**SENTIMENT ANALYSIS**

**Introduction:**

The application reads the hashtags from a CSV file and then downloads all tweets corresponding to those hashtags. The downloaded data is stored in database and then further analyzed.

**Tools Used:**

R Studio

Web browser

**Language:**

R

**Code:**

**Server.r**

**#importing the needed libraries**

library(shiny)

library(twitteR)

library(ROAuth)

library(rmongodb)

library(mongolite)

library(base64enc)

library(plyr)

require(RCurl)

library(stringr)

library(ggplot2)

library(rjson)

library(bit64)

library(httr)

**#settting up mongodb twitter credentials**

mongo <- mongo(collection = "filterstream", db = "twitter", url = "mongodb://localhost")

**#loading up positive and negative words**

pos <- scan('D:/positive.txt', what='character', comment.char=';')

neg <- scan('D:/negative.txt', what='character', comment.char=';')

**#adding more words to the positive and negative words list**

pos.words <- c(pos, 'upgrade')

neg.words <- c(neg, 'wtf', 'wait', 'waiting', 'epicfail')

**#calling the sentiment function**

score.sentiment <- function(sentences, pos.words, neg.words, .progress='none')

{

**#processing each tweets from collection of sentences**

scores <- laply(sentences, function(sentence, pos.words, neg.words){

**#cleaning the tweets by removing special characters**

sentence<- gsub("[\\s\\S]\*?\n", "", sentence)

sentence<- gsub("#\\w+ \*", "", sentence)

sentence<- gsub("@\\w+ \*", "", sentence)

sentence<- gsub(" ?(f|ht)(tp)(s?)(://)(.\*)[.|/](.\*)", "",sentence )

sentence <- gsub('[[:punct:]]', "", sentence)

sentence <- gsub('[[:cntrl:]]', "", sentence)

sentence <- gsub('\\d+', "", sentence)

sentence <- tolower(sentence)

**#Extracting words from each tweets and then comparing them with positive and negative list of words**

word.list <- str\_split(sentence, '\\s+')

words <- unlist(word.list)

pos.matches <- match(words, pos.words)

neg.matches <- match(words, neg.words)

pos.matches <- !is.na(pos.matches)

neg.matches <- !is.na(neg.matches)

score <- sum(pos.matches) - sum(neg.matches)

return(score)

},pos.words, neg.words, .progress=.progress)

scores.df <- data.frame(score=scores, text=sentences)

return(scores.df)

}

**#function to download tweets from twitter**

download <- function(hashtagss,num)

{

laply(hashtagss, function(hashtags,num){

tags <- hashtags

list <- searchTwitter(tags, num)

dataframe = twListToDF(list)

**#assigning an identification variable to each tweet**

tags<- gsub("#", "", tags)

tagname <- c(tags)

dataframe$tagname <- tagname

**#storing the downloaded data from twitter to MongoDB**

b=mongo.bson.from.df(dataframe)

mongo <- mongo.create()

icoll <- paste("twitter", "filterstream", sep=".")

mongo.insert.batch(mongo, icoll, b)

},num)

}

**#function that is executed on GUI activity**

shinyServer(function(input, output)

{

observeEvent(input$goButton,

{

**#setting up progress-bar**

progress <- Progress$new(min=1, max=4)

progress$set(value = 1)

progress$set(message = 'Downloading in progress', detail = 'This may take a while...')

**#reading tags from CSV file**

inFile <- input$file1

readtags <- read.csv(inFile$datapath)

readtags$c1 <- as.factor(readtags$tags)

readtags$c1 <- lapply(readtags$c1, as.character)

**#updating progress bar**

progress$set(value = 2)

download(readtags$c1,input$freq)

progress$set(value = 3)

progress$set(value = 4)

progress$set(message = 'Download completed successfully',detail = '')

})

**#the below code gets executed when Analyze button is clicked by User**

observeEvent(input$Calc,

{

**#Identifying tweets available in MongoDB**

agg\_df <- mongo$aggregate('[{ "$group" :

{ "\_id" : "$tagname",

"number\_records" : { "$sum" : 1}

}

}]')

row.names(agg\_df) <- NULL

rows <- nrow(agg\_df)

**#If database does not exist show appropriate message**

if((rows<1))

{

output$helptext <- renderUI({

helpText('Dataset is Empty')})

return(NULL)

}

**#Displaying available data in MongoDB in table to User**

output$mytable = renderDataTable({agg\_df})

**#Insert the right number of plot output objects into the web page**

output$plots <- renderUI({

plot\_output\_list <- lapply(1:rows, function(i)

{

plotname <- paste("plot", i, sep="")

tabPanel(agg\_df[i,1],plotOutput(plotname, height = 600, width = 600))

