

# **Classification of YouTube videos based on sentiments by implementing Bayes algorithm**

Yatish Jain

Bioinformatics, College of Science, Northeastern University

## **ABSTRACT**

I introduce a novel classification approach of YouTube videos based on the sentiments. These YouTube videos are classified as Joy, Anger, Sadness, Fear, Disgust, Surprise after a particular keyword is searched. This classification is based on the sentiment analysis performed on the description scraped using the YouTube API. This will be especially beneficial for the companies planning to broadcast their advertisements on YouTube videos. YouTube API designed for the purpose of this project takes a search keyword, which then access the data on YouTube matching the keyword under all headers like Videos, Channels and Playlists etc. On the extracted data, applied Bayes algorithm to classify the data based on the emotions and polarity. Validation of the classified videos was done by actually looking up the videos and manually looking at the comments to check the actions of users on that video and validate the classification. Future scope of this project can involve fetching the data on the same topic from twitter and put some hash tag on the classified YouTube videos to better represent these classified videos. Also the API itself can be modified to fetch comments as well from YouTube API and then sentiment analysis can be performed on combination of comments and description.

## **INTRODUCTION**

Whenever I search anything on YouTube it gives me varied result on the searched topic. I want to see an additional level of filter whenever anybody searches anything on YouTube. This additional filter will generate the webpage in such a way that whatever is searched by the user can be filtered based on the mood/sentiment of user. For example If I just type song then using this filter I can search sad, happy or any other mood songs. In my code I am fetching the data of Starbucks from YouTube and using sentiment analysis to classify the data based on emotion as well as polarity (i.e. negative or positive).

Sentiment analysis aims to determine the mood, tone and attitude of the speaker. Sentiment analysis is currently used to extract public opinion from social media and categorize how general media reacts to any particular event. The semantic features consist of the semantic concepts (e.g. "person", "company", "city") that represent the entities (e.g. "Steve Jobs", "Vodafone", "London") extracted from videos [1]. Semantic

feature is the part of Naïve Bayes algorithm. The canonical need for proper negation detection in sentiment analysis can be expressed as the fundamental difference in semantics inherent in the phrases, “this is great,” versus, “this is not great.”[2]. Naïve Bayes algorithm is used to perform sentiment analysis based on emotions and polarity. Classify\_emotion function of sentiment package of R uses Naïve Bayes algorithm to classify the data based on emotions. Classify\_polarity function of sentiment package of R uses Naïve Bayes algorithm to classify data into Positive, Negative and Neutral.

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes’ theorem with strong (naive) independence assumptions between the features. This means that under the above independence assumptions, the conditional distribution over the class variable C is:

$$p(C_x|x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^n p(x_i|C_k)$$

where the evidence  $Z = p(x)$  is a scaling factor dependent only on  $x_i, \dots, x_n$ , that is, a constant if the values of the feature variables are known.

