Twitter Data Analysis

Final Project Report

CSC 590 1 | Capstone Project | Spring 2018

Under the guidance of

Alex Wu

Submitted by,

Anu Mehndiratta

Student Id: 92336

ACKNOWLEDGEMENT

I would like to express my heartfelt appreciation and sincere gratitude to our Professor, Alex Wu for giving me this opportunity and encouraging my Project idea. I would also like to thank International Technological University for providing me with resources necessary for the Project.

Table of Contents

1.0 Introduction

Social media has become one of the most powerful tools in recent times for marketing and research that help organizations grow their business. To leverage the power of social media, we need to know how its users react when an event occurs (Presidential elections, Super Bowl, Apple product launch etc.). For this, we need to understand and analyze the data generated in the form of tweets, status updates etc.

Following the twitter hashtags on a particular trending issue can provide valuable business intelligence data such as mentioned below:

i. The public sentiments on an issue which in turn can help determine their attitude about the company. This aids the brand monitoring and in case of any negative tweets, the organization can take corrective measures to do immediate damage control.

ii. Demography of the community following the tweets and analysis of this data can provide valuable insights to outline target audience. Twitter marketing can thus help define the profiles of potential customers, their interests, locations etc.

iii. Performance of the hashtag content used in ad campaigns that can help find out what really resonates with the online community.

In this project, we plan to take twitter data samples with the requisite hashtags and provide useful results in the form of the number of users who tweeted using the hashtag, number of tweets containing media, number of tweets per users(celebrities or non-celebrities) etc. This data can very well be used in various data analysis models.

There are many instances in the business markets which have harnessed the power of the Twitter platform for marketing their products by reaching out to a global audience

2.0 Data Requirements

Since user profile data and hashtags from Twitter would be unstructured, data cannot be ordered in a relational or a hierarchical database. The data for our project will be extracted from Twitter using Apache Flume. The data would be in the JSON format and will be made available to the application logic. Data like

username, twitter handle, location from where the user tweeted, number of followers etc. constitute our data.

Steps for Data Retrieval:

- Provide desired key words in the Flume configuration file and provide the Access Token and Secret Token provided by Twitter.
- Run the Flume agent using the open source jar provided in the readme manual.
- The data is then stored on HDFS in the form of JSON. This data is completely unstructured for the following reasons:
    - All the users may not be verified users. So the verified field of that particular tweet would be empty.
    - Some users may not have followers or may be following very few people.
    - All this data cannot be stored in the form of relational data tables.

This raw data is converted into Hive Database, which is an eco-system of Hadoop and has in-built NoSQL implementation. Since major chunk of current days' data is schema less and we wanted to make use of a database or a data warehouse that provides this functionality, we chose Hive.

3.0 Design Diagrams

Hadoop Map-Reduce framework is built as Master-Slave architecture, with name node controlling the data nodes for data storage and job tracker controlling the task trackers for data processing purposes. Our system is a single node Hadoop cluster with a name node and a data node. The application architecture of the system developed is Layered architecture. We have chosen this architecture as this correctly fits in with our business requirements. Also since we would like to expand our system later, this type of architecture is featured to be scalable. Since data is added to our system every day, data can be added to the system without changing the application logic. The components in the figure 3.1.1 are explained briefly below:

Web Users: The users interact with the system with the front end provided. The user selects the operations of his choice to perform on the application, in this case – selecting the location wise tweet information or other metrics and submitting the options.

Presentation Layer: This layer provides the user interface that the web users interact with. Users provide the input and get the corresponding output, i.e the metric details the requested details.

Business Layer: The business layer for this project consists of the application logic to fetch raw data from HDFS and store it in Hive understandable format, which is later used for querying the results. The output is provided to the presentation layer after the operations are performed on the Hive database.

Datastore Layer: The datastore layer has the HDFS data repository to store the raw data that is fetched from Twitter. We have created a configuration file that has important information about what data should be retrieved and where the data should be stored on HDFS. An open source Flume jar is tailored to the configuration file that fetches the data from Twitter based on the parameters given in the config file.

