```
# Group 2 PROJECT: XBOX VS PLAYSTATION TWITTER ENVIRONMENT INSPECTION
# Members:
# Mario Cortez
# Tatiane DutraBruno
# Fangda Fan
# Aazad Ghoslya


# Note: All functions using API to get Twitter data in this file are commented after the data was obtained
# Note: All data obtained using API is then converted to .rdata or .csv file for later usage




#-------------------------------------------------------------------#
#                                                                   #
# Who are the followers and what do they tweet about?               #
#                                                                   #
#-------------------------------------------------------------------#

##### References:

# Social Media Analytics Course Material
# https://developer.twitter.com/en/docs/twitter-api
# https://cran.r-project.org/web/packages/rtweet/rtweet.pdf
# https://stackoverflow.com/questions/24829027/unimplemented-type-list-when-trying-to-write-table


#-------------------------------------------------------------------#
# Setting the environment                                           #
#-------------------------------------------------------------------#
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
#-------------------------------------------------------------------#
#                                                                   #
# Who are the followers and what do they tweet about?               #
#                                                                   #
#-------------------------------------------------------------------#

##### References:

# Social Media Analytics Course Material
# https://developer.twitter.com/en/docs/twitter-api
# https://cran.r-project.org/web/packages/rtweet/rtweet.pdf
# https://stackoverflow.com/questions/24829027/unimplemented-type-list-when-trying-to-write-table


#-------------------------------------------------------------------#
# Setting the environment                                           #
#-------------------------------------------------------------------#

# setwd("C:/Users/tdutrabruno/OneDrive - IESEG/IESEG/Social Media Analytics/Group Project")
# source("C:/Users/tdutrabruno/OneDrive - IESEG/IESEG/Social Media Analytics/Group Project/tokens.R")


oad("C:/Users/ffan1/OneDrive - IESEG (1)/Social Media Analytics/Group Project/tatimarioaazad/tati_Xbox Twitter - Followers Analysi

#-------------------------------------------------------------------#
# Downloading required libraries and tokens                         #                                   #
#-------------------------------------------------------------------#

if(!require("httr")) install.packages("httr"); library("httr")
if(!require("jsonlite")) install.packages("jsonlite"); library("jsonlite")
if(!require("data.table")) install.packages("data.table"); library("data.table")
if(!require("dplyr")) install.packages("dplyr"); library("dplyr")
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
f(!require("data.table")) install.packages("data.table"); library("data.table")
f(!require("dplyr")) install.packages("dplyr"); library("dplyr")

# Library to extract followers/user information
f(!require("academictwitteR")) install.packages("academictwitteR"); library("academictwitteR")

# Library Rtweets
f(!require("rtweet")) install.packages("rtweet"); library("rtweet")

# Libraries to text analysis
or (i in c('SnowballC','slam','tm','Matrix','tidytext','dplyr','hunspell','purrr')){
 if (!require(i, character.only=TRUE)) install.packages(i, repos = "http://cran.us.r-project.org")
 require(i, character.only=TRUE)


# Library to create word clouds
f (!require("wordcloud")) install.packages("wordcloud"); library("wordcloud")

# Library to plot graphs
ibrary(ggplot2)

# Library to execute the annotatation step
f (!require("udpipe")) install.packages("udpipe", quiet=TRUE) ; library("udpipe")
# Dowloading english model basis
ud_model <- udpipe_download_model(language = "english")
ud_model <- udpipe_load_model(ud_model$file_model)

# Create the Twitter Token to use with the library Rtweets
twitter_token <- create_token(
  app = appname,
  consumer_key = consumer_key,
  consumer_secret = consumer_secret,
  access_token = access_token,
```

Ln

```
  consumer_key = consumer_key,
  consumer_secret = consumer_secret,
  access_token = access_token,
  access_secret = access_secret,
  set_renv=FALSE)


#------------------------------------------------------------------#
# Import data from Twitter API                                     #
#------------------------------------------------------------------#

## 1. Profile to be analyzed
company <- "Xbox"

## Get the Profile ID
url_for_id <- modify_url(
  url = "https://api.twitter.com",
  path = c("2", "users","by","username",company)
)
resUser <- GET(url = url_for_id,add_headers(authorization = paste0("Bearer ",BearerToken)))
userid<- fromJSON(httr::content(resUser, "text"))

# ID Value
id <- get_user_id(company,bearer = BearerToken)

## 2. Get lists with company as a member
company_lists <- lists_memberships(company, n = 1000)

## 3. Get followers List (ID) - Library Rtweets
# For the sake of the analysis and computer limitations it will be consider 5000 users
user_followers <- get_followers(company,token=bearer_token(twitter_token))

## Transform user_followers in data frame and vector to be used in the further queries
```

```
user_followers <- get_followers(company,token=bearer_token(twitter_token))

## Transform user_followers in data frame and vector to be used in the further queries
select_followers <- user_followers[,1]
select_followers_df <- as.data.frame(select_followers)
select_followers_vt <- select_followers_df["user_id"]

## 4. Get Followers Last Tweets
followers_last_tweets <- lookup_users(select_followers_vt)

## 5. Get Followers general information (ID, Name, Username, location, description etc)
followers_data <- users_data(followers_last_tweets)

## 6. Select some customers to be analysed
## It is necessary first to filter the tweets in english, and then, get the general data from the respective user
followers_last_tweets_en <-followers_last_tweets %>%
  filter(lang == "en")
followers_data_en <- users_data(followers_last_tweets_en)

# Due to API limitations, we need to reduce the number of users to be analysed,
# Then, we will arrange the customers according the number of lists and number of statuses,
# Due to API limitations, we will slice the first 50 customers to be further analyzed, :
select_followers_en <- followers_data_en %>%
  arrange(desc(listed_count), desc(statuses_count))
select_followers_en <- followers_data_en[1:50,1]
select_followers_en_df <- as.data.frame(select_followers_en)
select_followers_en_vt <- select_followers_en_df["user_id"]

## 7. Get the tweets liked by followers
followers_likes <- get_favorites(select_followers_en_vt, n = 10)


## 8. Get the lists that the followers are members
```

