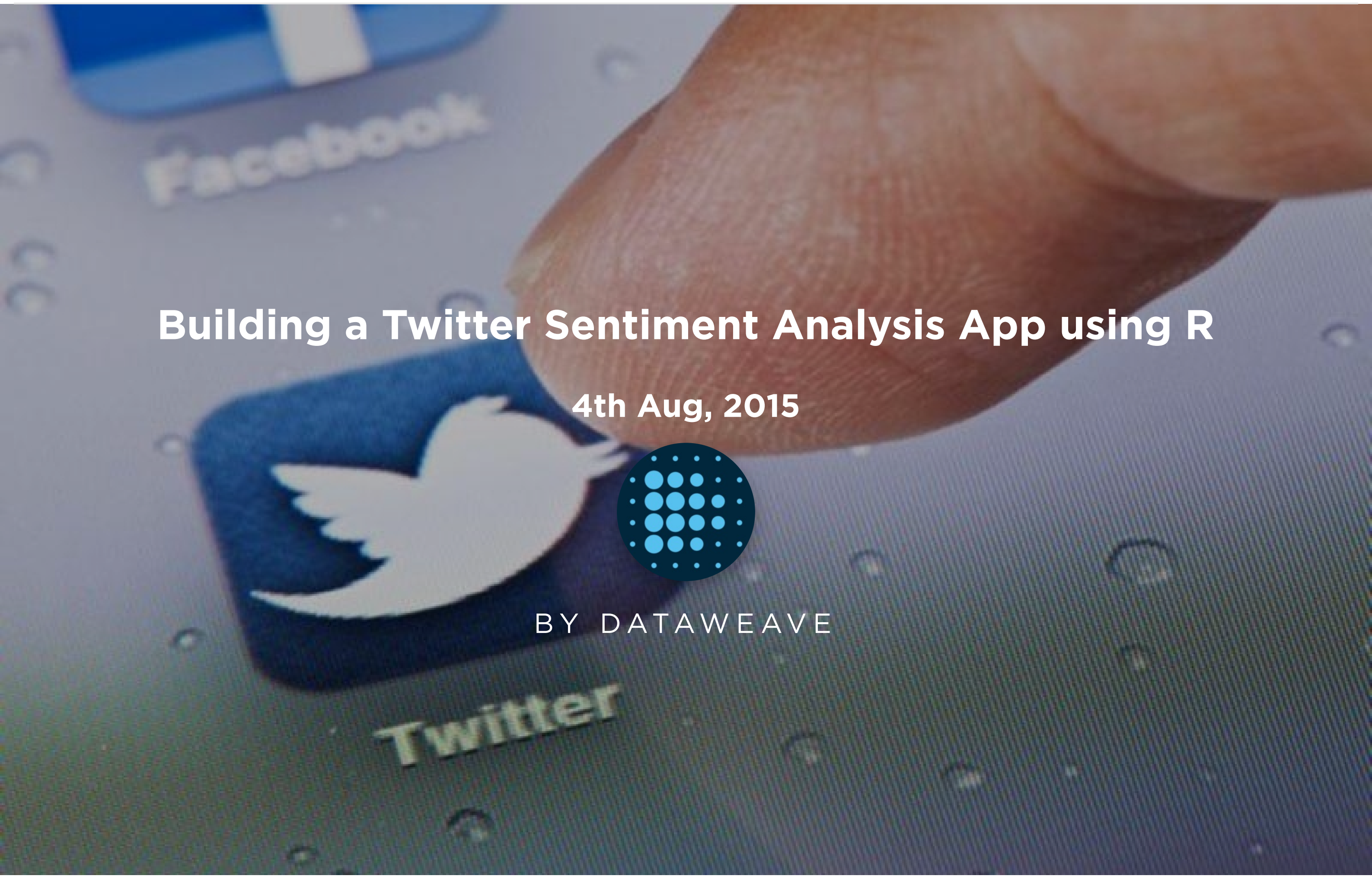


Building a Twitter Sentiment Analysis App using R

4th Aug, 2015

BY DATAWEAVE

Twitter, as we know, is a highly popular social networking and micro-blogging service used by millions worldwide. Each status or tweet as we call it is a 140 character text message. Registered users can read and post tweets, but unregistered users can only view them. Text mining and sentiment analysis are some of the hottest topics in the analytics domain these days. Analysts are always looking to crunch thousands of tweets to gain insights on different topics, be it popular sporting events such as the FIFA World Cup or to know when the next product is going to be launched by Apple.

