TWEET BASED INTELLIGENT MOVIE MARKETING

BY

SHREYA JAIN

SHRUTI MANDKE

MUSHTAQ RIZVI

INDEX

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.

```
key <- "brFgnrk6dFewBOWBiD2m0tANA"
secret <- "kvByOQydL4AIFnMnmAVVYMF4klgSNJwtkNbA5qmTCAisqv6QAT"
secrettk <- "7XfA0v9j0utKeUuf44n2YEB3AtzqVlMM0ue4IrJC0v2cK"
mytoken <- "708481334482698240-QTn0EaokD6IVWFH0ZUhzlW48rdl42Qt"

setup_twitter_oauth(key,secret,mytoken,secrettk)
```

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.

```
captainamericatweets = searchTwitter("Captain America",n = 5000, lang = "en",geocode='40.712940,-73.987920,100mi')
head(captainamericatweets)
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1		descriptior	statusesCc	followers(	favoritesC	friendsCou	url	name	created	protected	verified	captainam	location	lang	id	listedCoun	followReq	profileImageUrl
2	1	Breaking n	154532	10844314	1296	1063	https://t.c	Wall Stree	#######	FALSE	TRUE	WSJ	New York,	en	3108351	98357	FALSE	http://pbs.twimg.com/profile_images/685113343204585473/jV72Z
3	2	All the fas	82498	4048896	1200	2825	https://t.c	InStyle	#######	FALSE	TRUE	InStyle	New York,	en	14934818	20448	FALSE	http://pbs.twimg.com/profile_images/644873046046736384/tNXF3
4	3	The officia	37625	3762693	3195	719	http://t.cc	Marvel Ent	#######	FALSE	TRUE	Marvel	New York,	en	15687962	19411	FALSE	http://pbs.twimg.com/profile_images/573984336271122432/k8vEB
5	4	The latest	154673	2305185	1447	902	http://t.cc	USA TODA	#######	FALSE	TRUE	USATODA'	USA TODA	en	15754281	28436	FALSE	http://pbs.twimg.com/profile_images/726119684261744641/Y0uJv
6	5	An art, cul	61515	592916	79	11	http://t.cc	Laughing S	#######	FALSE	TRUE	LaughingS	New York	en	2172	12248	FALSE	http://pbs.twimg.com/profile_images/418456260971732992/By5Kk
7	6	Men's Fitn	34958	582422	1154	440	http://t.cc	Men's Fitn	#######	FALSE	TRUE	MensFitne	New York,	en	26259480	4390	FALSE	http://pbs.twimg.com/profile_images/610844529449676800/Im1NI
8	7	Tweets	69002	506444	1284	83196	https://t.c	Everything	#######	FALSE	FALSE	GAFollowe	Georgia, U	en	1.22E+08	1175	FALSE	http://pbs.twimg.com/profile_images/710676635553439744/9MCS
9	8	Everything	86725	362234	149	16295	http://t.cc	Movie TV	#######	FALSE	FALSE	movietvte	New York	en	18372616	411	FALSE	http://pbs.twimg.com/profile_images/643908040631054336/rstkT7
10	9	Liderazgo	11184	317196	3147	5218	http://t.cc	Audi de Mé	#######	FALSE	TRUE	AudideMe:	México	es	3.65E+08	306	FALSE	http://pbs.twimg.com/profile_images/445958709470969856/YYuH(
11	10	Toronto's	94859	260069	499	401	http://t.cc	Toronto Su	#######	FALSE	TRUE	TheToront	Toronto, C	en	24700876	1660	FALSE	http://pbs.twimg.com/profile_images/664793937064816640/OEHis
12	11	Follow us,	9607	259475	163	2732	http://t.cc	MovieRoo	#######	FALSE	FALSE	movieroor	New York,	en	15464215	112	FALSE	http://pbs.twimg.com/profile_images/2972155661/e872e0e719fcca
13	12	The Voice	112850	257197	4470	7452	https://t.c	JOE.ie	#######	FALSE	TRUE	JOEdotie	Ireland	en	1.32E+08	634	FALSE	http://pbs.twimg.com/profile_images/715127734842875905/yrkgqu
14	13	Toronto's	36261	248662	0	22	https://t.c	Breakfast	#######	FALSE	TRUE	BTtoronto	Toronto, C	en	22930902	1560	FALSE	http://pbs.twimg.com/profile_images/710669705850298370/ceJBz
15	14	The officia	13957	225078	16442	1116	http://t.cc	MR PORTE	#######	FALSE	TRUE	MRPORTE	London &	en	1.52E+08	2288	FALSE	http://pbs.twimg.com/profile_images/702838036342697984/qTPD-