File Edit Format View Help

```
#-------------------------------------------------------------#
# Save backup data in csv format                              #
#-------------------------------------------------------------#

write.csv(company_lists, "company_lists.csv")
backup_user_lists <- read.csv("company_lists.csv")

write.csv(followers_data, "followers_data.csv")
backup_user_followers_data <- read.csv("followers_data.csv")

fwrite(followers_last_tweets, file ="followers_last_tweets.csv")
backup_user_followers_last_tweets <- read.csv("followers_last_tweets.csv")

fwrite(followers_likes, file ="followers_likes.csv")
backup_user_followers_likes <- read.csv("followers_likes.csv")

fwrite(follower_lists, file ="follower_lists.csv")
backup_user_follower_lists <- read.csv("follower_lists.csv")

#-------------------------------------------------------------#
# Preprocessing datasets that will be used in the analysis    #
#-------------------------------------------------------------#

# General information about the followers (5000 followers)
followers_data <- followers_data %>%
  select(user_id, screen_name, name, location, description, url, protected, followers_count, friends_count, listed_count, statuses_cou
account_created_at, verified)

# Last 10 tweets liked by followers (50 followers)
followers_likes <- followers_likes %>%
  select (user_id, status_id,created_at, screen_name, text, reply_to_screen_name, is_quote, is_retweet, favorite_count, retweet_coun
ext_media_type, lang, quoted_text, quoted_name, quoted_statuses_count, quoted_description, quoted_verified)
```

```
followers_last_tweets <- followers_last_tweets %>%
 select(user_id, created_at, screen_name, text, reply_to_screen_name, is_retweet, favorite_count, retweet_count, hashtags, r

# Lists that has the company as member
company_lists <- company_lists %>%
 select(name, uri, subscriber_count, member_count, description)

# Lists that has the company's followers as member (50 followers)
follower_lists <- follower_lists %>%
 select(name, uri, subscriber_count, member_count, description)


#-----------------------------------------------------------------#
# Data Analysis
#-----------------------------------------------------------------#

## Correct spelling function
correct_spelling <- function(input) {
 output <- case_when(
  # any manual corrections
  input == 'license' ~ 'licence',
  # check and (if required) correct spelling
  !hunspell_check(input, dictionary('en_GB')) ~
   hunspell_suggest(input, dictionary('en_GB')) %>%
   # get first suggestion, or NA if suggestions list is empty
   map(1, .default = NA) %>%
   unlist(),
  TRUE ~ input # if word is correct
 )
 # if input incorrectly spelled but no suggestions, return input word
 ifelse(is.na(output), input, output)
}
```

```
# General Followers Information (FOLLOWERS_DATA)
#-----------------------------------------------------------------#

#*****************************************************************
# FOLLOWERS PLOT 1: Word Cloud with the Description of Followers Profile

txt_profile <- followers_data %>%
  select (user_id, description)

# 1. Remove punctuation and numbers with regular expressions
txt_profile <- mutate(txt_profile, message = gsub(x = description, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))

# 2. Annotationa
txt_profile <- udpipe_annotate(ud_model, x =txt_profile$description)
txt_profile <- as.data.frame(txt_profile)

# filtering only nouns and adjectives
txt_profile <- subset(txt_profile, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
txt_profile_DTM <- txt_profile %>% count(doc_id,lemma)
txt_profile_DTM <- txt_profile_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
txt_profile_Freq <- txt_profile_DTM %>% group_by(word) %>%
  summarize(freq = n()) %>%
  arrange(-freq)

# 5. Word Cloud
wordcloud(txt_profile_Freq$word, txt_profile_Freq$freq,
          max.words=40,
          scale=c(3,1))
```

```
wordcloud(txt_profile_Freq$word, txt_profile_Freq$freq,
          max.words=40,
          scale=c(3,1))


#*********************************************************************
# FOLLOWERS PLOT 2: Bar Graph with The distribution of the number of Followers of users (company's follower)

# Distribution Number of Followers

ggplot(followers_data, aes(x = followers_count)) +
 geom_histogram (binwidth = 100,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "followers_count", y = "frequency",
      title ="Distribution of the number of Followers of users")

followers_data %>% summarize(mean = mean(followers_count))

#*********************************************************************
# FOLLOWERS PLOT 3: Bar Graph with The distribution of the number of Friends (people he/she is following) of Users (company's

# Distribution Number of Friends

ggplot(followers_data, aes(x = friends_count)) +
 geom_histogram (binwidth = 100,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "friends_count", y = "frequency",
      title ="Distribution of the number of Friends of users")
```

```
 labs(x = "friends_count", y = "frequency",
      title ="Distribution of the number of Friends of users")

followers_data %>% summarize(mean = mean(friends_count))
#**********************************************************************
# FOLLOWERS PLOT 4: Bar Graph with The distribution of the number of lists the users are member

ggplot(followers_data, aes(x = listed_count)) +
 geom_histogram (binwidth = 5,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "listed_count", y = "frequency",
      title ="Distribution of the number of lists the users are member")

followers_data %>% summarize(mean = mean(listed_count))
#**********************************************************************
# FOLLOWERS PLOT 5: Bar Graph with The distribution of the number of users' tweets

list_statutes_qnt <- followers_data %>%
 filter(statuses_count < 30000)

ggplot(dist_statutes_qnt, aes(x = statuses_count)) +
 geom_histogram (binwidth = 100,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "statuses_count", y = "frequency",
      title ="Distribution of the number of users' tweets")

#**********************************************************************
# FOLLOWERS PLOT 6: Bar Graph with The number of verified users
```

```
#*********************************************************************
# FOLLOWERS PLOT 6: Bar Graph with The number of verified users

dist_verified <- followers_data %>%
 group_by(verified) %>%
 summarize(ncust = n())

ggplot(dist_verified, aes(x = verified, y = ncust)) +
 geom_bar(stat = "identity", na.rm = TRUE, fill="#69b3a2") +
 theme_light()+
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
 labs(x = "verified", y = "Frequency",
     title ="Number of verified users")


#*********************************************************************
# FOLLOWERS PLOT 7: Word Cloud with The location of users

location <- termFreq(followers_data$location)
wordcloud(names(location),location,
        max.words=40,
        scale=c(3,1))



#-------------------------------------------------------------------#
# Tweets liked by company's Followers (FOLLOWERS_LIKES)
#-------------------------------------------------------------------#

#*********************************************************************
# FOLLOWERS PLOT 8: Word Cloud with tweets liked by users

# Analysis likes
```

```
#-------------------------------------------------------------------------#
# Tweets liked by company's Followers (FOLLOWERS_LIKES)
#-------------------------------------------------------------------------#

#*************************************************************************
# FOLLOWERS PLOT 8: Word Cloud with tweets liked by users

# Analysis likes
txt_likes <- followers_likes %>%
 filter (lang=="en") %>%
 select (status_id, text)

# 1. Remove punctuation and numbers with regular expressions
txt_likes <- mutate(txt_likes, message = gsub(x = text, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))

# 2. Annotation

txt_likes <- udpipe_annotate(ud_model, x =txt_likes$text)
txt_likes <- as.data.frame(txt_likes)

# filtering only nouns and adjectives
txt_likes <- subset(txt_likes, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
txt_likes_DTM <- txt_likes %>% count(doc_id,lemma)
txt_likes_DTM <- txt_likes_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
likes_Freq <- txt_likes_DTM %>% group_by(word) %>%
  summarize(freq = n()) %>%
  arrange(-freq)
```

```
 summarize(freq = n()) %>%
 arrange(-freq)

# 5. Word Cloud
wordcloud(likes_Freq$word, likes_Freq$freq,
        max.words=40,
        scale=c(3,1))

#*********************************************************************
# FOLLOWERS PLOT 9: Word Cloud with the hashtags of tweets liked by users

txt_tags <- followers_likes %>%
 filter (!is.na(hashtags)) %>%
 select (status_id, hashtags)

txt_tags <- unlist(txt_tags$hashtags)

txt_tags <- as.data.frame(txt_tags)

txt_tags <- termFreq(txt_tags$txt_tags)
wordcloud(names(txt_tags),txt_tags,
        max.words=40,
        scale=c(3,1))

#*********************************************************************
# FOLLOWERS PLOT 10: Bar Graph with the number of tweets liked by users that were a quote

dist_quote <- followers_likes %>%
 group_by(is_quote) %>%
 summarize(ncust = n())

ggplot(dist_quote, aes(x = is_quote, y = ncust)) +
 geom_bar(stat = "identity", na.rm = TRUE, fill="#69b3a2") +
```

```
      title ="Retweet tweets liked by users")

#*********************************************************************
# FOLLOWERS PLOT 12: Bar Graph with the number of tweets liked by users and different types of media type they used

dist_media <- followers_likes %>%
 group_by(ext_media_type) %>%
 summarize(ncust = n())

ggplot(dist_media, aes(x = ext_media_type, y = ncust)) +
 geom_bar(stat = "identity", na.rm = TRUE, fill="#69b3a2") +
 theme_light()+
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