})

do.call(tabsetPanel, plot\_output\_list)

})

**# Call renderPlot for each one. Plots are only actually generated when they are visible on the web page.**

for (i in 1:rows)

{

print(agg\_df[i,1])

qry <- paste0('{ "tagname" : "',agg\_df[i,1] , '"}')

Datasettw <- mongo$find(qry)

Datasettw$text <- sapply(Datasettw$text,function(row) iconv(row, "latin1", "ASCII", sub=""))

Datasettw$text <- as.factor(Datasettw$text)

tw.scores <- score.sentiment(Datasettw$text, pos.words, neg.words, .progress="text")

**# Need local so that each item gets its own number. Without it, the value of i in the renderPlot() will be the same across all instances, because of when the expression is evaluated.**

local({

my\_i <- i

ks <- tw.scores$score

print(my\_i)

plotname <- paste("plot", my\_i, sep="")

output[[plotname]] <- renderPlot({

(wwt <- hist(ks,plot = TRUE))

plot(wwt, border = "dark blue", col = "light blue",

main = agg\_df[my\_i,1], xlab = "Tweets Polarity")

})

})

}

})

})

**Ui.R**

**#importing the GUI library package**

library(shiny)

shinyUI(pageWithSidebar(

**#setting up header text**

headerPanel("Sentiment Analysis"),

**#setting up the slider**

sidebarPanel(sliderInput("freq", "How many tweets?",min = 10, max = 1000, step = 10, value = 10),

**#setting up file input dialog boxes**

fileInput("file1", "Choose CSV File",

accept = c("text/csv","text/comma-separated-values,text/plain",".csv")),

**#Setting up button and other GUI components**

uiOutput('helptext'),

actionButton("goButton", "Download Tweets"),

actionButton("Calc", "Analyze"),

br()),

mainPanel(dataTableOutput('mytable'),br(),br(),

uiOutput("plots"))

))

**#loading server file contents**

source('server.r')

