**TWEET BASED INTELLIGENT MOVIE MARKETING**

**BY**

**SHREYA JAIN**

**SHRUTI MANDKE**

**MUSHTAQ RIZVI**

**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr. No. | Heading | Page No. |
| 1. | Introduction | 3 |
| 2. | Solution | 4 |
| 3. | Libraries | 5 |
| 4. | Setting up twitter account | 6-8 |
| 5. | Data Cleaning | 9-10 |
| 6. | Sentiment Analysis using R | 11 |
| 7. | Visualization | 12-22 |
| 8. | Shiny | 23-24 |
| 9. | Power of Sentiments | 25-26 |
| 10. | Scope of Improvement | 27 |
| 11. | Conclusion | 28 |
| 12. | References | 29 |

**Introduction**

We have created an end to end user application for Movie Producers using movie's trailer tweets retrieved from Twitter. Our task is to analyze the feedback of a trailer (either positive or negative) based on tweets that have mentioned the movie trailer, for example, using hashtags. Twitter is one of the most popular social media where people post comments about their everyday life.   
Users share their reviews about a movie’s trailer on Twitter which possibly encourage or discourage their followers’ motivations of watching the movie. Since Twitter restricts each tweet to be less than 140 characters, users’ comments tend to be straightforward. In addition, because of its huge influences, many movies have started to include a hashtag in their trailers that tell people to tweet about them to attract social attention. Therefore, Twitter has become a great platform to examine people’s feedback on a trailer. By analyzing the sentiment of thousands of tweets by people who have watched the trailer, a general review of the movie can be made. Producers can then use the review to decide in what amount the movie should be screened in a particular city.

**Solution**

We choose Naive Bayes Network algorithm as our approach to this problem.

Naive Bayes Network is a simple and popular approach towards sentiment analysis. It is very easy to implement, and works well with big training data. The more of training dataset, the better of testing performance.

We choose each word of a movie review as an attribute, storing the times appears in positive or negative class.

●Implementation of an algorithm for automatic classification of tweet into positive, negative or neutral.

●Sentiment Analysis to determine the attitude of movie is positive, negative or neutral towards the subject of interest.

●Graphical representation of the sentiment in form of different charts

●Find the top influential users for a particular location

**Libraries**

The libraries used in the project at different places:

1) ***‘tm’*** to use all the functions for text mining.

2) ***‘twitteR’*** to scrape tweets and interact with twitter.

3) ***‘ROAuth’*** to create a handshake between twitter and R.

4) ‘***sentiment’***

5) ***"httr"***

6)’ ***NLP***’

7)’ ***wordcloud***’

8)’ ***RColorBrewer***’

9)’ ***RCurl***’

10)’ ***bitops***’

11)’ ***plyr***’

12)’ ***sentiment***’

13)’ ***ggplot2***’

14)’ ***stringr***’

15)’ ***lattice***’

16)’ ***Rstem***’

**Setting up a Twitter Account**

In this project, we utilize information available through the Twitter API to gather personal profile information about the users and their friends. Aside from just information about individuals we collected the tweets by these users, location from where they tweeted, their re-tweets, favorites, and friends.

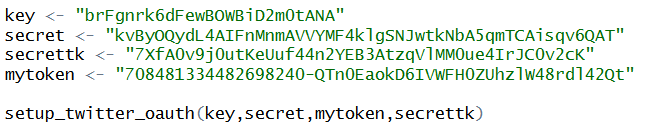
To perform above analysis, we have used the concept of Text Mining. We have installed following libraries in R for extracting the tweet data from twitter.

1) ***‘tm’*** to use all the functions for text mining.

2) ***‘twitteR’*** to scrape tweets and interact with twitter.