 labs(x = "ext_media_type", y = "Frequency",
      title ="Media type of tweets liked by users")


#-------------------------------------------------------------------#
# Analyzing Followers' tweets (FOLLOWERS_LAST_TWEETS)
#-------------------------------------------------------------------#

#*********************************************************************
# FOLLOWERS PLOT 13: Word Cloud with the last tweet of users

txt_last_tweets <- followers_last_tweets %>%
 filter (lang=="en") %>%
 select (user_id, text)

# 1. Remove punctuation and numbers with regular expressions
txt_last_tweets <- mutate(txt_last_tweets, message = gsub(x = text, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))

# 2. Annotation
```

```
# filtering only nouns and adjectives
txt_last_tweets <- subset(txt_last_tweets, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
txt_last_tweets_DTM <- txt_last_tweets %>% count(doc_id,lemma)
txt_last_tweets_DTM <- txt_last_tweets_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
txt_last_tweets_Freq <- txt_last_tweets_DTM %>% group_by(word) %>%
 summarize(freq = n()) %>%
 arrange(-freq)

# 5. Word Cloud
wordcloud(txt_last_tweets_Freq$word, txt_last_tweets_Freq$freq,
        max.words=40,
        scale=c(3,1),
        min.freq = 2)

#*********************************************************************
# FOLLOWERS PLOT 14: Percentage of Users who never tweeted

followers_active <- (sum(is.na(followers_last_tweets$text))/5000)*100
followers_active

#*********************************************************************
# FOLLOWERS PLOT 15: Word Cloud with the hashtags of the last tweet of users

txt_tweet_tags <- followers_last_tweets %>%
 filter (lang=="en") %>%
 filter (!is.na(hashtags)) %>%
 select (user_id, hashtags)
```

```
txt_tweet_tags <- as.data.frame(txt_tweet_tags)

txt_tweet_tags <- termFreq(txt_tweet_tags$txt_tweet_tags)
wordcloud(names(txt_tweet_tags),txt_tweet_tags,
          max.words=40,
          scale=c(3,1),
          min.freq = 2)


#*********************************************************************
# FOLLOWERS PLOT 16: Bar Graph with the number of last users' tweets that were a retweet

dist_foll_retweet <- followers_last_tweets %>%
  group_by(is_retweet) %>%
  summarize(ncust = n())

ggplot(dist_foll_retweet, aes(x = is_retweet, y = ncust)) +
  geom_bar(stat = "identity", na.rm = TRUE, fill="#69b3a2") +
  theme_light()+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(x = "is_retweet", y = "Frequency",
       title ="Users' last tweets - Retweet or not")

#---------------------------------------------------------------------#
# Analyzing the lists that have company as a member (COMPANY_LISTS)
#---------------------------------------------------------------------#

#*********************************************************************
# FOLLOWERS PLOT 17: Word Cloud with the description of the lists that the company is a member

txt_list <- company_lists %>%
  select (uri, description)
```

```
txt_list <- as.data.frame(txt_list)

# filtering only nouns and adjectives
txt_list <- subset(txt_list, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
txt_list_DTM <- txt_list %>% count(doc_id,lemma)
txt_list_DTM <- txt_list_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
txt_list_Freq <- txt_list_DTM %>% group_by(word) %>%
 summarize(freq = n()) %>%
 arrange(-freq)

# 5. Word Cloud
wordcloud(txt_list_Freq$word, txt_list_Freq$freq,
        max.words=40,
        scale=c(3,1))


#**********************************************************************
# FOLLOWERS PLOT 18: Bar Graph with the number of subscription of the lists that the company is a member

# Number of Subscription per lists

ggplot(company_lists, aes(x = subscriber_count)) +
 geom_histogram (binwidth = 10,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "subscriber_count", y = "frequency",
     title ="Number of subscribtions per lists")
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
ggplot(company_lists, aes(x = subscriber_count)) +
  geom_histogram (binwidth = 10,
                  center = 0.05,
                  fill="#69b3a2") +
  theme_light() +
  labs(x = "subscriber_count", y = "frequency",
       title ="Number of subscribtions per lists")

company_lists %>% summarize(mean = mean(subscriber_count))
#*********************************************************************
# FOLLOWERS PLOT 19: Word Cloud with the number of subscription of the lists that the company is a member

# Word cloud with list names

name <- termFreq(company_lists$name)
wordcloud(names(name),name,
          max.words=40,
          scale=c(3,1))

#-----------------------------------------------------------------------#
# Analyzing the lists that have company's followers as a member (FOLLOWERS_LISTS)
#-----------------------------------------------------------------------#

#*********************************************************************
# FOLLOWERS PLOT 20: Word Cloud with the description of the lists that the users are members

txt_list <- follower_lists %>%
  select (uri, description)

# 1. Remove punctuation and numbers with regular expressions
txt_list <- mutate(txt_list, message = gsub(x = description, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
name <- termFreq(company_lists$name)
wordcloud(names(name),name,
          max.words=40,
          scale=c(3,1))

#-----------------------------------------------------------------#
# Analyzing the lists that have company's followers as a member (FOLLOWERS_LISTS)
#-----------------------------------------------------------------#

#*****************************************************************
# FOLLOWERS PLOT 20: Word Cloud with the description of the lists that the users are members

xt_list <- follower_lists %>%
select (uri, description)

# 1. Remove punctuation and numbers with regular expressions
t_list <- mutate(txt_list, message = gsub(x = description, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))

# 2. Annotation

t_list <- udpipe_annotate(ud_model, x =txt_list$description)
t_list <- as.data.frame(txt_list)

Filtering only nouns and adjectives
t_list <- subset(txt_list, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
Get the number of times a word occurred in each document (or status, tweet)
t_list_DTM <- txt_list %>% count(doc_id,lemma)
t_list_DTM <- txt_list_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
t_list_Freq <- txt_list_DTM %>% group_by(word) %>%
```

```
# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
txt_list_DTM <- txt_list %>% count(doc_id,lemma)
txt_list_DTM <- txt_list_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
txt_list_Freq <- txt_list_DTM %>% group_by(word) %>%
 summarize(freq = n()) %>%
 arrange(-freq)

# 5. Word Cloud
wordcloud(txt_list_Freq$word, txt_list_Freq$freq,
        max.words=40,
        scale=c(3,1),
        min.freq = 1)


#**********************************************************************
# FOLLOWERS PLOT 21: Bar Graph with the number of subscription of the lists that the users are members

ggplot(follower_lists, aes(x = subscriber_count)) +
 geom_histogram (binwidth = 10,
              center = 0.05,
              fill="#69b3a2") +
 theme_light() +
 labs(x = "subscriber_count", y = "frequency",
     title ="Number of subscribtions per lists (Followers List")

#**********************************************************************
# FOLLOWERS PLOT 22: Word Cloud with the number of subscription of the lists that the users are members

name <- termFreq(follower_lists$name)
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
ggplot(follower_lists, aes(x = subscriber_count)) +
 geom_histogram (binwidth = 10,
                 center = 0.05,
                 fill="#69b3a2") +
 theme_light() +
 labs(x = "subscriber_count", y = "frequency",
     title ="Number of subscribtions per lists (Followers List")

#*****************************************************************
# FOLLOWERS PLOT 22: Word Cloud with the number of subscription of the lists that the users are members

name <- termFreq(follower_lists$name)
wordcloud(names(name),name,
          max.words=40,
          scale=c(3,1),
          min.freq = 2)

#-----------------------------------------------------------------#
#                                                                 #
#                    Xbox Own Tweets                              #
#                                                                 #
#-----------------------------------------------------------------#

library(rtweet)

Xbox <- read.csv(file = "Xbox.csv")
library(data.table)

# Xbox <- get_timeline("Xbox", n=3200)

glimpse(Xbox)
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
#                    Xbox Own Tweets                          #
#                                                  #
#-----------------------------------------------------------#

library(rtweet)

Xbox <- read.csv(file = "Xbox.csv")
library(data.table)

# Xbox <- get_timeline("Xbox", n=3200)

glimpse(Xbox)

Xbox <- Xbox %>% filter(lang == "en" )

sapply(Xbox, function(x) length(unique(x)))
#create NA marker for Xbox retweets
#0 when value is missing and 1 when it exists
Xbox$reply_to_user_id<-ifelse(is.na(Xbox$reply_to_user_id), 0, 1)
XboxOriginals <- Xbox[Xbox$reply_to_user_id == 0,]
XboxReplies <- Xbox[Xbox$reply_to_user_id == 1,]
#make sure the originals and replies table are correct
sapply(XboxReplies, function(x) length(unique(x)))

# load some packages that we will use
for (i in c('SnowballC','slam','tm','Matrix','tidytext','dplyr','hunspell','purrr')){
  if (!require(i, character.only=TRUE)) install.packages(i, repos = "http://cran.us.r-project.org")
  require(i, character.only=TRUE)
}

# Library to execute the annotatation step
if (!require("udpipe")) install.packages("udpipe", quiet=TRUE) ; library("udpipe")
# Dowloading english model basis
```

