## IMPLEMENTATION

The Dataset contains tweets retrieved based on the topic the user gives. The dataset is dynamic and gets updated each time when a new input is given.

### Analytical methods:

The methodology used is as follows:

1. Retrieve the data from twitter regarding a particular topic
2. Perform the pre-processing steps on the text data, which involves removing of URLs, typecasting, removing stop words etc.
3. Create a term-document matrix to visualize the counts of words.
4. Create a word cloud of the words to represent the dominance of words
5. Use different libraries in R to perform sentiment analysis on the tweets
6. Rate the tweets on 10 emotional polarities and represent in the form of a Histogram.

### Execution of the code:

We have Given Long Topic Comments in the code indicating which part has what functionality!

Let us briefly explain each segment:

1. **Libraries Required**

   These libraries contain all the functions I used in the code, we need to initialize them in the code in order to use them.

2. **Setup connection with Twitter**

   To connect with Twitter API, we need an account in apps.twitter.com where we will get Tokens and secret keys. We need to initialize them in the code and use them to connect to API using R. The library required is (twitter)

3. **Front end code** We don't need to write any html, css code to develop UI in Rshinyapps, there are inbuilt functions to take care of that. If we run that part of the code in Terminal we can see the html output.

4. **Backend code**

   Here is where the magic takes place, here contains the retrieving Tweets, preprocessing them and showing the outputs. I added some reactive variables to show only certain output according to the user's wish. You can see that I used "if" conditions to implement that. Detailed explanation for each segment is given below.

5. **Clean Text**

   Here many preprocessing steps like Cast typing, removing punctuation, White spaces, removing url's etc are carried out. To make sure the data we are using for the analysis is useful and not of unnecessary text.

6. **Term Document matrix**

   This basically is matrix consisting of the tweets on the columns and terms on the rows. It contains of "0" and "1", 1 means that that term is in that tweet "0" means that the word is not in the tweet.

7. **Network Diagram, Histogram**

   This is formed by the Term-Document-Matrix we created in the previous step. The network diagram consists of nodes(terms) which contains connections based on the

Presence of them in the same tweet. The histogram is then plotted from this network diagram; it shows how many Nodes are present with what degree.

8. **Community Detection**

   I have implemented 3 different algorithms for finding communities between the terms namely edge-betweeness, label-propagation, Fast-greedy.
   I have included the labels of the terms here so that the user can clearly see what terms are grouped together.

