Data Analysis Final Paper

Caleb Kreatschman

05/09/2018

## Issue Description

My original intention was to analyze my facebook messenger chat history in a general sort of scope, since I didn’t really know what I was looking for. After playing with the data a bit, I realized that there is quite a lot of potential in such a simple data set and I had to narrow my focus to something that I could easily accomplish. I settled on identifying patterns in the different types of people I message as well as the most used words for every group.

## Questions

Who do I message most? When do most of my conversations occur? Does message traffic with high or low density correspond with particular life events? What words are used most?

## Data Source

Personal faceook archive. There’s a lot of interesting data in the archive, but I only used the message portion. Every conversation existed as an independent html page.

## Documentation

No documentation aside from the data itself.

## Description of the Data

I used the ‘read\_html()’ function from the rvest package to load each file manually. I later realized that it wasn’t necessary to make each conversation into a dataframe, and that I could just combine them all as part of the read process. I will keep that in mind next time I do a project like this. Once the filed were read into RStudio, there wasn’t much structure. Every file was just a long string of barely readable html code.

## Cleaning and Preparation

This part of the project took, by far, the most effort. I learned a lot of lessons along the way that I will certainly speed my processes in the future. But, having managed to make a working data set, I kept going with what I had.

First step is to load the necessary packages.

library(tidyverse)

## -- Attaching packages --------------------------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 2.2.1 v purrr 0.2.4  
## v tibble 1.4.2 v dplyr 0.7.4  
## v tidyr 0.8.0 v stringr 1.2.0  
## v readr 1.1.1 v forcats 0.2.0

## -- Conflicts ------------------------------------------------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(rvest)

## Warning: package 'rvest' was built under R version 3.4.4

## Loading required package: xml2

##   
## Attaching package: 'rvest'

## The following object is masked from 'package:purrr':  
##   
## pluck

## The following object is masked from 'package:readr':  
##   
## guess\_encoding

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

library(tidytext)

## Warning: package 'tidytext' was built under R version 3.4.4

library(tokenizers)

## Warning: package 'tokenizers' was built under R version 3.4.4

library(tm)

## Warning: package 'tm' was built under R version 3.4.4

## Loading required package: NLP

##   
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':  
##   
## annotate

The next step was getting the data into a readable format. Since I failed to import all of the html files as a single data frame, I had to clean every data frame individually. Thus, I made a function that would break the html code of each page into three columns consisting of user, time, and message.

organize <- function(x){  
 organizedData <- data.frame(sender = c(x %>% html\_nodes(".user") %>% html\_text()),   
 time = c(x %>% html\_nodes(".meta") %>% html\_text()),  
 message = c(x %>% html\_nodes("p") %>% html\_text()))  
 return(organizedData)  
}  
speiden <- read\_html("Speiden.html") %>% organize()  
saraJean <- read\_html("SaraJean.html") %>% organize()  
joanna <- read\_html("Joanna.html")%>% organize()  
schuh <- read\_html("Schuh.html")%>% organize()  
electricShop <- read\_html("620.html")%>% organize()  
angelina <- read\_html("Angelina.html")%>% organize()  
ashley <- read\_html("Ashley.html")%>% organize()  
atkins <- read\_html("Atkins.html")%>% organize()  
austin <- read\_html("Austin.html")%>% organize()  
bria <- read\_html("Bria.html")%>% organize()  
candelaria<- read\_html("Candelaria.html")%>% organize()  
carly<- read\_html("Carly.html")%>% organize()  
carri<- read\_html("Carri.html")%>% organize()  
casson<- read\_html("Casson.html")%>% organize()  
christopherson<- read\_html("Christopherson.html")%>% organize()  
clara<- read\_html("Clara.html")%>% organize()   
david<- read\_html("David.html")%>% organize()  
destiny<- read\_html("Destiny.html")%>% organize()  
diandra<- read\_html("Diandra.html")%>% organize()  
eiland<- read\_html("Eiland.html")%>% organize()  
emmie<- read\_html("Emmie.html")%>% organize()  
erik<- read\_html("Erik.html", stringsAsFactors = F)%>% organize()  
family1<- read\_html("Family.html")%>% organize()  
family2<- read\_html("Family2.html")%>% organize()  
family3<- read\_html("Family3.html")%>% organize()  
family4<- read\_html("family4.html")%>% organize()  
fisher<- read\_html("Fisher.html")%>% organize()  
flag<- read\_html("Flag.html")%>% organize()  
glenn<- read\_html("Glenn.html")%>% organize()  
haines<- read\_html("Haines.html")%>% organize()  
hannahS<- read\_html("HannahS.html")%>% organize()  
hilton<- read\_html("Hilton.html")%>% organize()  
holt<- read\_html("Holt.html")%>% organize()  
johnT<- read\_html("JohnT.html")%>% organize()  
jordan<- read\_html("Jordan.html")%>% organize()  
josiah<- read\_html("Josiah.html")%>% organize()  
kaleigh<- read\_html("Kaleigh.html")%>% organize()  
kallie<- read\_html("Kallie.html")%>% organize()  
kalun<- read\_html("Kalun.html")%>% organize()  
kat<- read\_html("Kat.html")%>% organize()  
kelsey<- read\_html("Kelsey.html")%>% organize()  
kevin<- read\_html("Kevin.html")%>% organize()  
kurganov<- read\_html("Kurganov.html")%>% organize()  
lavolpa<- read\_html("LaVolpa.html")%>% organize()  
madison<- read\_html("Madison.html")%>% organize()  
mccain<- read\_html("Mccain.html")%>% organize()  
mccausland<- read\_html("Mccausland.html")%>% organize()  
mercado<- read\_html("Mercado.html")%>% organize()  
mother<- read\_html("Mother.html")%>% organize()  
mother2<- read\_html("mother2.html")%>% organize()  
msSpeiden<- read\_html("MsSpeiden.html")%>% organize()  
nathan<- read\_html("Nathan.html")%>% organize()  
pouliot<- read\_html("pouliot.html")%>% organize()  
precious<- read\_html("Precious.html")%>% organize()  
precious2<- read\_html("Precious2.html")%>% organize()  
rachel<- read\_html("Rachel.html")%>% organize()  
taira<- read\_html("Taira.html")%>% organize()  
taylor<- read\_html("Taylor.html")%>% organize()  
theresa<- read\_html("Theresa.html")%>% organize()