DataSource: Twitter Inc. is our data source. Twitter provides an option to its users to access the public data by creating an application. This process is explained in details in the subsequent section.

Layered Architecture for Twitter Data Retrieval and Analytics

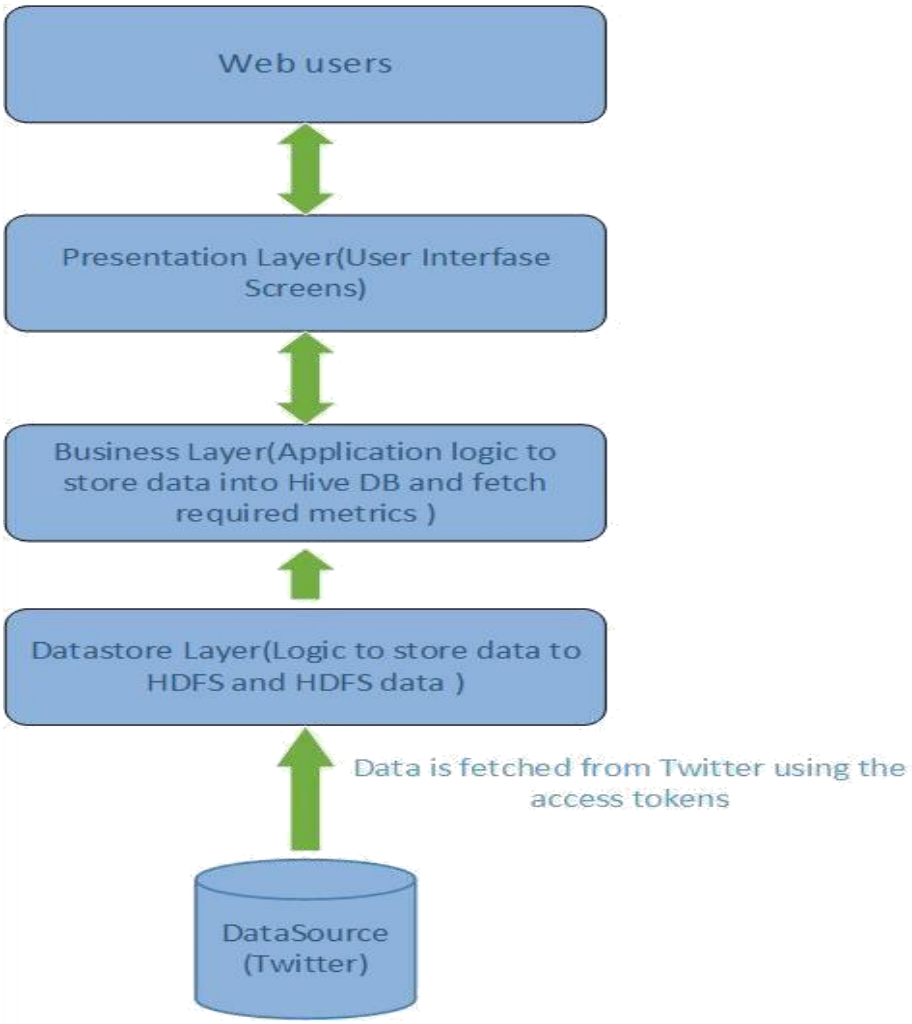


Figure 3.1.1 Layered Architecture Diagram

4.0 Twitter Data Analysis System:

As per the context diagram shown below, our system interacts with Twitter and gets the required data. The above shown boundary is considered because all the operations that are performed on the system begin after Twitter releases its data. Also, since Twitter releases data six hours after the tweet has been posted, our system does not have control over that time interval. Hence, Twitter API is external to our system.

Twitter API: All the information that the system gets is from Twitter, like the tweet text, the metadata of the tweet like tweet ID, user screen name, user geographic location, verified status, various counts related to the user etc. All the data that is fetched from Twitter API is mentioned as licensed data in its Developer Agreement and Policy as we have used one of our twitter accounts.

