Web Mining - IS 688

Sentiment Analysis On Dataset Containing Tweets on ISIS

Final project IS-688 Spring 2018

Project Report(Group E)

Guide - Prof. Christopher Markson

Team Members:

Sucharita Das
David Guardia
Sonya Hidar
Zhoujan Cai

Intro to the Analysis

This project is based on the interpretation of data retrieved from Twitter through the use of a sentiment analyzer in R. Twitter has become a popular medium across the world to express opinions on various topics. The data on this application is invaluable to governments, business, and data analyzers alike. Sentiment analysis was used to investigate the emotion behind tweets and group similarities between opinions on the internet on the topic of ISIS.

Sentiment analysis is typically used in data mining and business to improve the customer experience, improve marketing strategy, influence general attitude towards a topic or product, while also helping to mitigate communication crisis over social media. It has been become increasingly difficult to gauge opinions over the internet due to shorthand and issues with natural language, such as false negation.

Through the enhancement of sentiment analysis methodologies and algorithms, we are able to break through those barriers and understand the general opinion of internet users, enabling us to act quickly, if need be. Bar charts and pie charts were used to display the amount of similar emotions expressed in tweets and a word cloud was also used to visually represent the main words used to describe the topic at hand. Word clouds are great visual representations to display trends/patterns found in data that can otherwise be hidden in tabular data upon first glance.

This comes in handy when analyzing large datasets, as we did in this project. From our word cloud, the words such as “attack”, “kill”, “terror”, and “suicide” can be seen to have been mentioned by most users. These words indicate an overall dismal opinion on the topic.

