Whatsapp Chat Analysis

Kariuki

1/19/2024

## Introduction

* In this chatpter I am going to deal with Whatsapp data analysis and visualization.Whatsap is a means of communicating in the modern world using phones and computers. Whatsapp is used in modern world as a mean of online help desk by most service companies. Most companies and individuals uses Whatsapp for purposes like project management , product advertisement , group communications, etc.The case study is for one of the “——” club group in Nairobi-Kenya.
* I am going to restrict this topic to only text messages exchange using Whatsapp for our analysis.
* Dowloading Whatsapp data from Adroid phones
* The structure of Whatsapp chat data
* Reading Whatsapp chat data using R
* Visualizing Whatsapp chat data.
* Message per day
* Message per weekday
* Radar charts
* Messages per hour
* Heat Map
* Messages per Author
* Emoji analysis
* Word analysis

##### Objectives

One should be able to download data from Whatsapp, Visualize the data download from whatsapp.

Whatsapp uses a customized version of Extension Messaging and Presence Protocol , Whatsapp has more than 2 billion active users in more than 180 countries as of 2020.

Whatsapp store all data in servers. It is possible to access this data using APIs exposed by whatsapp.

## Reading Whatsapp chat Data in R

One of the ways to read Whatsapp chat data using R is with the help of the library . This library provides a function-rwa\_read() to read the TXT file contaninig the Whatsapp chat data. The usage is shown as foolows.

#necessary libarries  
#install.packages("rwhatsapp")  
library(rwhatsapp)  
  
v\_chats<-rwa\_read("C:/Users/User/OneDrive/Desktop/Whatsapp Data Chat Analysis/WhatsApp Chat with Friends of RCNC (1).txt")  
# We can view the read data using the head() function as shown below  
  
head(v\_chats,5)

## # A tibble: 5 × 6  
## time author text source emoji emoji\_name  
## <dttm> <fct> <chr> <chr> <list> <list>   
## 1 2021-10-02 10:36:00 <NA> " - ~ Ruth Kihor… C:/Us… <NULL> <NULL>   
## 2 2021-10-02 10:36:00 <NA> " - You were add… C:/Us… <NULL> <NULL>   
## 3 2023-09-14 16:15:00 +254 792 228943 "<Media omitted>" C:/Us… <NULL> <NULL>   
## 4 2023-09-14 16:55:00 +254 718 516109 "<Media omitted>" C:/Us… <NULL> <NULL>   
## 5 2023-09-14 17:53:00 Hellen Data "<Media omitted>" C:/Us… <NULL> <NULL>

## Messages and calls are end -to-end encrypted . No one outside of this chat , not even Whatsapp, can read or listen to them.

## Visualizing Whatsapp chat data

Before we dive deeper into visualizations, we discuss how to extract some information from the data. For this code to work we need some extra libraries such as lubridate and dplyr as shown below.

#install.packages("lubridate")  
#install.packages("tidyverse")  
#install.packages("viridis")  
  
#Extracting some important information from the data available  
  
library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.4.4 ✔ tibble 3.2.1  
## ✔ purrr 1.0.2 ✔ tidyr 1.3.0

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(viridis)

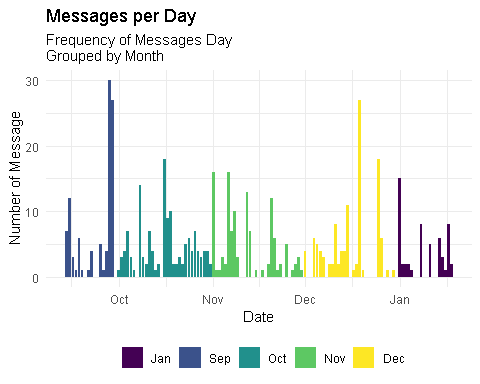
## Loading required package: viridisLite

v\_chats<- v\_chats %>%  
 mutate(message\_date= date(time))%>%  
 mutate(message\_month= month(time,label = TRUE))%>%  
 mutate(message\_month= factor(message\_month))%>%  
 mutate(message\_weekday\_number=wday(message\_date))%>%  
 mutate(message\_weekday\_name= weekdays(message\_date))%>%  
 mutate(message\_weekday\_name=factor(message\_weekday\_name))%>%  
 mutate(message\_hour=hour(time))%>%  
 filter(!is.na(author))

## Message Per Day

The first message we are going to check is the number of messages per day.