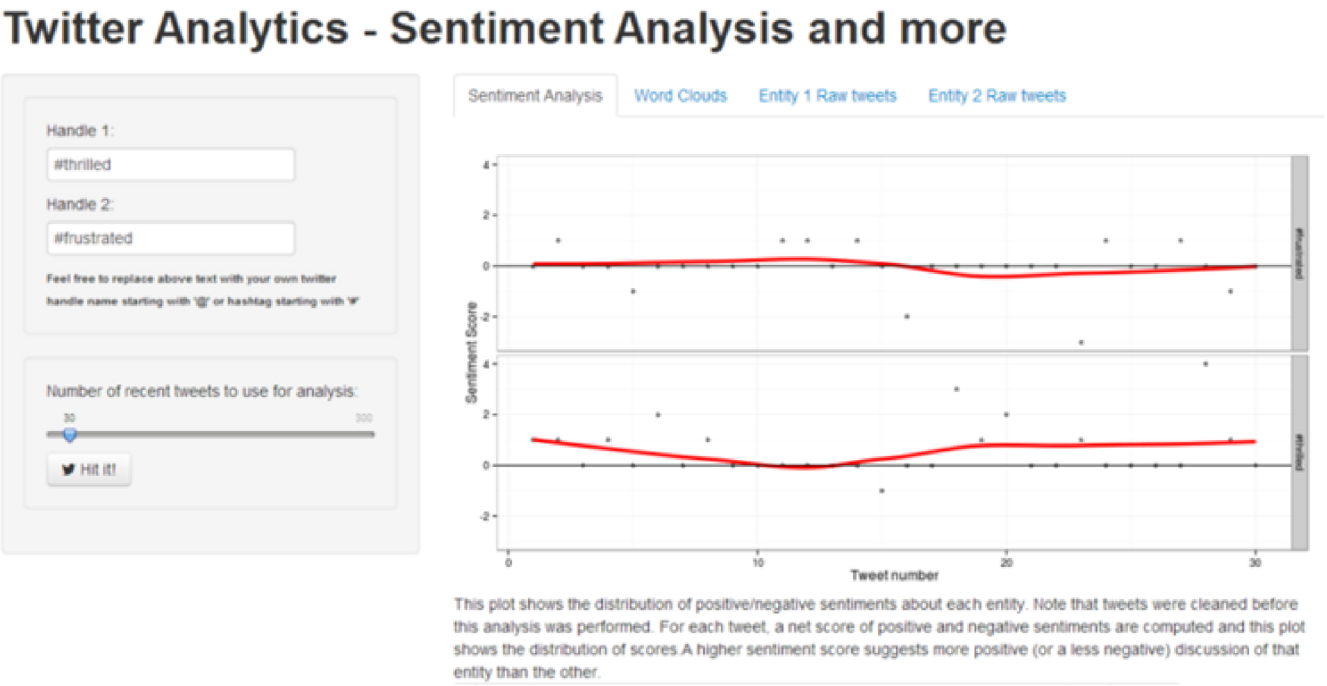
Today, we are going to see how we can build a web app for doing sentiment analysis of tweets using R, the most popular statistical language. For building the front end, we are going to be using the '**Shiny**' package to make our life easier and we will be running R code in the backend for getting tweets from twitter and analyzing their sentiment.

The first step would be to establish an authorized connection with Twitter for getting tweets based on different search parameters. For doing that, you can follow the steps mentioned in this document which includes the R code necessary to achieve that: **http://rpubs.com/dipanzan/twitterAccess**

After obtaining a connection, the next step would be to use the 'shiny' package to develop our app. This is a web framework for R, developed by **RStudio**. Each app contains a server file (server.R) for the backend computation and a user interface file (ui.R) for the frontend user interface.

You can get the code for the app from **my github repository here** which is fairly well documented but I will explain the main features anyway.

The first step would be to develop the UI of the application, you can take a look at the ui.R file, we have a left sidebar, where we take input from the user in two text fields for either twitter hashtags or handles for comparing the sentiment. We also create a slider for selecting the number of tweets we want to retrieve from twitter. The right panel consists of four tabs, here we display the sentiment plots, word clouds and raw tweets for both the entities in respective tabs as shown below.