Ln 77, Col 37 100% Wi

ProjectCode (1) - Notepad

File Edit Format View Help

```
# Library to execute the annotatsation step
if (!require("udpipe")) install.packages("udpipe", quiet=TRUE) ; library("udpipe")
# Dowloading english model basis
ud_model <- udpipe_download_model(language = "english")
ud_model <- udpipe_load_model(ud_model$file_model)

##### Analysing text of tweets liked by followers

# Analysis likes
tatiXboxReplies <- XboxReplies %>%
 select (status_id, text)

# 1. Remove punctuation and numbers with regular expressions
tatiXboxReplies <- mutate(tatiXboxReplies, message = gsub(x = text, pattern = "[0-9]+ | [[:punct:]] | \\(.*\\)", replacement = ""))

# 2. Annotation

tatiXboxReplies <- udpipe_annotate(ud_model, x =tatiXboxReplies$text)
tatiXboxReplies <- as.data.frame(tatiXboxReplies)

# filtering only nouns and adjectives
tatiXboxReplies <- subset(tatiXboxReplies, upos %in% c("NOUN","ADJ"))

# 3. Create the document-term matrix
# Get the number of times a word occurred in each document (or status, tweet)
tatiXboxReplies_DTM <- tatiXboxReplies %>% count(doc_id,lemma)
tatiXboxReplies_DTM <- tatiXboxReplies_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
likes_Freq <- tatiXboxReplies_DTM %>% group_by(word) %>%
 summarize(freq = n()) %>%
 arrange(-freq)
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
tatiXboxReplies_DTM <- tatiXboxReplies %>% count(doc_id,lemma)
tatiXboxReplies_DTM <- tatiXboxReplies_DTM %>% rename(word = lemma)

# 4. Investigate the most frequent terms
likes_Freq <- tatiXboxReplies_DTM %>% group_by(word) %>%
  summarize(freq = n()) %>%
  arrange(-freq)

# 5. Word Cloud
wordcloud(likes_Freq$word, likes_Freq$freq,
          max.words=40,
          scale=c(3,1))


###
# Preprocessing with tidytext
###

# 1. Remove punctuation and numbers with regular expressions
XboxReplies <- mutate(XboxReplies, text = gsub(x = text, pattern = "[0-9]+|[[:punct:]]|\\(.*\\)", replacement = ""))

# 2. Tokenization (+ going to lowercase)
XboxTokenized <- XboxReplies %>% unnest_tokens(output = "word", # how should the new column be named?
                                               input = text, # where can we find the text?
                                               token = "words", # which tokenization scheme should we follow?
                                               drop=FALSE,to_lower=TRUE) # drop=FALSE specifies that we want to keep our text; to_lower puts everyting to lowerc
# 3. Remove some other elements such as # and @ signs if they might occur
XboxTokenized <- filter(XboxTokenized, substr(word, 1, 1) != '#',
                        substr(word, 1, 1) != '@') # This compares for the first letter of a token# omit hashtags


# 5. remove stopwords
```