16	15	Your Celet	96275	210006	628	823	http://t.cc	Hollywooc	#######	FALSE	TRUE	Hollywooc	LA & NYC	en	21246105	1941	FALSE	http://pbs.twimg.com/profile_images/668263886282362881/bEUV-
17	16	FLEX is the	21796	190520	465	334	http://t.cc	FLEX	#######	FALSE	FALSE	FLEX_Mag	New York,	en	28449123	1168	FALSE	http://pbs.twimg.com/profile_images/461623483529564160/AOqH
18	17	What's rea	44858	185763	1230	1356	http://t.cc	The Village	#######	FALSE	TRUE	villagevoic	New York	en	22059385	5126	FALSE	http://pbs.twimg.com/profile_images/616282889135751168/LHSvU
19	18	The best ir	37855	161384	16751	4078	http://t.cc	Popular M	#######	FALSE	TRUE	PopMech	New York	en	23116280	4480	FALSE	http://pbs.twimg.com/profile_images/651094255503433728/-n0ju(
20	19	Writer. So	189536	160544	289	44705	http://t.cc	Stacie H C	#######	FALSE	FALSE	StacieinAtl	Atlanta, G	en	15293352	3278	FALSE	http://pbs.twimg.com/profile_images/662053179463475200/N0UA
21	20	Houston's	168101	137887	2543	812	http://t.cc	KBXX 97.9	#######	FALSE	FALSE	979TheBo:	Houston, T	en	1.58E+08	607	FALSE	http://pbs.twimg.com/profile_images/723228310453391361/kz2Dy
22	21	@Her0a |	192749	133015	43568	63235	https://t.c	Lennon <U	#######	FALSE	FALSE	LennonBo:	Maringá, P	en	2.28E+09	51	FALSE	http://pbs.twimg.com/profile_images/725760667681173504/6AOyr
23	22	Touchpoin	190093	131018	1127	1210	http://t.cc	Touchpoin	#######	FALSE	FALSE	TDG_1	New York	en	7.92E+08	504	FALSE	http://pbs.twimg.com/profile_images/584652445315239937/QAc1Y
24	23	The Three	46276	116919	173	13	http://t.cc	BGR.com	#######	FALSE	TRUE	BGR	New York,	en	18131006	6567	FALSE	http://pbs.twimg.com/profile_images/656444883444494337/Xv_ssl
25	24	Co-Host, B	154252	103783	18624	835	https://t.c	Dina Puglie	#######	FALSE	TRUE	DinaPuglie	Toronto, C	en	1.48E+08	832	FALSE	http://pbs.twimg.com/profile_images/722126265654841344/qW5D
26	25	Tweets frc	29969	102150	1901	2052	http://t.cc	comiXolog	#######	FALSE	FALSE	comiXolog	New York,	en	15135752	2567	FALSE	http://pbs.twimg.com/profile_images/621025521724317696/R1733
27	26	My name i	77110	100602	2320	90288	http://t.cc	Spaceships	#######	FALSE	FALSE	SpaceshipsLB		en	1.92E+08	1493	FALSE	http://pbs.twimg.com/profile_images/666141965814075392/oF3f7
28	27	Toronto's	189730	100252	22785	418	http://t.cc	KiSS 92.5	#######	FALSE	TRUE	KiSS925	Toronto	en	47618120	531	FALSE	http://pbs.twimg.com/profile_images/712748787274350593/AOIpo
29	28	All about E	670178	96207	4	92127	http://t.cc	Best Videc	#######	FALSE	FALSE	BestGame	Toronto	en	2.69E+08	514	FALSE	http://pbs.twimg.com/profile_images/531879213284880384/ZlsTCj
30	29	Boris take:	70977	95143	2307	1881	https://t.c	Joe Gibbs	#######	FALSE	TRUE	JoeGibbsR	Huntersvill	en	15169828	2469	FALSE	http://pbs.twimg.com/profile_images/691712832753143809/T_Y55
31	30	We air we	42332	95058	24903	336	http://t.cc	The Social	#######	FALSE	TRUE	TheSocial(	Toronto, C	en	1.25E+09	317	FALSE	http://pbs.twimg.com/profile_images/422746382584143873/4aT15
32	31	Cityline De	6437	94102	1912	301	https://t.c	Shai DeLuc	#######	FALSE	FALSE	ShaiDeLuc	Canada / I	en	93196250	38	FALSE	http://pbs.twimg.com/profile_images/662474368488833029/Z9_oq
33	32	Exhausted	196098	91249	2927	75554	https://t.c	2 Kids and	#######	FALSE	FALSE	2kidsanda	Iowa	en	2.19E+08	1386	FALSE	http://pbs.twimg.com/profile_images/716016786295029761/RGgU
34	33	Chicago Fi	29722	90710	1841	1476	https://t.c	Chicago Fi	#######	FALSE	TRUE	ChicagoFir	Bridgeview	en	21677316	2075	FALSE	http://pbs.twimg.com/profile_images/691095381795586048/xnKo7