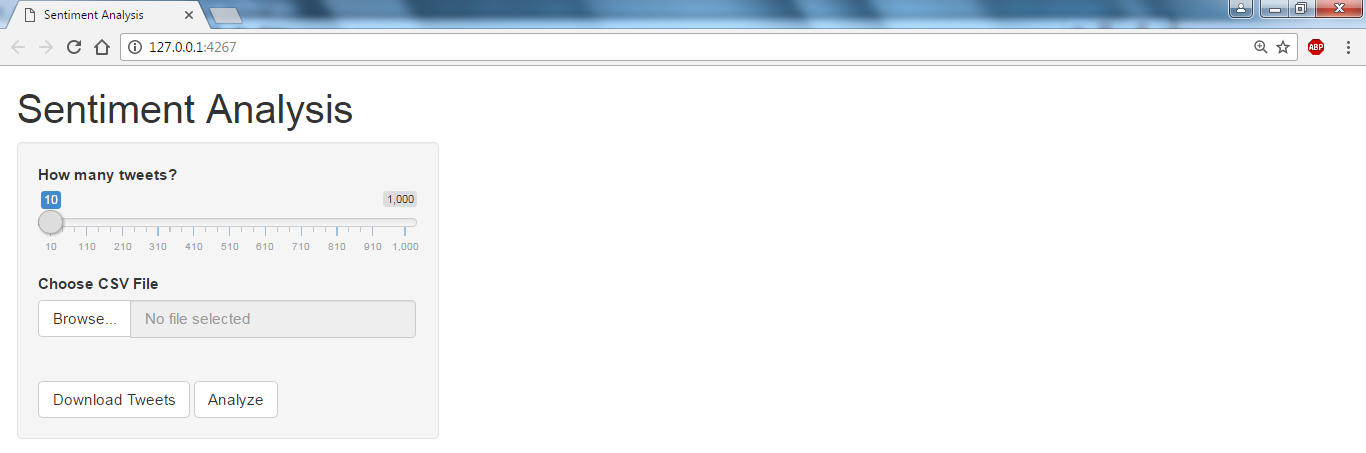
**SCREENSHOTS:**

**User interface of the Application:**

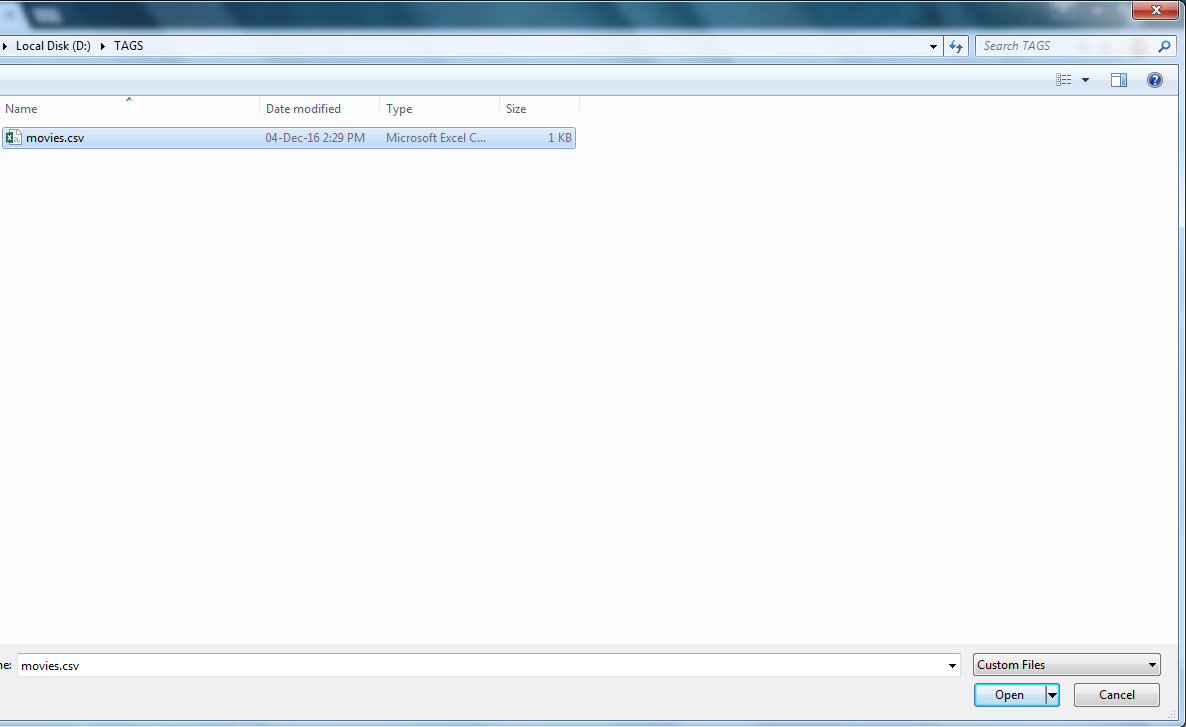
1. The tweets slider allows the user to select the frequency of twitter tags that are to be downloaded. Minimum frequency is 10 and maximum is 1,000.

The twitter API has limitations on the frequency of tweets that can be downloaded and is set max to around 1500. The tweets slider is set to maximum frequency of 1000 to avoid getting blocked by twitter as there is limitations on the tweets that can be downloaded

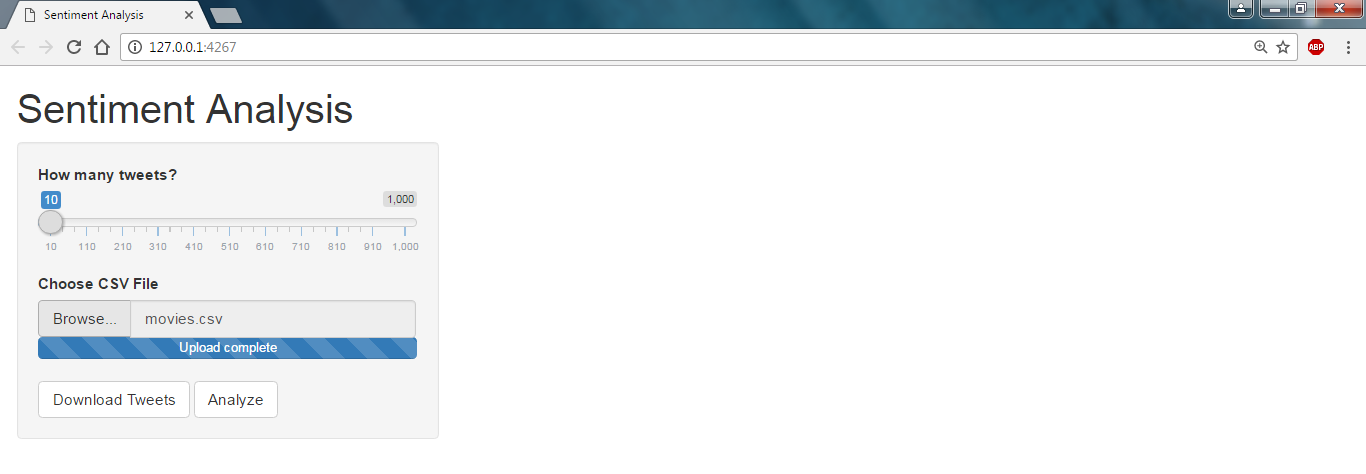
1. The File Input dialog(Choose CSV file) provides an file dialog box where we can pass the CSV file as input.
2. The CSV file would contain all the tags that are to be downloaded. The downloaded data is stored in MongoDB for further analysis.
3. On clicking “Download Tweets” button all the tweets corresponding to tags specified in CSV file are downloaded.