Ln 77, Col 37

```
# 5. remove stopwords

XboxTokenized <- XboxTokenized %>% anti_join(get_stopwords())

# 7. Create the document term matrix

# first, we need to get the number of times a word occurred in each document (or status, tweet)

XboxTokenized <- XboxTokenized %>% count(status_id,word)

# then, we could perform weighting (e.g., tfidf) using the bind_tf_idf(word,id,n) function
# however, we will integrate this directly when making the document term matrix

XboxDTM <- XboxTokenized %>% cast_dtm(status_id,word,n,weighting = tm::weightTfIdf)

######
### 8. inspect our text
######

# 1. we can look at associations/correlations between words (this is with the dtm):
findAssocs(XboxDTM, terms = "Xbox", corlimit = 0.1)


# 2. investigate the most frequent terms

XboxFreq <- XboxTokenized %>% group_by(word) %>% # for this, we need to have the sum over all documents
  summarize(freq = n()) %>%
  arrange(-freq)              # arrange = order, from most frequent term to lowest frequent
head(XboxFreq)

# 3. We can also build a wordcloud in order to give this insight visually
# what is the basis for a wordcloud? Term frequencies
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
XboxFreq <- XboxTokenized %>% group_by(word) %>% # for this, we need to have the sum over all documents
  summarize(freq = n()) %>%
  arrange(-freq)              # arrange = order; from most frequent term to lowest frequent
head(XboxFreq)

# 3. We can also build a wordcloud in order to give this insight visually
# what is the basis for a wordcloud? Term frequencies

# Load the package wordcloud
if (!require("wordcloud")) install.packages("wordcloud"); library("wordcloud")


# 1. Word cloud based on the original text
#
# use the termFreq of the tm package
# This also uses a tokenizer inside
tf <- termFreq(XboxReplies$text)
wordcloud(names(tf),tf,
          max.words=40,
          scale=c(3,1))

# 2. Word cloud based on the tibble and all text pre-processing

#create word cloud
wordcloud(XboxFreq$word, XboxFreq$freq,
          max.words=40,
          scale=c(3,1))

###### explore all other methods in session 2 ######
## bigrams ##

# Implementing bigrams
```

Ln 77, Col 37 100% Windows (CRLF)

ProjectCode (1) - Notepad

File Edit Format View Help

```
                                input = text,
                                token = "ngrams",n=2, drop=FALSE) %>%
 count(status_id,bigram)
XboxBigramDTM <- XboxBigramCount  %>% cast_dtm(status_id,bigram,n)

# make a wordcloud

XboxBiCount <- XboxBigramCount %>% group_by(bigram) %>% summarize(freq = n())
wordcloud(XboxBiCount$bigram,XboxBiCount$freq,max.words = 40)

# removing stopwords with bigrams
# two approaches - first remove stopwords
#                - first create bigram, and then look whether they contain stop words

ibrary(tidyr)
# remove the interesting words for negations
sw <- get_stopwords() %>% filter(!word %in% c("no","not","nor","in","the","to","for","a","on"))

bigrams_nosw <- XboxReplies %>% unnest_tokens(output = "bigram",
                                input = text,
                                token = "ngrams",n=2, drop=FALSE) %>%
 separate(bigram, c("word1", "word2"), sep = " ") %>% # Split bigram into 2
 filter(!word1 %in% sw$word) %>% # Check for each of the words whether it is a stopword
 filter(!word2 %in% sw$word) %>%  # if they are, delete
 unite(bigram, word1, word2, sep = " ")   %>% # unite the words again
 count(status_id,bigram)

XboxBigram_noswDTM <- bigrams_nosw  %>% cast_dtm(status_id,bigram,n)

bigrams4 <- XboxBigram_noswDTM$dimnames$Terms
#####
XboxbigramFreq <- bigrams_nosw %>% group_by(bigram) %>% # for this, we need to have the sum over all documents
 summarize(freq = n()) %>%
```

Ln 77, Col 17 100% Windows (CRLF)