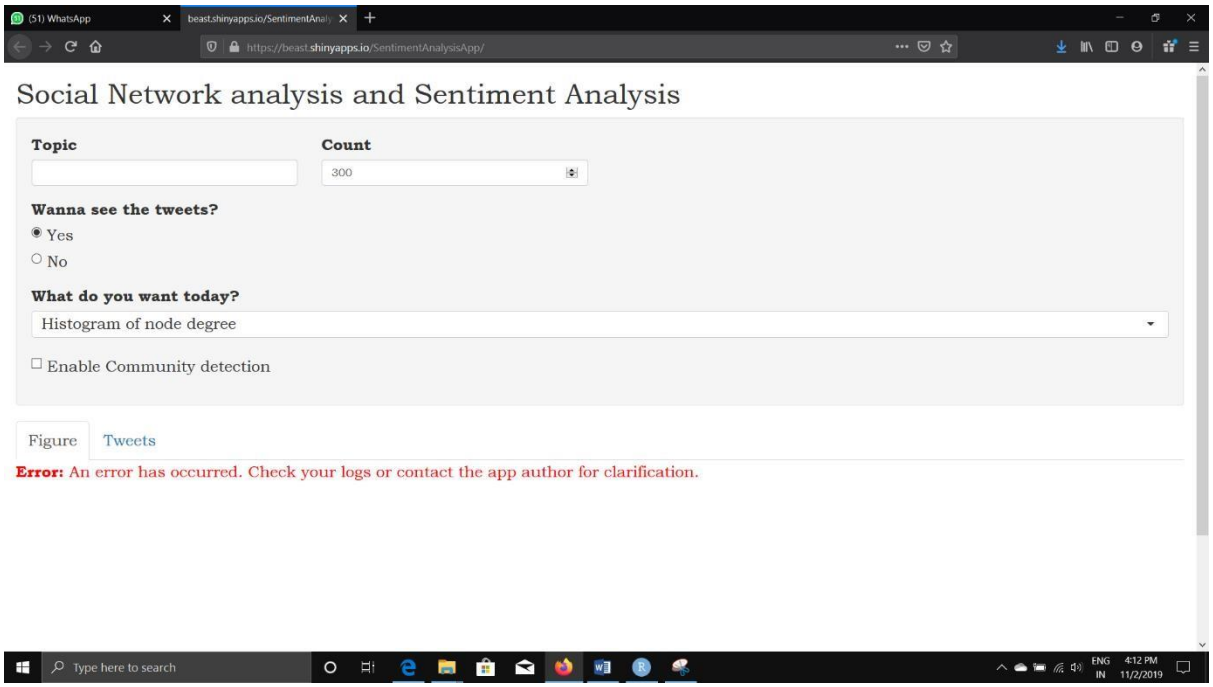
9. **Word-cloud**

   This is classic inference technique where we can see what terms are more dominant in the whole corpus (tweets). The size of the term in the word-cloud is based on the Frequency in the corpus

10. **Sentiment Analysis**

    This uses the Lexicon method. For every word in the English language, the library has categorized into 10 categories (1 or 0), now when we give the tweets, it will count the number of 1's in every category and then show a histogram of the output, we can get a basic inference by taking a look at the histogram. The parameters are "anger", "anticipation", "disgust", "fear", "joy", "sadness", "surprise", "trust", "negative", "positive".

**Explanation of vis idiom:**

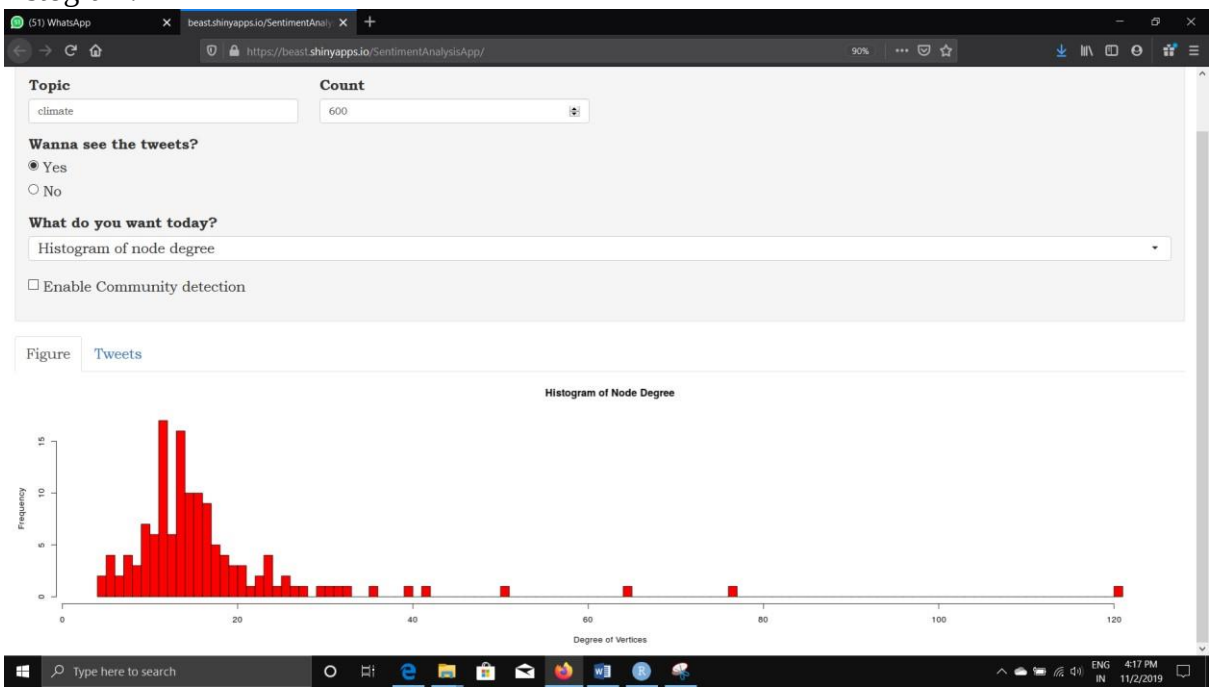



This is the Website which we made. Here we can:

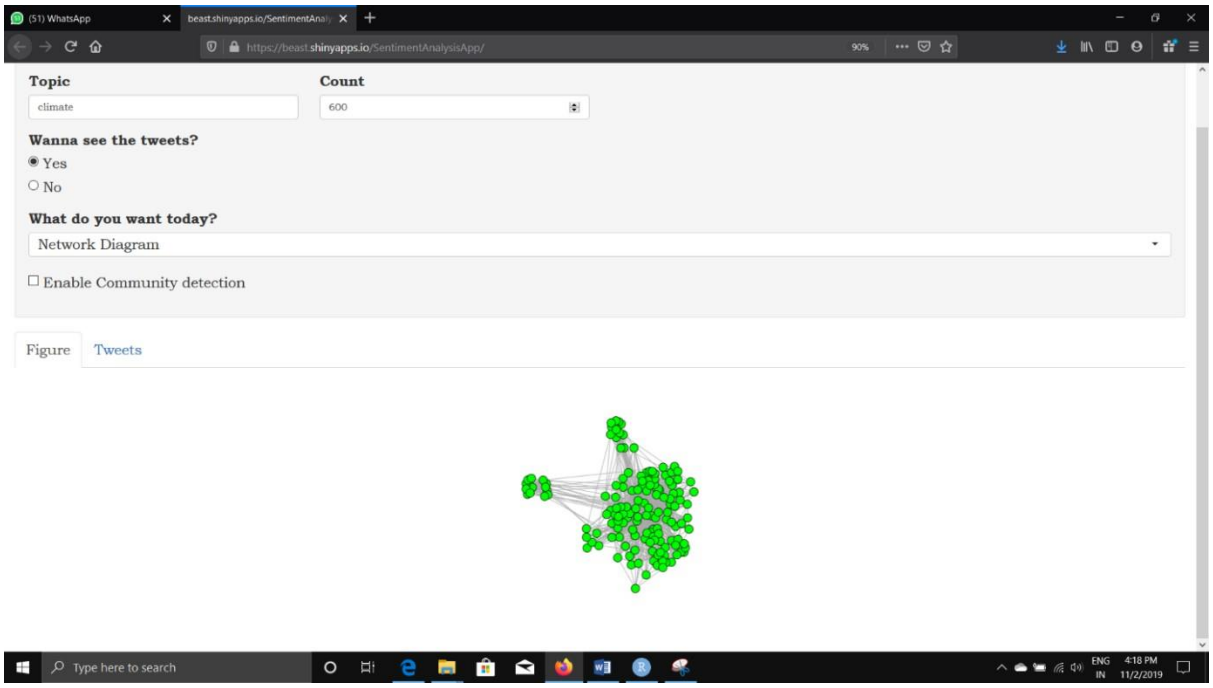
1. Enter a topic on which we want to retrieve the tweets from
2. Enter the number of tweets to extract
3. Select the Visualization technique