Loading Libraries

About the Dataset

- The dataset used for this project contains two files one containing over 17,000 tweets from 100+ pro-ISIS fanboys called Isis tweets which contains the different time and date when the tweet was published. The dataset link and project files are in Shared Google Drive The other file contains over 122,000 tweets collected from across the world. This file captures data from tweets containing any of the following terms, with no further editing or selection:
- isis
- isil
- daesh
- islamicstate
- raqqa
- Mosul
- islamic state This dataset was chosen as it contains information from pro-ISIS groups and those of the general public.

```r
setwd('/Users/dguardia/R/NJIT-WebMining688/final_project')
isisTweets <- read.csv("./data/AboutIsis.csv", stringsAsFactor=FALSE)
allNewTweets <- read.csv("./data/IsisTweets.csv", stringsAsFactor=FALSE)

str(allNewTweets)

## 'data.frame':    17410 obs. of  8 variables:
##  $ name          : chr  "GunsandCoffee" "GunsandCoffee" "GunsandCoffee"
"GunsandCoffee" ...
##  $ username      : chr  "GunsandCoffee70" "GunsandCoffee70"
"GunsandCoffee70" "GunsandCoffee70" ...
##  $ description   : chr  "ENGLISH TRANSLATIONS: http://t.co/QLdJ0ftews"
"ENGLISH TRANSLATIONS: http://t.co/QLdJ0ftews" "ENGLISH TRANSLATIONS:
http://t.co/QLdJ0ftews" "ENGLISH TRANSLATIONS: http://t.co/QLdJ0ftews" ...
##  $ location      : chr  "" "" "" "" ...
##  $ followers     : int  640 640 640 640 640 640 640 640 640 640 ...
##  $ numberstatuses: int  49 49 49 49 49 49 49 49 49 49 ...
##  $ time          : chr  "1/6/2015 21:07" "1/6/2015 21:27" "1/6/2015 21:29"
"1/6/2015 21:37" ...
##  $ tweets        : chr  "ENGLISH TRANSLATION: 'A MESSAGE TO THE TRUTHFUL
IN SYRIA - SHEIKH ABU MUHAMMED AL MAQDISI: http://t.co/73xFszsj"|
__truncated__ "ENGLISH TRANSLATION: SHEIKH FATIH AL JAWLANI 'FOR THE PEOPLE
OF INTEGRITY, SACRIFICE IS  EASY' http://t.co/uqqz"| __truncated__ "ENGLISH
TRANSLATION: FIRST AUDIO MEETING WITH SHEIKH FATIH AL JAWLANI (HA):
http://t.co/TgXT1GdGw7 http://t.co/ZuE8eisze6" "ENGLISH TRANSLATION: SHEIKH
NASIR AL WUHAYSHI (HA), LEADER OF AQAP: 'THE PROMISE OF VICTORY':
http://t.co/3qg5d"| __truncated__ ...

str(isisTweets)

## 'data.frame':    122619 obs. of  5 variables:
##  $ name    : chr  "Sean Ferigan" "m.zakariyya" "ちゃんゆず" "chutney" ...
##  $ username: chr  "ferigan" "mzakariyya5" "yuzuchaaan777" "plainparatha"
...
##  $ tweetid : num  7.52e+17 7.52e+17 7.52e+17 7.52e+17 7.52e+17 ...
##  $ time    : chr  "7/11/2016 8:45:39 AM" "7/11/2016 8:45:39 AM" "7/11/2016
8:45:38 AM" "7/11/2016 8:45:38 AM" ...
##  $ tweets  : chr  "ISIS influence on the decline as terrorists lose
Twitter battles     - CNET http://www.cnet.com/news/isis-influ"|
__truncated__ "RT @AyishaBaloch: #IndiaISISandBangladesh And world can ALSO
not ignore the truth revealing india 's role in pr"| __truncated__
"@Laika_isis @wink_BF  テラリアもってないいいい" "RT @KabirTaneja: Anti-ISIS
volunteer fighting with the Kurds. things are getting strange on planet
Earth.  #Pok"| __truncated__ ...

head(isisTweets)

##                     name      username      tweetid                 time
## 1           Sean Ferigan       ferigan 7.524236e+17 7/11/2016 8:45:39 AM
## 2            m.zakariyya   mzakariyya5 7.524236e+17 7/11/2016 8:45:39 AM
## 3              ちゃんゆず yuzuchaaan777 7.524236e+17 7/11/2016 8:45:38 AM
```

```
## 4             chutney  plainparatha 7.524236e+17 7/11/2016 8:45:38 AM
## 5            ॐ□□□□ ॐ     dharam_vj 7.524236e+17 7/11/2016 8:45:37 AM
## 6 Dipendra Dipzo Khati DipendraDipzo 7.524236e+17 7/11/2016 8:45:36 AM
##
tweets
## 1
ISIS influence on the decline as terrorists lose Twitter battles     - CNET
http://www.cnet.com/news/isis-influence-twitter-on-the-decline-us-state-
department/#ftag=CAD590a51e
## 2
RT @AyishaBaloch: #IndiaISISandBangladesh And world can ALSO not ignore the
truth revealing india 's role in providin explosive to ISIS http…
## 3
@Laika_isis @wink_BF  テラリアもってないいいい
## 4
RT @KabirTaneja: Anti-ISIS volunteer fighting with the Kurds. things are
getting strange on planet Earth.  #PokemonGO https://t.co/ARdBQ4…
## 5 RT @MrsGandhi: It 's Urdu dailies not internet alone that 's turning
Muslims into terrorists #MustRead @tufailelif
http://www.dailyo.in/politics/muslims-radicalisation-isis-hyderabad-ramzan-
internet-war-of-badr-prophet-muhammad-orlando-shooting/story/1/11599.html
## 6 RT @MrsGandhi: It 's Urdu dailies not internet alone that 's turning
Muslims into terrorists #MustRead @tufailelif
http://www.dailyo.in/politics/muslims-radicalisation-isis-hyderabad-ramzan-
internet-war-of-badr-prophet-muhammad-orlando-shooting/story/1/11599.html

## Create a date column
isisTweets$date <-as.Date(isisTweets$time, "%d/%m/%Y %H:%M:%S")
##head(isisTweets)


## get the date and info for the new file using lubridade we need to explain
each library
allNewTweets$created <- mdy_hm(allNewTweets$time)
allNewTweets$created <- with_tz(allNewTweets$created, "America/New_York")
##allNewTweets$time
```