ProjectCode (1) - Notepad

File Edit Format View Help

```
######

df <- as.data.frame(matrix(bigrams4, ncol = 1, byrow = TRUE))

df1 <- df %>%
 separate(V1, c("word1", "word2","word3","word4"), sep = " ")

df1 <- cbind(stack(df1[1:4]))

df1 <- df1 %>%
 group_by(values) %>%
 count(values) %>%
 arrange(n)

# make a wordcloud

XboxBiCount_nosw <- bigrams_nosw %>% group_by(bigram) %>% summarize(freq = n())
wordcloud(XboxBiCount_nosw$bigram,XboxBiCount_nosw$freq,max.words = 100)



########## sentiment analysis begins here ################################

library("textdata")

XboxSentiment <- inner_join(XboxTokenized,get_sentiments("bing"))

library("ggplot2")

# 3.1 get the most positive/negative words

summarySentiment <- XboxSentiment %>% count(word,sentiment,sort=TRUE) %>%
 group_by(sentiment) %>%
```

```
[illegible] <- [illegible] get_sentiments("bing"))

library("ggplot2")

# 2.1 get the most positive/negative words

summarySentiment <- [illegible] %>% count(word, sentiment, sort = TRUE) %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  arrange(n) %>%
  as.data.frame(stringsAsFactors = FALSE)

summarySentiment %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
    x = NULL) +
  coord_flip()

# 2.2 get a summary per post
library(tidyr)
statusSentiment <- [illegible] %>%
  count(status_id, sentiment) %>%           # count the positives and negatives per id (status)
  spread(sentiment, n, fill = 0) %>%     # we want overall sentiment, and here we have both (positive and negative), making it difficult to directly [illegible]
sentiment
  mutate(sentiment = positive - negative)

statusSentiment %>%

  [illegible] = mean(sentiment))
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
  count(status_id, sentiment) %>%          # count the positives and negatives per id (status)
  spread(sentiment, n, fill = 0) %>%     # we want overall sentiment, and here we have text (positive and negative), making it difficult to
sentiment.
  mutate(sentiment = positive - negative)

statusSentiment %>%

  summarize(mean = mean(sentiment))

# until here we did this with the bing dictionary
# do this with the afinn dictionary:
statusSentiment <- inner_join(XboxTokenized,get_sentiments("afinn")) %>%
  group_by(status_id) %>%
  summarize(Sentiment = sum(value))

mean(statusSentiment$Sentiment)
# mean sentiment 2.397 with AFINN (-5 to 5)

statusSentiment <- XboxReplies %>% left_join(statusSentiment,by="status_id") %>%
  mutate(Sentiment = ifelse(is.na(Sentiment),0,Sentiment))


# plot this over time

time  <- as.POSIXct(statusSentiment[,3], format="%Y-%m-%dT%H:%M:%OS",tz="UTC")

# order the time (normally already sorted, just to be sure)
sentiment <- statusSentiment[order(time),"Sentiment"]
lim <- max(abs(sentiment))

#Plot sentiment by time
plot(1:length(sentiment),
     sentiment,
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
# order the time (normally already sorted, just to be sure)
sentiment <- statusSentiment[order(time),"Sentiment"]
lim <- max(abs(sentiment))

#Plot sentiment by time
plot(1:length(sentiment),
    sentiment,
    xaxt="n",
    type="l",
    ylab="Valence",
    xlab="Time (days:hour:minute)",
    main="Sentiment",
    ylim=c(-lim,lim))
abline(h = 0, col = "red", lty = 3)

axis(1,at=1:length(sentiment),
    labels=paste0(substr(time[order(time)],1,10),"\n",substr(time[order(time)],12,16)))

# near 2021-11-11, 2021-12-05 something bad happened, Xbox responses had negative sentiment

library(tidytext)

# Searching for the tweets with  #Xbox
# Xbox_tweets <- search_tweets(q = "#Xbox" , n=1000)

Xbox_tweets <- as.data.frame(Xbox_tweets)
fwrite(Xbox_tweets, file ="Xbox_tweets.csv")
Xbox_tweets <- read.csv(file = "Xbox_tweets.csv")

#Checking for the top 5 tweets
head(Xbox_tweets , n = 5)
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
Xbox_tweets <- as.data.frame(Xbox_tweets)
fwrite(Xbox_tweets, file ="Xbox_tweets.csv")
Xbox_tweets <- read.csv(file = "Xbox_tweets.csv")

#Checking for the top 5 tweets
head(Xbox_tweets , n = 5)

#Getting the users tweeting for Xbox

Users <- Xbox_tweets$screen_name

#Getting the unique users tweeting for Xbox

Users <- unique ( Xbox_tweets$screen_name )

#Users tweets reagrding Xbox

Users_tweets <- search_users("#Xbox", n = 1000)

# Getting the data about the users

#Location
length(unique(Users_tweets$location))

#Remove the null values and get the visualization of the users
Users_tweets %>%
  count(location) %>%
  mutate(location = reorder(location, n)) %>%
  na.omit() %>%
  top_n(10) %>%
  ggplot(aes(x = location, y = n)) +
  geom_col() +
```

Ln 77, Col 37

```
 top_n(10) %>%
 ggplot(aes(x = location, y = n)) +
 geom_col() +
 coord_flip() +
 labs(x = "Count",
      y = "Location",
      title = "Location of the users")

# Source
length(unique(Users_tweets$source))

Users_tweets %>%
 count(source) %>%
 mutate(source = reorder(source, n)) %>%
 top_n(10) %>%
 ggplot(aes(x = source, y = n)) +
 geom_col() +
 coord_flip() +
 labs(x = "Count",
      y = "Source",
      title = "Source used by the users for tweet")



# Tweets from Xbox
# Xbox <- get_timelines("Xbox",n=3500)
# Xbox

# Xbox <- as.data.frame(Xbox)
# fwrite(Xbox, file ="Xbox.csv")
Xbox <- read.csv(file = "Xbox.csv")
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
    title = "Source used by the users for tweet")



# Tweets from Xbox
# Xbox <- get_timelines("Xbox",n=3500)
# Xbox

# Xbox <- as.data.frame(Xbox)
# fwrite(Xbox, file ="Xbox.csv")
Xbox <- read.csv(file = "Xbox.csv")

#Checking for the top 5 tweets
head(Xbox , n = 5)


#getting the retweets
Retweets_Xbox <- Xbox[Xbox$is_retweet == TRUE, ]

# Counting the retweets
count(Retweets_Xbox)

#Getting the replies
Replies_Xbox <- subset(Xbox, !is.na(Xbox$reply_to_status_id))

# Counting the retweets
count(Replies_Xbox)

# Removing the retweets and replies and getting the tweets only by xbox
# Remove retweets
From_Xbox <- Xbox[Xbox$is_retweet==FALSE, ]
```

Ln 77, Col 37

ProjectCode (1) - Notepad

File Edit Format View Help

```
# Counting the retweets
count(Retweets_Xbox)

#Getting the replies
Replies_Xbox <- subset(Xbox, !is.na(Xbox$reply_to_status_id))

# Counting the retweets
count(Replies_Xbox)

# Removing the retweets and replies and getting the tweets only by xbox
# Remove retweets
From_Xbox <- Xbox[Xbox$is_retweet==FALSE, ]

# Remove replies
From_Xbox <- subset(From_Xbox, is.na(From_Xbox$reply_to_status_id))

# Counting the Xbox tweets
count(From_Xbox)

count(Xbox)

From_Xbox <- From_Xbox %>% arrange(-favorite_count)
From_Xbox[1,5]

From_Xbox <- From_Xbox %>% arrange(-retweet_count)
From_Xbox[1,5]


# Creating a data frame
data <- data.frame(
  category=c("Orignal", "Retweets", "Replies"),
  count=c(2828, 198, 135)
)
```

```
From_Xbox[1,5]


# Creating a data frame
data <- data.frame(
  category=c("Orignal", "Retweets", "Replies"),
  count=c(2828, 198, 135)
)

# Adding columns
data$fraction = data$count / sum(data$count)
data$percentage = data$count / sum(data$count) * 100
data$ymax = cumsum(data$fraction)
data$ymin = c(0, head(data$ymax, n=-1))

# Rounding the data to two decimal points
install.packages("forestmangr")
library(forestmangr)
data <- round_df(data, 2)

# Showing the data of different types
Type_of_Tweet <- paste(data$category, data$percentage, "%")
ggplot(data, aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=Type_of_Tweet)) +
  geom_rect() +
  coord_polar(theta="y") +
  xlim(c(2, 4)) +
  theme_void() +
  theme(legend.position = "left")

#Cleaning the data

From_Xbox$text <-  gsub("https\\S*", "", From_Xbox$text)
From_Xbox$text <-  gsub("@\\S*", "", From_Xbox$text)
```

