1. Hourly Analysis of President's Tweets

A.

Methodology

Functions

getRawDateTime -> returns the date and time of each and every tweet
getPrezDateTime -> returns the date and time when president tweeted for each of his tweet
timeConvert function -> takes in the raw date and time returned from getPrezDateTime function and returns the hour for which president tweeted
dateConvert function -> takes in date and time and returns the date which has day, month and year only.

We are loading the tweets using sc.textFile function, using flatMaps to extract the hours President has tweeted and total number of days of twitter data parsed into two RDDs named hourFilter and rawDateTime. I am caching these RDDs, as I will be reusing them again.

We are returning **<hour, # of tweets in that hour by president>,** key-value pairs to tweetHour RDD, also returning **<unique date, # of tweets on that day>** - key-value pair to tweetDay RDD.

Instructions

Run the below command or execute the "execute.sh" script, it generates output.csv file in local directory.

```
spark-submit --master yarn-client sparkProgram.py hdfs://hadoop2-0-
0/twitter/*
```

We are using collect function to collect the spark program output locally and saving it in my local directory with name output.csv,

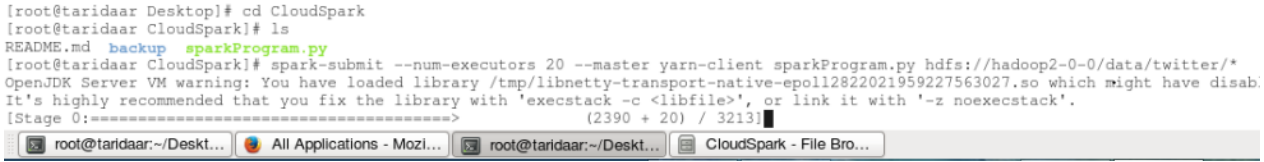
Using Excel, from output.csv file, we can plot the details.
From output.csv file, we can infer the following.

Total no. of days of twitter data parsed, total number of tweets made by president are available for each hour.

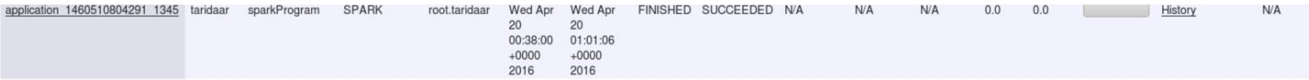
Using excel manipulations, I am calculating the average or expected number of tweets per hour by dividing the tweet count obtained by total no. of twitter data parsed.

Note: UTC hours are converted to Eastern Time, by reducing time by 4 hours, this calculation is done using excel. After conversion, following plots are plotted.

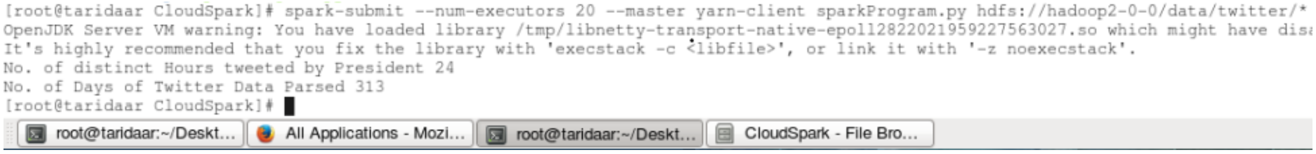
Output Screenshots



```
[root@taridaar Desktop]# cd CloudSpark
[root@taridaar CloudSpark]# ls
README.md  backup  sparkProgram.py
[root@taridaar CloudSpark]# spark-submit --num-executors 20 --master yarn-client sparkProgram.py hdfs://hadoop2-0-0/data/twitter/*
OpenJDK Server VM warning: You have loaded library /tmp/libnetty-transport-native-epoll2822021959227563027.so which might have disab
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
[Stage 0:=======================================>                 (2390 + 20) / 3213]
```