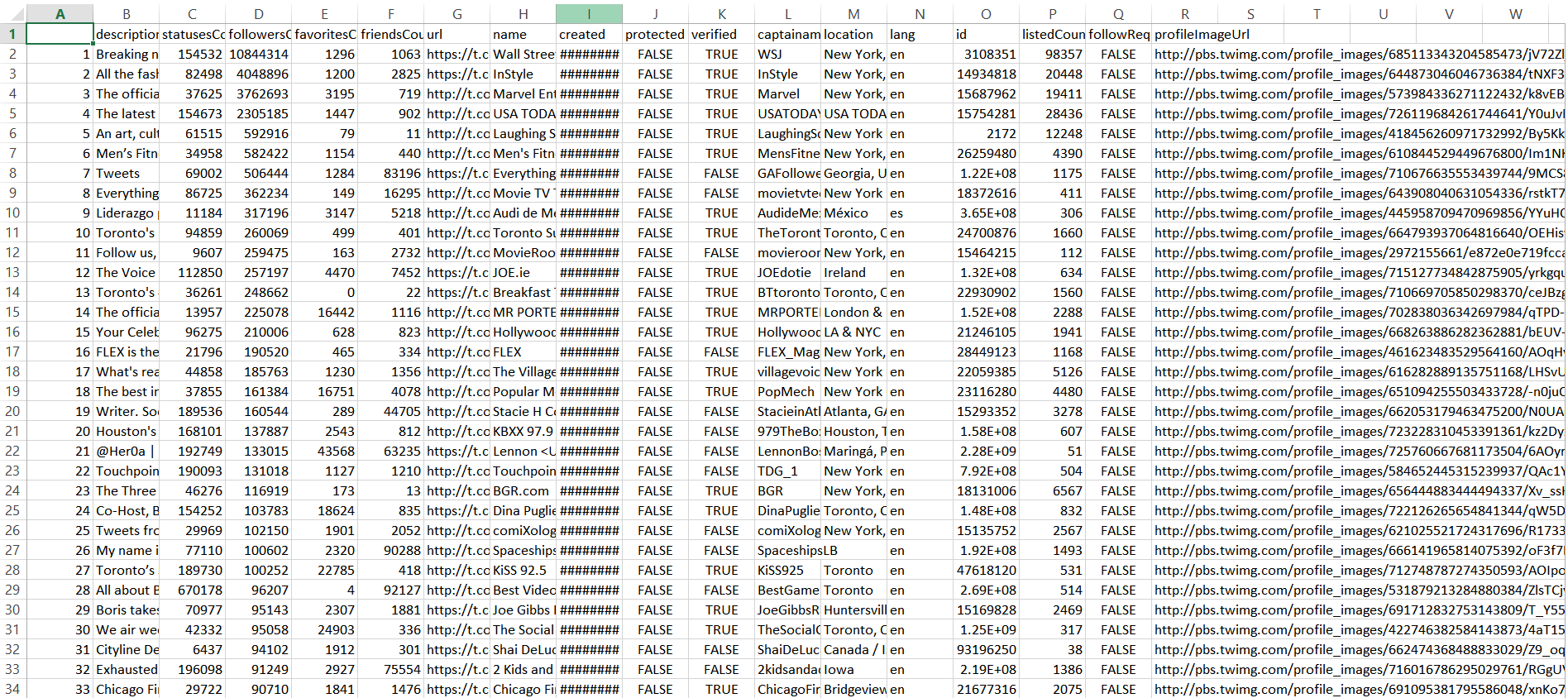
3) ***‘ROAuth’*** to create a handshake between twitter and R.

We need a standard Twitter account and then update it to a developer account to be able to use twitter API. Below are the twitter authorization account that we used to further extract the tweets from twitter.



We scrape tweets from twitter in real time which contains the movie name. It can be changed to any movie name that the user selects from the UI. We also restrict it by giving the geolocation based on the location that the user is looking for. As of now, we have selected MA location and have given the geo-coordinates of Northeastern University and taken a radius of 100 miles from there. Also, we can restrict the number of tweets by specifying ‘n’. In this case we have taken 5000 top tweets. And we restrict the language to English.





**Sample data that we got from twitter after using the tokens. The data set includes:**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Column Name** | **Description** |
|  | description | User Account Description |
|  | statusesCount | Number of tweets by the user |
|  | followersCount | Number of followers for the particular user |
|  | favoritesCount | Count of friends for that particular user |
|  | friendsCount | Number of friends of that particular user |
|  | url | URL for the users account |
|  | Name | Username |
|  | created protected | When account is created and is it protected or not. |
|  | Verified | Account is verified or not |
|  | Captainamericalistnames | Words related to particular movie |
|  | Location | User location |
|  | Lang | Language |
|  | Id | UserID |
|  | listedCount | Count of the tweets |
|  | followRequestSent | Request to follow sent by the user |
|  | profileImageUrl | User image profile url |

**Data Cleaning**

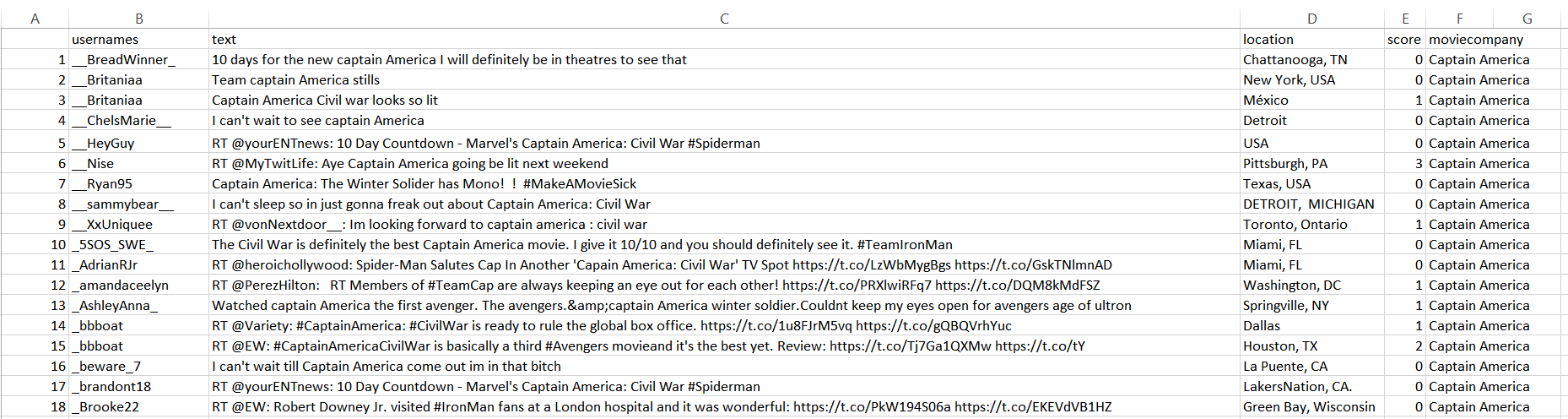
Data cleansing, is the process of amending or removing data in a database that is incorrect, incomplete, improperly formatted, or duplicated. After getting the tweets, we clean the data in various ways:

1. Extract text part from the tweet

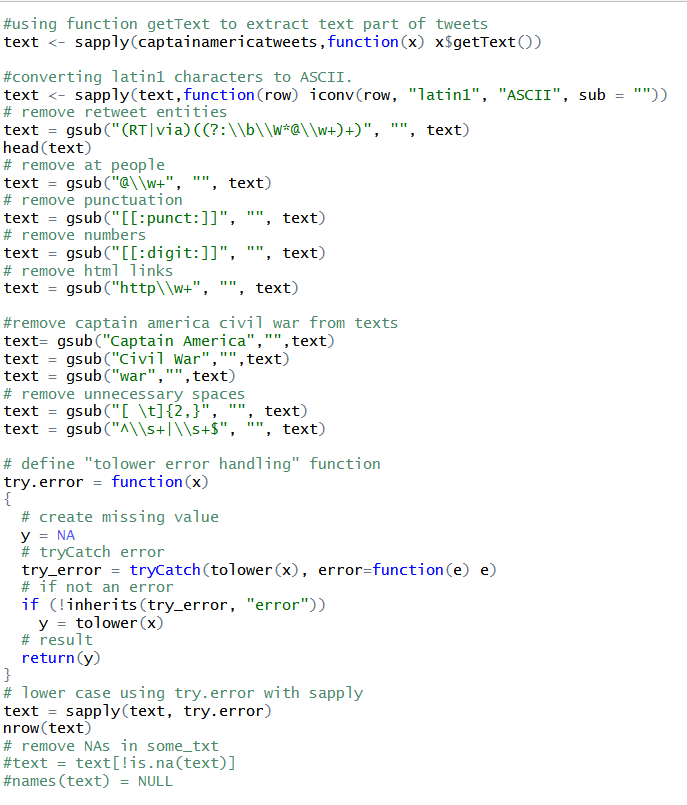
2. Converting latin characters to ASCII

3. Remove RT word from the tweet

4. Remove @, Remove Numbers, Remove Punctuations, Remove html links  
5. Remove Captain America, Civil Wars, Wars   
5. Remove emoticons – emoticons are present in latin format and thus are of no use for analysis.  
6. Tolower – we convert all the alphabets to lowercase for consistency.  
7. Remove text word, Remove NA’s, Remove Missing value



Sample data set after cleaning the data for the movie name “Captain America” with MA location and their respective sentiment scores.



**Sentimental Analysis using R:**

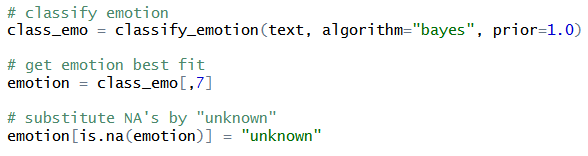
Before using sentiment analysis, we should know what exactly sentiment analysis is. It is using NLP, statistics, or machine learning methods to extract, identify, or otherwise characterize the sentiment content of a text unit.

Sentiment analysis is used to see if a text is neutral, positive or negative. Emotion analysis is used to see which emotion a text has (happy, fear, anger). Both are using similar codes but the comparison lexicon is different. Since Twitter restricts each tweet to be less than 140 characters, users’ comments tend to be straightforward. In addition, because of its huge influences, many movies have started to include a hashtag in their trailers that tell people to tweet about them to attract social attention. Therefore, Twitter has become a great platform to examine people’s feedback.

Thus twitter is full of sentiments.

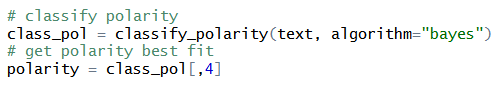
**Classifying emotion from the tweets:**

To classify the tweets based on emotions we use classify\_emotion function. It helps classify the emotion (e.g. anger, disgust, fear, joy, sadness, surprise) of a set of texts using a naive Bayes classifier trained on Carlo Strapparava and Alessandro Valitutti's emotions lexicon.



**Classifying polarity from the tweets:**

To classify the tweets based on polarity we use classify\_polarity function. Classifies the polarity (e.g. positive or negative) of a set of texts using a naive Bayes classifier trained on Janyce Wiebe's subjectivity lexicon.

