**A complete guide to text processing using Twitter data and R.**

# Why Text Processing using R?

With the increasing importance of computational text analysis in research , many researchers face the challenge of learning how to use advanced software that enables this text analysis. Currently, one of the most popular environments for computational methods and the emerging field of “data science” is the R statistical software. However, for researchers that are not well-versed in programming, learning how to use R can be a challenge, and performing text analysis in particular can seem daunting.

# **Why R?**

R was specifically designed for statistical analysis, which makes it highly suitable for data science applications. Although the learning curve for programming with R can be steep, especially for people without prior programming experience, the tools now available for carrying out text analysis in R make it easy to perform powerful, cutting-edge text analytics using only a few simple commands. One of the keys to R’s explosive growth has been its densely populated collection of extension software libraries, known in R terminology as packages, supplied and maintained by R’s extensive user community. Each package extends the functionality of the base R language and core packages, and in addition to functions and data must include documentation and examples, often in the form of vignettes demonstrating the use of the package. The best-known package repository, the Comprehensive R Archive Network (CRAN), currently has over 10,000 packages that are published.

Twitter("#technology", n=1000,lang = "en")

Convert this extracted data to a dataframe which makes it more readable and easier to work with.

amazon\_tweets <- twListToDF(tweets\_a)

google\_tweets <- twListToDF(tweets\_g)

facebook\_tweets <- twListToDF(tweets\_f)

tech\_tweets <- twListToDF(tweets\_tech)

View(amazon\_tweets)

View(google\_tweets)

View(facebook\_tweets)

View(tech\_tweets)

Below is a code to pre-process the data and remove tabs, blank spaces, links etc. This section can be modified according to one’s requirements.

google\_text<- google\_tweets$text

amazon\_text<- amazon\_tweets$text

facebook\_text<- facebook\_tweets$text

tech\_text<- tech\_tweets$text

#convert all text to lower case

google\_text<- tolower(google\_text)

amazon\_text<- tolower(amazon\_text)

facebook\_text<- tolower(facebook\_text)

tech\_text<- tolower(tech\_text)

# Replace blank space (“rt”)

google\_text <- gsub("rt", "", google\_text)

amazon\_text <- gsub("rt", "", amazon\_text)

facebook\_text <- gsub("rt", "", facebook\_text)

tech\_text <- gsub("rt", "", tech\_text)

# Replace @UserName

google\_text <- gsub("@\\w+", "", google\_text)

amazon\_text <- gsub("@\\w+", "", amazon\_text)

facebook\_text <- gsub("@\\w+", "", facebook\_text)

tech\_text <- gsub("@\\w+", "", tech\_text)

# Remove punctuation

google\_text <- gsub("[[:punct:]]", "", google\_text)

amazon\_text <- gsub("[[:punct:]]", "", amazon\_text)

facebook\_text <- gsub("[[:punct:]]", "", facebook\_text)

tech\_text <- gsub("[[:punct:]]", "", tech\_text)

# Remove links

google\_text <- gsub("http\\w+", "", google\_text)

amazon\_text <- gsub("http\\w+", "", amazon\_text)

facebook\_text <- gsub("http\\w+", "", facebook\_text)

tech\_text <- gsub("http\\w+", "", tech\_text)

# Remove tabs

google\_text <- gsub("[ |\t]{2,}", "", google\_text)

amazon\_text <- gsub("[ |\t]{2,}", "", amazon\_text)

facebook\_text <- gsub("[ |\t]{2,}", "", facebook\_text)

tech\_text <- gsub("[ |\t]{2,}", "", tech\_text)

# Remove blank spaces at the beginning

google\_text <- gsub("^ ", "", google\_text)

amazon\_text <- gsub("^ ", "", amazon\_text)

facebook\_text <- gsub("^ ", "", facebook\_text)

tech\_text <- gsub("^ ", "", tech\_text)

# Remove blank spaces at the end

google\_text <- gsub(" $", "", google\_text)

amazon\_text <- gsub(" $", "", amazon\_text)

facebook\_text <- gsub(" $", "", facebook\_text)

tech\_text <- gsub(" $", "", tech\_text)

# What are Stop Words?

When working with text mining applications, we often hear of the term “stop words” or “stop word list” or even “stop list”. Stop words are basically a set of commonly used words in any language, not just English. The reason why stop words are critical to many applications is that, if we remove the words that are very commonly used in a given language, we can focus on the important words instead.

Stop words are generally thought to be a **“single set of words”**. It really can mean different things to different applications.

#clean up by removing stop words

google\_tweets.text.corpus <- tm\_map(google\_tweets.text.corpus, function(x)removeWords(x,stopwords()))

amazon\_tweets.text.corpus <- tm\_map(amazon\_tweets.text.corpus, function(x)removeWords(x,stopwords()))

facebook\_tweets.text.corpus <- tm\_map(facebook\_tweets.text.corpus, function(x)removeWords(x,stopwords()))

tech\_tweets.text.corpus <- tm\_map(tech\_tweets.text.corpus, function(x)removeWords(x,stopwords()))

We are done pre-processing our data, and are ready to do some analysis.

