Twitter Data Clustering

With the absence of any significant cricket or any event (sporting or non sporting), it was difficult to gather data for writing an event based clustering program. In the meantime, I picked up a trending hashtag on Twitter and captured all tweets for a day. Using this data, detecting events did not make much sense, so the program only clusters data into different categories.

#tag

One of the trending hashtags have been #PappuCensored since a few days. Some of the other related trending hashtags were #OfficeOfRG and #Pappu. Due to the nature of tweets in this case the clustering that is expected to happen will not be based on events, but on nature of comments. Please note that there is no political intent. :-)

Input Characteristics

Variable	Value
Total Tweets	40,556
Total Tokens	483,034
Unique Significant Tokens	6110
Clusters	10 (Randomly Set)

Clusters

The number of tweets in each cluster is as follows. The uneven distribution indicates the clustering has not been done properly. A deeper analysis of the contents would help us better segregate these tweets.

```
8    24664
9     5327
7     3970
0     2424
6     1222
2      772
1      731
4      602
3      522
5      322
```

Top Cluster Words

The top words in each of the clusters are as follows. This gives us an idea about the discussion points in that cluster. It's a bit difficult to gauge in these clusters, but for example, cluster 5 groups tweets about Rahul Gandhi's comment on potato turning into gold.

```
Cluster 0 Top Words:
['@incindia', 'patra', '@officeofrg', 'https', '@sambit', '@hardikpatel']
Cluster 1 Top Words:
['@seems3r', '@scrapravi', '@seematrivedi6', '@sbhandari16', '@officeofrg', 'zindabad', '@sengarajay235']
Cluster 2 Top Words:
['different', 'group', 'country', 'met', '@officeofrg', 'fiis']
Cluster 3 Top Words:
['congress', 'gold', '@mediacrooks', 'defend', '@naina0806', 'stop', 'aloo', 'p', 'stupid']
Cluster 4 Top Words:
['equally', 'reveres', 'm', 'bhakt', 'know', 'shiv']
Cluster 5 Top Words:
['potatoes', 'finally', 'gold', 'mechanism', 'got']
Cluster 6 Top Words:
['#', '#pappucensored', '#18nov', '@officeofrg', '2014', 'blackday']
Cluster 7 Top Words:
['60', '1', 't', 'https', '6', '8', '@officeofrg']
Cluster 8 Top Words:
['#', '#pappucensored', '@shuklapinku', '@officeofrg', 'https', '@narendramodi', '@mayankforbjp', '#pappu', 't']
Cluster 9 Top Words:
['word', 'congress', 'ec', 't', 'https', 'bjp', 'pappu', 'rahul']
```

Programming

Live Tweet Reader

This is accomplished through Twitter's API - tweets are read live and stored into a MongoDB database.

Tweet Consumer

The program processes the collection containing all the streamed tweets, performing the following operations:

1. Read tweets from Mongo DB collection
2. Perform tokenization and stemming of tweets
3. Convert each tweet into a TF/IDF vector
4. Perform K Means Clustering
5. Store Clustered Timeline Data in File

Live Cricket Streaming

If a cricket match would have been going on, it would surely have got more interesting. While broadly, clustering would happen in a fashion similar to the above problem, there will be quite a few different aspects.

Cluster Based Events

The clustering of the tweets would be based on lexicon based event recognition. Each lexicon will stand for a particular event, like a wicket, boundary, six etc.

Eg. Boundary = [‘boundary’, ‘four’, ‘4’], Wicket = [‘catch’, ‘out’, ‘bowled’, ‘stumped’, ‘lbw’, ‘wicket’]

We have not created these defined lexicons in our solution, as there was no event to track as such. However, in a cricket match live feed, these lexicons will help us directly cluster our tweets, and we wouldn’t require a computational TF/IDF vector clustering. The advantage here is that, no training data will be required for this, any incoming tweet can easily be evaluated.

In order to improve upon this, lexicons can be introduced for players as well.

Eg. Rahul Dravid = [‘rahul’, ‘dravid’, ‘wall’]

Together with an event lexicon, it can describe an event with the players involved. In our timeline, each cluster detected would be one point on the timeline, i.e. one event on the timeline. In the current case, there is no such cluster based timeline, it is just a series of time sorted tweets within each cluster.

Event Triggers

An event trigger needs to be defined to identify relevant events in the game. This is essential as a cricket match initiates a lot of tweets commenting on various aspects of the game, but are unlikely to give us any information. Intuitively, an event will trigger when there is a sudden burst of tweets.

We can generally process tweets in the previous one minute at all times, and whenever there is a sudden jump in the number of tweets in the second half of the time window over the first half, we can declare an event, and move on to process the tweets. If there is no trigger, the tweets in the time window need not be processed.

$$\frac{tweets\ in\ the\ first\ 30\ seconds}{tweets\ in\ the\ next\ 30\ seconds} > 2\ (threshold)$$

Prevent Event Duplicates

With a huge chunk of data coming in, event duplicates are likely in our method. This can be reduced by ignoring all retweets.