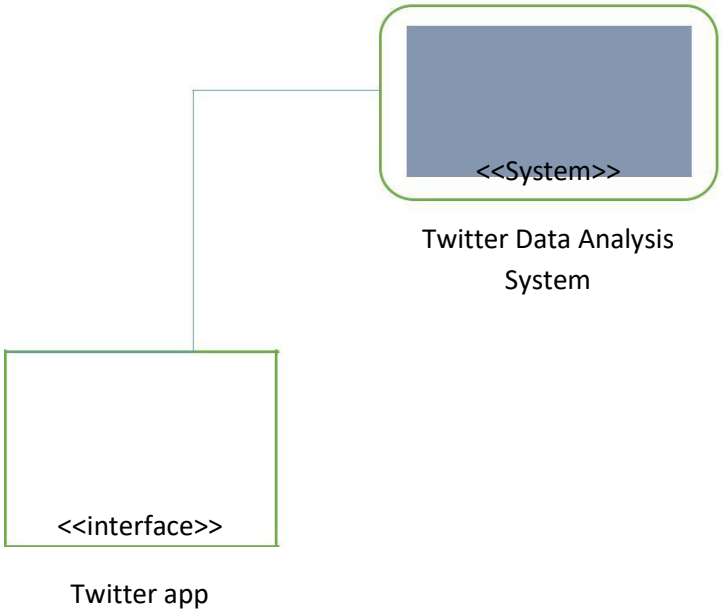
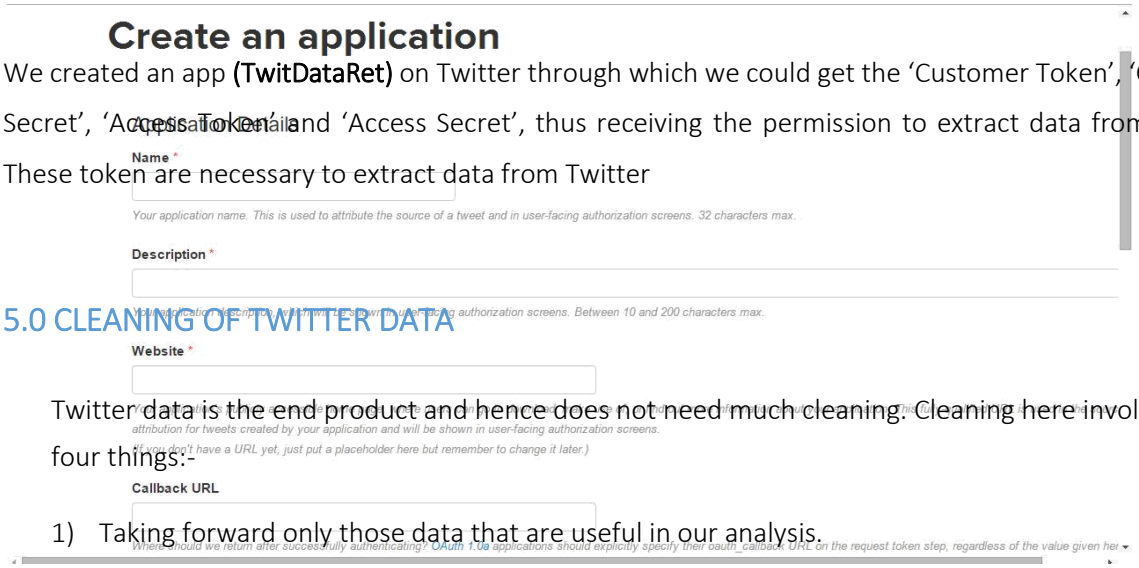


Figure 3.2.1 Context Diagram

4.1 Twitter App Creation

Each of the team members were successfully able to create the twitter account and post tweets so as to build on the data and get the most recent data. The limitation which twitter put recently on the use of its APIs is that the most recent data that can be extracted is the one posted 6 hours before the time of extraction.



We created an app **(TwitDataRet)** on Twitter through which we could get the 'Customer Token', 'Customer Secret', 'Access Token' and 'Access Secret', thus receiving the permission to extract data from Twitter. These token are necessary to extract data from Twitter

5.0 CLEANING OF TWITTER DATA

Twitter data is the end product and hence does not need much cleaning. Cleaning here involves four things:-

1) Taking forward only those data that are useful in our analysis.
2) Changing the date format from twitter format to UNIX format for processing in HIVE.
3) Removal of foreign language characters
4) Removal of stop words.