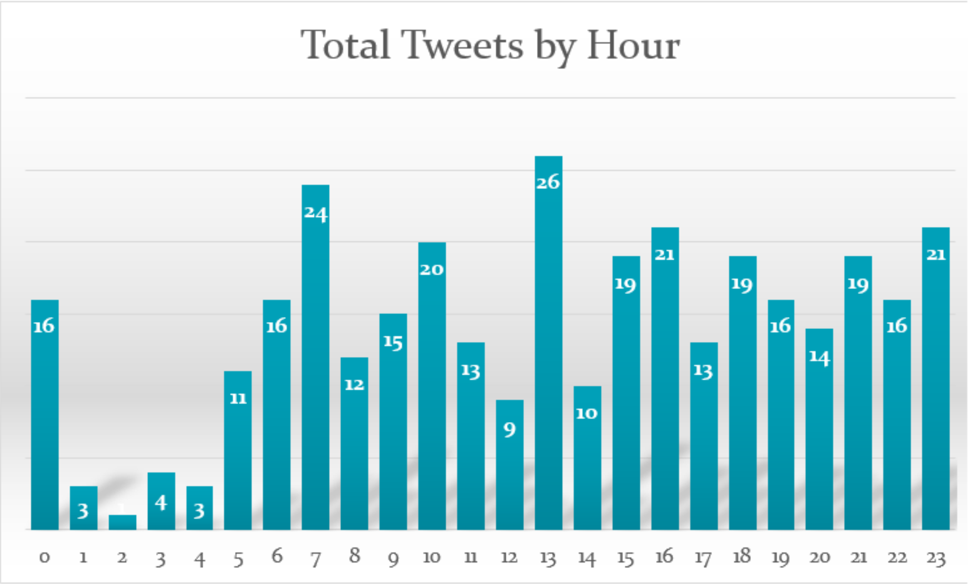


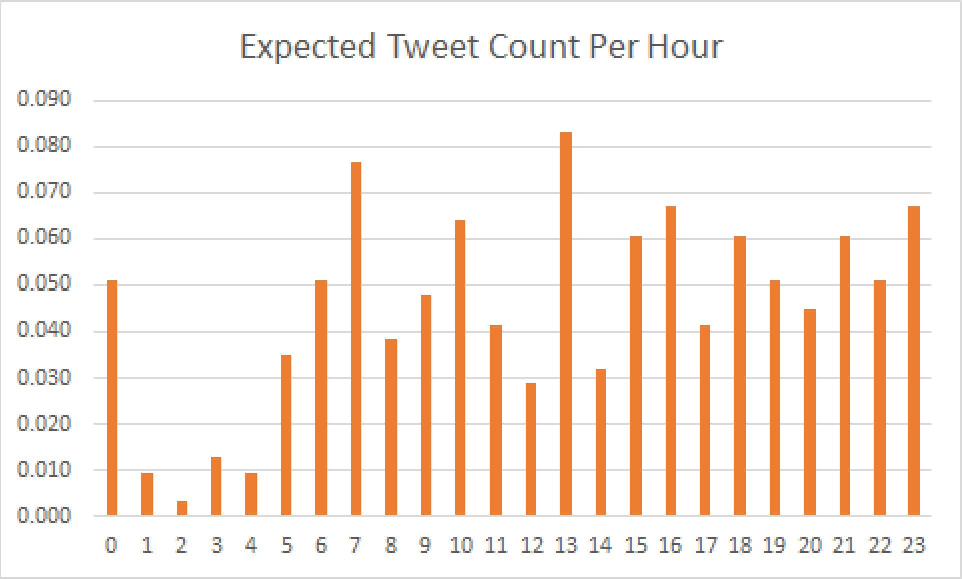
application_1460510804291_1345	taridaar	sparkProgram	SPARK	root.taridaar	Wed Apr 20 00:38:00 +0000 2016	Wed Apr 20 01:01:06 +0000 2016	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0		History	N/A



```
[root@taridaar CloudSpark]# spark-submit --num-executors 20 --master yarn-client sparkProgram.py hdfs://hadoop2-0-0/data/twitter/*
OpenJDK Server VM warning: You have loaded library /tmp/libnetty-transport-native-epoll2822021959227563027.so which might have dis
It's highly recommended that you fix the library with 'execstack -c <libfile>', or link it with '-z noexecstack'.
No. of distinct Hours tweeted by President 24
No. of Days of Twitter Data Parsed 313
[root@taridaar CloudSpark]# █
```

Excel Output





Analysis Inference: President tweets most at 1300 followed by 0700, 2300, 1600 and 1000 hours.

Hour	Tweet Count	Expected Number of Tweets
0	16	0.051
1	3	0.010
2	1	0.003
3	4	0.013
4	3	0.010
5	11	0.035
6	16	0.051
7	24	0.077
8	12	0.038
9	15	0.048
10	20	0.064
11	13	0.042
12	9	0.029
13	26	0.083
14	10	0.032
15	19	0.061
16	21	0.067
17	13	0.042
18	19	0.061
19	16	0.051
20	14	0.045
21	19	0.061
22	16	0.051
23	21	0.067

5. What twitter user tweeted the most? What is the top 5 longest tweeters over each's average tweet length? Bottom 5?(Twitter)

Sol:

Platform used: python on pyspark
Dataset used: Twitter data set

In this question we require the information about the twitter user name and his tweet text for which we can find his tweet length and average length of all his tweets. So we make use of two features: twitter user "screen-name" and tweet "text".

First we load the twitter data by using json loads function and then read the twitter username and his tweet text using a function. We return the username as key and (tweet length, 1) as value. Now we use reduceByKey function to add all his tweet lengths and get count by username.