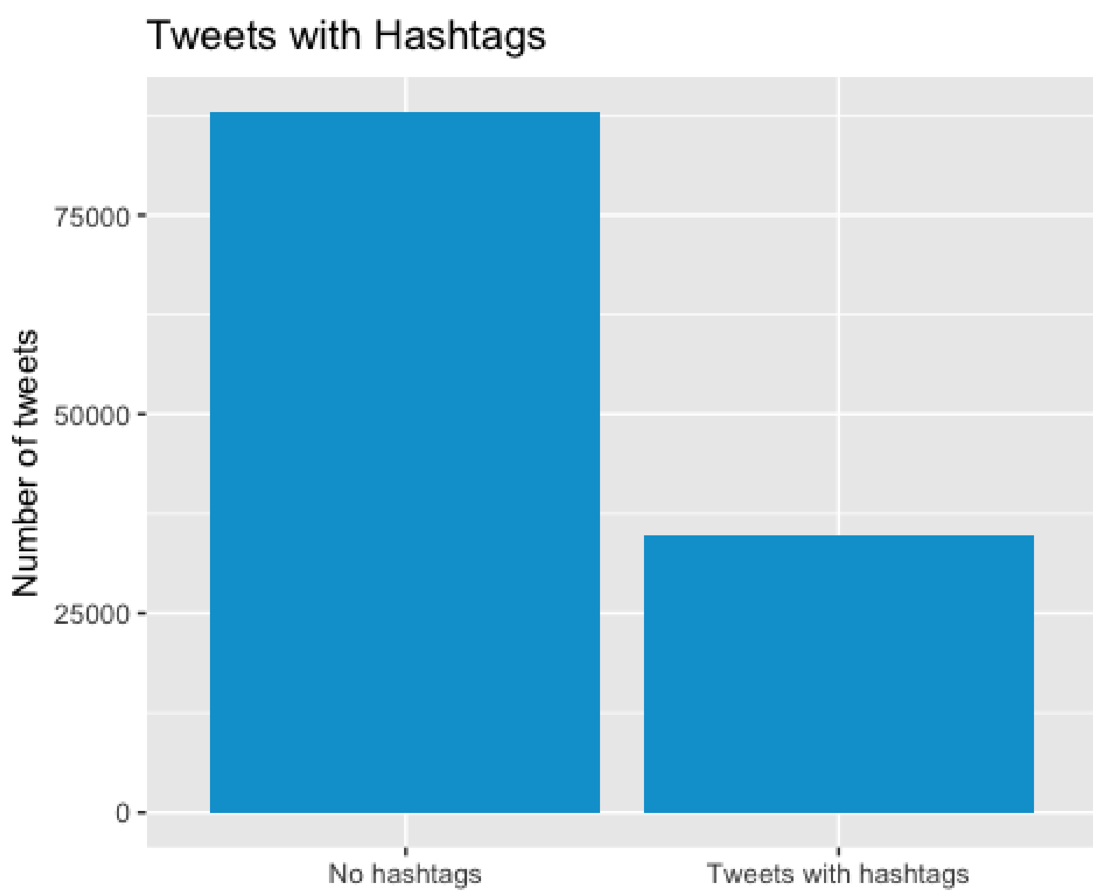
find all the Unique with hashtags

```
isisTweets %>%
  mutate(isHT = grepl("#", isisTweets$tweets)) -> aboutIsisHahstags

isisTweets %>%
  summarize("Tweets with No hashtags" = nrow(subset(aboutIsisHahstags, isHT
== FALSE)),
            "Tweets with Hashtags" = nrow(subset(aboutIsisHahstags, isHT ==
TRUE))) %>%
              formattable(align = "c")
```

Hashtag	
Tweets with No hashtags	Tweets with Hashtags
87888	34731

Here the all the hashtags graph

```
ggplot(isisTweets, aes(factor(grepl("#", isisTweets$tweets)))) +
  geom_bar(fill = "#00a0d3") +
  theme(legend.position="none", axis.title.x = element_blank()) +
  ylab("Number of tweets") +
  ggtitle("Tweets with Hashtags") +
  scale_x_discrete(labels=c("No hashtags", "Tweets with hashtags"))
```



Libraries

All the different package used

Function Name	Library	Description
ggplot	ggplot2	It initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.
aes	ggplot2	Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geoms.
factor	base	The function factor is used to encode a vector as a factor (the terms 'category' and 'enumerated type' are also used for factors).
grep1	base	It searches for matches to argument pattern within each element of a character vector: they differ in the format of and amount of detail in the results.