6.0 Hive

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data and makes querying and analyzing easy. Apache Hive (HiveQL) with Hadoop Distributed file System is used for Analysis of data. Hive provides a SQL-like interface to process data stored in HDP. Due its SQL-like interface, Hive is increasingly becoming the technology of choice for using Hadoop. To set up HIVE in Hadoop

Build or Download the JSON file Before we can query the data, we need to ensure that the Hive table can properly interpret the JSON data. By default, Hive expects that input files use a delimited row format, but our Twitter data is in a JSON format, which will not work with the defaults. And we can use the Hive SerDe interface to specify how to interpret what we've loaded. SerDe stands for Serializer and Deserializer, which are interfaces that tell Hive how it should translate the data into something that Hive can process.

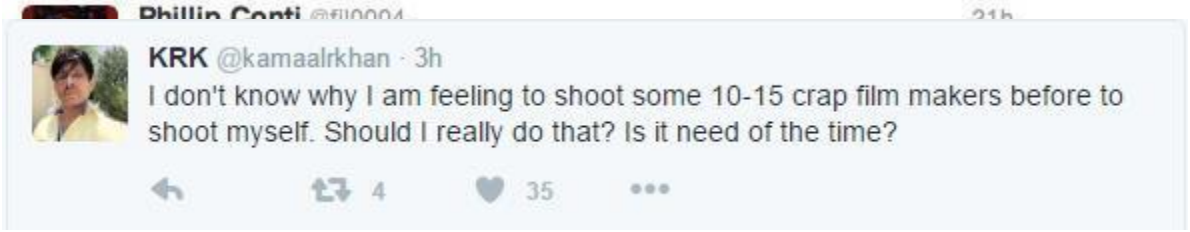
7.0 SENTIMENT ANALYSIS AND RATING CALCULATION

What is Sentiment Analysis?

• It's a classification of polarity of a given text in the document, sentence or phrase • The goal is to determine whether the expressed opinion in the text is positive, negative or neutral

Positive tweet: -

Negative tweet: -



Neutral tweets:

Dictionary AFINN is a list of English words rated for valence with an integer between minus five (negative) and plus five (positive). The words have been manually labeled by Finn Årup Nielsen in 2009-2011. The file is tab-separated. There are two versions:

AFINN-111: Newest version with 2477 words and phrases.

AFINN-96: 1468 unique words and phrases on 1480 lines. Note that there are 1480 lines, as some words are listed twice. The word list in not entirely in alphabetic ordering.

Loaded the word into dictionary hive table to do the sentiment analysis

1. create table dictionary (word string,rating int) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
2. LOAD DATA INPATH '/AFINN.txt' into TABLE dictionary;

```
1 | create table dictionary(word string,rating int) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
```

```
hive> create table dictionary(word string,rating int) ROW FORMAT DELIMITED FIELDS TERMINATED BY '/t';
OK
Time taken: 0.244 seconds
hive>
```

Now, let's load the AFINN dictionary into the table by using the following command:

```
1 | LOAD DATA INPATH '/AFINN.txt' into TABLE dictionary;
```

We have this AFINN dictionary in the root directory of HDFS.

```
hive> LOAD DATA INPATH '/AFINN.txt' into TABLE dictionary;
Loading data to table default.dictionary
Table default.dictionary stats: [numFiles=1, totalSize=28094]
OK
Time taken: 0.21 seconds
```

We can view the contents of the dictionary table by using this command:

```
1 | select * from dictionary;
```

```
hive> select * from dictionary;
OK
abandon -2      NULL
abandoned       -2      NULL
abandons        -2      NULL
abducted        -2      NULL
abduction       -2      NULL
abductions      -2      NULL
abhor   -3      NULL
abhorred        -3      NULL
abhorrent       -3      NULL
abhors  -3      NULL
abilities       2       NULL
ability 2       NULL
aboard  1       NULL
absentee        -1      NULL
absentees       -1      NULL
absolve 2       NULL
absolved        2       NULL
absolves        2       NULL
absolving       2       NULL
absorbed        1       NULL
abuse   -3      NULL
abused  -3      NULL
abuses  -3      NULL
abusive -3      NULL
accept  1       NULL
accepted        1       NULL
accepting       1       NULL
```

