# Sentiment Analysis Project in R

## I. Introduction

Sentiment Analysis is a text analysis process that aims at categorizing words in text data. In this project, we will perform sentiment analysis on Amazon products reviews, using Machine learning, to see how customers view them (positively, negatively, neutral).

The dataset we haave has 31000 reviews, however for the sake of reducing running time we will use 350 observation in this project.

```
## Libraries
library(xlsx)
library(tm)
library(tokenizers)
library(dplyr)
library(tidytext)
library(ggplot2)
library(devtools)
if(!require(Rstem)) install_url("http://cran.r-project.org/src/contrib/Archive/Rstem/Rstem_0.4-1.tar.gz
if(!require(sentiment)) install_url("http://cran.r-project.org/src/contrib/Archive/sentiment/sentiment_
library(Rstem)
library(sentiment)
library(SnowballC)
library(wordcloud)
library(RColorBrewer)
```

```
## Reading the data file
df <- read.xlsx("Reviews1.xlsx", sheetName = "Sheet1",header=F)
df <- df[1:350,]
str(df)
```

```
## 'data.frame':    350 obs. of  2 variables:
##  $ X1: num  1 2 3 4 5 6 7 8 9 10 ...
##  $ X2: chr  "It is good if you have internet than you can download the stuff, else, you can't" "The
```

```
#Adding Column names to the dataframe
colnames(df) = c("ID","Reviews")
```

## II. Text processing

Before we proced to sentiment analysis, we start by processing the text data by converting text to lowercase, removing punctuation and stop words and white space and letters. Finally we split the text into words through tokenization.

```
corpus = Corpus(VectorSource(df$Reviews))
inspect(corpus[3])

### Removing junk values###
for (j in seq(corpus)){
  corpus[[j]] <- gsub("&quot;", "", corpus[[j]])
  corpus[[j]] <- gsub("<p>", "", corpus[[j]])
  corpus[[j]] <- gsub("<br>", "", corpus[[j]])
  }

#text pre-processing
strwrap(corpus[[1]])
corpus=tm_map(corpus, tolower)
corpus=tm_map(corpus, removePunctuation)
corpus=tm_map(corpus, removeWords, stopwords("english"))
### Strip whitespaces ###
corpus <- tm_map(corpus, stripWhitespace)
corpus <- tm_map(corpus, removeWords, letters)
dictCorpus <- corpus
corpus=tm_map(corpus, stemDocument)
class(corpus)
inspect(corpus[3])
```

```
# concatenate tokens by document, create data frame
myDf <- data.frame(text = sapply(corpus, paste, collapse = " "), stringsAsFactors = FALSE)
```

```
text <- myDf %>%unnest_tokens(words, text, token = "words") # split into tokens
head(text)
```

```
##      words
## 1     good
## 2 internet
## 3      can
## 4 download
## 5    stuff
## 6      els
```

# III. Sentiment Analysis

## Emotions & polarity classifications

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

```
## Warning in TermDocumentMatrix.SimpleCorpus(x, control): custom functions are
## ignored
```

```
head(class_emo)
```

```
##      ANGER              DISGUST            FEAR
## [1,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
## [2,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
## [3,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
## [4,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
## [5,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
## [6,] "1.46871776464786" "3.09234031207392" "2.06783599555953"
##      JOY                SADNESS            SURPRISE           BEST_FIT
## [1,] "7.34083555412328" "1.7277074477352" "2.78695866252273" "joy"
## [2,] "1.02547755260094" "1.7277074477352" "2.78695866252273" NA
## [3,] "1.02547755260094" "1.7277074477352" "2.78695866252273" NA
## [4,] "1.02547755260094" "1.7277074477352" "2.78695866252273" NA
## [5,] "1.02547755260094" "1.7277074477352" "2.78695866252273" NA
## [6,] "1.02547755260094" "1.7277074477352" "2.78695866252273" NA
```

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

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

# classify polarity
class_pol = classify_polarity(text, algorithm="bayes")
```

```
## Warning in TermDocumentMatrix.SimpleCorpus(x, control): custom functions are
## ignored
```