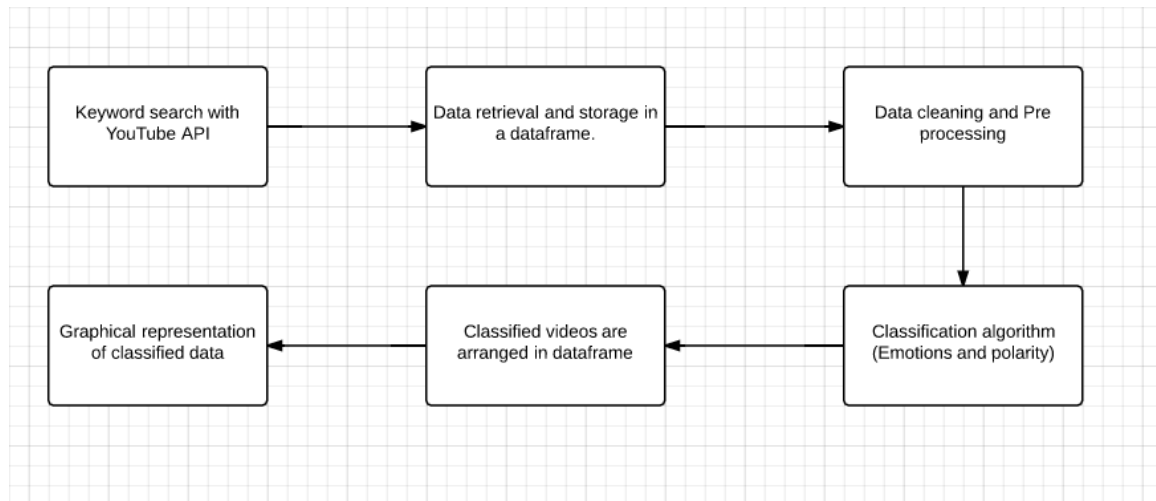


Fig. 1.1 Entire Workflow

The proposed solution involves the use of API to fetch the data from YouTube videos. Retrieved data is then stored into a dataframe that makes the cleaning process easier. Lot of pre processing and cleaning is done to get rid of additional spaces, escape characters, punctuations etc. Sentiment package of R is used to classify the data based on emotions and polarity. Classified data is again stored in dataframe to make the graphical representations easier.

## MATERIALS AND METHODS

### Code with documentation

```
# Load required libraries
library(rjson)
library(RCurl)
library(sentiment)
library(ggplot2)

#Generate your own key as per the instructions in document and paste it here to
configure the API
key<- "AIzaSyCtXqcXod2qpAaDpWs5-PKYT7dymMTBbUA"

#Funtion to check connection. This getStats function will fetch the statistics
of any video given the video ID and key

getStats <- function(id,key){
  url=paste("https://www.googleapis.com/youtube/v3/videos?id=",id,"&key=",key,"&p
art=statistics,snippet",sep="")
  raw.data <- getURL(url)
  rd <- fromJSON(raw.data,unexpected.escape = "skip")
  if(length(rd$items)!= 0){
    title<- rd$items[[1]]$snippet$title
    channelTitle<- rd$items[[1]]$snippet$channelTitle
    description<-rd$items[[1]]$snippet$description
    views<- rd$items[[1]]$statistics$viewCount
    likes<- rd$items[[1]]$statistics$likeCount
    if(is.null(likes)){
      likes<- "Info Not available"
    }
    dislikes<- rd$items[[1]]$statistics$dislikeCount
    if(is.null(dislikes)){
      dislikes<- "Info Not available"
    }
    fav<- rd$items[[1]]$statistics$favoriteCount
    if(is.null(fav)){
      fav<- "Info Not available"
    }
    comments<- rd$items[[1]]$statistics$commentCount
    if(is.null(comments)){
      comments<- "Info Not available"    }

  return(data.frame(title,description,channelTitle,views,likes,dislikes,fav,comme
nts))
}
}

#getVideos function return the list of videos along with their statistics given
the channelID and key.
```

```

getVideos<- function(channelID,key){
url=paste("https://www.googleapis.com/youtube/v3/search?key=",key,"&channelId=",
channelID,"&part=snippet,id&order=date&maxResults=10",sep="")
raw.data <- getURL(url)
rd <- fromJSON(raw.data)
perPage<- rd$pageInfo$resultsPerPage
totalResults<-rd$pageInfo$totalResults
totalVideos<-min(perPage,totalResults)
stats<c(as.character(),as.character(),as.character(),as.integer(),as.integer(),
as.integer(),as.integer(),as.integer())

for (i in 1:totalVideos){
kind<- rd$items[[i]]$id$kind
if(kind == "youtube#video"){
videoID<- rd$items[[i]]$id$videoId
print(videoID)
stats<-rbind(stats,getStats(videoID,key))
}
else if(kind == "youtube#playlist"){
playlistID<- rd$items[[i]]$id$playlistId
url=paste("https://www.googleapis.com/youtube/v3/playlistItems?part=snippet&2ContentDetails&maxResults=10&playlistId=", playlistID,"&key=",key,sep="")
raw.data <- getURL(url)
rd1 <- fromJSON(raw.data)
perPage<- rd1$pageInfo$resultsPerPage
totalResults<-rd1$pageInfo$totalResults
totalVideos<-min(perPage,totalResults)
for(i in 1:totalVideos){
videoID<-rd1$items[[i]]$contentDetails$videoId
print(videoID)
stats<-rbind(stats,getStats(videoID,key))
}
}
}
return(stats)
}