3. Convert Sentences into array of words

```
-- Compute sentiment
create view l1 as select id, words from Mytweets_raw lateral view explode(sentences(lower(text))) dummy as words;
create view l2 as select id, word from l1 lateral view explode( words ) dummy as word;
```

4. Look up from the dictionary and find out its polarity

```
-- CHANGED!!!
create view l3 as select
    id,
    l2.word,
    case d.polarity
      when  'negative' then 0
      when 'positive' then 5
      else 2.5 end as polarity
 from l2 left outer join dictionary d on l2.word = d.word;
```

5. Find the Rating

```
--(WORKED - overall rating)
select
  sum(a.polarity) s,
  count(b.polarity) c,
  sum(a.polarity)/ count(b.polarity) r
from
  (select * from l3 where polarity=0 or polarity=5) a
join
  (select * from l3 where polarity=0 or polarity=5) b
on (a.id = b.id);
```

8.0 ANALYSIS

1. Country-wise Rating:

```
create view l4 as select
tweets_clean.country,
avg(l3.polarity) as averg
from l3 left outer join tweets_clean on l3.id = tweets_clean.id group by tweets_clean.country having tweets_clean.country IS NOT NULL order by averg desc;
```

2. Top Hashtags

```
create view l5 as
select ht,count(ht) as countht
from Mytweets_raw lateral view
explode(entities.hashtags.text) dummy as ht
group by ht
order by countht desc;
```

3. Ratings by different sources

```
create view l6 as select
substr(tweets_clean.source,instr(tweets_clean.source,">"),instr(tweets_clean.source,"</a>") - instr(tweets_clean.source,">")),
avg(l3.polarity) as averg
from l3 left outer join tweets_clean on l3.id = tweets_clean.id group by tweets_clean.source;
```

4. Users with most tweets

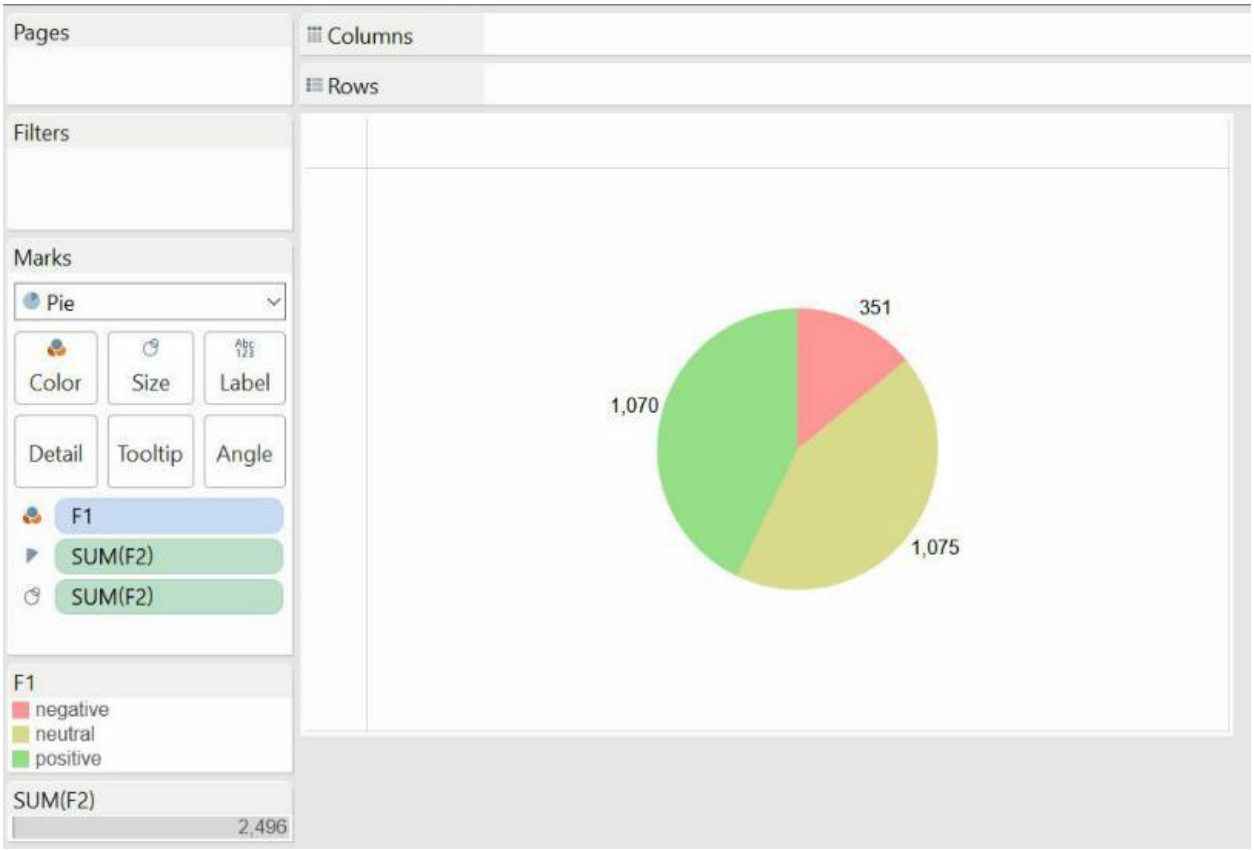
```
create view l7 as
select screen_name,count(id) as cnt
rom tweets_simple
group by screen_name
having screen_name IS NOT NULL
order by cnt desc;
```

