P2 – Indexing and Searching for hot topics with ElasticSearch, Trident, CountMin and HeavyHitters

---

- Please make sure you have unzipped the 'project' folder to location '~/apache-storm-0.9.3/examples/storm-starter/src/jvm/storm/starter/trident/'

- Open terminal and change your current working directory to directory where pom.xml resides, i.e. '~/apache-storm-0.9.3/examples/storm-starter'

- Run following command on terminal to compile and include dependencies [if any]

```
> mvn package
```

- Make sure correct source file 'stopwords.txt' containing stop-words is present under directory '~/apache-storm-0.9.3/examples/storm-starter/data'

Part B

1. Install latest version of ElasticSearch. In a new terminal, start using following commands:

```
> cd [ElasticSearch_directory]
> ./bin/elasticsearch
```

This will keep elastic search running in the background.

2. Switch back to first terminal where the current working directory contains pom.xml. You will require Twitter API keys to execute this code.

3. Run topology using following command

```
> storm jar target/storm-starter-0.9.3-jar-with-dependencies.jar storm.starter.trident.project.RealTimeElasticSearchTopology [CONSUMER_KEY] [CONSUMER_SECRET_KEY] [ACCESS_TOKEN] [ACCESS_TOKEN_SECRET] > op.txt &
```

All four keys can be obtained from twitter developer's website.

4. Please wait for sometime for program to execute and generate output. All the output will be redirected to file op.txt. To verify output please run below command

```
> cat op.txt | grep DRPC
```

Note: I have used following queries:
- *Query1*: termQuery → checks for word 'man' in the tweet.
- *Query2*: wildcardQuery → checks for tweet containing '*er*', * can be anything.
- *Query3*: termsQuery → checks if the tweet contains at least 2 words from following list of words : ["love","hate","man","moon","apples","very", "like"].
- *Query4*: fuzzyQuery → checks if the tweet contains 'RT', this is an easy way to get retweets.
- *Query5*: matchQuery → checks for tweets containing word 'hate'.
- *Query6*: boolQuery → finds tweets cotaining word 'love' but does not have word 'hate'.

Part C

1. Make sure your current working directory to directory where pom.xml resides, i.e. '~/apache-storm-0.9.3/examples/storm-starter' and file 'stopwords.txt' exists at location '~/apache-storm-0.9.3/examples/storm-starter/data/'

2. Run topology using following command

```
> storm jar target/storm-starter-0.9.3-jar-with-dependencies.jar storm.starter.trident.project.countmin.CountMinSketchTopology [CONSUMER_KEY] [CONSUMER_SECRET_KEY] [ACCESS_TOKEN] [ACCESS_TOKEN_SECRET] [K]
```

The keys are same as we used to run the part B of the project.
The K parameter value decides the number of top items we wish to display.

Please monitor the terminal as the top-K items will be displayed in every 4-seconds.

Note:
- The code currently fetches top-K items in the stream.
- Current implementation fetches top 5 items.

Following files were added/modified:

FilterStopWords
Location: projects/filters/FilterStopWords.java
-checks if word is present in bloom filter. If present it is made unavailable for further processing.

TopList
Location: project/countmin/TopList.java
- adds elements into priority queue to keep track of top-K items in the stream

CountMinTopK
Location: project/countmin/state/CountMinTopK.java
- query function which returns top K items stored in priority queue

CountMinSketchState
Location: project/countmin/state/CountMinSketchState.java
- modified to accpet parameter k, which is used to initialize priority queue

CountMinsSketchState
Location: project/countmin/state/CountMinSketchState.java
- initialies priority queue with size K.
- adds a functions to return top-K items in the priority queue