```
head(class_pol)
```

```
##      POS                NEG                  POS/NEG             BEST_FIT
## [1,] "8.78232285939751" "0.445453222112551" "19.7154772340574"  "positive"
## [2,] "1.03127774142571" "0.445453222112551" "2.31512017476245"  "positive"
## [3,] "1.03127774142571" "0.445453222112551" "2.31512017476245"  "positive"
## [4,] "1.03127774142571" "0.445453222112551" "2.31512017476245"  "positive"
## [5,] "1.03127774142571" "9.47547003995745"  "0.108836578774127" "negative"
## [6,] "1.03127774142571" "0.445453222112551" "2.31512017476245"  "positive"
```

```
# 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=text, emotion=emotion, polarity=polarity, stringsAsFactors=FALSE)

head(sent_df)
```

```
##      words emotion polarity
## 1     good     joy positive
## 2 internet neutral positive
## 3      can neutral positive
## 4 download neutral positive
## 5    stuff neutral negative
## 6      els neutral positive
```

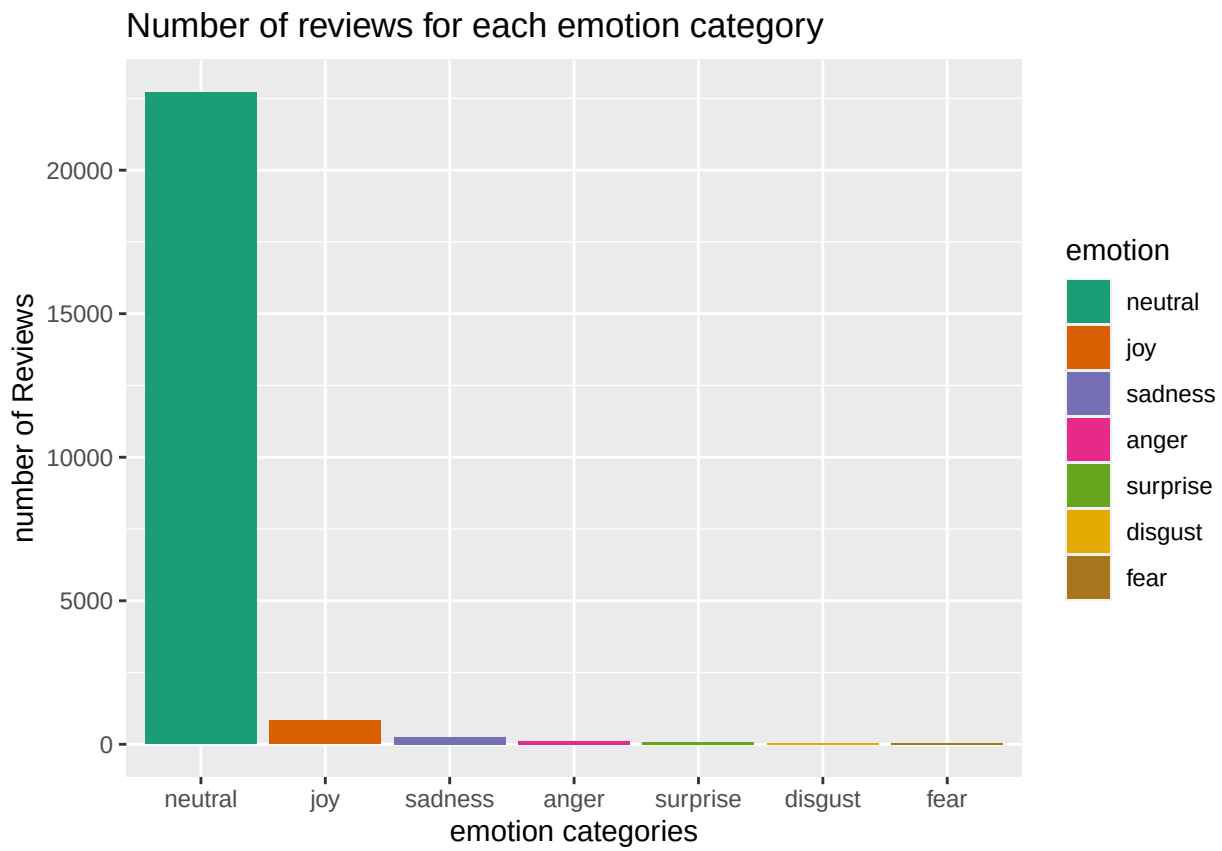
```
# sort data frame
sent_df6 = within(sent_df,emotion <- factor(emotion, levels=names(sort(table(emotion), decreasing=TRUE)))
head(sent_df6)
```