File Edit Format View Help

```
  xlim(c(2, 4)) +
  theme_void() +
  theme(legend.position = "left")

#Cleaning the data

From_Xbox$text <- gsub("https\\S*", "", From_Xbox$text)
From_Xbox$text <- gsub("@\\S*", "", From_Xbox$text)
From_Xbox$text <- gsub("amp", "", From_Xbox$text)
From_Xbox$text <- gsub("[\r\n]", "", From_Xbox$text)
From_Xbox$text <- gsub("[[:punct:]]", "", From_Xbox$text)


#Tokenization

tweets_Xbox <- From_Xbox %>%
  select(text) %>%
  unnest_tokens(word, text)

#Remove Stopwords
tweets_Xbox <- tweets_Xbox %>%
  anti_join(stop_words)


# gives you a bar chart of the most frequent words found in the tweets

tweets_Xbox %>%
  count(word, sort = TRUE) %>%
  top_n(10) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(x = word, y = n)) +
  geom_col() +
  xlab(NULL) +
```

```
tweets_Xbox %>%
 count(word, sort = TRUE) %>%
 top_n(10) %>%
 mutate(word = reorder(word, n)) %>%
 ggplot(aes(x = word, y = n)) +
 geom_col() +
 xlab(NULL) +
 #coord_flip() +
 labs(y = "Count",
      x = "Word",
      title = "Most frequent words ")

#WordCloud

if (!require("wordcloud")) install.packages("wordcloud"); library("wordcloud")

# Word Cloud for #Xbox
wordcloud(From_Xbox$hashtags, min.freq=2, scale=c(1, 1), random.order=FALSE,
          colors=brewer.pal(8, "Set1"))

# Word Cloud for re-tweets
wordcloud(Retweets_Xbox$retweet_screen_name, min.freq=3, scale=c(1, 1), random.order=FALSE,
          colors=brewer.pal(8, "Set1"))


#Sentiment Analysis

install.packages("syuzhet")
library(syuzhet)

# Converting tweets to ASCII to tackle strange characters
tweets <- iconv(tweets, from="UTF-8", to="ASCII", sub="")
```

```
wordcloud(Retweets_Xbox$retweet_screen_name, min.freq=3, scale=c(1, 1), random.order=FALSE,
        colors=brewer.pal(8, "Set1"))


#Sentiment Analysis

install.packages("syuzhet")
library(syuzhet)

# Converting tweets to ASCII to tackle strange characters
tweets <- iconv(tweets, from="UTF-8", to="ASCII", sub="")

# removing retweets, in case needed
tweets <-gsub("(RT|via)((?:\\b\\w*@\\w+)+)","",tweets)

# removing mentions, in case needed
tweets <-gsub("@\\w+","",tweets)

ew_sentiment<-get_nrc_sentiment((tweets))

sentimentscores<-data.frame(colSums(ew_sentiment[,]))

names(sentimentscores) <- "Score"
sentimentscores <- cbind("sentiment"=rownames(sentimentscores),sentimentscores)
rownames(sentimentscores) <- NULL

ggplot(data=sentimentscores,aes(x=sentiment,y=Score))+
  geom_bar(aes(fill=sentiment),stat = "identity")+
  theme(legend.position="none")+
  xlab("Sentiments")+ylab("Scores")+
  ggtitle("Total sentiment based on scores")+
```

ProjectCode (1) - Notepad

File Edit Format View Help

```
tweets <-gsub("(RT | via)((?:\\b\\w*@\\w+)+)","",tweets)

# removing mentions, in case needed
tweets <-gsub("@\\w+","",tweets)

ew_sentiment<-get_nrc_sentiment((tweets))

sentimentscores<-data.frame(colSums(ew_sentiment[,]))

names(sentimentscores) <- "Score"
sentimentscores <- cbind("sentiment"=rownames(sentimentscores),sentimentscores)
rownames(sentimentscores) <- NULL

ggplot(data=sentimentscores,aes(x=sentiment,y=Score))+
  geom_bar(aes(fill=sentiment),stat = "identity")+
  theme(legend.position="none")+
  xlab("Sentiments")+ylab("Scores")+
  ggtitle("Total sentiment based on scores")+
  theme_minimal()

#-----------------------------------------------------------------#
#                                                                 #
#                  Xbox vs PlayStation                            #
#                                                                 #
#-----------------------------------------------------------------#


if(!require("httr")) install.packages("httr"); library("httr")
if(!require("jsonlite")) install.packages("jsonlite"); library("jsonlite")
if(!require("data.table")) install.packages("data.table"); library("data.table")
if(!require("dplyr")) install.packages("dplyr"); library("dplyr")
library("readr")
library("tidytext")
```

File Edit Format View Help

```
#-------------------------------------------------------------#
#                                                         #
#                  Xbox vs PlayStation                        #
#                                                         #
#-------------------------------------------------------------#


if(!require("httr")) install.packages("httr"); library("httr")
if(!require("jsonlite")) install.packages("jsonlite"); library("jsonlite")
if(!require("data.table")) install.packages("data.table"); library("data.table")
if(!require("dplyr")) install.packages("dplyr"); library("dplyr")
library("readr")
library("tidytext")
library("RTwitterV2")
if (!require("ggplot2")) install.packages("ggplot2", quiet=TRUE) ; library("ggplot2")
library("wordcloud")
if(!require("syuzhet")) install.packages("syuzhet"); library("syuzhet")
if(!require("Unicode")) install.packages("Unicode"); library("Unicode")
if(!require("tidyverse")) install.packages("tidyverse"); library("tidyverse")
if(!require("devtools")) install.packages("devtools"); library("devtools")
devtools::install_github("MaelKubli/RTwitterV2")
library(RTwitterV2)
if(!require("tm")) install.packages("tm"); library("tm")

#Getting the tweets with #Xbox and #Playstation

##########################################
#       Reading the data          #
##########################################

setwd("C:/Users/[illegible]/Documents/GitHub/SMA Yes We Can")

Xbox_25Jan_200tweets <- read.csv("C:/Users/[illegible]/Documents/GitHub/SMA Yes We Can/Xbox_25Jan_200tweets.csv")
```