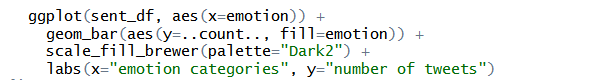


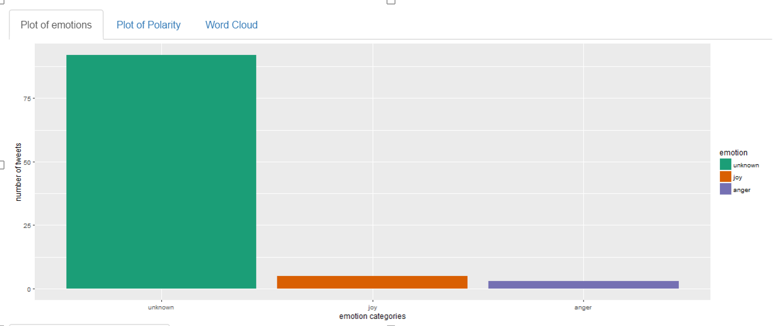
**Creating a data frame:**

We create a data frame that contains the tweet, emotions and polarity and sort it. The resulting data frame would look like this:

**Creating emotion plot:**

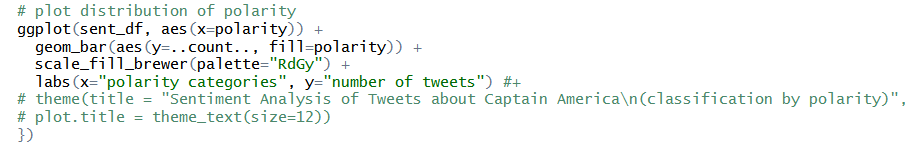
Obtain emotion plot of the extracted emotions with the help of following code:

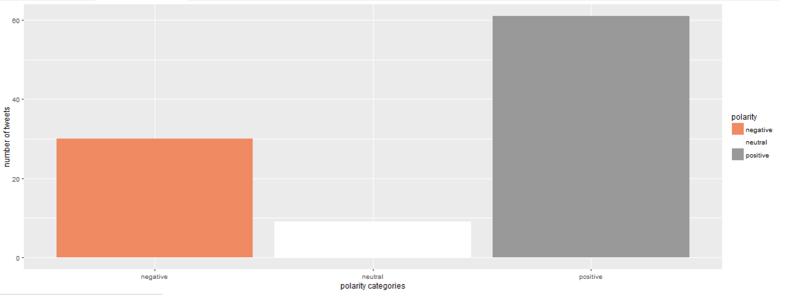




**Creating polarity plot:**

We follow similar procedure to obtain polarity plot:



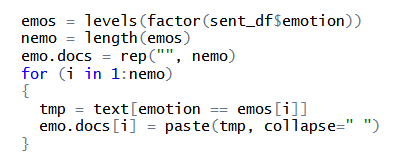


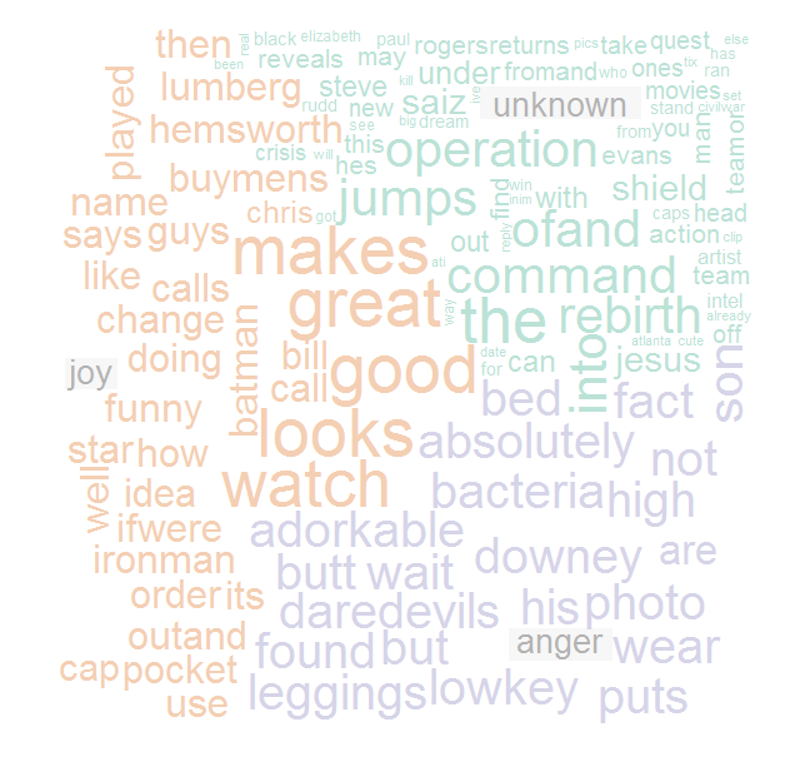
**Creating emotions word cloud:**

We follow similar procedure to clean the data. We remove numbers, punctuations, and some special characters and extract plane text. The code to perform this data cleansing is already mentioned above.

