Cloud Computing
P4 Spark for Real Problems

Group Members
Renukalakshmi Dasari
Sania Alex
Susmitha Kandula
Shilpa Batchu

Twitter Question 2:

What day of the week does @PrezOno tweet the most on average? Use the same example as in #1 but for days of the week.

First, I am retrieving the screen names of the users and the day on which the tweet is created.

As the day and time format is in the following manner

Tue Feb 11 04:45:51 +0000 2014

In order to retrieve the day I took (json.loads(line)['created_at'].split()[0] and to retrieve the screen names I took json.loads(line)['user']['screen_name']

Next I filtered the screen names with PrezOno and the days on which PrezOno tweeted.

To find which day of the week does PrezOno tweeted the most, I divided the count of each day on which PrezOno tweeted with 52 (since there are 365 days, and Sundays will be 52, Mondays will be 52 likewise)

Finally the output is:

(u'Sun', 1.134615384615385)

(u'Mon', 0.92307692307692357)

(u'Tue', 0.63461538461538469)

(u'Wed', 1.0576923076923079)

(u'Thu', 1.038461538461539)

(u'Fri', 0.75000000000000022)

(u'Sat', 1.0192307692307701)

From the output, I can say that PrezOno tweeted the most on Sundays.

Twitter Question 3

How does @PrezOno's tweet length compare to the average of all others? What is his average length? All others?

The Twitter data has been loaded using Spark Context. As the twitter data is in json format, from each tweet the text field can be retrieved with the function json.loads()["text"] .

I found the average tweet length of each user. And then I retrieved the average tweet length of PrezOno using the Spark filter(). To find the average tweet length of all others, I found the average tweet length of all others excluding PrezOno.

Compared the PrezOno tweet length with others by calculating the ratio of PrezOno and others.

Difficulties: Once I found the average length of all others and extracted PrezOno tweets, it became difficult to extract the tweets of all others. I solved it by using another filter which filters tweets which are not tweeted by PrezOno. In this way I calculated the average tweet length of other tweets other than PrezOno.

Results:

Average tweet length of PrezOno : 104

Average tweet length of others: 86

Ratio: 104/86

= 1.2(approx)

So , PrezOno's tweet length is approximately 1.2 times others average length.

Twitter Question 7th

For those tweets with location information, what lat/long (or city/state) is the centroid? What was the proportion of tweets with location to those without?

Explanation/Approach:

The input given to the program is the entire twitter dataset, for the question to be answered we need the geographic details from where the tweet was posted. By using the keyword "geo" I extracted all the details out of the tweets. This will give us a dictionary which consists of keys type, point, co-ordinates. We need the key coordinates and its value (tuple of latitude and longitude). Using the "coordinates" key word Latitude and Longitude values were extracted and stored into separate RDD's. The tweets without location were filtered using filter function out and centroid is found out by adding up the sum of all the latitudes and longitudes values, finally divide the sum with the count of tweets which has location information.

The key value pairs issued through map function look like (Latitude", value/count), where all the values with key as "Latitude" got added up. Same is the case with Longitude.

The count variable has got count of tweets with location information and I declared another variable called c which stores the count of all the tweets. With the command **rdd.count ()** we can simply find out the count which goes into the variables "count" and "c".

The proportion of tweets with location to those without is calculated using the simple logic

count: c-count

Results:

When the program has been run on cluster with twitter data as input, the following results were recorded

Location Centroid [38.990277950025686, -85.452802750937991]

The proportion of tweets with location to those without 1868302: 4212000

Twitter Question 10-
Detect the proportion of bad words in a tweet. Plot bad word proportion by hour for all 24 hours.

Input data files-

The twitter data files had multiple tweets where every tweet consisted of keys such as text, time at which it was created, ID etc. I needed text and time at which each tweet was created.

1.The text was in plain text format consisting of words. I split it to get it in following format-
[u'Dog', u'Mising', u'bastard', u'rascal', u'fuck', u'for', u'a', u'Year', u'Found', u'by', u'Good', u'Samaritans', u'--', u'Big', u'Surprise', u'at', u'the', u'End!', u'http://t.co/541xBpHOOY', u'|', u'#dogs', u'#pets']

2.Time consisted of day, date and time. I split it to get the following-
[u'Wed', u'Sep', u'24', u'03:45:31', u'+0000', u'2014']
From the above, I took the 3rd index which gave me the time.