5. Sentiment count

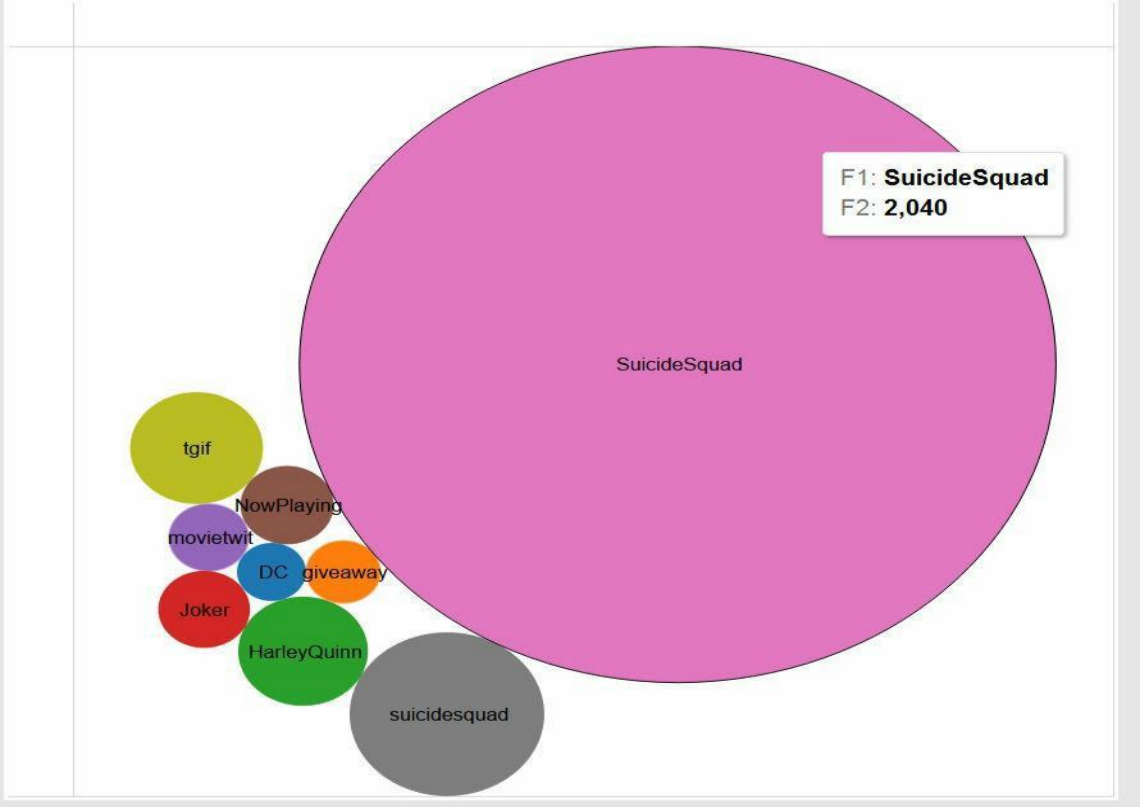
```
create view l8 as
select sentiment, count(id)
from tweets_sentiment ts
group by sentiment;
```