Twitter Analytics - Sentiment Analysis and more

Handle 1: #thrilled

Handle 2: #frustrated

Feel free to replace above text with your own twitter handle name starting with '@' or hashtag starting with '#'

Number of recent tweets to use for analysis: 30

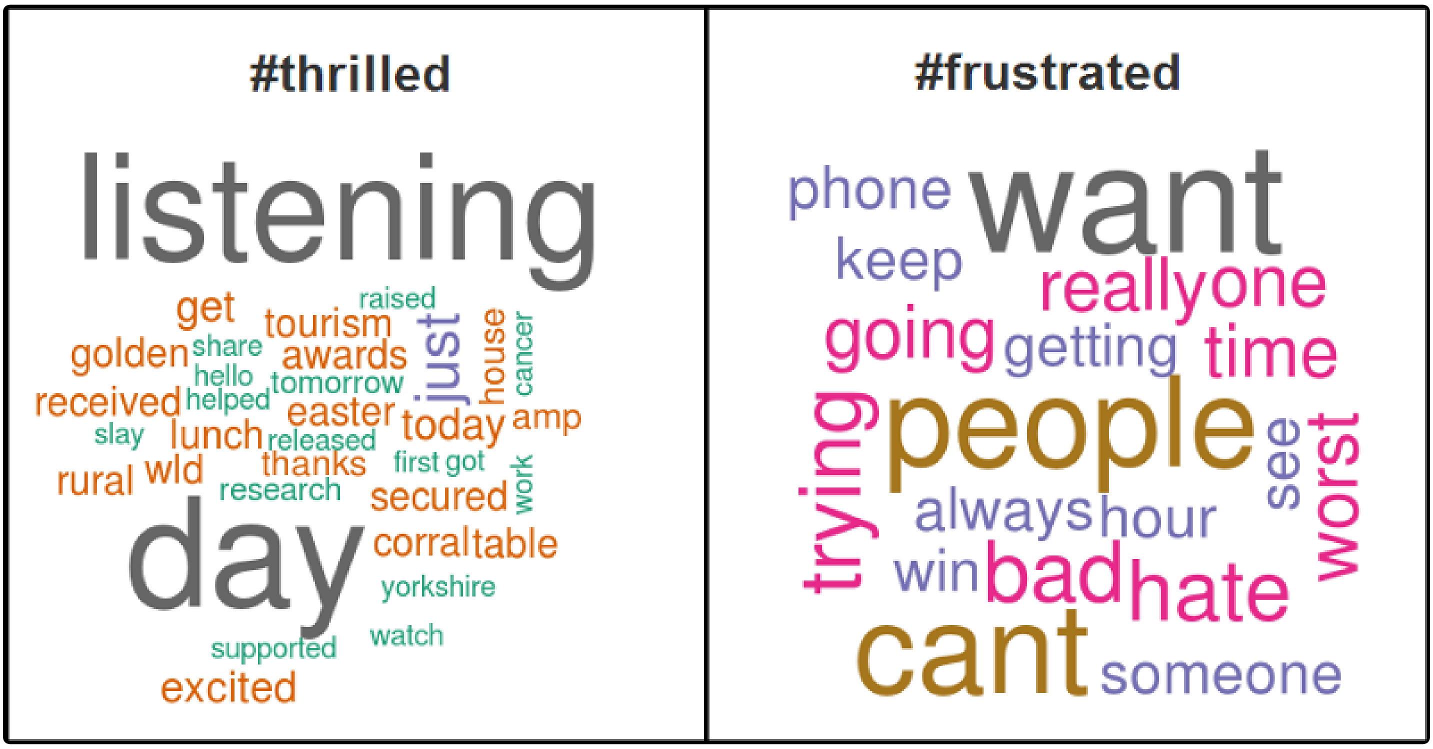
This plot shows the distribution of positive/negative sentiments about each entity. Note that tweets were cleaned before this analysis was performed. For each tweet, a net score of positive and negative sentiments are computed and this plot shows the distribution of scores.A higher sentiment score suggests more positive (or a less negative) discussion of that entity than the other.

Coming to the backend, remember to also copy the two dictionary files, 'negative_words.txt' and 'positive_words.txt' from the repository because we will be using them for analyzing and scoring terms from tweets. On taking a close look at the server.R file, you can notice the following operations taking place.