# **What are Word Clouds?**

Word clouds (also known as text clouds or tag clouds) work in a simple way: the more a specific word appears in a source of textual data (such as a speech, blog post, or database), the bigger and bolder it appears in the word cloud.

So let’s generate some word clouds and find out some of the frequent and important terms being used in the tweets we have extracted.

library("wordcloud")

#generate wordcloud

wordcloud(google\_tweets.text.corpus,min.freq = 10,colors=brewer.pal(8, "Dark2"),random.color = TRUE,max.words = 500)

wordcloud(amazon\_tweets.text.corpus,min.freq = 10,colors=brewer.pal(8, "Dark2"),random.color = TRUE,max.words = 500)

wordcloud(facebook\_tweets.text.corpus,min.freq = 10,colors=brewer.pal(8, "Dark2"),random.color = TRUE,max.words = 500)

wordcloud(tech\_tweets.text.corpus,min.freq = 10,colors=brewer.pal(8, "Dark2"),random.color = TRUE,max.words = 500)

# What is Sentiment Analysis?

Sentiment essentially relates to feelings; attitudes, emotions and opinions. Sentiment Analysis refers to the practice of applying Natural Language Processing and Text Analysis techniques to identify and extract subjective information from a piece of text. A person’s opinion or feelings are for the most part subjective and not facts. Which means to accurately analyze an individual’s opinion or mood from a piece of text can be extremely difficult. With Sentiment Analysis from a text analytics point of view, we are essentially looking to get an understanding of the attitude of a writer with respect to a topic in a piece of text and its polarity; whether it’s positive, negative or neutral.

In recent years there has been a steady increase in interest from brands, companies and researchers in Sentiment Analysis and its application to business analytics. The business world today, as is the case in many data analytics streams, are looking for “business insight.”

In relation to sentiment analysis, I am talking about insights into consumer behavior, what customers want, what are customers like and dislike about the products, what their buying signals are, what their decision process looks like etc because in the end of the its the customers for whose satisfaction these businesses work for.

#getting emotions using in-built function

mysentiment\_google<-get\_nrc\_sentiment((google\_text))

mysentiment\_amazon<-get\_nrc\_sentiment((amazon\_text))

mysentiment\_facebook<-get\_nrc\_sentiment((facebook\_text))

mysentiment\_tech<-get\_nrc\_sentiment((tech\_text))

#calculationg total score for each sentiment

Sentimentscores\_google<-data.frame(colSums(mysentiment\_google[,]))

Sentimentscores\_amazon<-data.frame(colSums(mysentiment\_amazon[,]))

Sentimentscores\_facebook<-data.frame(colSums(mysentiment\_facebook[,]))

Sentimentscores\_tech<-data.frame(colSums(mysentiment\_tech[,]))

names(Sentimentscores\_google)<-"Score"

Sentimentscores\_google<-cbind("sentiment"=rownames(Sentimentscores\_google),Sentimentscores\_google)

rownames(Sentimentscores\_google)<-NULL

names(Sentimentscores\_amazon)<-"Score"

Sentimentscores\_amazon<-cbind("sentiment"=rownames(Sentimentscores\_amazon),Sentimentscores\_amazon)

rownames(Sentimentscores\_amazon)<-NULL

names(Sentimentscores\_facebook)<-"Score"

Sentimentscores\_facebook<-cbind("sentiment"=rownames(Sentimentscores\_facebook),Sentimentscores\_facebook)

rownames(Sentimentscores\_facebook)<-NULL

names(Sentimentscores\_tech)<-"Score"

Sentimentscores\_tech<-cbind("sentiment"=rownames(Sentimentscores\_tech),Sentimentscores\_tech)

rownames(Sentimentscores\_tech)<-NULL

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#plotting the sentiments with scores

ggplot(data=Sentimentscores\_google,aes(x=sentiment,y=Score))+geom\_bar(aes(fill=sentiment),stat = "identity")+

theme(legend.position="none")+

xlab("Sentiments")+ylab("scores")+ggtitle("Sentiments of people behind the tweets on tech giant GOOGLE")

ggplot(data=Sentimentscores\_amazon,aes(x=sentiment,y=Score))+geom\_bar(aes(fill=sentiment),stat = "identity")+

theme(legend.position="none")+

xlab("Sentiments")+ylab("scores")+ggtitle("Sentiments of people behind the tweets on ecomerce giant AMAZON")

ggplot(data=Sentimentscores\_facebook,aes(x=sentiment,y=Score))+geom\_bar(aes(fill=sentiment),stat = "identity")+

theme(legend.position="none")+

xlab("Sentiments")+ylab("scores")+ggtitle("Sentiments of people behind the tweets on Social Netwoking site FACEBOOK")

ggplot(data=Sentimentscores\_tech,aes(x=sentiment,y=Score))+geom\_bar(aes(fill=sentiment),stat = "identity")+

theme(legend.position="none")+

xlab("Sentiments")+ylab("scores")+ggtitle("Sentiments of people behind the tweets on tech as a whole")