File Edit Format View Help

```
  ggtitle("Top 5 Most Liked Tweets")+
  labs(x="Organic Tweets", y=("Favorite Count"))+
  theme(axis.text.y = element_text(face="bold", color="black", size=8, hjust = 1),
        plot.title = element_text(hjust = 0.5))+
  coord_flip()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 80))

# TOP 5 MOST LIKED TWEETS (TO REPORT)

# Sort the organic tweets by retweet count, and then select only first 10 tweets
dt_tweets_organic <- dt_tweets_organic %>% arrange(-retweet_count)
tmp=dt_tweets_organic[1:5,]
ggplot(tmp, aes(x=reorder(text, retweet_count), y=retweet_count))+
  geom_bar(stat="identity", fill =  "#377F97")+
  ggtitle("Top 5 Most Retweeted Tweets")+labs(x="Organic Tweets", y=("Retweet Count"))+
  theme(axis.text.y = element_text(face="bold", color="black", size=8, hjust = 1),
        plot.title = element_text(hjust = 0.5))+coord_flip()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 80))

# TOP 5 MOST RT TWEETS (TO REPORT)


##########################################################################################################
# Organic Tweets, Retweets and Replies #PLAYSTATION
##########################################################################################################



dt_tweets_organic = Playstation_22Jan_2000tweets[Playstation_22Jan_2000tweets$is_retweet==FALSE,]
dt_tweets_organic <- subset(dt_tweets_organic,
                            is.na(dt_tweets_organic$replyToSN))
dt_tweets_organic <- dt_tweets_organic %>% arrange(desc(like_count))
tmp=dt_tweets_organic[1:5,]
```

Ln 77, Col 37

```
                    is.na(dt_tweets_organic$replyToSN))
dt_tweets_organic <- dt_tweets_organic %>% arrange(desc(like_count))
tmp=dt_tweets_organic[1:5,]
ggplot(tmp, aes(x=reorder(text, like_count), y=like_count))+
  geom_bar(stat="identity", fill =  "#377F97")+
  ggtitle("Top 5 Most Liked Tweets")+
  labs(x="Organic Tweets", y=("Favorite Count"))+
  theme(axis.text.y = element_text(face="bold", color="black", size=8, hjust = 1),
        plot.title = element_text(hjust = 0.5))+
  coord_flip()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 80))

# TOP 5 MOST LIKED TWEETS (TO REPORT)

# Sort the organic tweets by retweet count, and then select only first 10 tweets
dt_tweets_organic <- dt_tweets_organic %>% arrange(-retweet_count)
tmp=dt_tweets_organic[1:5,]
ggplot(tmp, aes(x=reorder(text, retweet_count), y=retweet_count))+
  geom_bar(stat="identity", fill =  "#377F97")+
  ggtitle("Top 5 Most Retweeted Tweets")+labs(x="Organic Tweets", y=("Retweet Count"))+
  theme(axis.text.y = element_text(face="bold", color="black", size=8, hjust = 1),
        plot.title = element_text(hjust = 0.5))+coord_flip()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 80))

# TOP 5 MOST RT TWEETS (TO REPORT)


##############################################################################################################
# Tweets Vs Retweets Vs Replies #XBOX
##############################################################################################################
retweets = Xbox_22Jan_2000tweets[Xbox_22Jan_2000tweets$is_retweet==TRUE,]
replies <- subset(Xbox_22Jan_2000tweets, !is.na(Xbox_22Jan_2000tweets$quoted_reply_count))
# Create Dataframe
```

```
File  Edit  Format  View  Help
 theme(plot.title = element_text(hjust = 0.5))+
 coord_polar("y", start=0)

##########################################################################################################
# Tweets Vs Retweets Vs Replies #PLAYSTATION
##########################################################################################################
retweets = Playstation_22Jan_2000tweets[Playstation_22Jan_2000tweets$is_retweet==TRUE,]
replies <- subset(Playstation_22Jan_2000tweets, !is.na(Playstation_22Jan_2000tweets$quoted_reply_count))
# Create Dataframe
tmp <- data.frame(
 type=c("Organic", "Retweets", "Replies"),
 count=c(dim(dt_tweets_organic)[1], dim(retweets)[1], dim(replies)[1]))
ggplot(tmp, aes(x="", y=count, fill=type))+
 geom_bar(stat="identity")+
 ggtitle("Organic Tweets Vs Retweets Vs Replies")+
 labs(x="Tweets Vs Retweets", y="Frequency")+
 theme(plot.title = element_text(hjust = 0.5))+
 coord_polar("y", start=0)

##########################################################################################################
# When People tweeted or Retweeted most - HOUR BY HOUR Analysis #XBOX
##########################################################################################################

Time_Analysis = lapply(Xbox_22Jan_2000tweets$created_at, FUN=function(x) substr(x[[1]], 11,13))
tmp=data.frame(table(unlist(Time_Analysis)))
# Plot
ggplot(data=tmp, aes(x=Var1, y=Freq, group=1)) +
 geom_line(size=1, color="red")+
 geom_point(size=3, color="red")+
 ggtitle("Number of tweets by Hour")+
 labs(x="TimeStamp", y="Frequency - Tweets/Retweets/Replies")+
 theme(plot.title = element_text(hjust = 0.5),
       axis.text.x = element_text(size=10, angle=35))
```

```
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()

# PLOT CONTRIBUTION TO SENTIMENT

# 3.2 get a summary per post
Playstation_22Jan_2000tweets$status_id
statusSentiment <- oxfamSentiment %>%
  count(status_id, sentiment) %>%          # count the positives and negatives per id (status)
  spread(sentiment, n, fill = 0) %>%      # we want overall sentiment, and here we have text (positive and negative), making it difficult t
sentiment.
  mutate(sentiment = positive - negative)
# note that we only have sentiment for 73 posts in this way, since the others are considered neutral

# until here we did this with the bing dictionary
# do this with the afinn dictionary:
statusSentiment <- inner_join(oxfamTokenized,get_sentiments("afinn")) %>%
  group_by(status_id) %>%                  # here we get numeric values, so we can just sum them per post
  summarize(Sentiment = sum(value))

mean(statusSentiment$Sentiment)
# the mean sentiment seems to be somewhat positive BUT MORE POSITIVE THAN XBOX
```