```
##      words emotion polarity
## 1     good     joy positive
## 2 internet neutral positive
## 3      can neutral positive
## 4 download neutral positive
## 5    stuff neutral negative
## 6      els neutral positive
```

At the end of this step, we got a data frame of words from customers reviews classified according to the emotion d=they express (jot, sad, angry...) and their polarity (positive, negative, neutral). In the next steps, we will vizualise the data to get more insights on customers feedback.

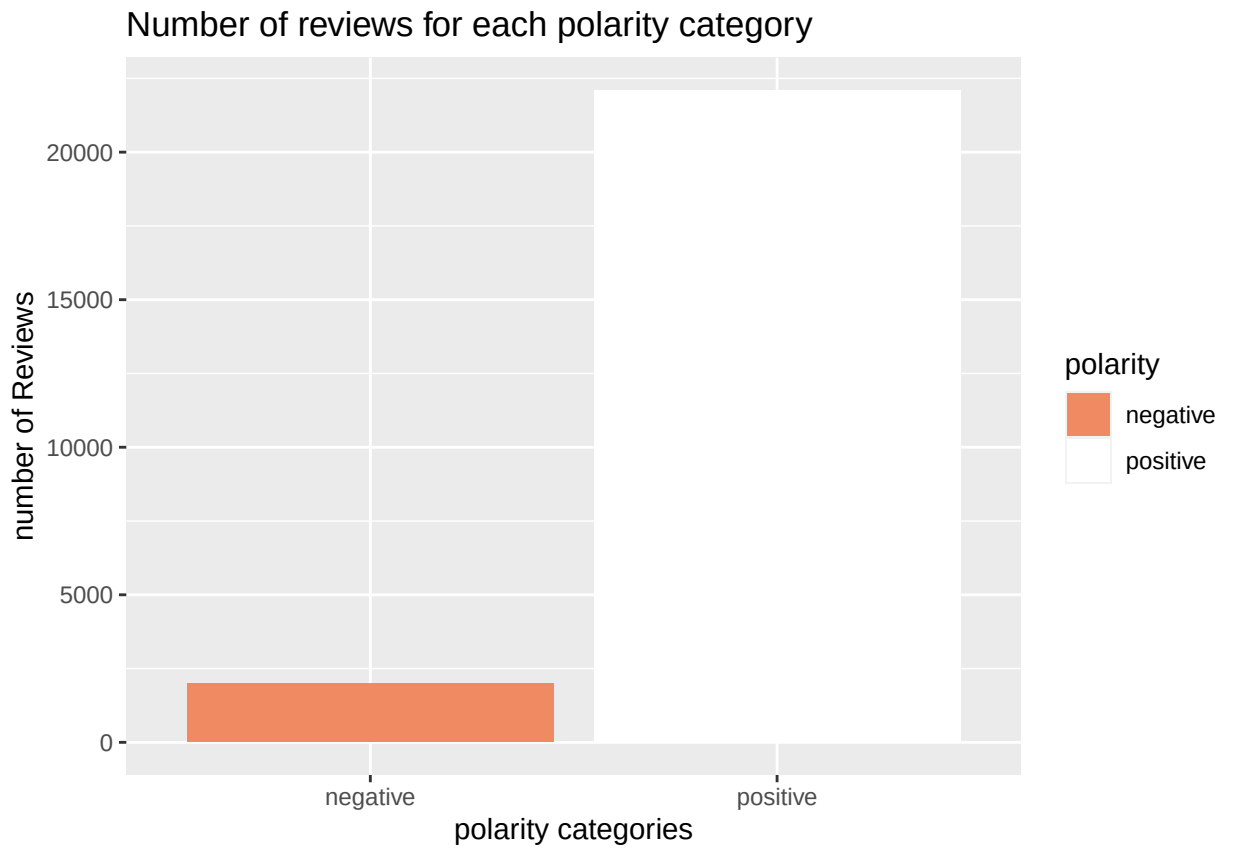
**Visualizing**