9.0 RESULTS

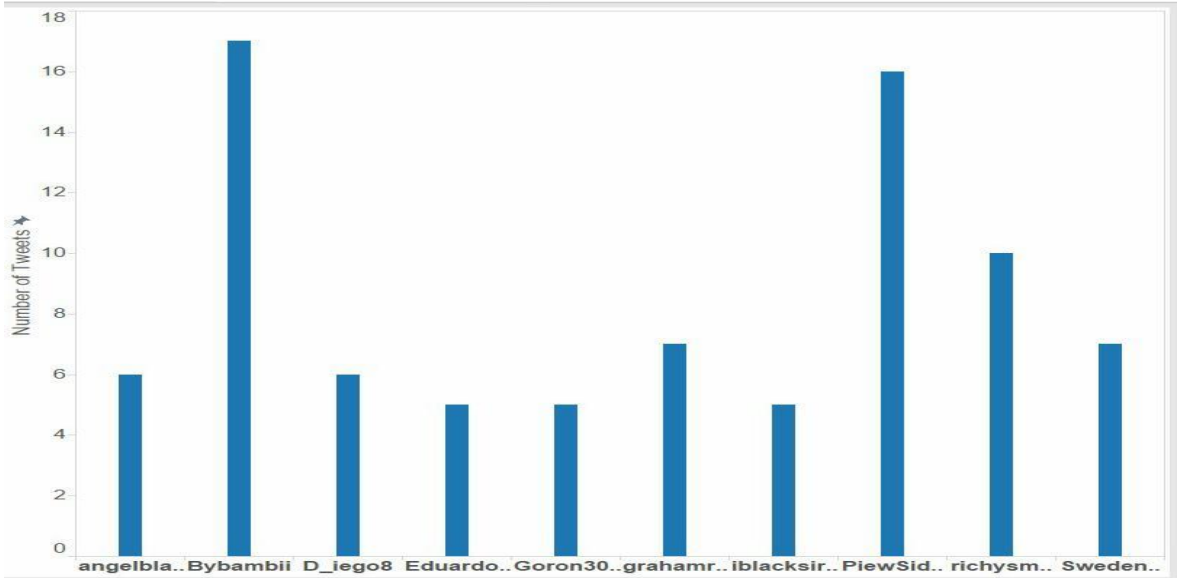
Sentiments as positive, negative and neutral



Top ten #tags State



Top users who tweeted about the product



10.0 PROBLEMS FACED

1. Retrieving the twitter data was difficult as twitter have not made their data public. Data is provided only to twitter apps.
2. Flume setup needed a lot of complex classpath JARs and configurations.
3. Creating a customized JSON SerDe for conversion of unstructured JSON data to structured Hive tables.
4. Setting up Amazon Web Services for running HIVE queries.
5. Connecting HIVE with Tableau.