I realized that, once I combined all of the data frames, there would be no way to determine which person I was talking to. So I added a ‘recipient’ column. Originally, I included this column as part of the ‘organize’ function, but it kept throwing an error when I tried to knit the document. So I had to do it manually, one by one. I wish I could have found a better way to create the recipient column based off of the sender column, but there wasn’t much consistency to work with. Repeating this much code is certainly not ideal, but it gets the job done.

I didn’t bother correcting the recipient column for any conversations involving more than two people, the low message count in those data frames has little effect on the overall report.

speiden$recipient <- c("Caleb Kreatschman")  
speiden$recipient[speiden$sender[] %in% c("Caleb Kreatschman")] <- as.character(speiden$sender[2])  
saraJean$recipient <- c("Caleb Kreatschman")  
saraJean$recipient[saraJean$sender[] %in% c("Caleb Kreatschman")] <- as.character(saraJean$sender[2])  
joanna$recipient <- c("Caleb Kreatschman")  
joanna$recipient[joanna$sender[] %in% c("Caleb Kreatschman")] <- as.character(joanna$sender[3])  
schuh$recipient <- c("Caleb Kreatschman")  
schuh$recipient[schuh$sender[] %in% c("Caleb Kreatschman")] <- as.character(schuh$sender[2])  
electricShop$recipient <- c("Caleb Kreatschman")  
angelina$recipient <- c("Caleb Kreatschman")  
angelina$recipient[angelina$sender[] %in% c("Caleb Kreatschman")] <- as.character(angelina$sender[1])  
ashley$recipient <- c("Caleb Kreatschman")  
ashley$recipient[ashley$sender[] %in% c("Caleb Kreatschman")] <- as.character(ashley$sender[1])  
atkins$recipient <- c("Caleb Kreatschman")  
atkins$recipient[atkins$sender[] %in% c("Caleb Kreatschman")] <- as.character(atkins$sender[2])  
austin$recipient <- c("Caleb Kreatschman")  
austin$recipient[austin$sender[] %in% c("Caleb Kreatschman")] <- as.character(austin$sender[2])  
bria$recipient <- c("Caleb Kreatschman")  
bria$recipient[bria$sender[] %in% c("Caleb Kreatschman")] <- as.character(bria$sender[2])  
candelaria$recipient <- c("Caleb Kreatschman")  
candelaria$recipient[candelaria$sender[] %in% c("Caleb Kreatschman")] <- as.character(candelaria$sender[1])  
carly$recipient <- c("Caleb Kreatschman")  
carly$recipient[carly$sender[] %in% c("Caleb Kreatschman")] <- as.character(carly$sender[2])  
carri$recipient <- c("Caleb Kreatschman")  
carri$recipient[carri$sender[] %in% c("Caleb Kreatschman")] <- as.character(carri$sender[1])  
casson$recipient <- c("Caleb Kreatschman")  
casson$recipient[casson$sender[] %in% c("Caleb Kreatschman")] <- as.character(casson$sender[2])  
christopherson$recipient <- c("Caleb Kreatschman")  
christopherson$recipient[christopherson$sender[] %in% c("Caleb Kreatschman")] <- as.character(christopherson$sender[1])  
clara$recipient <- c("Caleb Kreatschman")  
clara$recipient[clara$sender[] %in% c("Caleb Kreatschman")] <- as.character(clara$sender[2])  
david$recipient <- c("Caleb Kreatschman")  
david$recipient[david$sender[] %in% c("Caleb Kreatschman")] <- as.character(david$sender[1])  
destiny$recipient <- c("Caleb Kreatschman")  
destiny$recipient[destiny$sender[] %in% c("Caleb Kreatschman")] <- as.character(destiny$sender[1])  
diandra$recipient <- c("Caleb Kreatschman")  
diandra$recipient[diandra$sender[] %in% c("Caleb Kreatschman")] <- as.character(diandra$sender[2])  
eiland$recipient <- c("Caleb Kreatschman")  
eiland$recipient[eiland$sender[] %in% c("Caleb Kreatschman")] <- as.character(eiland$sender[2])  
emmie$recipient <- c("Caleb Kreatschman")  
emmie$recipient[emmie$sender[] %in% c("Caleb Kreatschman")] <- as.character(emmie$sender[1])  
erik$recipient <- c("Caleb Kreatschman")  
erik$recipient[erik$sender[] %in% c("Caleb Kreatschman")] <- as.character(erik$sender[1])  
family1$recipient <- c("Caleb Kreatschman")  
family2$recipient <- c("Caleb Kreatschman")  
family3$recipient <- c("Caleb Kreatschman")  
family4$recipient <- c("Caleb Kreatschman")  
fisher$recipient <- c("Caleb Kreatschman")  
fisher$recipient[fisher$sender[] %in% c("Caleb Kreatschman")] <- as.character(fisher$sender[2])  
flag$recipient <- c("Caleb Kreatschman")  
flag$recipient[flag$sender[] %in% c("Caleb Kreatschman")] <- as.character(flag$sender[2])  
glenn$recipient <- c("Caleb Kreatschman")  