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

Sr. No.	Column Name	Description
1.	description	User Account Description
2.	statusesCount	Number of tweets by the user
3.	followersCount	Number of followers for the particular user
4.	favoritesCount	Count of friends for that particular user
5.	friendsCount	Number of friends of that particular user
6.	url	URL for the users account
7.	Name	Username
8.	created protected	When account is created and is it protected or not.
9.	Verified	Account is verified or not
10.	Captainamericalistnames	Words related to particular movie
11.	Location	User location
12.	Lang	Language
13.	Id	UserID
14.	listedCount	Count of the tweets
15.	followRequestSent	Request to follow sent by the user
16.	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

A	B	C	D	E	F
	usernames	text	location	score	moviecompany
1	__BreadWinner_	10 days for the new captain America I will definitely be in theatres to see that	Chattanooga, TN	0	Captain America
2	__Britaniaa	Team captain America stills	New York, USA	0	Captain America
3	__Britaniaa	Captain America Civil war looks so lit	México	1	Captain America
4	__ChelsMarie__	I can't wait to see captain America	Detroit	0	Captain America
5	__HeyGuy	RT @yourENTnews: 10 Day Countdown - Marvel's Captain America: Civil War #Spiderman	USA	0	Captain America
6	__Nise	RT @MyTwitLife: Aye Captain America going be lit next weekend	Pittsburgh, PA	3	Captain America
7	__Ryan95	Captain America: The Winter Solider has Mono! ! #MakeAMovieSick	Texas, USA	0	Captain America
8	__sammybear__	I can't sleep so in just gonna freak out about Captain America: Civil War	DETROIT, MICHIGAN	0	Captain America
9	__XxUniquee	RT @vonNextdoor__: Im looking forward to captain america : civil war	Toronto, Ontario	1	Captain America
10	_5SOS_SWE_	The Civil War is definitely the best Captain America movie. I give it 10/10 and you should definitely see it. #TeamIronMan	Miami, FL	0	Captain America
11	_AdrianRJr	RT @heroichollywood: Spider-Man Salutes Cap In Another 'Capain America: Civil War' TV Spot https://t.co/LzWbMygBgs https://t.co/GskTNlmnAD	Miami, FL	0	Captain America
12	_amandaceelyn	RT @PerezHilton: RT Members of #TeamCap are always keeping an eye out for each other! https://t.co/PRXlwiRFq7 https://t.co/DQM8kMdFSZ	Washington, DC	1	Captain America
13	_AshleyAnna_	Watched captain America the first avenger. The avengers.&captain America winter soldier.Couldnt keep my eyes open for avengers age of ultron	Springville, NY	1	Captain America
14	_bbboat	RT @Variety: #CaptainAmerica: #CivilWar is ready to rule the global box office. https://t.co/1u8FJrM5vq https://t.co/gQBQVrhYuc	Dallas	1	Captain America
15	_bbboat	RT @EW: #CaptainAmericaCivilWar is basically a third #Avengers movieand it's the best yet. Review: https://t.co/Tj7Ga1QXMw https://t.co/tY	Houston, TX	2	Captain America
16	_beware_7	I can't wait till Captain America come out im in that bitch	La Puente, CA	0	Captain America
17	_brandont18	RT @yourENTnews: 10 Day Countdown - Marvel's Captain America: Civil War #Spiderman	LakersNation, CA.	0	Captain America
18	_Brooke22	RT @EW: Robert Downey Jr. visited #IronMan fans at a London hospital and it was wonderful: https://t.co/PkW194S06a https://t.co/EKEVdVB1HZ	Green Bay, Wisconsin	0	Captain America

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