Tweets and Retweets

The bellow bar chart indicates that most tweets analyzed did not use hashtags. Hashtags are denoted by a "#" sign on twitter and is used to group together similar tweets. For example, if multiple people used the hashtag "#war", a user would presented these tweets when searching for "war". This could indicate that most people did not respond to any particular current event, news, or group topic, and instead took to twitter to voice a general statement or opinion ISIS related.

```
isisTweets %>%
  mutate(isRT = grepl("^\\RT\\b", isisTweets$tweets)) -> aboutIsis

isisTweets %>%
  summarize("Tweets" = nrow(subset(aboutIsis, isRT == FALSE)),
            "Retweets" = nrow(subset(aboutIsis, isRT == TRUE)))
%>%formattable(align = "c")
```

Tweets

Retweets

36702

85917

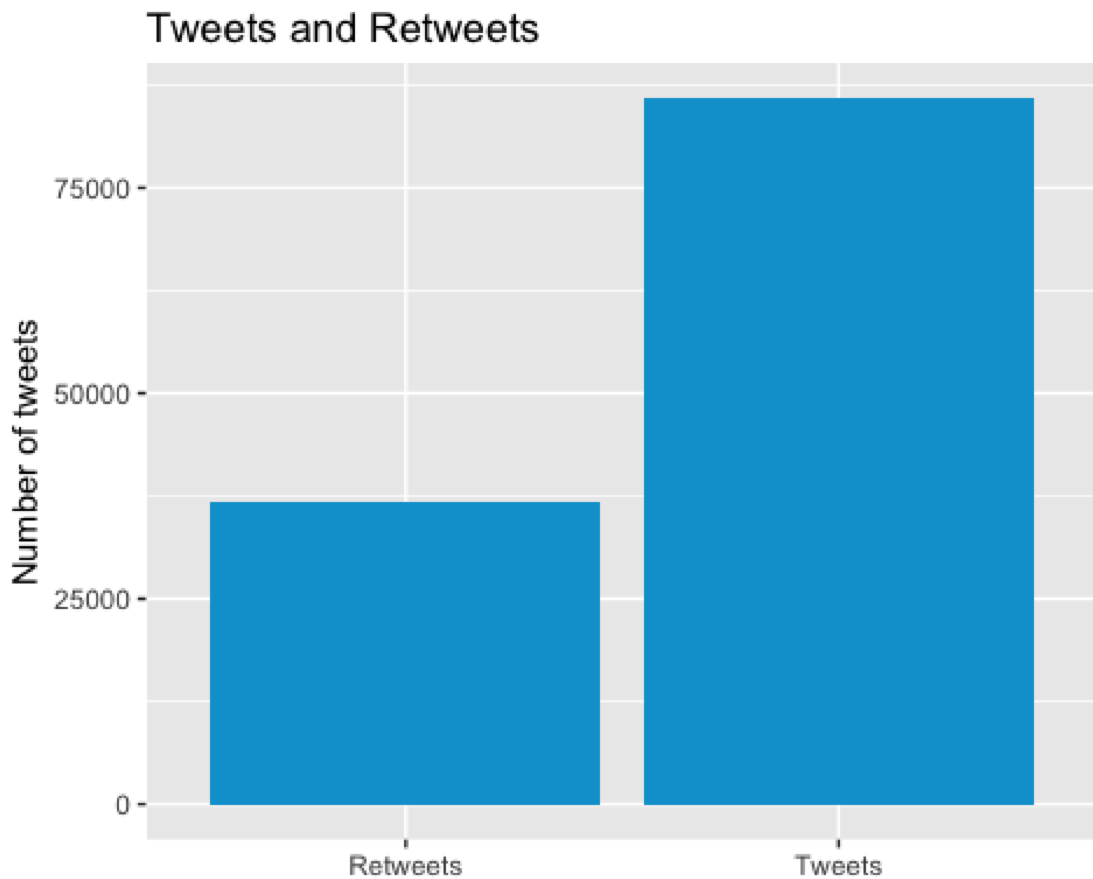
Libraries

All the different package used