glenn$recipient[glenn$sender[] %in% c("Caleb Kreatschman")] <- as.character(glenn$sender[1])  
haines$recipient <- c("Caleb Kreatschman")  
haines$recipient[haines$sender[] %in% c("Caleb Kreatschman")] <- as.character(haines$sender[2])  
hannahS$recipient <- c("Caleb Kreatschman")  
hannahS$recipient[hannahS$sender[] %in% c("Caleb Kreatschman")] <- as.character(hannahS$sender[1])  
hilton$recipient <- c("Caleb Kreatschman")  
hilton$recipient[hilton$sender[] %in% c("Caleb Kreatschman")] <- as.character(hilton$sender[1])  
holt$recipient <- c("Caleb Kreatschman")  
holt$recipient[holt$sender[] %in% c("Caleb Kreatschman")] <- as.character(holt$sender[2])  
johnT$recipient <- c("Caleb Kreatschman")  
johnT$recipient[johnT$sender[] %in% c("Caleb Kreatschman")] <- as.character(johnT$sender[1])  
jordan$recipient <- c("Caleb Kreatschman")  
jordan$recipient[jordan$sender[] %in% c("Caleb Kreatschman")] <- as.character(jordan$sender[1])  
josiah$recipient <- c("Caleb Kreatschman")  
josiah$recipient[josiah$sender[] %in% c("Caleb Kreatschman")] <- as.character(josiah$sender[1])  
kaleigh$recipient <- c("Caleb Kreatschman")  
kaleigh$recipient[kaleigh$sender[] %in% c("Caleb Kreatschman")] <- as.character(kaleigh$sender[1])  
kallie$recipient <- c("Caleb Kreatschman")  
kallie$recipient[kallie$sender[] %in% c("Caleb Kreatschman")] <- as.character(kallie$sender[2])  
kalun$recipient <- c("Caleb Kreatschman")  
kalun$recipient[kalun$sender[] %in% c("Caleb Kreatschman")] <- as.character(kalun$sender[2])  
kat$recipient <- c("Caleb Kreatschman")  
kat$recipient[kat$sender[] %in% c("Caleb Kreatschman")] <- as.character(kat$sender[1])  
kelsey$recipient <- c("Caleb Kreatschman")  
kelsey$recipient[kelsey$sender[] %in% c("Caleb Kreatschman")] <- as.character(kelsey$sender[1])  
kevin$recipient <- c("Caleb Kreatschman")  
kevin$recipient[kevin$sender[] %in% c("Caleb Kreatschman")] <- as.character(kevin$sender[1])  
kurganov$recipient <- c("Caleb Kreatschman")  
kurganov$recipient[kurganov$sender[] %in% c("Caleb Kreatschman")] <- as.character(kurganov$sender[1])  
lavolpa$recipient <- c("Caleb Kreatschman")  
lavolpa$recipient[lavolpa$sender[] %in% c("Caleb Kreatschman")] <- as.character(lavolpa$sender[1])  
madison$recipient <- c("Caleb Kreatschman")  
madison$recipient[madison$sender[] %in% c("Caleb Kreatschman")] <- as.character(madison$sender[2])  
mccain$recipient <- c("Caleb Kreatschman")  
mccain$recipient[mccain$sender[] %in% c("Caleb Kreatschman")] <- as.character(mccain$sender[1])  
mccausland$recipient <- c("Caleb Kreatschman")  
mccausland$recipient[mccausland$sender[] %in% c("Caleb Kreatschman")] <- as.character(mccausland$sender[1])  
mercado$recipient <- c("Caleb Kreatschman")  
mercado$recipient[mercado$sender[] %in% c("Caleb Kreatschman")] <- as.character(mercado$sender[2])  
mother$recipient <- c("Caleb Kreatschman")  
mother$recipient[mother$sender[] %in% c("Caleb Kreatschman")] <- as.character(mother$sender[1])  
mother2$recipient <- c("Caleb Kreatschman")  
mother2$recipient[mother2$sender[] %in% c("Caleb Kreatschman")] <- as.character(mother2$sender[4])  
msSpeiden$recipient <- c("Caleb Kreatschman")  
msSpeiden$recipient[msSpeiden$sender[] %in% c("Caleb Kreatschman")] <- as.character(msSpeiden$sender[1])  
nathan$recipient <- c("Caleb Kreatschman")  
nathan$recipient[nathan$sender[] %in% c("Caleb Kreatschman")] <- as.character(nathan$sender[2])  
pouliot$recipient <- c("Caleb Kreatschman")  
pouliot$recipient[pouliot$sender[] %in% c("Caleb Kreatschman")] <- as.character(pouliot$sender[1])  
precious$recipient <- c("Caleb Kreatschman")  
precious$recipient[precious$sender[] %in% c("Caleb Kreatschman")] <- as.character(precious$sender[1])  
precious2$recipient <- c("Caleb Kreatschman")  
precious2$recipient[precious2$sender[] %in% c("Caleb Kreatschman")] <- as.character(precious2$sender[1])  
rachel$recipient <- c("Caleb Kreatschman")  
rachel$recipient[rachel$sender[] %in% c("Caleb Kreatschman")] <- as.character(rachel$sender[2])  
taira$recipient <- c("Caleb Kreatschman")  
taira$recipient[taira$sender[] %in% c("Caleb Kreatschman")] <- as.character(taira$sender[1])  
taylor$recipient <- c("Caleb Kreatschman")  
taylor$recipient[taylor$sender[] %in% c("Caleb Kreatschman")] <- as.character(taylor$sender[1])  
theresa$recipient <- c("Caleb Kreatschman")  
theresa$recipient[theresa$sender[] %in% c("Caleb Kreatschman")] <- as.character(theresa$sender[1])