```

*#getChannelsOrPlaylists function return the list of videos and their statistics associated with a keyword search on YouTube. # When you search a keyword on youtube sometimes playlists also end up in search and we fetch the data from those playlists as well which might not be directly related to our search keyword, but it will fetch the data of similar searches.*

```

getChannelsOrPlaylists<- function(search, key){
search<-URLencode(search)
url<paste("https://www.googleapis.com/youtube/v3/search?q=",search,"&key=",key,
"&type=channel&part=snippet&maxResults=50",sep="")
raw.data <- getURL(url)
rd <- fromJSON(raw.data)
perPage<- rd$pageInfo$resultsPerPage
totalResults<-rd$pageInfo$totalResults
totalChannels<- min(perPage,totalResults)

data<c(as.character(),as.character(),as.character(),as.integer(),as.integer(),a
s.integer(),as.integer(),as.integer())

```

```

for(i in 1:totalChannels){
channelID<- rd$items[[i]]$id$channelId
print(channelID)
if(!is.null(channelID)){
data<-rbind(data, getVideos(channelID, key))
}
}

data<- data[complete.cases(data), ]

data<-unique(data)
write.csv(data, "dataTest.csv", row.names=F)
return(data)
}

#retrieved data from API is stored in a vector named data
search<-"starbucks"

data<-getChannelsOrPlaylists(search, key)

#Data preprocessing

some_txt = as.character(data$description)
head(some_txt)

if (!require("pacman")) install.packages("pacman")
pacman::p_load(devtools, installr)

##please install the following to run this code #install.Rtools()
#install_url("http://cran.r-project.org/src/contrib/Archive/Rstem/Rstem_0.4-
1.tar.gz")
#install_url("http://cran.r-
project.org/src/contrib/Archive/sentiment/sentiment_0.2.tar.gz")

# remove punctuation
some_txt = gsub("[[:punct:]]", "", some_txt)
# remove numbers
some_txt = gsub("[[:digit:]]", "", some_txt)
# remove html links
some_txt = gsub("http\\w+", "", some_txt)
# remove unnecessary spaces
some_txt = gsub("[ \\t]{2,}", "", some_txt)
some_txt = gsub("^\\s+|\\s+$", "", some_txt)

# 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
some_txt = sapply(some_txt, try.error)
# remove NAs in some_txt
some_txt = some_txt[!is.na(some_txt)] names(some_txt) = NULL

#Classifying YouTube data based on emotions
class_emo = classify_emotion(some_txt, algorithm="bayes", prior=1.0)
emotion = class_emo[,7]
# substitute NA's by "unknown"
emotion[is.na(emotion)] = "unknown"
# classify polarity
class_pol = classify_polarity(some_txt, algorithm="bayes")
# get polarity best fit
polarity = class_pol[,4]

# Create data frame with the results and obtain some general statistics # data
frame with results
sent_df = data.frame(text=some_txt, emotion=emotion,
polarity=polarity, stringsAsFactors=FALSE)
# sort data frame sent_df = within(sent_df,

emotion <- factor(emotion, levels=names(sort(table(emotion),
decreasing=TRUE)))

# plot distribution of emotions

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 videos") + ggtitle("Sentiment Analysis of videos about
Starbucks\n(classification by emotion)") + theme(plot.title =
element_text(size=12, face="bold"))

# 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 videos") + ggtitle("Sentiment
Analysis of videos about Starbucks\n(classification by polarity)") +
theme(plot.title = element_text(size=12, face="bold"))

emos = levels(factor(sent_df$emotion))
nemo = length(emos)
emo.docs = rep("", nemo)
for (i in 1:nemo) {
tmp = some_txt[emotion == emos[i]]
emo.docs[i] = paste(tmp, collapse=" ")
}
# create corpus
corpus = Corpus(VectorSource(emo.docs))
tdm = TermDocumentMatrix(corpus)
tdm = as.matrix(tdm)
colnames(tdm) = emos

```

```
# comparison word cloud
comparison.cloud(tdm, colors = brewer.pal(nemo, "Dark2"),
scale = c(3,.5), random.order = FALSE, title.size = 1.5) data<-
cbind(data,emotion=sent_df$emotion,polarity=sent_df$polarity)
```

## RESULTS

We have got a final dataframe which has two more column of emotions and polarity. We can put a filter in this final dataframe to just select videos with any particular emotions or polarity.