We use the top function on count to get the user who tweeted the most.
Next we use the map function to find the average tweet length of each user. Finally we print the top 5 and bottom 5 average tweet length users using top function.

Input command:

```
spark-submit --master yarn-client –-num-executors 40
tweetsLen.py hdfs://hadoop2-0-0/twitter/*
```

Created output.sh file to execute the above query and fetch the output file.

Output:

```
Top tweeting User: 'marilyn9743' with 3419 tweets
Top 5 Users and AvgTweets: [(u'Huntersweat', 416), (u'RoyalEliteKiva', 350), (u'blackxhole', 320),
(u'KelleeMichele', 272), (u'pizzadellarry', 253)]
Bottom 5 Users and AvgTweets: [(u'ShakeIt4Rome', 1), (u'Im_Lil_Wanie', 1), (u'Abby_Palmiter', 1),
(u'HannahGarwood', 1), (u'Oliviacouss', 1)
```

1.For each year available, plot the size of the set of words used. Year on the x-axis, number of words on y-axis. (Google onegram)

Sol:

Platform used: python on pyspark
Dataset used: Google onegram

Here, we need to find the number of unique words used in each year. We try to plot the number of words used against the year in which they were used.
First we read each line that has the format "word year count unique_count". Then we use the split function to divide them into components. Next we use map function to pass year as the key and 1 as value for each word. Now we use reduceByKey to get the total unique words in each year. Finally we sorted the years so that we can plot.

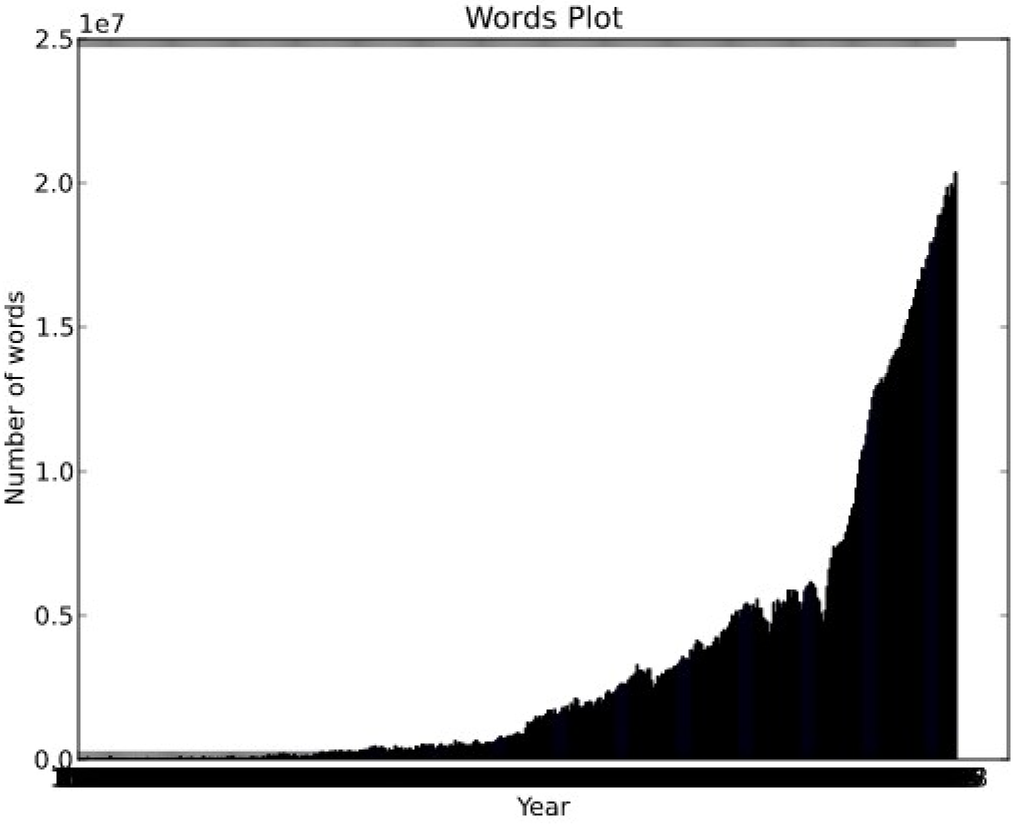
The output plot graph was plotted using the matplotlib.pyplot . Years from 1505 to 2008 was taken on x-axis vs number of unique words on y-axis. The plot shown below is very dense and compact as we try to fit 503 years in the plot.

Input command:

```
spark-submit --master yarn-client –-num-executors 40
ogramoq.py hdfs://hadoop2-0-0/data/1gram/
```

Created output.sh file to execute the above query and fetch the output file.