Once I had each data frame in four columns with accurate recipient columns, I combined them all into a single data frame.

finalData <- rbind(angelina, ashley, atkins, austin, bria, candelaria, carly, carri, casson, christopherson, clara, david, destiny, diandra, eiland, electricShop, emmie, erik, family1, family2, family3, family4, fisher, flag, glenn, haines, hannahS, hilton, holt, joanna, johnT, jordan, josiah, kaleigh, kallie, kalun, kat, kelsey, kevin, kurganov, lavolpa, madison, mccain, mccausland, mercado, mother, mother2, msSpeiden, nathan, pouliot, precious, precious2, rachel, saraJean, schuh, speiden, taira, taylor, theresa)

At this point, I wasn’t happy with the way the ‘time’ column was formatted (i.e.“Saturday, January 22, 2011 at 10:49am PST”). I used the following code to clean it up.

finalData$time <- as.character(finalData$time) #Changed to character data so I can work on it as a string.  
finalData$message <- as.character(finalData$message)  
  
#I want to separate the values into weekday, date, year, and time. I can also drop the "PDT" because it isn't relevant. gsub time!  
  
finalData$time <- gsub(pattern = " at", replacement = ",", finalData$time, fixed = TRUE)   
  
#This will make the "at" into a comma.  
#Need to put a space before "at" or it murders all the "saturday" values.  
  
finalData$time <- gsub(pattern = "PDT", replacement = " ", finalData$time, fixed = TRUE)   
#Get rid of the "PDT"  
finalData$time <- gsub(pattern = "PST", replacement = " ", finalData$time, fixed = TRUE)   
#Get rid of the "PST"  
  
finalData <- within(finalData, time <- data.frame(do.call('rbind', strsplit(as.character(time), ",", fixed = TRUE))))   
#Turns cleaned up time strings into 4 separate columns.  
  
#Let's get the date values into a single column.  
finalData$time$X1 <- as.character(finalData$time$X1)  
finalData$time$X2 <- as.character(finalData$time$X2) #Easier to paste when they're strings.  
finalData$time$X3 <- as.character(finalData$time$X3)  
finalData$date <- paste(finalData$time$X2, ", ", finalData$time$X3)  
  
#Don't really need the two separate columns anymore.  
finalData$time <- within(finalData$time, rm(X2, X3))

Unfortunately, cleaning up the time left all of the new values as subcolumns of the ‘time’ column. I wanted them to each have their own column in the main data frame.

finalData$weekday <- paste(finalData$time$X1)  
finalData$timeSent <- paste(finalData$time$X4)  
finalData <- within(finalData, rm(time))  
finalData$weekday <- as.factor(finalData$weekday)  
finalData$date <- mdy(finalData$date)  
finalData$timeSent <- parse\_date\_time(finalData$timeSent, orders = "IMp")  
head(finalData)