For example: The following code gives the videos with emotion disgust. There are only 3 instances of such emotion:

```
dataDisgust<-data[which(data$emotion=="disgust"),]
str(dataDisgust)
'data.frame': 3 obs. of 10 variables:
 $ title : Factor w/ 422 levels "Meet A Starbucks Barista Who Loves to
Bake with Coffee",...: 92 273 327
 $ description : Factor w/ 349 levels "Meet Claire. She's one of our amazing
baristas who's rediscovered her love for coffee in a delicious way. She bakes
with it. Bu"| __truncated__,...: 81 225 278
 $ channelTitle: Factor w/ 126 levels "Starbucks Coffee",...: 10 44 90
 $ views : Factor w/ 325 levels "796","2366","1320",...: 86 220 86
 $ likes : Factor w/ 140 levels "Info Not available",...: 13 13 15
 $ dislikes : Factor w/ 98 levels "Info Not available",...: 2 6 6
 $ fav : Factor w/ 1 level "0": 1 1 1
 $ comments : Factor w/ 122 levels "Info Not available",...: 14 12 17
 $ emotion : Factor w/ 7 levels "unknown","joy",...: 6 6 6
 $ polarity : Factor w/ 3 levels "negative","neutral",...: 2 1 3
```

The following code gives the videos with negative polarity. There are 61 instances of such emotion:

```
dataNegative<-data[which(data$polarity=="negative"),]
str(dataNegative)

'data.frame': 61 obs. of 10 variables:
 $ title : Factor w/ 422 levels "Meet A Starbucks Barista Who Loves to
Bake with Coffee",...: 2 7 8 20 25 28 32 33 34 42 ...
 $ description : Factor w/ 349 levels "Meet Claire. She's one of our amazing
baristas who's rediscovered her love for coffee in a delicious way. She bakes
with it. Bu"| __truncated__,...: 2 7 8 20 25 28 32 33 34 42 ...
 $ channelTitle: Factor w/ 126 levels "Starbucks Coffee",...: 1 1 1 2 3 3 4 4 5
...
 $ views : Factor w/ 325 levels "796","2366","1320",...: 2 7 8 20 25 28 32
33 34 42 ...
 $ likes : Factor w/ 140 levels "Info Not available",...: 1 1 1 4 12 13 9
14 10 18 ...
```

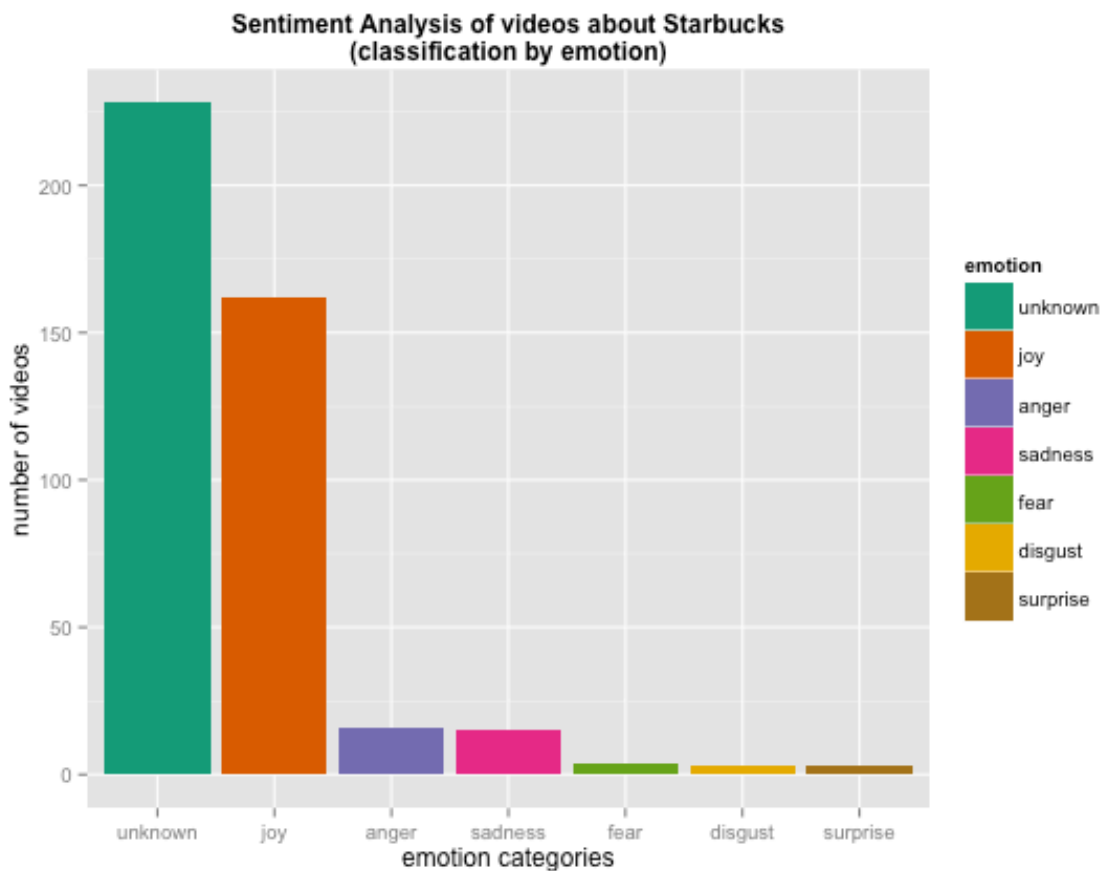
```

$ dislikes      : Factor w/ 98 levels "Info Not available",...: 1 1 1 5 6 6 6 6 6
5 ...
$ fav           : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
$ comments      : Factor w/ 122 levels "Info Not available",...: 1 1 1 11 12 13 13
12 14 16 ...
$ emotion       : Factor w/ 7 levels "unknown","joy",...: 1 1 2 2 2 4 1 1 1 1 ...
$ polarity      : Factor w/ 3 levels "negative","neutral",...: 1 1 1 1 1 1 1 1 1 1
...