We follow similar procedure to generate the data frame

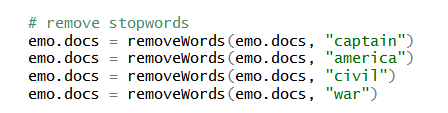
To generate word cloud, we first extract emotion variable from the data frame and store its length in a new variable. Then we paste the factored value of each emotion in emo.docs.





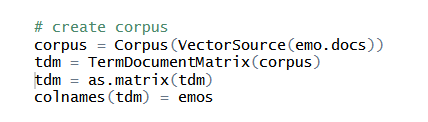
**Removing Stop words:**

There are certain words like captain, America, civil and war which do not contribute to the word cloud generation. We remove these words so that they won’t occupy most words as they occur most number of times.



**Generating corpus and Term Document Matrix:**

Corpus specifies a collection of text documents. We generate Term Document Matrix as it facilitates multiple R functions to be used on it. The generated corpus is converted into Term Document Matrix using TermDocumentMatrix function.



**Generating histogram to show sentiment analysis based on scores for movie entered by user:**

In the screenshot shown below, searchTwitter function takes Captain America as input to extract 5000 tweets for sentiment analysis.

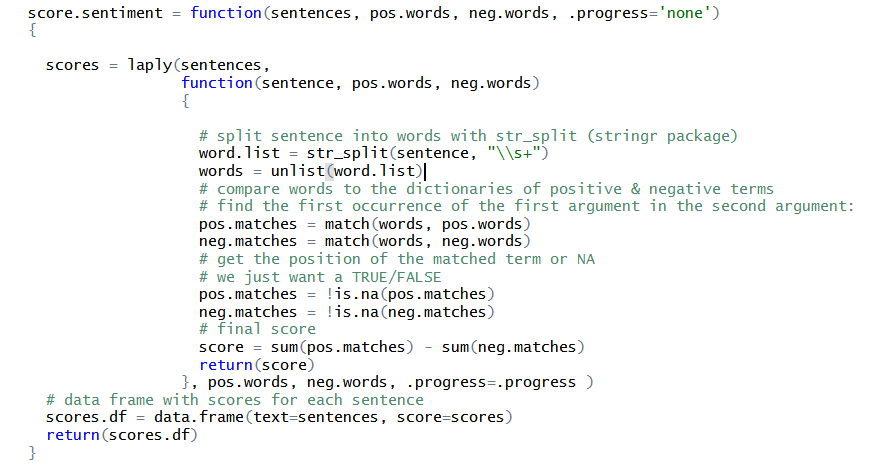


We have kept list of positive words and negative words in separate files. The score.sentiment function does the task of computing the sentiment score for each tweet.

***Sentiment Lexicon*** is a list of words which we can use to compare any scraped text. Hu Liu Lexicon made a standard of sentiment analysis by manually creating a list of positive and negative words. Combined it consists of approximately 6800 words. A lexicon does not provide any mechanism for storing definitions; the lexicon contains only words, with no associated information. It is therefore similar to a set of strings, but with a different internal representation. The Lexicon class supports efficient lookup operations for words and prefixes.

We have downloaded the lexicons and imported into our R environment.





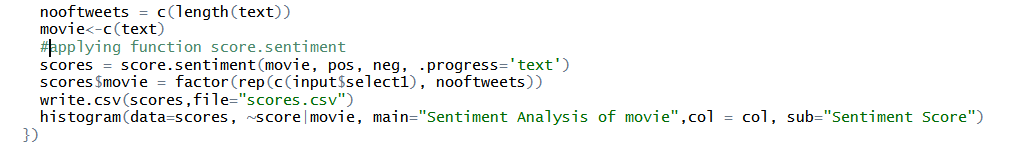
The laply function applies the function to each of the tweet listed in the list. Str.list splits each sentence into words whenever is sees a white space. The unlist function breaks down the words into separate words. The pos.matches is an array, where it stores the index of each positive word along with a ‘na’.

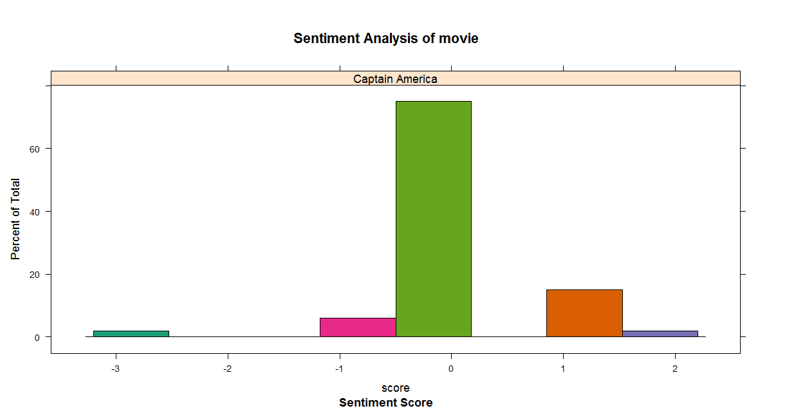
The is.na (pos.matches) will return true whenever it finds a match for positive word, otherwise it returns False. True is considered as +1 while False is considered as -1.