## sender  
## 1 Angelina Dolf  
## 2 Caleb Kreatschman  
## 3 Angelina Dolf  
## 4 Caleb Kreatschman  
## 5 Caleb Kreatschman  
## 6 Angelina Dolf  
## message  
## 1 Cool! Let me know if you find something amazing.  
## 2 I'll keep an eye out, I've heard that Seattle has a decent coffee scene  
## 3 Yaaay! I'm glad it didn't take too long to get to you. If you find any coffee beans near you that are to die for feel free to send them my way. I could always use some more on deployment. <f0><U+009F><U+0098><U+0084>  
## 4 Got the coffee! I owe you big time :)  
## 5 Fo sho  
## 6 That sounds a little pricey. Haha I'll think of something eventually. I'll let you know when I do.  
## recipient date weekday timeSent  
## 1 Caleb Kreatschman 2016-05-27 Friday 0000-01-01 17:27:00  
## 2 Angelina Dolf 2016-05-27 Friday 0000-01-01 17:00:00  
## 3 Caleb Kreatschman 2016-05-27 Friday 0000-01-01 16:49:00  
## 4 Angelina Dolf 2016-05-27 Friday 0000-01-01 16:02:00  
## 5 Angelina Dolf 2016-05-20 Friday 0000-01-01 18:27:00  
## 6 Caleb Kreatschman 2016-05-20 Friday 0000-01-01 18:21:00

str(finalData)

## 'data.frame': 45570 obs. of 6 variables:  
## $ sender : Factor w/ 70 levels "Angelina Dolf",..: 1 2 1 2 2 1 2 2 1 2 ...  
## $ message : chr "Cool! Let me know if you find something amazing." "I'll keep an eye out, I've heard that Seattle has a decent coffee scene" "Yaaay! I'm glad it didn't take too long to get to you. If you find any coffee beans near you that are to die fo"| \_\_truncated\_\_ "Got the coffee! I owe you big time :)" ...  
## $ recipient: chr "Caleb Kreatschman" "Angelina Dolf" "Caleb Kreatschman" "Angelina Dolf" ...  
## $ date : Date, format: "2016-05-27" "2016-05-27" ...  
## $ weekday : Factor w/ 7 levels "Friday","Monday",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ timeSent : POSIXct, format: "0000-01-01 17:27:00" "0000-01-01 17:00:00" ...

summary(finalData)

## sender message recipient   
## Caleb Kreatschman :22460 Length:45570 Length:45570   
## Clara Mailloux :15909 Class :character Class :character   
## Kallie Barnes : 965 Mode :character Mode :character   
## Sarah Jean Wilson : 952   
## Journey Lewis : 860   
## Precious Kreatschman: 850   
## (Other) : 3574   
## date weekday timeSent   
## Min. :2011-02-19 Friday :6060 Min. :0000-01-01 00:00:00   
## 1st Qu.:2017-04-30 Monday :7649 1st Qu.:0000-01-01 11:39:00   
## Median :2017-07-24 Saturday :5278 Median :0000-01-01 17:02:00   
## Mean :2017-07-04 Sunday :6364 Mean :0000-01-01 14:55:27   
## 3rd Qu.:2018-02-19 Thursday :5543 3rd Qu.:0000-01-01 20:31:00   
## Max. :2018-04-17 Tuesday :7440 Max. :0000-01-01 23:59:00   
## Wednesday:7236

Data is in a readable and workable format.

I wanted to breakdown the conversations into meaningful groups. I decided to separate each individual into the categories of family, friends, navy, my girlfriend, and me. Some individuals occupy more than one group.

navyFolks <- c("Nicole Atkins", "James Candelaria", "William Casson","Tim Christopherson","Hunter Eiland","Adam Speedy","Andrew Jepson","CaseyandBrenna Reeves",  
 "Genesis Amon", "Jacob Linden", "Jessica St. John Budd", "Marc LaVolpa", "Ryan Boswell", "Steven De Leon", "Zakir Ismagilov", "Jeffrey Daniel Rufin",  
 "Joshua Paul Hannaman","Richard Hutchins", "Savannah Meekins", "Thomas Payne", "Alec Fisher", "Sarah Flaig", "Tyler Haines", "Cassandra Hilton", "Jordan Tucker",   
 "James Kelsey", "K. J. Holt", "Josiah Greklek", "Trathen McCausland", "Brandon Schuh", "Joe Da F Merc", "Marcus Pouliot")  
friends <- c("Taylor Kenneth Shideler", "Brandon Schuh", "Nathan Carey", "James Kelsey", "Kalun Patrick",   
 "Josiah Greklek", "Kevin McCarthy", "Altaira Dactal", "Joanna Lloyd", "Erik Sullivan", "Hannah Shideler",   
 "John Thompson", "Diandra Totten","Hunter Eiland","Carri Wingreen","Carly Wells","Austin Johnson",  
 "Bria Eickholt", "Ashley Everett", "Angelina Dolf", "Rachel Morris", "Kallie Barnes")  
family <- c("Haven Rohme", "Kaliegh Masters", "Journey Lewis", "Madison Kreatschman", "Emilie Woodward",   
 "Alicia Cox McKee", "Precious Kreatschman", "Hannah McKee", "Donda Lewis Kreatschman")  
girlfriend <- c("Clara Mailloux")  
me <- c("Caleb Kreatschman")