```
#using function getText to extract text part of tweets
text <- sapply(captainamericatweets,function(x) x$getText())

#converting latin1 characters to ASCII.
text <- sapply(text,function(row) iconv(row, "latin1", "ASCII", sub = ""))
# remove retweet entities
text = gsub("(RT|via)((?:\\b\\W*@\\w+)+)", "", text)
head(text)
# remove at people
text = gsub("@\\w+", "", text)
# remove punctuation
text = gsub("[[:punct:]]", "", text)
# remove numbers
text = gsub("[[:digit:]]", "", text)
# remove html links
text = gsub("http\\w+", "", text)

#remove captain america civil war from texts
text= gsub("Captain America","",text)
text = gsub("Civil War","",text)
text = gsub("war","",text)
# remove unnecessary spaces
text = gsub("[ \t]{2,}", "", text)
text = gsub("^\\s+|\\s+$", "", text)

# define "tolower error handling" function
try.error = function(x)
{
  # create missing value
  y = NA
  # tryCatch error
  try_error = tryCatch(tolower(x), error=function(e) e)
  # if not an error
  if (!inherits(try_error, "error"))
    y = tolower(x)
  # result
  return(y)
}
# lower case using try.error with sapply
text = sapply(text, try.error)
nrow(text)
# remove NAs in some_txt
#text = text[!is.na(text)]
#names(text) = NULL
```

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.

```
# classify emotion
class_emo = classify_emotion(text, algorithm="bayes", prior=1.0)

# get emotion best fit
emotion = class_emo[,7]

# substitute NA's by "unknown"
emotion[is.na(emotion)] = "unknown"
```

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.

```
# classify polarity
class_pol = classify_polarity(text, algorithm="bayes")
# get polarity best fit
polarity = class_pol[,4]
```

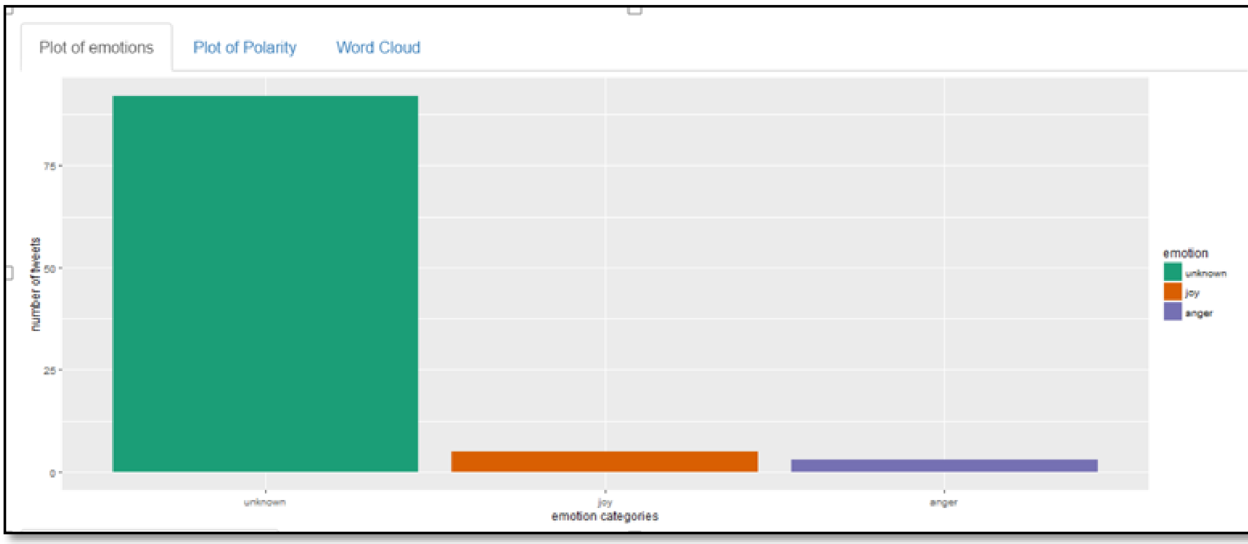
Creating a data frame:

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:

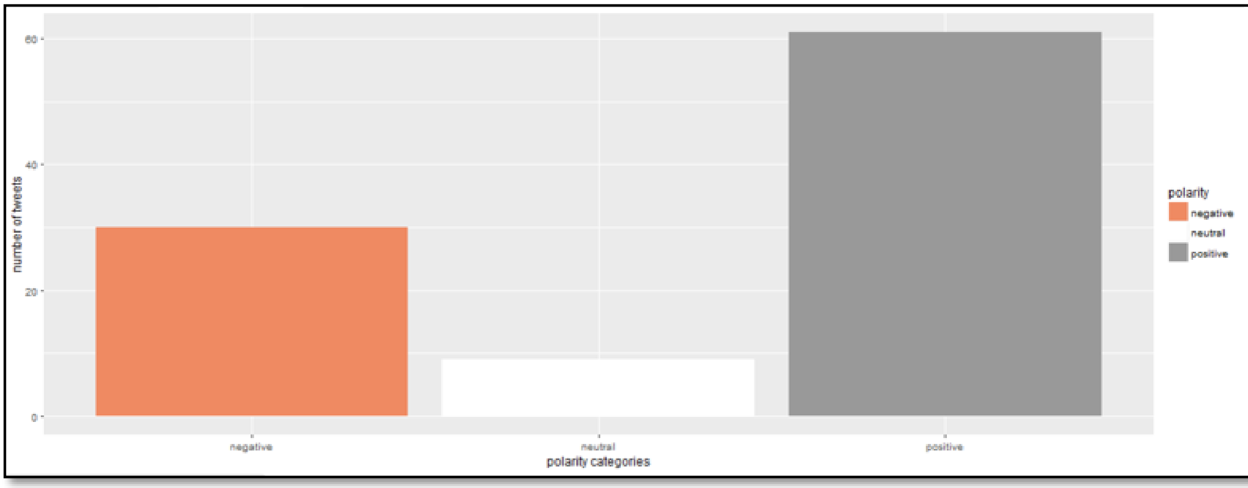
```
ggplot(sent_df, aes(x=emotion)) +
  geom_bar(aes(y=..count.., fill=emotion)) +
  scale_fill_brewer(palette="Dark2") +
  labs(x="emotion categories", y="number of tweets")
```



Creating polarity plot:

We follow similar procedure to obtain polarity plot:

```
# plot distribution of polarity
ggplot(sent_df, aes(x=polarity)) +
  geom_bar(aes(y=..count.., fill=polarity)) +
  scale_fill_brewer(palette="RdGy") +
  labs(x="polarity categories", y="number of tweets") #+
# theme(title = "Sentiment Analysis of Tweets about Captain America\n(classification by polarity)",
# plot.title = theme_text(size=12))
})
```



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.

```
emos = levels(factor(sent_df$emotion))
nemo = length(emos)
emo.docs = rep("", nemo)
for (i in 1:nemo)
{
  tmp = text[emotion == emos[i]]
  emo.docs[i] = paste(tmp, collapse=" ")
}
```


then
black
elizabeth
paul
rogers
returns
take
quest
reveals
may
lumberg
steve
under
fromand
ones
played
hemsworth
saiz
unknown
movies
operation
evans
buymens
jumps
shield
name
chris
with
head
says
guys
makes
ofand
action
like
calls
command
great
the
rebirth
change
batman
into
joy
doing
bill
good
jesus
son
funny
call
bed.
fact
star
how
looks
absolutely
not
well
idea
watch
bacteria
high
ifwere
adorkable
ironman
butt
wait
downey
are
order
its
daredevils
his
photo
outand
found
but
anger
wear
cap
pocket
use
leggings
lowkey
puts

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.

```
# remove stopwords
emo.docs = removeWords(emo.docs, "captain")
emo.docs = removeWords(emo.docs, "america")
emo.docs = removeWords(emo.docs, "civil")
emo.docs = removeWords(emo.docs, "war")
```

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.

```
# create corpus
corpus = Corpus(VectorSource(emo.docs))
tdm = TermDocumentMatrix(corpus)
tdm = as.matrix(tdm)
colnames(tdm) = emos
```

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.

```
captainamericatweets = searchTwitter("Captain America",n = 5000, lang = "en",geocode='40.712940,-73.987920,100mi')
head(captainamericatweets)
```

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.

```
#importing positive and negative lexicon
pos = readLines("positive_words.txt")
neg = readLines("negative_words.txt")
```

```
score.sentiment = function(sentences, pos.words, neg.words, .progress='none')
{