The sum (pos.matches) returns the sum of all the values corresponding to true. The sum (neg.matches) returns sum of all values corresponding to false.

Score is the difference between sum (pos.matches) and sum (neg.matches) which results in a score being generated.

The below code shows the percentage progress of the function score.sentiment. The histogram function plots the histogram for the scores calculated by the function score.sentiment.

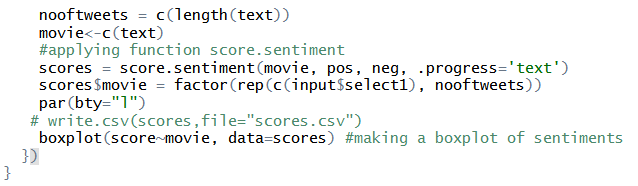




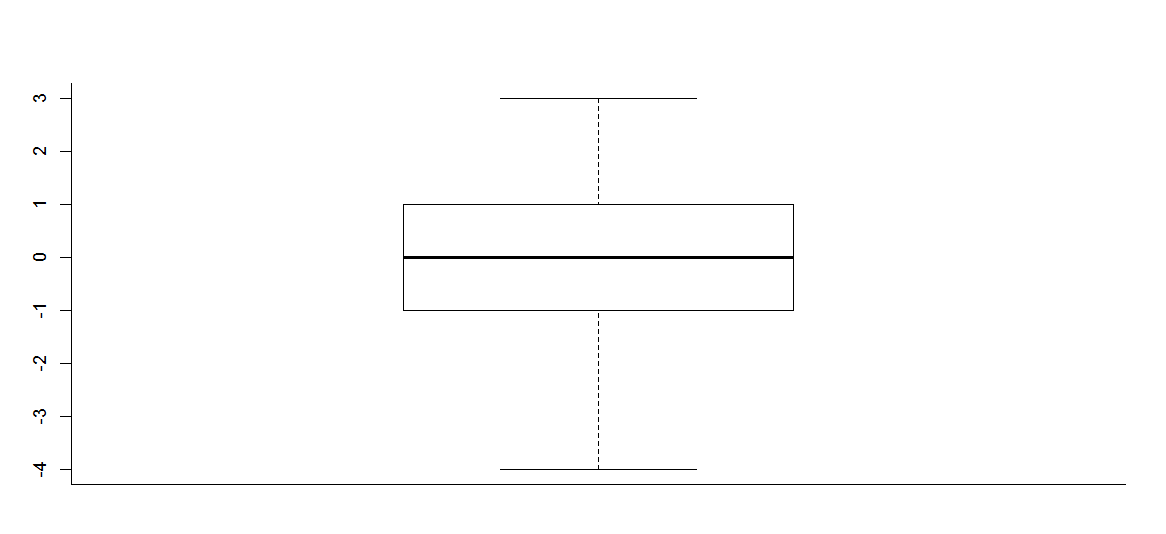
**Generating boxplot to show sentiment analysis:**

We create a boxplot to show overall distribution of the score based on sentiment analysis performed by function score.sentiment.

Box Plot function:



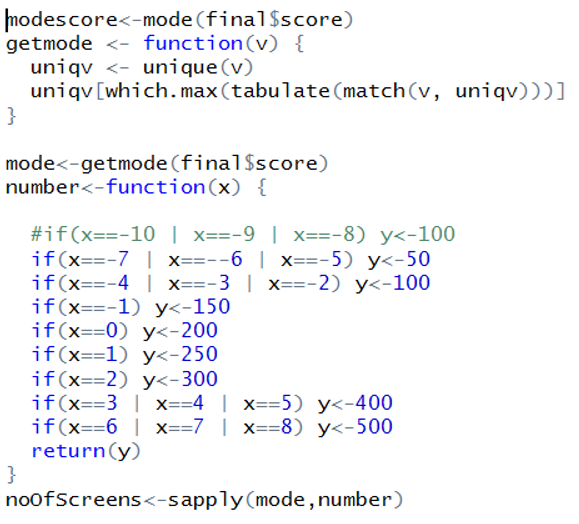
The boxplot function returns box plot of the scores generated by the function. The box plot shows that the overall score ranges between -1 and +1, median being at 0. This shows that most sentiments associated with the movie are neutral. The distribution of the plot shows the presence of some very good and very bad sentiments associated with it too.



**Calculating mode to predict the number of movie screens:**

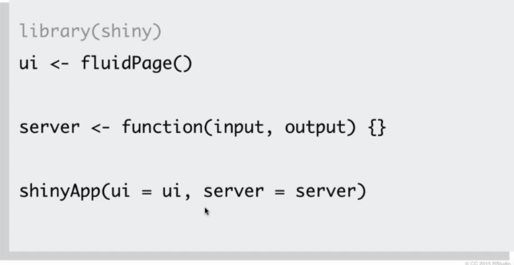
We calculated the mode by using the scores of all the tweets for a particular movie. And then used that mode to predict the number of screens to screen for that movie in that particular location.