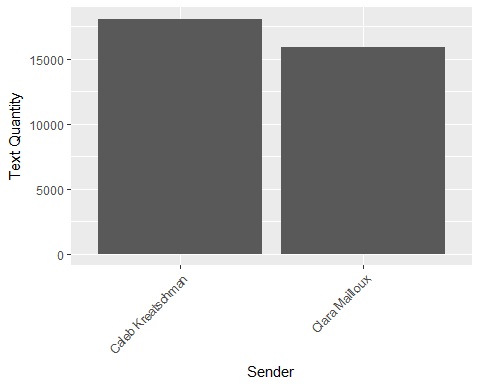
I then converted each group into a data frame.

gfData <- finalData %>% filter(sender %in% girlfriend | recipient %in% girlfriend)  
familyData <- finalData %>% filter(sender %in% family | recipient %in% family)  
friendsData <- finalData %>% filter(sender %in% friends | recipient %in% friends)  
navyData <- finalData %>% filter (sender %in% navyFolks | recipient %in% navyFolks)

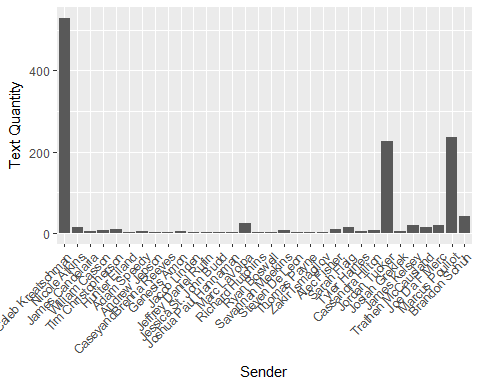
## Final Results

First off, I wanted to determine the quantity of texting that was done within each group. I did this by making a bar graph function and applying it to each group data frame.

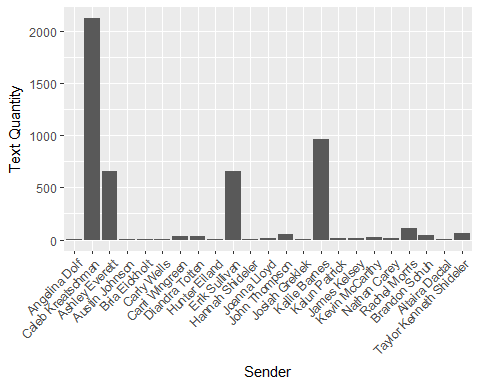
dfBar <- function(df) {ggplot() + geom\_bar(data = df, aes(sender)) + xlab("Sender") + ylab("Text Quantity") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))}  
  
dfBar(gfData)



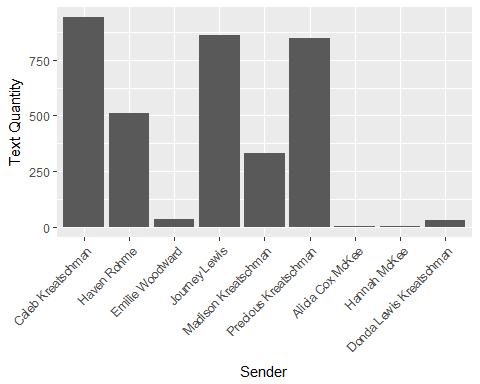
dfBar(navyData)



dfBar(friendsData)



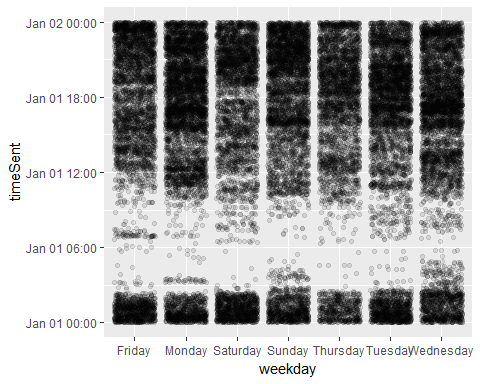
dfBar(familyData)



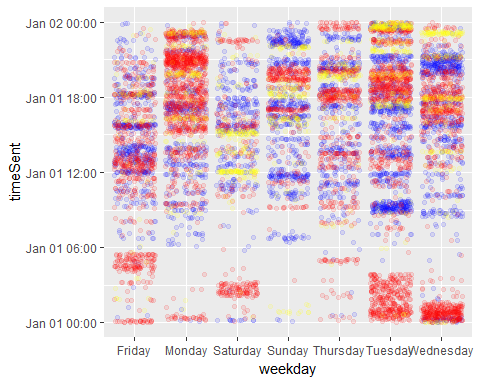
It seems that I’m the more talkative one between my girlfriend and I. I also note that, among my friends, most converstations are brief aside from Ashley and Erik standing out very strongly. Among family, my mother and younger sister are the winners. And, among my Navy acquaintances, Jordan and Marcus stand out.