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

                   # split sentence into words with str_split (stringr package)
                   word.list = str_split(sentence, "\\s+")
                   words = unlist(word.list)
                   # compare words to the dictionaries of positive & negative terms
                   # find the first occurrence of the first argument in the second argument:
                   pos.matches = match(words, pos.words)
                   neg.matches = match(words, neg.words)
                   # get the position of the matched term or NA
                   # we just want a TRUE/FALSE
                   pos.matches = !is.na(pos.matches)
                   neg.matches = !is.na(neg.matches)
                   # final score
                   score = sum(pos.matches) - sum(neg.matches)
                   return(score)
                 }, pos.words, neg.words, .progress=.progress )
  # data frame with scores for each sentence
  scores.df = data.frame(text=sentences, score=scores)
  return(scores.df)
}
```

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'.

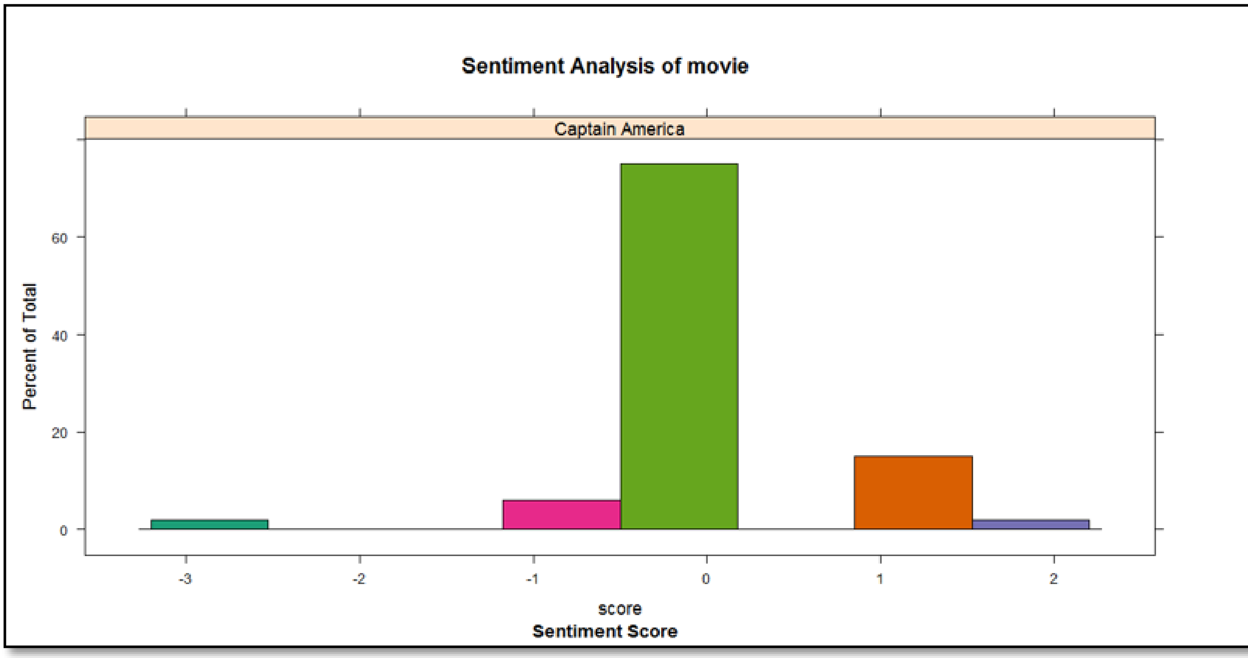
The is.na (pos.matches) will return true whenever it finds a match for positive word, otherwise it returns False. True is considered as +1 while False is considered as -1.

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.