Output plot:



3. Plot the average word length for all unique words for all years available. Year on x-axis, average word-length on y-axis. (Google 1gram)
Sol:
Platform used: python-spark
Dataset used: Google 1gram

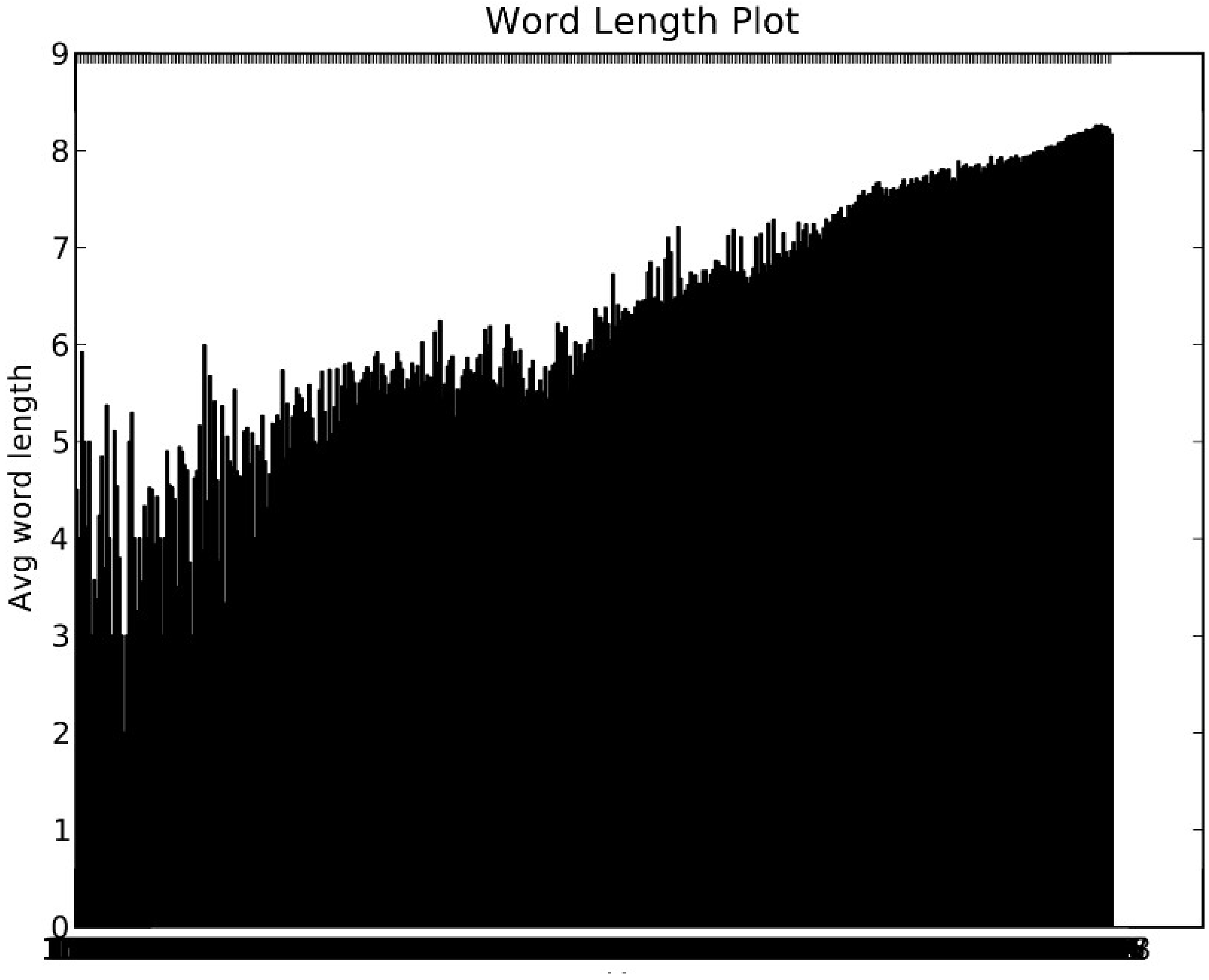
All the 1gram was split up and mapped to extract all the words and their occurrence. The data obtained is transformed to obtain the year and the word then the data is reducedByKey, key here is year. All the data obtained and the average word length is obtained for each year and sorted output is displayed using matplotlib library and the figure (barchar.pdf) is saved in the current directory.

Since the range of years are taken from 1505 to 2008, the number of years considered are 503, they cannot be shown clearly on the plot. So the output plot shown below is compact and dense.

Input command: `spark-submit --master yarn-client –-num-executors 40 ogramoq3q.py hdfs://hadoop2-0-0/data/1gram/`

Created output.sh file to execute the above query and fetch the output file.

Output:


Word Length Plot
Avg word length