Function Name	Library	Description
mutate	dplyr	Mutate adds new variables and preserves existing; transmute drops existing variables.
summarize	hmisc	It is used for producing stratified summary statistics and storing them in a data frame for plotting.

Graph of Retweets

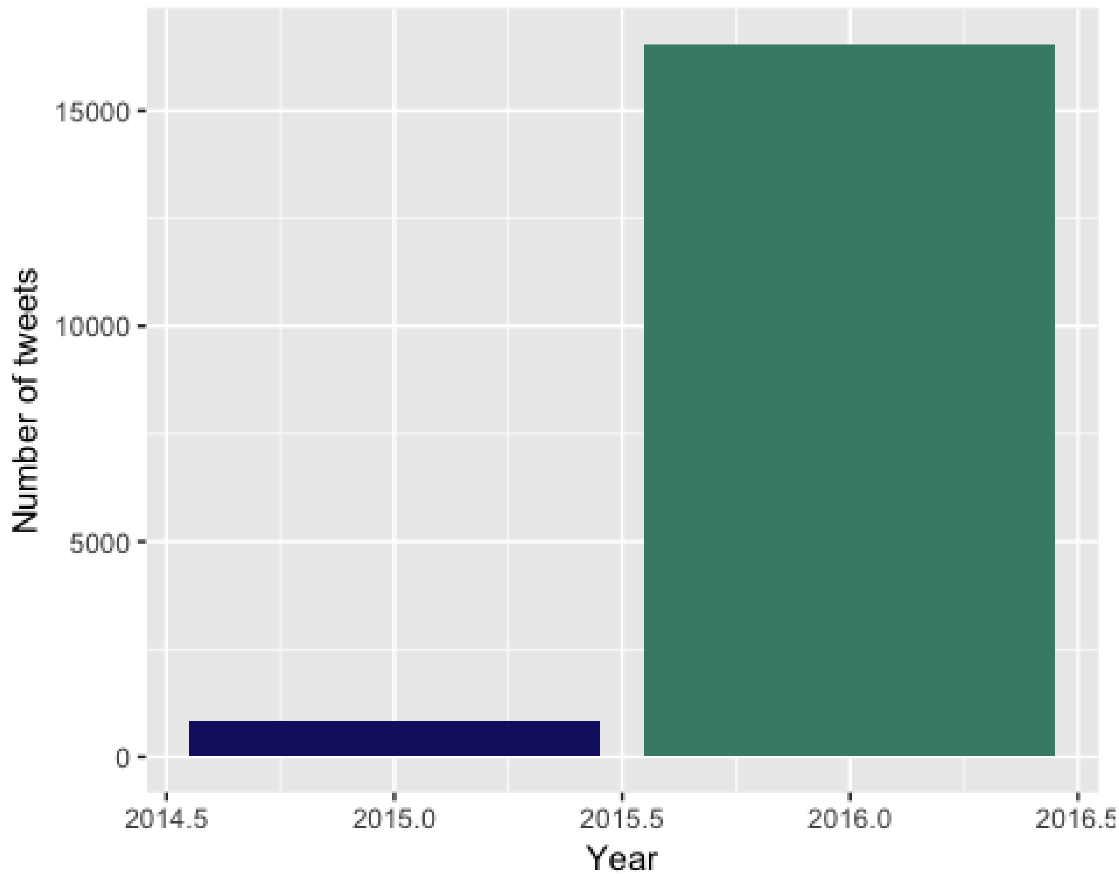
```
ggplot(isisTweets, aes(factor(grepl("^\\RT\\b", isisTweets$tweets)))) +
  geom_bar(fill = "#00a0d3") +
  theme(legend.position="none", axis.title.x = element_blank()) +
  ylab("Number of tweets") +
  ggtitle("Tweets and Retweets") +
  scale_x_discrete(labels=c("Retweets", "Tweets"))
```



This shows that most tweets were crafted individually by users and not copied from another user's statements, which are called "re-tweets". This means that a variety of unique natural language statements were passed through the sentiment analyzer.