**File Dialog Box:**



**Once the CSV file is selected its tags are read and the status of progress bar changes to Upload complete.**



**Now, We can analyze the data that exists in MongoDB database.**

To do so **click on Analyze button**, the data present in MongoDB is displayed in table format and its sentiment analysis is displayed in tabs below in the form of histogram.

In the below output there are four hash-tags present in MongoDB database

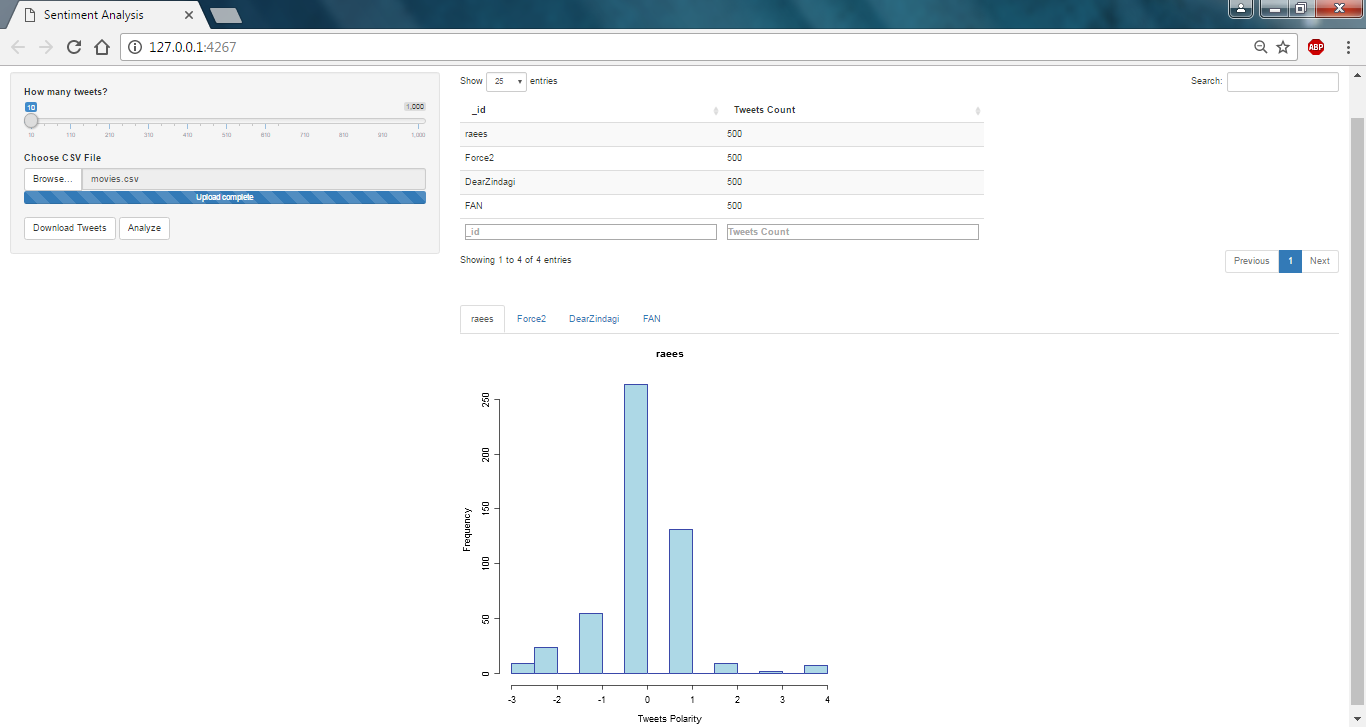
1.raees

2.force2

3.dearZindagi

4.FAN

The no. of tweets present for each hash-tag is also displayed. All tags here have 500 tweets. i.e 500 rows for each hash-tag



**Histogram Explanation:**

1. The tweets that are negative are plotted on X-axis **below zero** with their frequency been represented on by Y-axis.
2. The tweets that are positive are plotted on X-axis **above zero** with their frequency been represented on by Y-axis.
3. The tweets that are neutral(neither positive nor negative) are plotted on X-axis **on zero** with their frequency been represented on by Y-axis.
4. The score of positive and negative tweets is represented by X-axis
5. The co-ordinate (-1) is less negative as compared to (-3)
6. The co-ordinate (3) is more positive as compared to (1)

In the above graph we can say that the “raees” movie has more positive tweets as compared to negative tweets.