For example, say the mode of scores is +2, for which the output will be predicted as 300. Which implies that the user can screen the movie at 300 individual theatres.



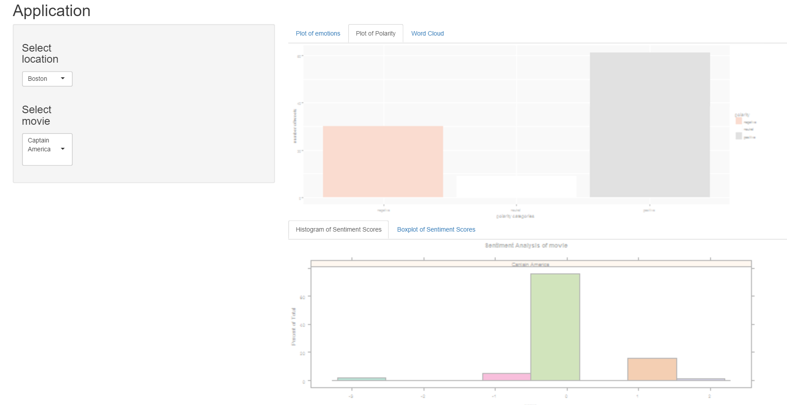
**Using shiny server to integrate front end with the analysis performed at back end:**

Generating a simple web app in shiny:



Three components mentioned above in screenshot are required to deploy an application in shiny. The ui component can have any number of ui components like textboxes, drop down menus, radio buttons etc. The server function takes input and output as arguments and call to the shinyApp function.

In our application, we have majorly used reactivity functionality of shiny to dynamically find output based on user input. Incorporating the reactivity, we implement 2 use cases.



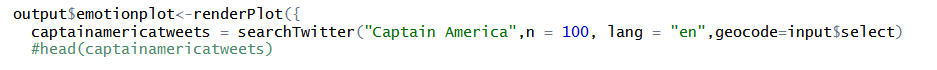
**1) User inputs city of his choice to view public opinion in that city:**

We consider 3 cities in the United States namely Boston, New York and Los Angeles but the functionality can be applied to any number of cities.



We give a drop down menu in the front end where we hardcode the geolocation coordinates for the 3 cities. The selected input from the down list get stored in a reactive element **input$select** where select refers to the id of selectInput function.

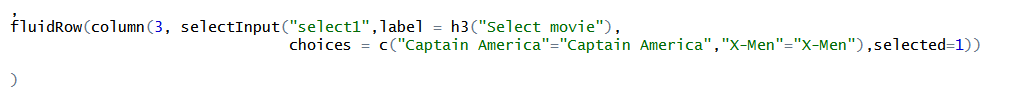
This **input$select** can bedynamically used in any function so as to render results corresponding to it.



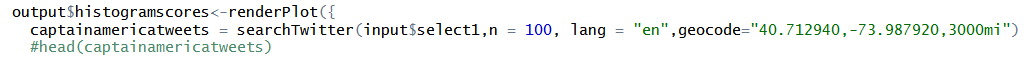
The above function extracts tweets dynamically for the movie Captain America for the city entered by user and print different user friendly plots so that user can get fair idea about the ideal location for the movie release.

**2) User inputs movie of his choice to view public opinion about that movie:**

We consider two movies namely Captain America and X-Men but the functionality can be applied to perform sentiment analysis for any movie to be released.



We give a drop down menu in the front end where we hardcode the movie names for 2 movies. The selected input from the down list get stored in a reactive element **input$select1** where select1 refers to the id of selectInput function.