I obtained the list of bad words banned by google and put it on the cluster.
Download link- http://www.freewebheaders.com/full-list-of-bad-words-banned-by-google/

Problems faced and tricks used-

1. The biggest problem I faced was that the list of bad words was not clean. It had characters which were not understandable to the program. I had to clean the data by keeping only alphanumeric characters.
2. Next trouble was that when I imported the list of bad words, split it and stored it in an RDD, every single word got saved as an item in the list. This bad word was not matching with an equivalent word in tweet text. I had to convert every item in bad word list to a string and append it.

Formula used-

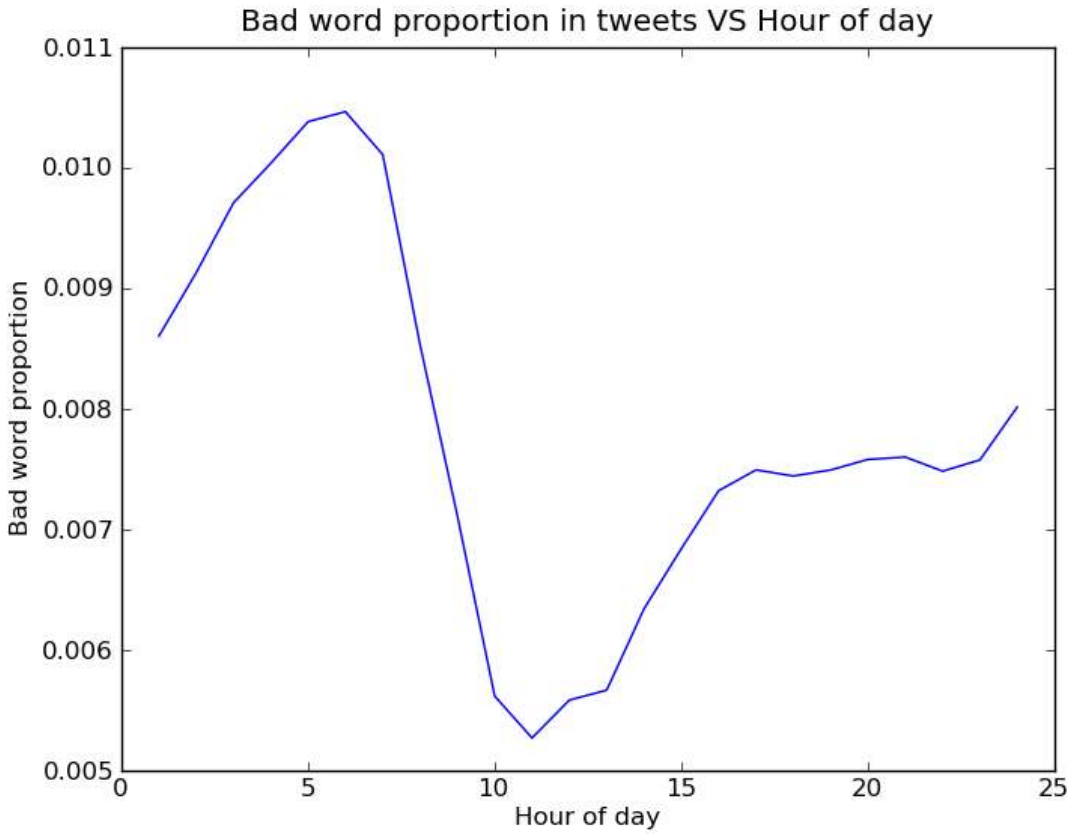
First, I calculated the number of bad words in each tweet and the total number of words in a tweet.
According to the time at which the tweet was made, I calculated the proportion of bad words in every hour.
Bad word proportion= Number of bad words used in an hour/ Total number of words in tweets made in an hour

Commands to run code-

To run badword_twitter.py- spark-submit --master yarn-client badword_twitter.py hdfs://hadoop2-0-0/data/twitter
To run plot.py- cat output_badwordProportion | ./plot.py

Plot:
The plot has hours on X-axis and bad word proportion on Y-axis.

X Axis- Hour of day
Y Axis- Bad word proportion



In the above graph-
I have made the following assumption-

12am-1am : 1st hour
1am-2am: 2nd hour
2am-3am- 3rd hour
..
11pm-12am: 24th hour

Analysis- As seen in the plot the proportion of bad words to total number of words is quite small. It is maximum during night time and is least during noon time.