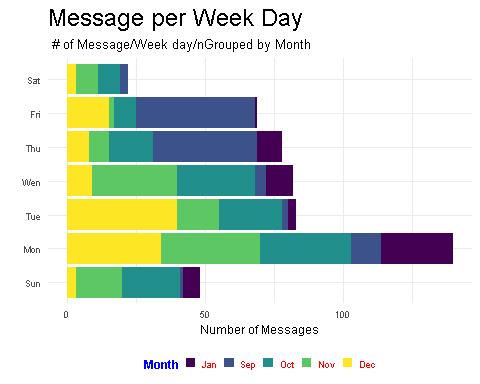
v\_chats %>%  
 group\_by(message\_month)%>%  
 count(message\_date)%>%  
 ggplot(aes(x=message\_date, y =n , fill= message\_month))+ geom\_bar(stat = "identity")+  
 scale\_fill\_viridis(discrete=TRUE)+  
 labs(x="Date", y ="Number of Message", fill = "Month")+  
 ggtitle("Messages per Day","Frequency of Messages Day\nGrouped by Month")+  
 theme\_minimal()+  
 theme(legend.title = element\_blank(),  
 legend.position = "bottom")

 ’ September’23 was the month with most messages per day, followed by December’23. For September this was due to plans that were taking place and Dec people had much funs being a festival season. Start of January ’24 the messages per day have reduced but there is anticipation of rise in chats as the club prepare for 2024 initiatives.

## Message per Weekday

Visualizing the messages per week days using bar-chart.

v\_chats%>%  
 group\_by(message\_month,message\_weekday\_number,message\_weekday\_name)%>%  
 count()%>%  
 ggplot(aes(x=reorder(message\_weekday\_name,message\_weekday\_number),y=n,fill=message\_month))+  
 geom\_bar(stat = "identity")+  
 scale\_x\_discrete(labels=c("Monday"="Mon",  
 "Tuesday"="Tue",  
 "Wednesday"="Wen",   
 "Thursday"="Thu",  
 "Friday"="Fri",   
 "Saturday"="Sat",  
 "Sunday"="Sun"))+  
 scale\_color\_viridis(discrete = TRUE)+  
 labs(x="", y="Number of Messages", fill="Month")+  
 coord\_flip()+  
 ggtitle("Message per Week Day"," # of Message/Week day/nGrouped by Month")+ theme\_minimal()+  
 theme(legend.title = element\_text(color= "blue", size=9, face="bold"),  
 legend.text = element\_text(color="red",size = 7),  
 legend.position = "bottom",  
 legend.key.size = unit(0.3,"cm"),  
 legend.key.width = unit(0.3,"cm"),  
 axis.text.x = element\_text(size = 7),  
 axis.text.y = element\_text(size = 7),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10))

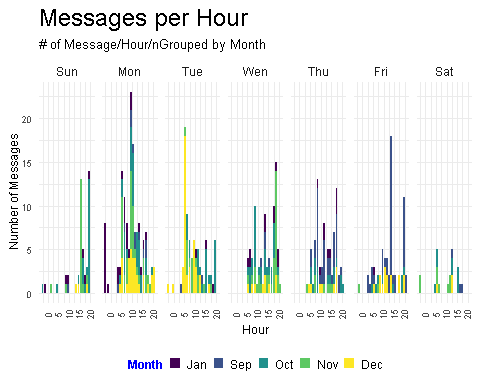


We can see that in month of September Thursday and Friday were the days with most chats/messages. October most conversation was done on Monday.November most conversation were taking place in Wensday and Monday. While December was unique one with most conversation happening in Tuesday. New year 2024 , most conversation have started to happen on Monday but still with another one week to close the first month in year 2024.

## Messages per Hour

one can visualize these messages per hour and grouping by month and hour.Want to see how messages were exchanged in every month for every weekday in every hour.

v\_weekdays<-c("Sun","Mon","Tue","Wen","Thu","Fri","Sat")  
names(v\_weekdays)<-1:7  
v\_chats%>%  
 group\_by(message\_month,message\_weekday\_number,message\_weekday\_name,message\_hour)%>%  
 count()%>%  
 ggplot(aes(x=message\_hour,y=n,fill=message\_month))+  
 geom\_bar(stat = "Identity")+  
 scale\_fill\_viridis(discrete = TRUE)+  
 labs(x="Hour", y = "Number of Messages",fill="Month")+  
 facet\_wrap(~message\_weekday\_number,ncol = 7,  
 labeller = labeller(message\_weekday\_number=v\_weekdays))+  
 ggtitle("Messages per Hour","# of Message/Hour/nGrouped by Month")+  
 theme\_minimal()+  
 theme(legend.title = element\_text(color="blue",size =9,face = "bold"),  
 legend.position = "bottom",  
 legend.key.size = unit(0.3,"cm"),  
 legend.key.width = unit(0.3,"cm"),  
 axis.text.x = element\_text(size = 7,angle = 90),  
 axis.text.y = element\_text(size = 7),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10))