```

Classification of Starbucks videos is done based on the following emotions:

1. Joy
2. Anger
3. Sadness
4. Fear
5. Disgust
6. Surprise
7. Unknown

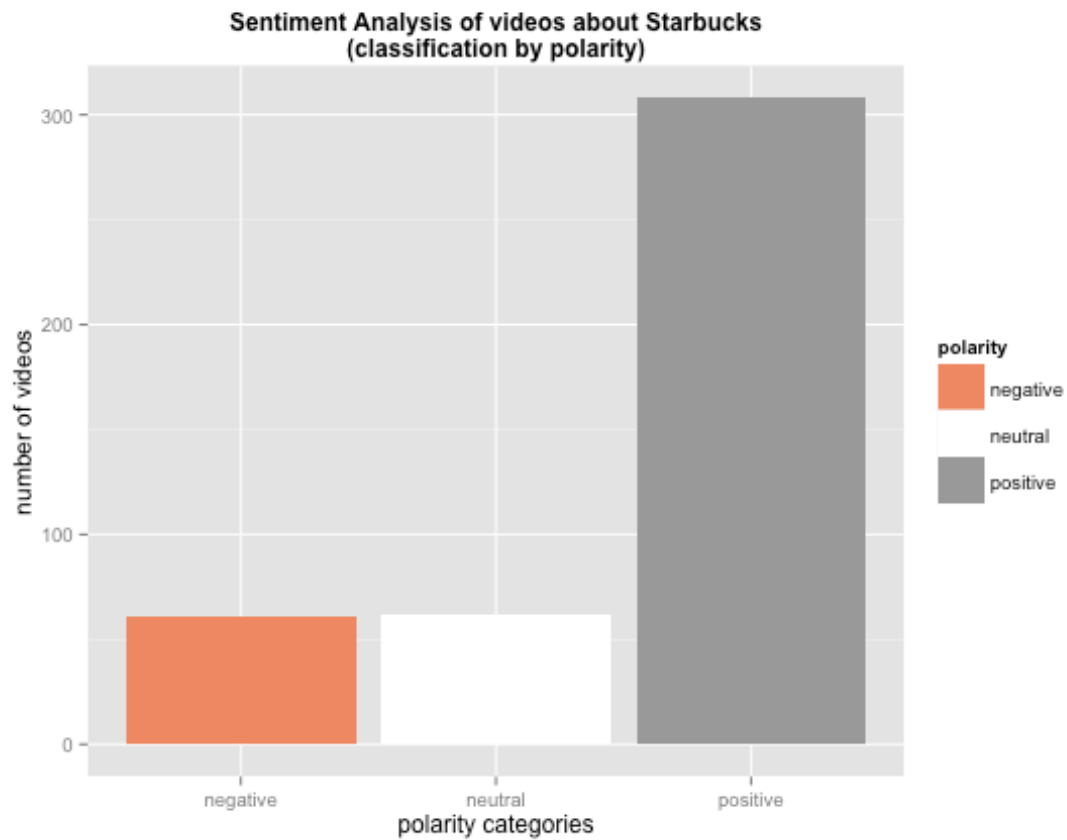


Classification of Starbucks videos is done based on the following polarity:

1. Positive



2. Negative
3. Neutral



Word Cloud can also be generated from the corpus of descriptions of YouTube videos, which gives the majority of words under each emotions:



4. Fear – 2 + videos
5. Disgust – 2 + videos
6. Surprise – 2 + videos
7. Unknown – 250 + videos


From the 2nd graph we can make the following classification based on polarity:


1. Positive – 305 + videos
2. Negative – 60 + videos
3. Neutral – 60 + videos

To validate the classification we are taking one of the classified videos and then reading the comments on that video to infer the emotion behind that video.

Title	Channel Title	Views	Likes	Dislikes	Fav	Comments	Emotion	Polarity
My Starbucks Rewards	Starbucks Singapore	2701	10	0	0	2	Joy	Neutral


Lets take a look at the comments of this video:






SHOW MORE



ALL COMMENTS (5)




Top comments ▾






**superchef2331** 1 year ago
 There is this iOS and android app that let's you get free starbucks money just for downloading free apps! Follow this link to get started: <http://featu.re/ABB4G7>

Reply ·  






**Ryan Doonan** 1 year ago
 I love Starbucks they know wht I get everytime I go really great drinks :)

Reply ·  






**Love2 Loveu** 1 year ago
 I love Starbucks. Had to take a sabbatical--I was getting obsessed!

Reply · 1  





**Selena Pedroza** 1 year ago
 Yall need to make a starbucks in Denver city Texas please

Reply ·  



**cheryl Duszynski** 1 year ago
 Love Starbucks♥

Reply ·  

From the comments above we can validate the working of our classification based on sentiment. All the comments on this video seems to be excited with joy and hence the video is classified as Joy.

Note1: - Note that polarity of this video is considered as Neutral based on the description of the video.

Note2: - Note that the count of comments fetched from my API is just two and we can see 5 comments on this video as the YouTube API data is not updated but the console developer project is gaining momentum and hopefully this will be refreshed frequently when it gains popularity.

### **Strengths of this approach**

This approach will create an additional filter on YouTube pages, which will bring sentiment analysis to user and this added filter would enhance the usability of YouTube search.

Additional polarity feature can set the tone of the video and further enhance the feature. Channel owners can actually broadcast the videos with positive tone above the videos of negative tone and can take advantage of this usability.

### **Weakness of this approach**

As we can see in the classification of emotions 250 + videos are classified under the unknown emotions this issue can be overcome by applying the concept from Hassan Saif et. al. paper incorporation of semantic feature to Naïve Bayes algorithm by training the model for sentiment analysis[1].

### **Future steps**

We can further increase this concept of videos classification to actually incorporate Twitter hash tags to the videos, which will increase the searchability of any video. Next step will involve enhancing the YouTube API to include search list as well along with videos, channels and playlist. This will fetch more related data of the keyword. Also along with the other fields we can also scrape comments on any videos and make a separate dataframe of comments and then perform sentiment analysis on them.

## **LITERATURE CITED**

[1] Hassan Saif, Yulan He and Harith Alani, Knowledge Media Institute, The Open University, United Kingdom. **Semantic Sentiment Analysis of Twitter.**

[2] Isaac G. Councill, Ryan McDonald, Leonid Velikovich “**What’s Great and What’s Not: Learning to Classify the Scope of Negation for Improved Sentiment Analysis**”, Proceedings of the Workshop on Negation and Speculation in Natural Language Processing, pages 51–59, Uppsala, July 2010.

Adiya Abisheva et. al. "Who watches (and shares) what on youtube? and when?: using twitter to understand youtube viewership." ACM 978-1-4503-2351-2/14/02

<http://www.r-bloggers.com/twitter-sentiment-analysis-with-r/>

<http://www.slideshare.net/sumit786raj/sentiment-analysis-of-twitter-data/6>

<http://www.inside-r.org/howto/mining-twitter-airline-consumer-sentiment>

<http://analyzecore.com/2014/04/28/twitter-sentiment-analysis/>

Rudy Prabowo, Mike Thelwall "**Sentiment Analysis: A Combined Approach**",  
School of Computing and Information Technology University of Wolverhampton

Alec Go, Richa Bhayani et.al. "Twitter Sentiment Classification using Distant Supervision", Stanford University