



```
-----
Time: 2019-11-01 04:02:00
-----
@livdgr: This is the skin that I deserve https://t.co/oJ80dkaAT2RT @wikyymarchut: @Yungin4rmBRLA Look up the movie on google with the words google docs after itRT @smilefoun
dation: This 88th birthday of India's most loved & respected 'People's President' Dr. APJ Abdul Kalam, children a
cross #Missio_RT @THM_Off: One Kollywood tracker is very busy in degrading the #BIGIL movie without any proper facts
& data.. To hide his unethical work,..RT @Exofanbase: [STREAM LINK] 191101 [Heart 4 U #CHEN] #.Preview of next epi
sode # Can we perform?

12:10NN KST

https://t.co/IUnDpgo4Rb...First of all DO NOT ENGAGE! IN ANY WAY JUST REPORT AND BLOCK PLEASE
Some people are trying to start an # for an ot6... https://t.co/23gzbyvCgbRT @iamsampoline: Apart from ARM (x3) & a
remake with Shankar, who's those big established directors Thalapathy has worked with in last 10...# ketodiet " Best ke
to diet for fat loss is available here" https://t.co/9GU8ds9nvH https://t.co/4whs4C6hKSRT @HooRej: It was jus a milli
on yesterday. 24 hours later and we've reached 1.4M views!! Thank you so much to everyone who watched The Pre_RT @All
ahCanceledMe: your wcv getting fucked to an a24 movie tonight it's over broRT @davedrawsgood: Why isn't the movie cal
led "Bride of Frankenstein's Monster"?RT @jaylabrenae: A GOOFY MOVIE! Happy Halloween from Max, Roxanne and Powerlin
e! https://t.co/NLLM25r9OeRT @ishrajinderjeet: "We are a business and social services-sharing platform, leveraging #
ArtificialIntelligence to connect you LIVE with t...RT @Chris_Martin_95: >Sam Raimi Directing
>Freddy vs Jason writers

If this isn't secretly Freddy vs Jason vs Ash..... https://t.co/Ed7...These movie objects are a small price to pay fo
r salvation. https://t.co/xgrwgexB1MRT @jaylabrenae: A GOOFY MOVIE! Happy Halloween from Max, Roxanne and Powerlin
e! https://t.co/NLLM25r9OeRT @jaylabrenae: A GOOFY MOVIE! Happy Halloween from Max, Roxanne and Powerline! https://
t.co/NLLM25r9OeRT @HumbleTeej: A lot of action sequels just raise the stakes (and the budget) and think that that wil
l make a better movie. You don't have...RT @juicy_tay: Tb to last Halloween when I took a selfie on this dudes phone c
ause he had a light up case n gave him my # in exchange for...disney's sky high is the best movie of all time.[SPONSORE
D] Portfolio Tracking Platform Acointing Launches a New App
#blockchain #airdrop #AI #shill #lambo... https://t.co/3g7uWceHfmRT @ThaigerNews: Child's bruises spark suit against sc
hool - more at https://t.co/HB5ccuVOKy #Thailand
...

-----
Time: 2019-11-01 04:02:00
-----
('movie', 151, '2019-11-01 04:02:00')
('ai', 9, '2019-11-01 04:02:00')
('good', 6, '2019-11-01 04:02:00')
('spark', 2, '2019-11-01 04:02:00')
('data', 5, '2019-11-01 04:02:00')
```

2. Screenshot of data stored in BigQuery

2.1 Table Hashtags

First, I count total number of distinct hashtags appeared in the last 600 seconds.

There are 890 hashtags.

Query history

Saved queries

Job history

Transfers

Scheduled queries

BI Engine

Resources

+ ADD DATA

Search for your tables and datasets

apt-momentum-254514

twapi

twgqy

Unsaved query Edited

1

select distinct count(hashtags) FROM apt-momentum-254514.twapi.hashtags

Run

Save query

Save view

Schedule query

More

Query results

SAVE RESULTS

EXPLORE WITH DATA STUDIO

Query complete (1.1 sec elapsed, 12.1 KB processed)

Job information

Results

JSON

Execution details

Row	f0_
1	890

Secondly, we list top 10 hashtags with the highest frequency. As is seen from the figure below, **#bigil** is the one appeared most in last 10 minutes.

BigQuery

FEATURES & INFO

SHORTCUTS

+ COMPOSE NEW QUE

Query history

Saved queries

Job history

Transfers

Scheduled queries

BI Engine

Resources

+ ADD DATA

Search for your tables and datasets

apt-momentum-254514

twapi

twgqy

Query editor

1

SELECT * FROM apt-momentum-254514.twapi.hashtags order by count desc LIMIT 15

Run

Save query

Save view

Schedule query

More

SAVE RESULTS

EXPLORE WITH DATA STUDIO

Query complete (0.8 sec elapsed, 19.1 KB processed)

Job information

Results

JSON

Execution details

Row	count	hashtags
1	128	#bigil
2	82	#ai
3	63	#valimai
4	47	#thalapathy
5	46	#powermonster
6	46	#umidigipower3
7	39	#bigdata
8	27	#kaithi
9	26	#machinelearning
10	22	#datascience
11	21	#vino_rt
12	16	#moodttaeyong
13	16	#robotics
14	16	#analytics
15	14	#artificialintelligence

2.1 Table Wordcount

In this part, I filter 5 chosen words and calculate the appearance frequency of them with the interval of 60 seconds and length of 600 seconds. Next, I will show each of these five words one by one.