```
# plot distribution of emotions
ggplot(sent_df6, aes(x=emotion)) +
  geom_bar(aes(y=..count.., fill=emotion)) +
  scale_fill_brewer(palette="Dark2") +
  labs(x="emotion categories", y="number of Reviews",
       title="Number of reviews for each emotion category")
```



The barplot above shows that the majority of words in reviews are neutral which could be related to generic words in the reviews like computer or music, but the second most popular emotion is joy, and it seems to be more dominant compared to the other emotions, which shows that there are more happy customers than sad and angry ones.

```
# plot distribution of polarity
ggplot(sent_df6, aes(x=polarity)) +
  geom_bar(aes(y=..count.., fill=polarity)) +
  scale_fill_brewer(palette="RdGy") +
  labs(x="polarity categories", y="number of Reviews",
       title="Number of reviews for each polarity category")
```




The polarity barplot shows that there are clearly more positive reviews, more than 70%, than negative ones, which means that most customers are satisfied with the product and services.

```
#Separate the text by emotions and visualize the words
#with a comparison cloud separating text by emotion
emos = levels(factor(sent_df$emotion))
nemo = length(emos)
emos
```

```
## [1] "anger"    "disgust"  "fear"     "joy"      "neutral"  "sadness"  "surprise"
```

```
nemo
```

```
## [1] 7
```

```
emo.docs = rep("", nemo)

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

```
#table showing polarity and emotion
table(sent_df$polarity,sent_df$emotion)
```

```
##
##            anger disgust  fear   joy neutral sadness surprise
##   negative    40       3     4    49    1674     224        2
##   positive    81      45    39   782   21038      26       82
```

```
table(sent_df$emotion)
```

```
##
##    anger  disgust     fear      joy  neutral  sadness surprise
##      121       48       43      831    22712      250       84
```

```
table(sent_df$polarity)
```

```
##
## negative positive
##     1996    22093
```

```
# remove stopwords
emo.docs = removeWords(emo.docs, stopwords("english"))
# create corpus
corpus = Corpus(VectorSource(emo.docs))
#remove white spaces
corpus<- tm_map(corpus, stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation drops
## documents
```

```
#convert to lower case
corpus<- tm_map(corpus, tolower)
```

```
## Warning in tm_map.SimpleCorpus(corpus, tolower): transformation drops documents
```

```
#stop word removal
corpus<- tm_map(corpus, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeWords, stopwords("english")):
## transformation drops documents
```

```r
#apply stemming function
corpus<- tm_map(corpus, stemDocument,language = ("english"))
```

```
## Warning in tm_map.SimpleCorpus(corpus, stemDocument, language = ("english")):
## transformation drops documents
```

```r
#remove punctuation
corpus<- tm_map(corpus, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation drops
## documents
```

```r
#remove numbers
corpus<- tm_map(corpus, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(corpus, removeNumbers): transformation drops
## documents
```

```r
#command to change the class of the corpus
corpus2 <- tm_map(corpus, PlainTextDocument)
```

```
## Warning in tm_map.SimpleCorpus(corpus, PlainTextDocument): transformation drops
## documents
```

```r
corpus <- Corpus(VectorSource(corpus2))
#create Term document matrix
tdm = TermDocumentMatrix(corpus)
```

```r
#Simple word cloud
#tdm = DocumentTermMatrix(corpus)
tdm = as.matrix(tdm)
tdm = data.frame(tdm)

freq<-sort(rowSums(tdm), decreasing=TRUE)
myNames <- names(freq)
k <- which(names(freq)=="long")
myNames[k] <- "long"
d <- data.frame(word=myNames, freq)

allwords<-d


#creating a word cloud
wordcloud(d$word, d$freq, scale=c(5,0.3),min.freq=50,
          rot.per=.15,max.words=Inf,
          random.order=F,font=2,colors = brewer.pal(8, "Dark2"))
```


The word cloud gives us insights about how products features are expressed in emotions. This visualization enables us to see groups of words expressed by customers and the correspondent sentiments.