```
  nooftweets = c(length(text))
  movie<-c(text)
  #applying function score.sentiment
  scores = score.sentiment(movie, pos, neg, .progress='text')
  scores$movie = factor(rep(c(input$select1), nooftweets))
  write.csv(scores,file="scores.csv")
  histogram(data=scores, ~score|movie, main="Sentiment Analysis of movie",col = col, sub="Sentiment Score")
})
```


Sentiment Analysis of movie
Captain America
Percent of Total
60
40
20
0
-3
-2
-1
0
1
2
score
Sentiment Score

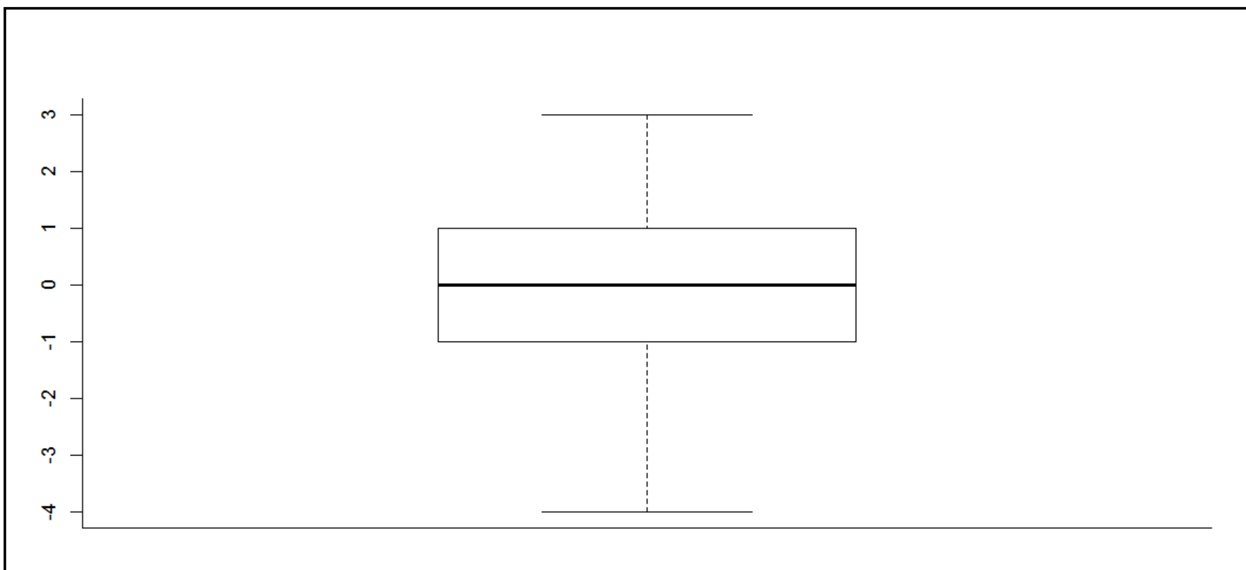
Generating boxplot to show sentiment analysis:

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

Box Plot function:

```
    nooftweets = c(length(text))
    movie<-c(text)
    #applying function score.sentiment
    scores = score.sentiment(movie, pos, neg, .progress='text')
    scores$movie = factor(rep(c(input$select1), nooftweets))
    par(bty="l")
   # write.csv(scores,file="scores.csv")
    boxplot(score~movie, data=scores) #making a boxplot of sentiments
  })
}
```

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.

```
modescore<-mode(final$score)
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

mode<-getmode(final$score)
number<-function(x) {

  #if(x==-10 | x==-9 | x==-8) y<-100
  if(x==-7 | x==--6 | x==-5) y<-50
  if(x==-4 | x==-3 | x==-2) y<-100
  if(x==-1) y<-150
  if(x==0) y<-200
  if(x==1) y<-250
  if(x==2) y<-300
  if(x==3 | x==4 | x==5) y<-400
  if(x==6 | x==7 | x==8) y<-500
  return(y)
}
noOfScreens<-sapply(mode,number)
```

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

Generating a simple web app in shiny:

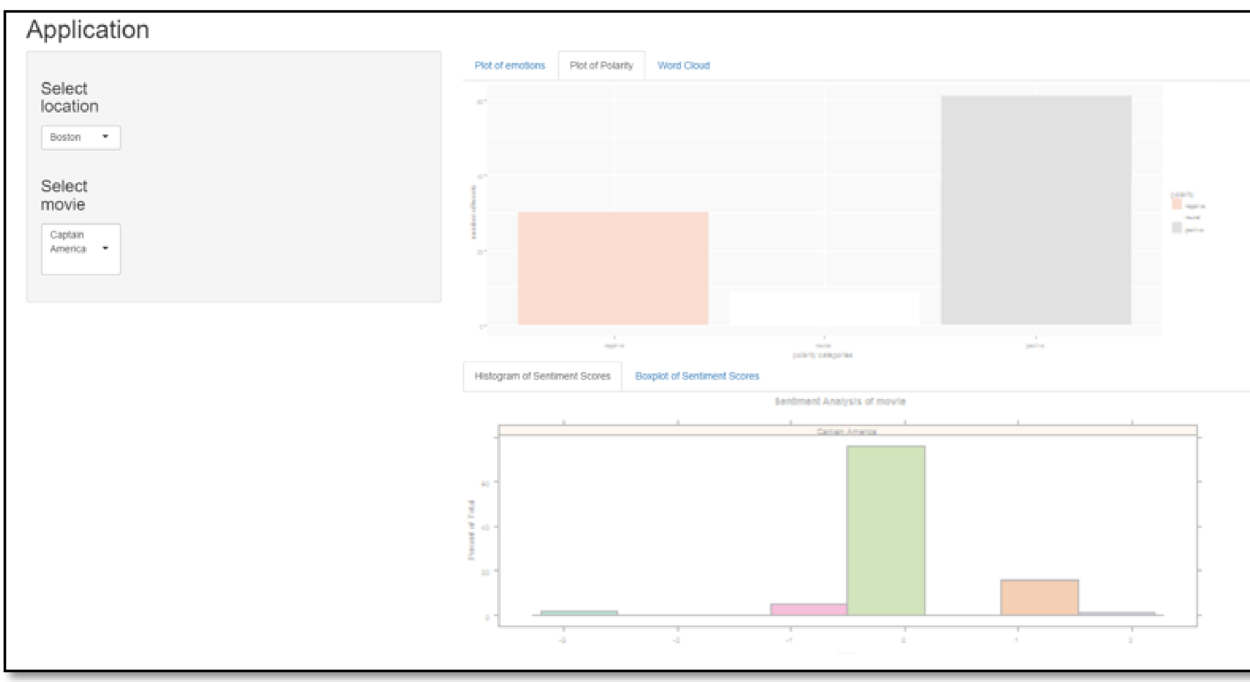
```
library(shiny)
ui <- fluidPage()