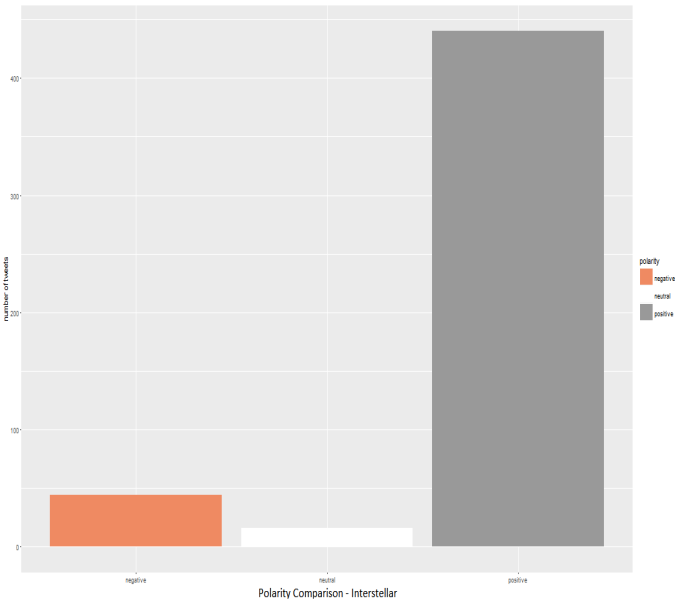
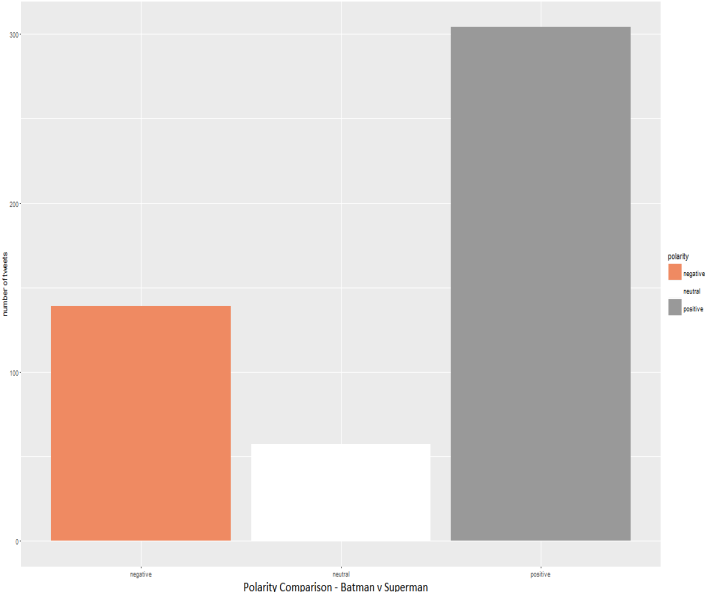
Here we use searchTwitter function to dynamically extract tweets for movie selected by user in the radius of 3000 miles from Northeastern University and print various user friendly plots.

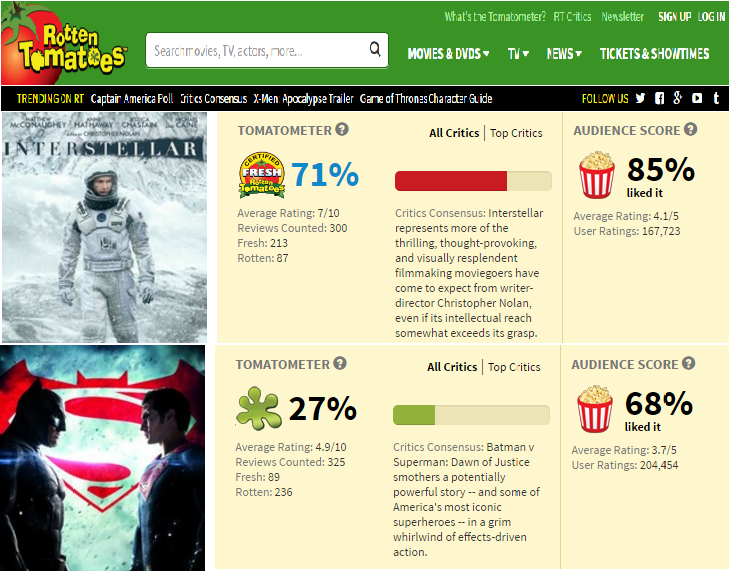
**Power of Sentiments**

We performed sentimental analysis on two already released movies with recent 1000 tweets and compared their movie ratings on Rotten Tomatoes. Our Analysis is as follows:-

The tallest bar which is seen represents positive polarity and the white one is neutral polarity which should be ignored. As it can be seen in the below graph, Interstellar has most number of tweets on the positive side. Thus we rank Interstellar most positive.

Batman v Superman has lowest positive tweets thus we rank it low.

****

The comparison on rotten tomato for Interstellar and Batman v Superman shows higher rating for Interstellar and lower rating in comparison to Batman v Superman.

**Scope of Improvement**

1. Correct prediction of the oxymoron statements, as lexicon library fails to do that and ignores that part.

2. Deciding the number of screens on a more firmed basis.

3. Currently, we are not using interests and activities from a user’s Twitter profile as it is very hard to standardize. In the future, we can semantically cluster them to use as features in our algorithms.

**Conclusions**

From our results, we can make several conclusions:

1. Learned how to integrate R and Shiny for text mining, sentiment analysis, visualization and deployment on the web. Using these tools together enables us to answer detailed questions.

1. It can be easily visualized that based on the sentiment analysis performed on the movie trailer the distributor can get a fair idea about the ideal location and number of screens to distribute the movie.

2. The opinion mining can be performed for any movie at any location, even for entire globe. By inputting any movie, a distributor can view the public opinion for a movie of his/her counterpart too. Based on that, he can make strategic decisions and do the needful to improve public feedback for his own movie.

**References**

1. Shiny.rstudio.com
2. <https://gist.github.com/wch/5436415/>
3. <https://gist.github.com/wch/5436415/>
4. <https://cran.r-project.org/web/packages/shiny/shiny.pdf>
5. <http://rstudio.github.io/shiny/tutorial/#ui-and-server>
6. <http://datascienceplus.com/sentiment-analysis-with-machine-learning-in-r/>