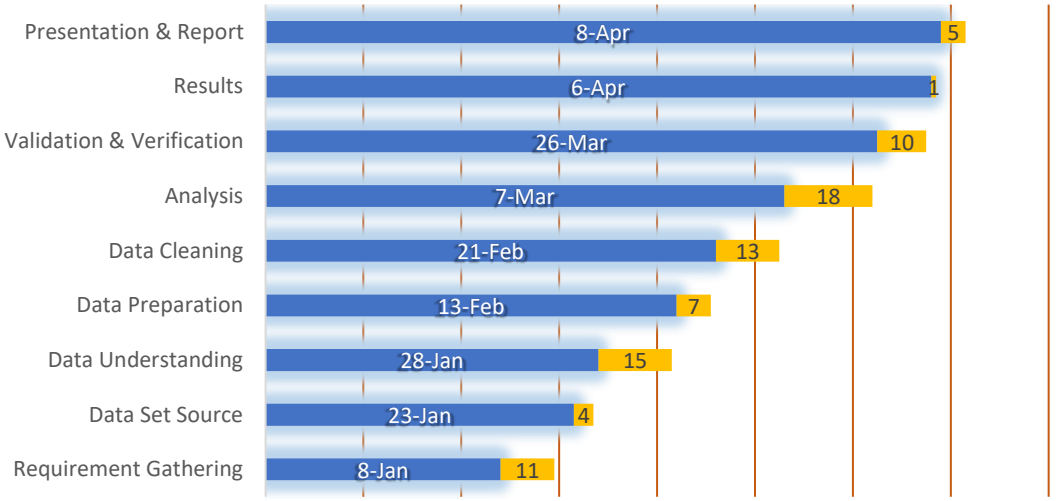
11. 0 CONCLUSION

Apache FLUME is a powerful technology for data streaming. We have combined the real stream data with our spam detection function for classification of tweets as Spam. We have combined real stream data with NLP dataset to get sentiment analysis We successfully calculated the Rating System and carried out analysis on the ratings.

12. 0 Future Scope

1. **Negation Handling -** Words like 'no', 'not', and 'never' are difficult to handle properly. This can be handled by using n-grams which is an inbuilt function in **HIVE** which can analyze two or three words together and then look up in the dictionary look up.
2. **Sarcasm Detection** – Sarcasm is tough to detect in written form unless you know the context. When speaking, the tone usually gives away the Sarcasm. That's not the case in written text though. This can be handled to almost the accuracy of 80% by machine learning techniques and by NLP by analyzing the smileys and words like 'lol','wow','(not)','!!!'

13.0 GANTT CHART



	Requirement Gathering	Data Set Source	Data Understanding	Data Preparation	Data Cleaning	Analysis	Validation & Verification	Results	Presentation & Report
Start Date	8-Jan	23-Jan	28-Jan	13-Feb	21-Feb	7-Mar	26-Mar	6-Apr	8-Apr
Duration	11	4	15	7	13	18	10	1	5

14.0 REFERENCES

1. Agarwal, A., Xie, B., Vovsha, I., Rambow, O., Passonneau, R.: Sentiment analysis of twitter data. In: Proc. ACL 2011 Workshop on Languages in Social Media. pp. 30–38 (2011)
2. Barbosa, L., Feng, J.: Robust sentiment detection on twitter from biased and noisy data. In: Proceedings of COLING. pp. 36–44 (2010)
3. Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., Smith, N.: Part-of-speech tagging for twitter: Annotation, features, and experiments. Tech. rep., DTIC Document (2010)
4. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford (2009)
5. Guerra, P., Veloso, A., Meira Jr, W., Almeida, V.: From bias to opinion: A transfer-learning approach to real-time sentiment analysis. In: Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (2011)
6. ACM conference on Information and knowledge management. pp. 375–384. ACM (2009)

7. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Proceedings of the European Conference on Machine Learning. pp. 318–329 (2006)
8. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. Proceedings of LREC 2010 (2010)
9. Rizzo, G., Troncy, R.: Nerd: Evaluating named entity recognition tools in the web of data. In: Workshop on Web Scale Knowledge Extraction (WEKEX11). vol. 21 (2011)
10. Saif, H., He, Y., Alani, H.: Semantic Smoothing for Twitter Sentiment Analysis. In: Proceeding of the 10th International Semantic Web Conference (ISWC) (2011)
11. Grant Stafford, Louis Lei Yu : An Evaluation of the Effect of Spam on Twitter Trending Topics. In: Gustavus Adolphus College Conference

15.0 Git Hub Link

https://github.com/amehendiratta/Anu_Bigdata