server <- function(input, output) {}

shinyApp(ui = ui, server = server)
```

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.

```
fluidRow(column(3, selectInput("select", label = h3("Select location"),
                             choices = c("Boston"="40.698470,-73.951442,50mi", "NewYork"="40.698470,-73.951442,50mi", "Los Angeles"="3
                             selected = 1)  ) )
```

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 be dynamically used in any function so as to render results corresponding to it.

```
output$emotionplot<-renderPlot({
  captainamericatweets = searchTwitter("Captain America",n = 100, lang = "en",geocode=input$select)
  #head(captainamericatweets)
```

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.

```
,
fluidRow(column(3, selectInput("select1",label = h3("Select movie"),
                               choices = c("Captain America"="Captain America","X-Men"="X-Men"),selected=1))

)
```

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.

```
output$histogramscores<-renderPlot({
  captainamericatweets = searchTwitter(input$select1,n = 100, lang = "en",geocode="40.712940,-73.987920,3000mi")
  #head(captainamericatweets)
```

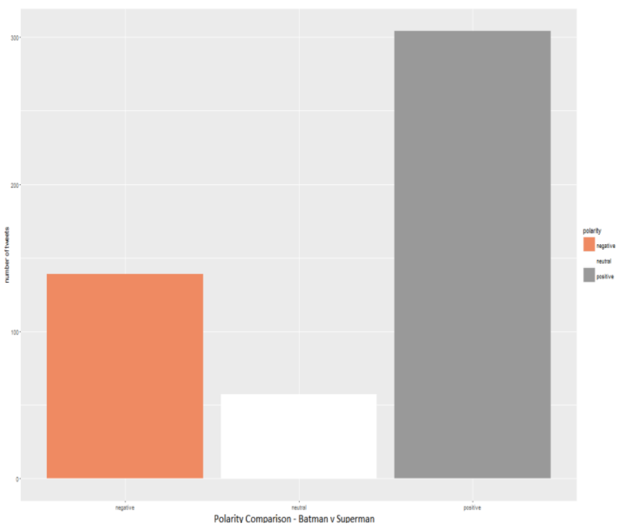
Here we use searchTwitter function to dynamically extract tweets for movie selected by user in the radius of 3000 miles from Northeastern University and print various user friendly plots.

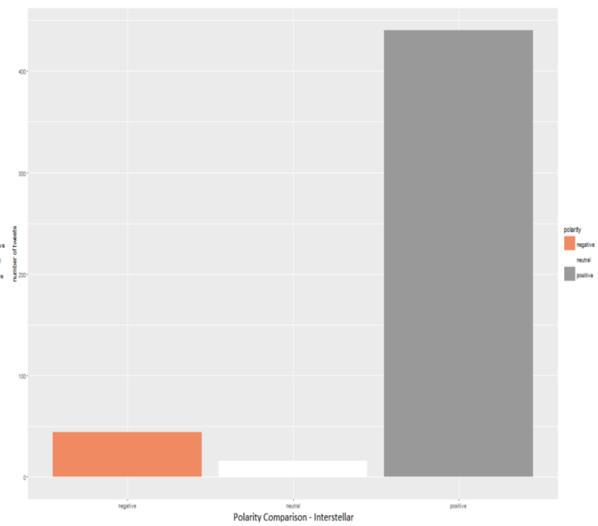
Power of Sentiments

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.

Link to the web application - https://shruti.shinyapps.io/shiny/

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/