Twitter Data Analytics using Spark

Prudhvi Raj (16208160), Bhargav(16207553), Vipin (16208781), Sudhakar (16209800)

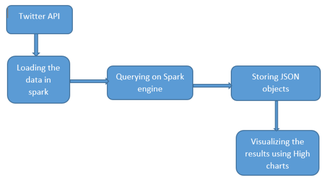
Introduction

- Big data analytics is the use of analytical techniques used against large diverse data to uncover hidden patterns, unknown correlations, market trends, customer preferences and other useful business information.
- Twitter streaming data is stored, processed to observe the current prevailing trends in the tweets.
- Twitter data was filtered using various filter criteria such as - thanksgiving, balckfriday, Christmas.
- The recent trend with most tweets at the time was thanksgiving.
- To perform analytics we have retrieved data from twitter using API on thanksgiving
- Web UI was designed for the user to visualize the data in the form of graphs for better understanding.

Aims & Objectives

- Collect sufficient tweet data for analytics.
- Dump the data into a database for querying.
- Write sparkSQL queries to retrieve meaningful data from the collected data
- Visualize the results using graphs using interactive User Interface.

Architecture



Query Results

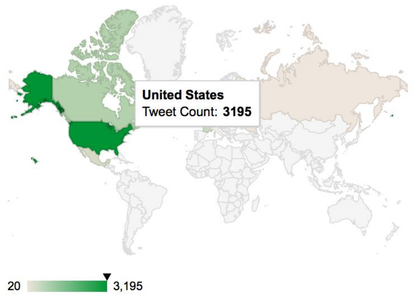
- **Query 1**

Location based tweet analysis with the count of tweets from each location

SQL Query:-

SELECT place.country AS country ,COUNT(*) AS country_count from tweets WHERE place.country is not null GROUP by place.country order by country_count desc limit 10

Graph:-



Query 2:

Most frequent text in user tweets

SQL Query:

SELECT aWord, COUNT(*) AS WordOccuranceCount FROM (SELECT SUBSTRING_INDEX(SUBSTRING_INDEX(concat(user.description, ' '), ' ', aCnt), ' ', -1) AS aWord FROM tweets CROSS JOIN (SELECT a.i+b.i*10+c.i*100 + 1 AS aCnt FROM integers a, integers b, integers c) Sub1 WHERE (LENGTH(text) + 1 - LENGTH(REPLACE(text, ' ', ''))) >= aCnt) Sub2 WHERE Sub2.aWord != '' GROUP BY aWord ORDER BY WordOccuranceCount DESC LIMIT 6

Graph:

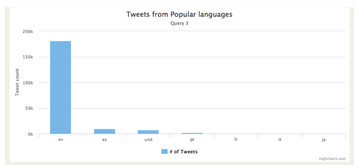
- **Query 3**

Distinct languages used in tweets

SQL Query:

val q3 = sqlContext.sql("SELECT DISTINCT lang,COUNT(lang) AS tweet_count FROM
tweets GROUP BY lang LIMIT 7") save output file:

Graph:



- **Query 3**

Retrieve the count of people who are talking about marts/shopping

SQL Query:

val q4 = sqlContext.sql("SELECT COUNT(text) AS Walmart_visitors from tweets Where text regexp('[*mart]')")

Graph:-

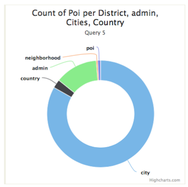
- **Query 4**

Discover the location type from which the user is active

SQL Query:

val q6 = sqlContext.sql("SELECT DISTINCT place.place_type,COUNT(place.place_type) AS tweet_count FROM tweets GROUP BY place.place_type")

Graph:-



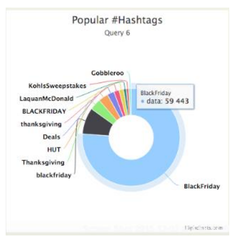
- **Query 6**

To get the famous hashtags used by the tweeters from the tweets

SQL Query:

val q7 = sqlContext.sql("SELECT entities.hashtags[0].text, count(entities.hashtags[0].text) as famous_tags FROM tweets group by entities.hashtags[0].text order by famous_tags desc limit 10");

Graph:-



Query 7:-

Time zone, Tweet count and retweet count from the data

SQL Query:

val x1 = sqlContext.sql("select user.time_zone as time_zone, count(*) as Tweet_count from tweets where user.time_zone is not null group by user.time_zone order by Tweet_count desc")

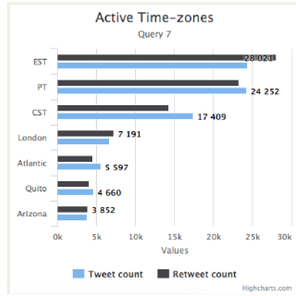
x.registerTempTable("x1")

val x2 = sqlContext.sql("select retweeted_status.user.time_zone as time_zone, count(*) as Retweet_count from tweets where retweeted_status.user.time_zone is not null group by retweeted_status.user.time_zone order by Retweet_count desc")

x2.registerTempTable("x2")

val Query5 = sqlContext.sql("select x1.time_zone, x1.Tweet_count, x2.Retweet_count from x1 inner join x2 on x1.time_zone = x2.time_zone order by x1.Tweet_count desc")

Graph:-



Conclusion

This web UI can be made more interactive by providing the facility for the user to specify his own custom queries and plot the results in graphs.

Further this can be used for sentiment analysis by improvising our query8.

Our UI can be integrated with a third party application for more complicated analysis on the data.

UMKC
UNIVERSITY OF MISSOURI-KANSAS CITY