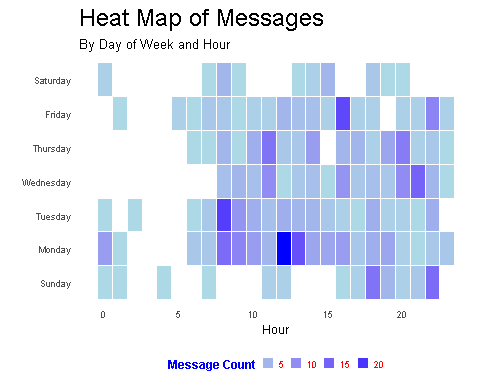


What clear is that most conversation happens around 5-10 hrs , taking into account in a day we have 24hrs. Saturday has the least number of messages per hour and also the whole day. Monday ,Tuesday and Friday top as the days with most messages per hour .January most conversation are taking place on Monday around 0-10 hrs.

## Heat Map

One can visualize this data regarding messages per hour in the form of heat map.

v\_chats%>%  
 group\_by(message\_weekday\_number,message\_weekday\_name,message\_hour)%>%  
 count()%>%  
 ggplot(aes(x=message\_hour,y=reorder(message\_weekday\_name,message\_weekday\_number)))+  
 geom\_tile(aes(fill= n), color= "white",na.rm = TRUE)+   
 scale\_fill\_gradient(low="lightblue",high="blue")+  
 guides(fill=guide\_legend(title = "Message Count"))+  
 labs(title = "Heat Map of Messages", subtitle = "By Day of Week and Hour",  
 x="Hour",y= "")+  
 theme\_minimal()+  
 theme(legend.title = element\_text(colour = "blue",size = 9,face = "bold"),  
 legend.text = element\_text(color = "red",size = 7),  
 legend.position = "bottom",  
 legend.key.size = unit(0.3,"cm"),  
 legend.key.width = unit(0.3,"cm"),  
 axis.text.x = element\_text(size = 7),  
 axis.text.y = element\_text(size = 7),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank())

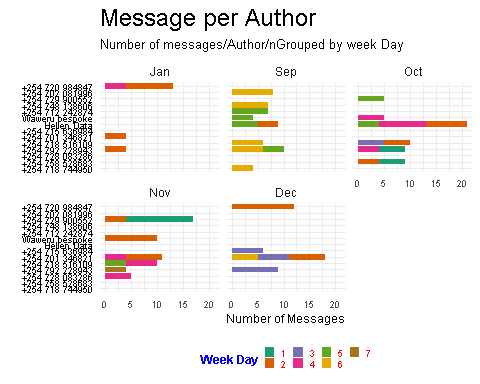


Clear that Monday, Friday and Tuesday are the days of week with the most messages count at around 12hrs.

## Messages per Author Just diving deeper to see messages per author, per weekdays across all the months.

#install.packages("RColorBrewer")  
library(RColorBrewer)  
library(dplyr)  
library(ggplot2)  
  
threshold <- 4 # Set your minimum threshold here  
  
v\_chats %>%  
 group\_by(message\_month, message\_weekday\_number, message\_weekday\_name, author) %>%  
 count() %>%  
 filter(n >= threshold) %>% # Exclude authors with fewer than 'threshold' messages  
 ggplot(aes(x = reorder(author, n),  
 y = n,  
 fill = reorder(message\_weekday\_number, message\_weekday\_name))) +  
 geom\_bar(stat = "Identity") +  
 scale\_fill\_manual(labels = c("Monday" = "Mon",  
 "Tuesday" = "Tue",  
 "Wednesday" = "Wen",  
 "Thursday" = "Thus",  
 "Friday" = "Fri",  
 "Saturday" = "Sat",  
 "Sunday" = "Sun"),  
 values = brewer.pal(8, "Dark2")) +  
 facet\_wrap(~message\_month) +  
 labs(x = "", y = "Number of Messages", fill = "Week Day") +  
 coord\_flip() +  
 ggtitle("Message per Author", "Number of messages/Author/nGrouped by week Day") +  
 theme\_minimal() +  
 theme(legend.title = element\_text(colour = "blue", size = 9, face = "bold"),  
 legend.text = element\_text(color = "red", size = 7),  
 legend.position = "bottom",  
 legend.key.size = unit(0.3, "cm"),  
 legend.key.width = unit(0.3, "cm"),  
 axis.text.x = element\_text(size = 7),  
 axis.text.y = element\_text(size = 7, color = "black"),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10))