Next, I was curious what times of the day and week most of my messages were concentrated. I used a jittergraph with the weekday on the x-axis and the time sent on the y-axes. Once again, I utilized functions. As my messages to the navy, family, and friends groups were sparse relative to my girlfriend, I combined them all into a single plot.

jitterGraph <- function(df){  
 ggplot() + geom\_jitter(data = df, aes(x = weekday, y = timeSent), alpha = 0.1)  
}  
#Plus another one that allows all of my category inputs:  
jitterGraph2 <- function(df,df2, df3){  
 tempDf <- ggplot() +  
 geom\_jitter(data = df, aes(x = weekday, y = timeSent), alpha = 0.1, color = "blue") +  
 geom\_jitter(data = df2, aes(x = weekday, y = timeSent), alpha = 0.1, color = "red") +  
 geom\_jitter(data = df3, aes(x = weekday, y = timeSent), alpha = 0.1, color = "yellow")  
 tempDf + scale\_fill\_discrete(breaks = c("1","2","3"))  
}  
jitterGraph(gfData)



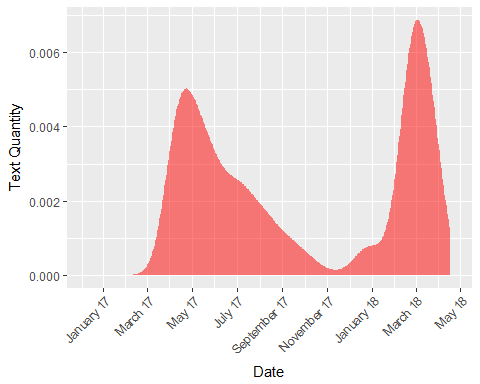
jitterGraph2(familyData, friendsData, navyData)



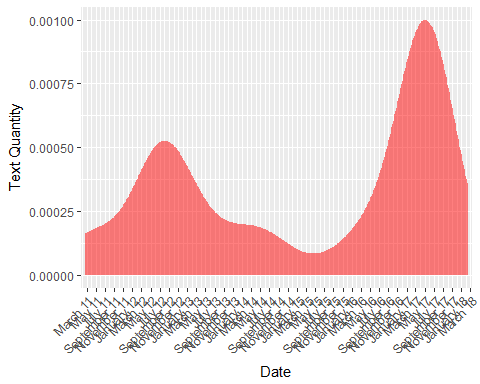
No real surpises there. I don’t tend to message while I’m sleeping. Though there were some post-midnight spikes among the ‘friends’ group on Tuesdays, Wednesdays, and Saturdays.

Next, I wanted to see how the message density corresponded to a longer timeline. For this I used density plots.

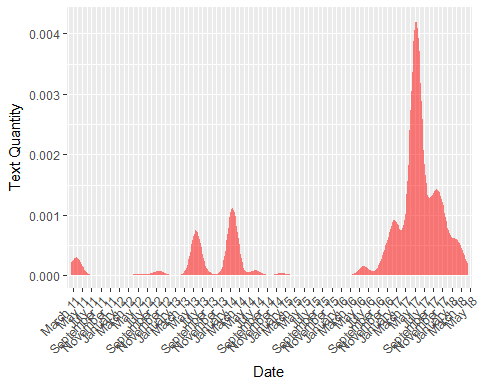
densityGraph <- function(df) {  
 ggplot(df, aes(x = date)) +  
 stat\_density(alpha = 0.5, fill = "red") +  
 scale\_x\_date(date\_breaks = "2 month",  
 date\_labels = "%B %y", expand=c(0,30)) + xlab("Date") + ylab("Text Quantity") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
}  
densityGraph(gfData)



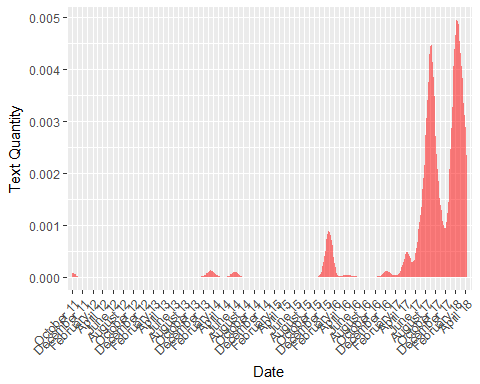
densityGraph(navyData)



densityGraph(friendsData)



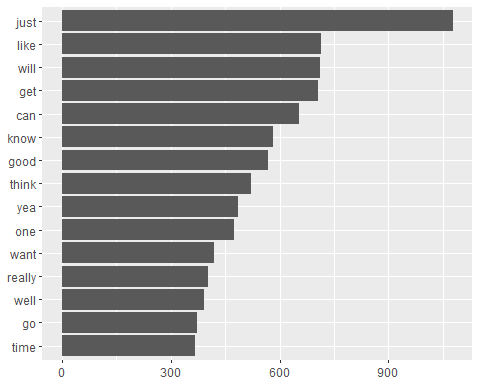
densityGraph(familyData)