One thing I want to mention here is that some of words may have no

appearance in some of intervals.

a. ai

Query editor				
1 SELECT * FROM apt-momentum-254514.twapi.wordcount where word="ai" order by time				
<div>Run Save query Save view Schedule query More</div>				
Query results				
Query complete (0.7 sec elapsed, 1 KB processed)				
Job information Results JSON Execution details				
Row	time	count	word	
1	2019-11-01 02:34:00 UTC	4	ai	
2	2019-11-01 02:35:00 UTC	13	ai	
3	2019-11-01 02:36:00 UTC	7	ai	
4	2019-11-01 02:37:00 UTC	14	ai	
5	2019-11-01 02:38:00 UTC	11	ai	
6	2019-11-01 02:39:00 UTC	16	ai	
7	2019-11-01 02:40:00 UTC	13	ai	
8	2019-11-01 02:41:00 UTC	15	ai	
9	2019-11-01 02:42:00 UTC	13	ai	
10	2019-11-01 02:43:00 UTC	15	ai	

b. data

Query editor				
1 SELECT * FROM apt-momentum-254514.twapi.wordcount where word="data" order by time				
<div>Run Save query Save view Schedule query More</div>				
Query results				
Query complete (0.8 sec elapsed, 1 KB processed)				
Job information Results JSON Execution details				
Row	time	count	word	
1	2019-11-01 02:34:00 UTC	1	data	
2	2019-11-01 02:36:00 UTC	2	data	
3	2019-11-01 02:37:00 UTC	2	data	
4	2019-11-01 02:38:00 UTC	1	data	
5	2019-11-01 02:39:00 UTC	1	data	
6	2019-11-01 02:42:00 UTC	2	data	
7	2019-11-01 02:43:00 UTC	2	data	

c. good

Query editor				
1 SELECT * FROM apt-momentum-254514.twapi.wordcount where word="good" order by time				
<div>Run Save query Save view Schedule query More</div>				
Query results				
Query complete (0.8 sec elapsed, 1 KB processed)				
Job information Results JSON Execution details				
Row	time	count	word	
1	2019-11-01 02:34:00 UTC	3	good	
2	2019-11-01 02:35:00 UTC	13	good	
3	2019-11-01 02:36:00 UTC	14	good	
4	2019-11-01 02:37:00 UTC	11	good	
5	2019-11-01 02:38:00 UTC	8	good	
6	2019-11-01 02:39:00 UTC	6	good	
7	2019-11-01 02:40:00 UTC	14	good	
8	2019-11-01 02:41:00 UTC	21	good	
9	2019-11-01 02:42:00 UTC	5	good	
10	2019-11-01 02:43:00 UTC	12	good	

d. movie

Query editor

1

SELECT

FROM

apt-momentum-254514.tvapi.wordcount

where word="movie" order by time

Run

Save query

Save view

Schedule query

More

Query results

SAVE RESULTS

EXPLORE WITH DATA STUDIO

Query complete (0.6 sec elapsed, 1 KB processed)

Job information

Results

JSON

Execution details

Row	time	count	word
1	2019-11-01 02:34:00 UTC	51	movie
2	2019-11-01 02:35:00 UTC	152	movie
3	2019-11-01 02:36:00 UTC	156	movie
4	2019-11-01 02:37:00 UTC	131	movie
5	2019-11-01 02:38:00 UTC	171	movie
6	2019-11-01 02:39:00 UTC	140	movie
7	2019-11-01 02:40:00 UTC	146	movie
8	2019-11-01 02:41:00 UTC	140	movie
9	2019-11-01 02:42:00 UTC	124	movie
10	2019-11-01 02:43:00 UTC	141	movie

e. spark

Query editor

1

SELECT

FROM

apt-momentum-254514.tvapi.wordcount

where word="spark" order by time

Run

Save query

Save view

Schedule query

More

Query results

SAVE RESULTS

EXPLORE WITH DATA STUDIO

Query complete (0.9 sec elapsed, 1 KB processed)

Job information

Results

JSON

Execution details

Row	time	count	word
1	2019-11-01 02:34:00 UTC	1	spark
2	2019-11-01 02:35:00 UTC	1	spark
3	2019-11-01 02:36:00 UTC	8	spark
4	2019-11-01 02:37:00 UTC	7	spark
5	2019-11-01 02:38:00 UTC	3	spark
6	2019-11-01 02:39:00 UTC	6	spark
7	2019-11-01 02:40:00 UTC	6	spark
8	2019-11-01 02:41:00 UTC	6	spark
9	2019-11-01 02:42:00 UTC	6	spark
10	2019-11-01 02:43:00 UTC	5	spark