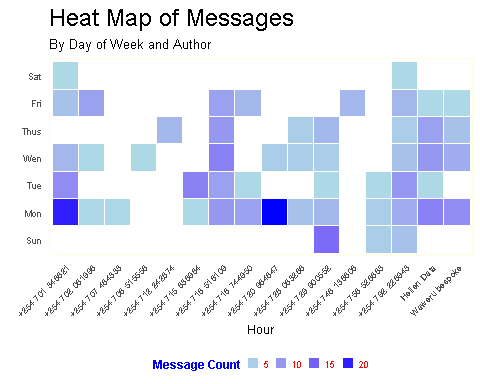
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA  
  
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:  
## returning NA



What you are able to see is the number of messages per Author. The threshold shows authors with more than 4 text/messages in last 5 months. These are likely to be leaders of these clubs.

Creating similar thing but now using a Heat map- Messages per author per weekday.

# Set your minimum threshold here  
threshold <- 4   
  
v\_chats %>%  
 group\_by(message\_weekday\_number, message\_weekday\_name, author) %>%  
 count() %>%  
 filter(n >= threshold) %>% # Exclude authors with fewer than 'threshold' messages  
 ggplot(aes(x = author, y = reorder(message\_weekday\_name, message\_weekday\_number))) +  
 geom\_tile(aes(fill = n), color = "white", na.rm = TRUE) +   
 scale\_fill\_gradient(low = "lightblue", high = "blue") +  
 scale\_y\_discrete(labels = c("Monday" = "Mon",  
 "Tuesday" = "Tue",  
 "Wednesday" = "Wen",  
 "Thursday" = "Thus",  
 "Friday" = "Fri",  
 "Saturday" = "Sat",  
 "Sunday" = "Sun")) +  
 guides(fill = guide\_legend(title = "Message Count")) +  
 labs(title = "Heat Map of Messages", subtitle = "By Day of Week and Author",  
 x = "Hour", y = "") +  
 theme\_minimal() +  
 theme(legend.title = element\_text(colour = "blue", size = 9, face = "bold"),  
 legend.text = element\_text(color = "red", size = 7),  
 legend.position = "bottom",  
 legend.key.size = unit(0.3, "cm"),  
 legend.key.width = unit(0.3, "cm"),  
 axis.text.x = element\_text(size = 7, angle = 45, hjust = 1), # Adjust angle and hjust  
 axis.text.y = element\_text(size = 7),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.background = element\_rect(colour = "lightyellow"))

 There are two authors with highest number of chats/ messages on monday.

## Emoji Analysis

Whatsapp data provides two fields with informations on the emojis used in the chats. These fields are emoji and so a lot of analysis is possible on the emoji used in the chats.

The main purpose of this data preparation is to get images of the emojis as the emojis are coded in whatsapp chat data.

head(v\_chats[,c("emoji","emoji\_name")]%>%  
 filter(!is.null(emoji\_name))%>%  
 filter(emoji\_name !="NULL"))

## # A tibble: 6 × 2  
## emoji emoji\_name  
## <list> <list>   
## 1 <chr [1]> <chr [1]>   
## 2 <chr [1]> <chr [1]>   
## 3 <chr [1]> <chr [1]>   
## 4 <chr [1]> <chr [1]>   
## 5 <chr [1]> <chr [1]>   
## 6 <chr [2]> <chr [2]>

## clean data   
  
v\_Emojis <- v\_chats %>%  
unnest(c(emoji, emoji\_name)) %>%  
mutate(emoji = str\_sub(emoji, end = 1)) %>%  
mutate(emoji\_name = str\_remove(emoji\_name, ":.\*")) %>%  
mutate(emoji\_url = map\_chr(emoji,  
~paste0("https://abs.twimg.com/emoji/v2/72x72/",  
as.hexmode(utf8ToInt(.x)), ".png"))  
) %>%  
filter(!is.null(emoji\_name)) %>%  
filter(emoji\_name != "NULL")  
  
head(v\_Emojis)