## RESULT

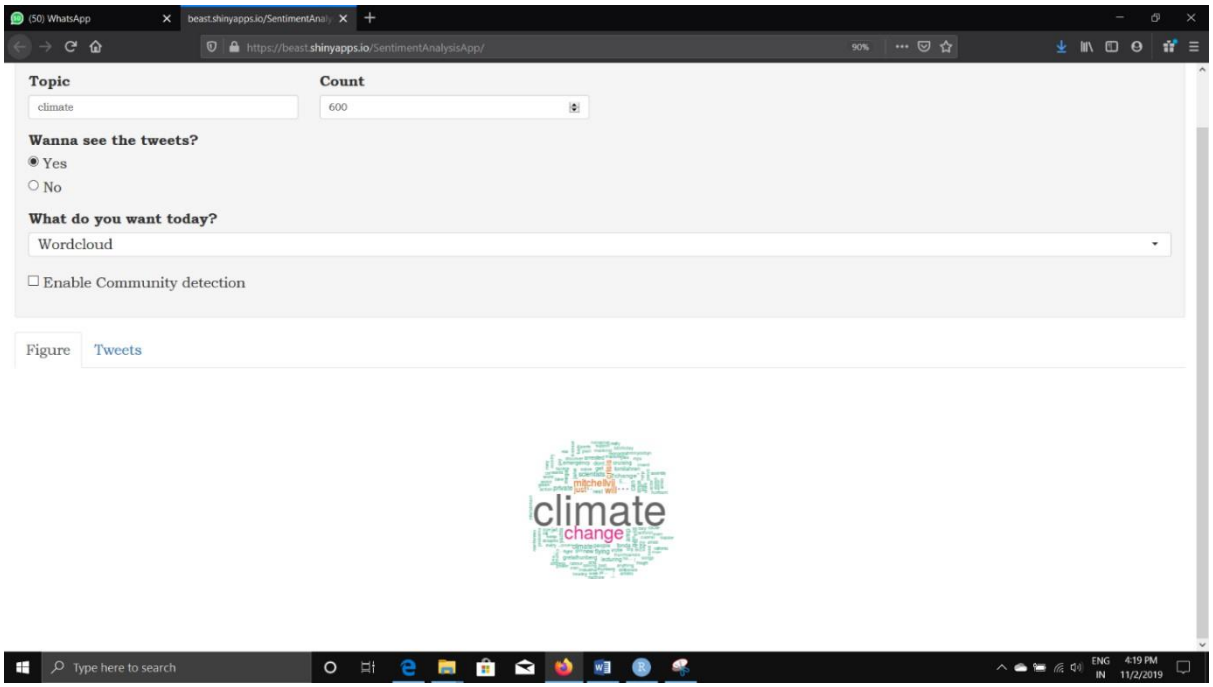
Histogram:



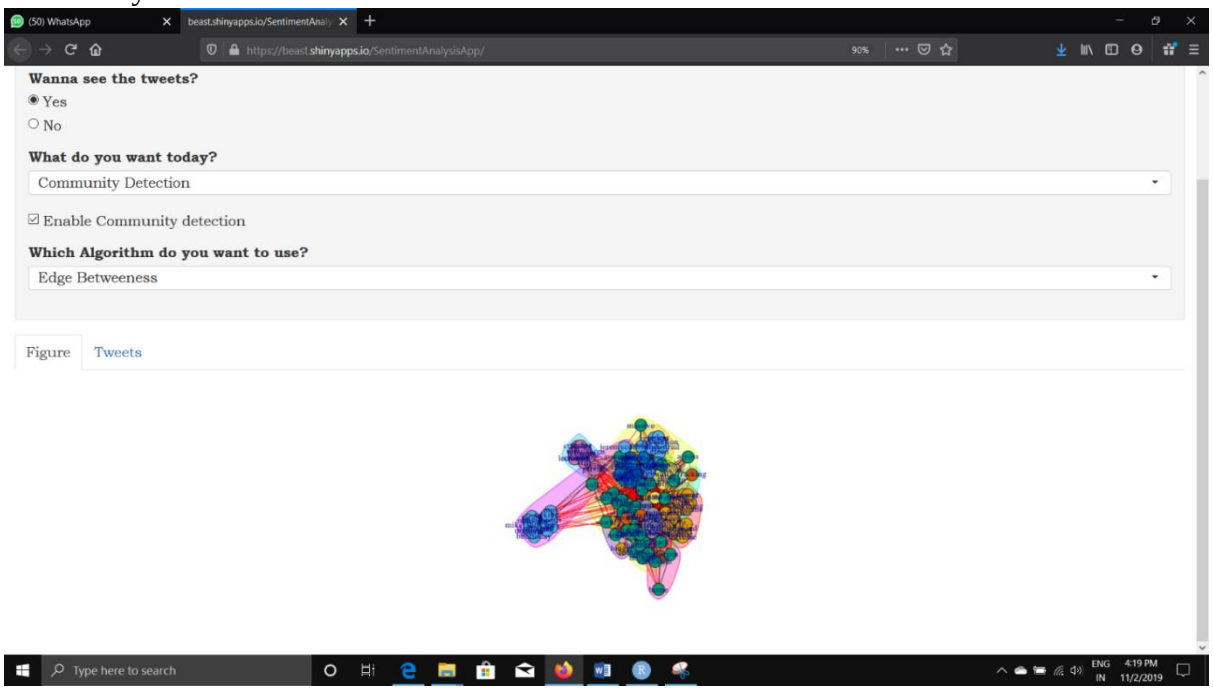

Network:



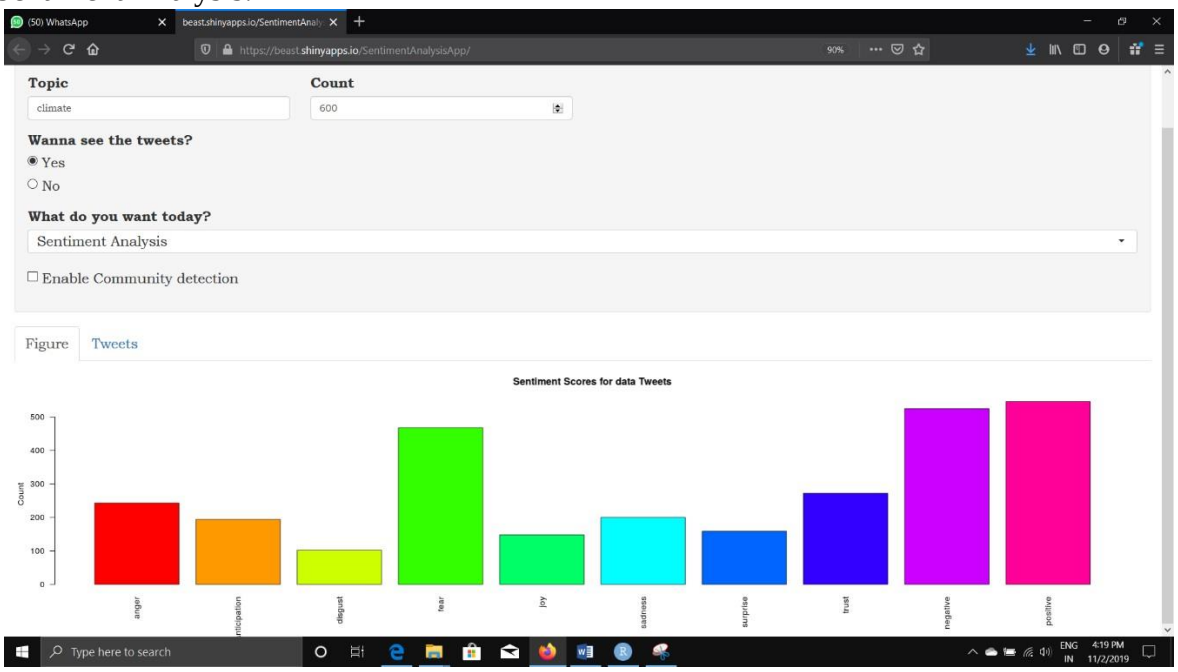
Word cloud:




Community detection:



Sentiment Analysis:




Till now we have seen many different types of visualizations and plots from which we can infer some clues. For example, in the Plot of "Histogram of node degree" we can see that the better the corpus (many number of tweets) the more left shifted the bar plot. So, we can say that the "more left shifted" the histogram looks the more "rich data" we have.

As you can clearly see in the Network diagram in have disabled the node labels as it would be very clumsy to view and infer anything but it is a very important component in the Community detection phase where we use 3 different algorithms to find communities in the network diagram. The 3 Algorithms give different results but we can see almost always the similar set of terms are grouped together. Also the it can also be seen from the "word cloud" that it is quite a fancy way of representing data, but there is more to it, the size of the term in the word cloud represents the dominance or the frequency of that particular term which can be seen from the term-document matrix (used to create the Network Diagram).

The sentiment analysis works in a pretty straight forward manner, the function rates every word in the English language on 10 parameters which you can see the plot. For example the word "happy" may have the 0 in the emotion "anger' and 1 in the emotion "excited", like this they are assigned 0's or 1 in 10 parameters. The plot basically shows how many terms are there in the corpus which portray the respective emotion.

While the website execution time depends on the user's internet bandwidth, the processing time is quite slow as the backend runs on the RStudio Shiny apps server.

The final of this project is a live and fully functioning website which can carry out all the functionalities mentioned in the Reviews of this project. The Website can be accessed from here. https://beast.shinyapps.io/SentimentAnalysisApp/