 The graph for my girlfriend is very interesting, considering that she left for deployment in march of 2017 and returned in October of 2017. She left for another deployment in February of 2018 and returned at the end of April. It would see that our messages tend to spike initially when we’re separated, but calm down once we’ve adjusted. My conversation with Navy folks, which most likely refers to the conversations with Marcus and Jordan, were most heavily concentrated in May. I believe I was giving advice to Marcus about boats during that time, but I don’t know about Jordan. The friends group seems a bit more consistent, but also has a spike in May. Family has a strong spike from June-September, then sharply declines to November. But quickly rises again at the end of December. The second spike correlates with my time in the MSSA program, so it’s possible that I often spoke with my family about that. The rise in conversation with friends, navy, and (to some extent) family are almost inversely proportional to a decline in conversation with my girlfriend during her first deployment. I don’t remember anything specific happening at that point, so perhaps I should ask her.

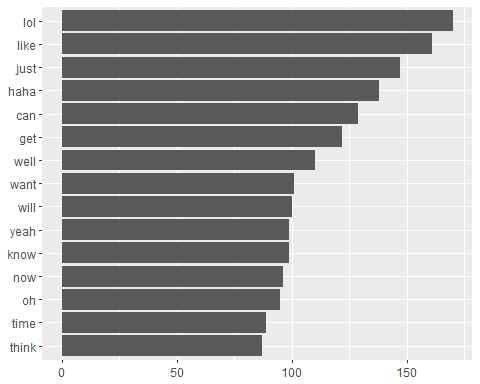
Finally, I wanted to identify the words used the most by each group - with myself in a separate graph.

topWords <- function(group) {  
 words.j <- tokenize\_words(finalData$message[finalData$sender %in% group], stopwords = stopwords("en"))  
 wordsa.j <- words.j[[1]]  
 for (i in 2:length(words.j)){  
 wordsa.j <- c(wordsa.j, words.j[[i]])  
 }  
 words.j <- data.frame(word = wordsa.j, frq = 1)  
 words.j <- aggregate(. ~ word, words.j, sum)  
 words.j <- words.j[order(-words.j$frq),]  
 rm(wordsa.j, i)  
  
 p9j <- ggplot(head(words.j, n = 15), aes(y = frq, x = reorder(word, frq))) +  
 geom\_bar(stat = "identity") +  
 coord\_flip() +  
 theme(axis.title = element\_blank())  
 return(p9j)  
}  
  
#Top words by group:

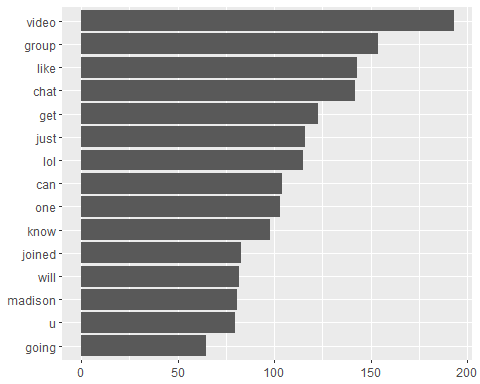
topWords(girlfriend)



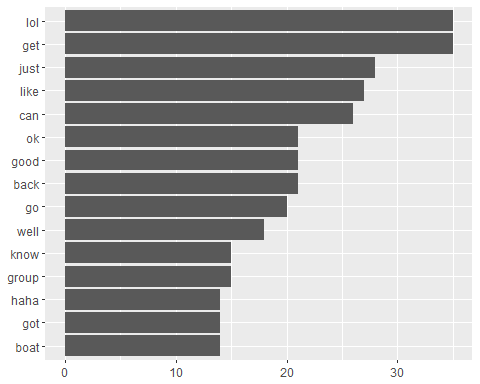
topWords(friends)



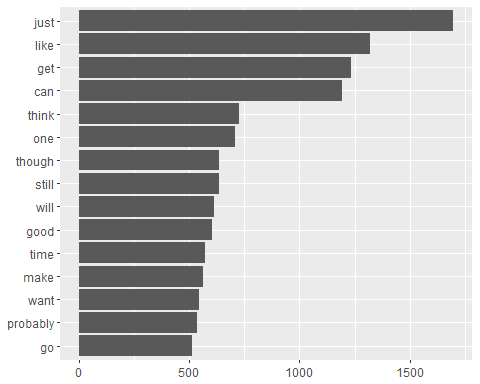
topWords(family)



topWords(navyFolks)



topWords(me)



It’s interesting that my girlfriend and I have such similar top words. Specifically, I never would have guessed that we both use the word ‘just’ so much.

In conclusion, this was a much more fascinating data set to work with than I initially anticipated. Cleaning and organizing that data was very challenging and I learned a lot about several packages that we didn’t use in class. I intend to keep working on this as a pet project after graduation as I know there are lots of spots where my code should be more efficient. Ultimately, I’d like to develop a simple program that allows a user to plug in their own archive file and receive a series of graphs detailing their message information.