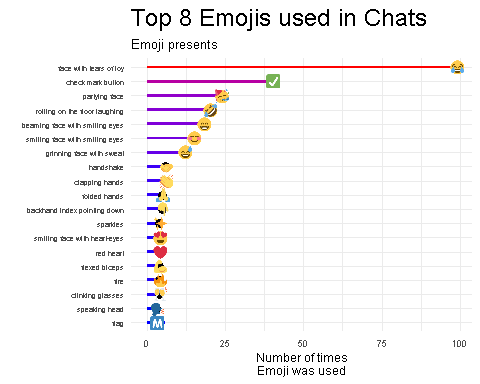
## # A tibble: 6 × 12  
## time author text source emoji emoji\_name message\_date  
## <dttm> <fct> <chr> <chr> <chr> <chr> <date>   
## 1 2023-09-14 21:26:00 +254 718 744950 "Hey… C:/Us… "\U0… backhand … 2023-09-14   
## 2 2023-09-14 21:27:00 +254 718 744950 "\*Th… C:/Us… "\U0… smiling f… 2023-09-14   
## 3 2023-09-15 05:43:00 +254 780 462036 "\*Th… C:/Us… "\U0… smiling f… 2023-09-15   
## 4 2023-09-15 07:14:00 +254 718 744950 "\*Th… C:/Us… "\U0… smiling f… 2023-09-15   
## 5 2023-09-15 14:29:00 +254 704 053256 "Big… C:/Us… "\U0… partying … 2023-09-15   
## 6 2023-09-16 19:58:00 +254 718 744950 "Tod… C:/Us… "\U0… folded ha… 2023-09-16   
## # ℹ 5 more variables: message\_month <ord>, message\_weekday\_number <dbl>,  
## # message\_weekday\_name <fct>, message\_hour <int>, emoji\_url <chr>

# time author text source  
  
head(v\_Emojis$emoji\_url)

## [1] "https://abs.twimg.com/emoji/v2/72x72/1f447.png"  
## [2] "https://abs.twimg.com/emoji/v2/72x72/1f60a.png"  
## [3] "https://abs.twimg.com/emoji/v2/72x72/1f60a.png"  
## [4] "https://abs.twimg.com/emoji/v2/72x72/1f60a.png"  
## [5] "https://abs.twimg.com/emoji/v2/72x72/1f973.png"  
## [6] "https://abs.twimg.com/emoji/v2/72x72/1f64f.png"

## Emoji Visualizations

# Install the ggimage package if you haven't already  
#install.packages("ggimage")  
  
# Load the ggimage package  
library(ggimage)  
library(ggplot2)  
library(ggtext)  
  
# Set your threshold here  
threshold <- 20   
  
v\_Emojis %>%  
 count(emoji, emoji\_name, emoji\_url) %>%  
 arrange(desc(n)) %>%  
 filter(row\_number() <= threshold) %>% # Keep only the top 8 emojis  
 ggplot(aes(x = reorder(emoji\_name, n), y = n)) +  
 geom\_col(aes(fill = n), show.legend = FALSE, width = 0.2) +  
 geom\_point(aes(color = n), show.legend = FALSE, size = 3) +  
 geom\_image(aes(image = emoji\_url), size = 0.05) +  
 scale\_fill\_gradient(low = "blue", high = "red") +  
 scale\_color\_gradient(low = "black", high = "brown") +  
 coord\_flip() +  
 labs(title = "Top 8 Emojis used in Chats", subtitle = "Emoji presents",  
 x = "", y = "Number of times\nEmoji was used") +  
 theme\_minimal() +  
 theme(  
 axis.text.x = element\_text(size = 7),  
 axis.text.y = element\_text(size = 5),  
 axis.title = element\_text(size = 9),  
 plot.title = element\_text(size = 18),  
 plot.subtitle = element\_text(size = 10)  
 )



Here are the top 5 emojis used by authors in this group

* Face with tears of joy
* Check mark button
* Partying face
* Rolling on floor laughing
* Beaming face with smilling face

## Conclusion

The analysis of WhatsApp data provides a wealth of information.There are hardly any products available in the world as on date (end of 2024) for analysis of WhatsApp data. So, this is an areawhere useful products could be created, and businesses could beformed around the same.