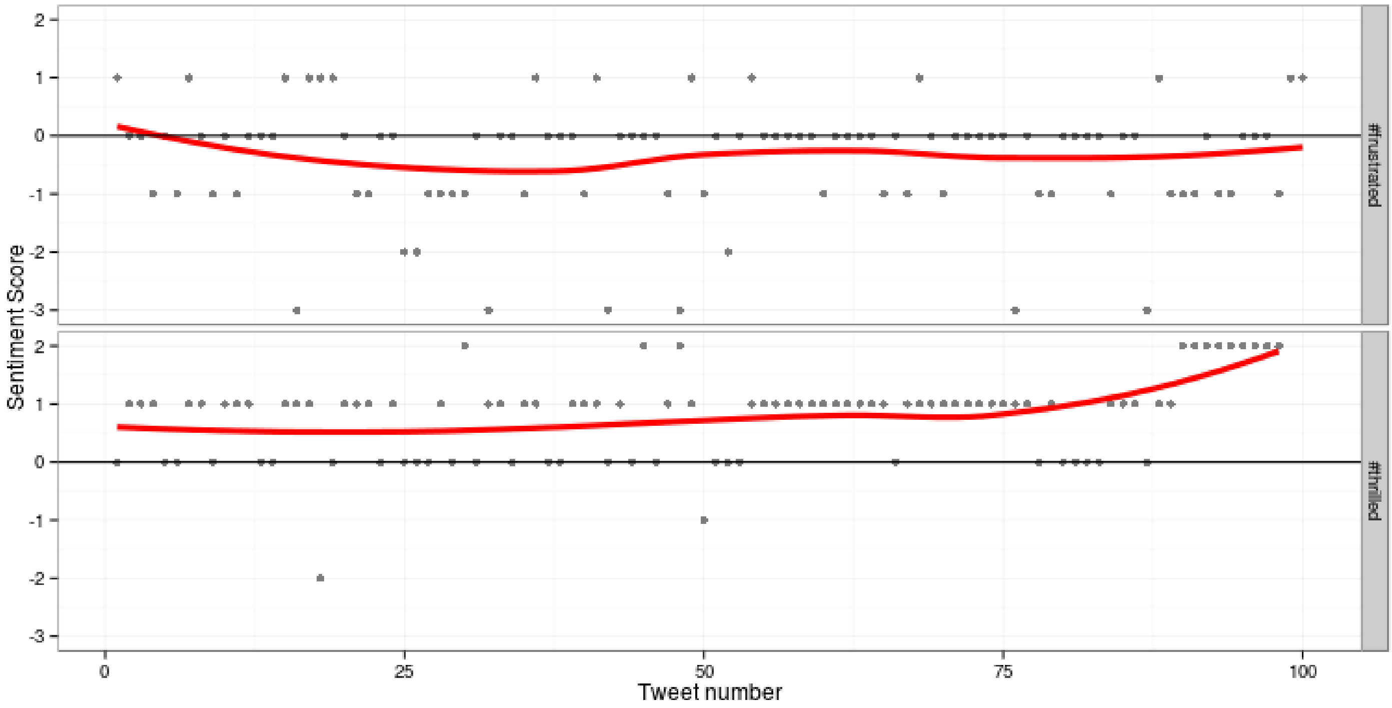
– The 'TweetFrame' function sends the request query to Twitter, retrieves the tweets and aggregates it into a data frame. — The 'CleanTweets' function runs a series of regexes to clean tweets and extract proper words from them. — The 'numoftweets' function calculates the number of tweets. — The 'wordcloudentity' function creates the word clouds from the tweets. — The 'sentimentalanalysis' and 'score.sentiment' functions performs the sentiment analysis for the tweets.

These functions are called in reactive code segments to enable the app to react instantly to change in user input. The functions are documented extensively but I'll explain the underlying concept for sentiment analysis and word clouds which are generated.

For word clouds, we get the text from all the tweets, remove punctuation and stop words and then form a term document frequency matrix and sort it in decreasing order to get the terms which occur the most frequently in all the tweets and then form a word cloud figure based on those tweets. An example obtained from the app is shown below for hashtags '#thrilled' and '#frustrated'.



For sentiment analysis, we use Jeffrey Breen's **sentiment analysis algorithm cited here**, where we clean the tweets, split tweets into terms and compare them with our positive and negative dictionaries and determine the overall score of the tweet from the different terms. A positive score denoted positive sentiment, a score of 0 denotes neutral sentiment and a negative score denotes negative sentiment. A more extensive and advanced n-gram analysis can also be done but that story is for another day. An example obtained from the app is shown below for hashtags '#thrilled' and '#frustrated'.



After getting the server and UI code, the next step is to deploy it in the server, we will be using shinyapps.io (**https://www.shinyapps.io/**) server which allows you to host your R web apps free of charge. If you already have the code loaded up in RStudio, you can deploy it from there using the 'deployApp()' command. For more details on app building and hosting check out the **official tutorial here.**

You can check out a live demo of the app here: **https://dipanjan.shinyapps.io/twitter-analysis/**

It's still under development so suggestions are always welcome.

Originally published at **blog.dataweave.in**.

- DataWeave Marketing

4th Aug, 2015

BRAND PERCEPTION