Graph by year

```
ggplot(data = allNewTweets, aes(x = year(created))) +
  geom_bar(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Year") + ylab("Number of tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```

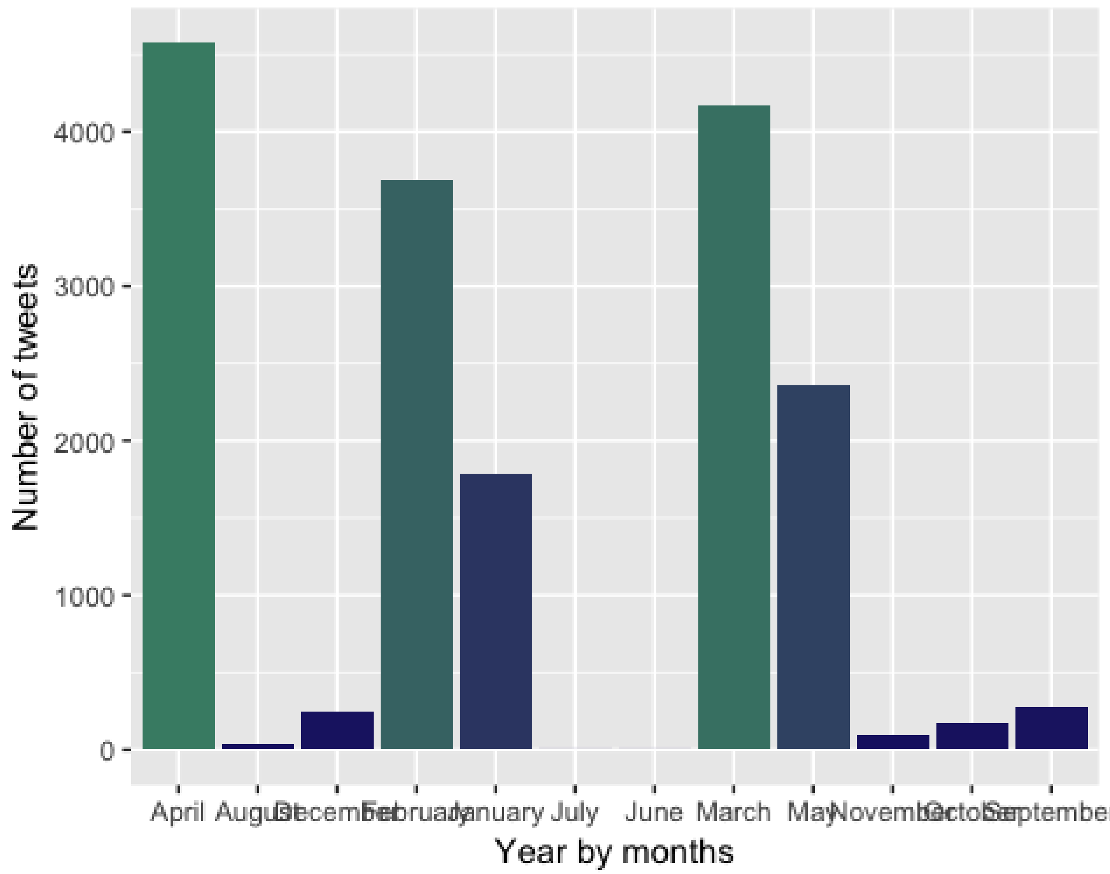


*****

The above graph represents the amount of tweets collected from each year. As can be seen, the majority of tweets were collected during 2016, particularly during the January - July period.

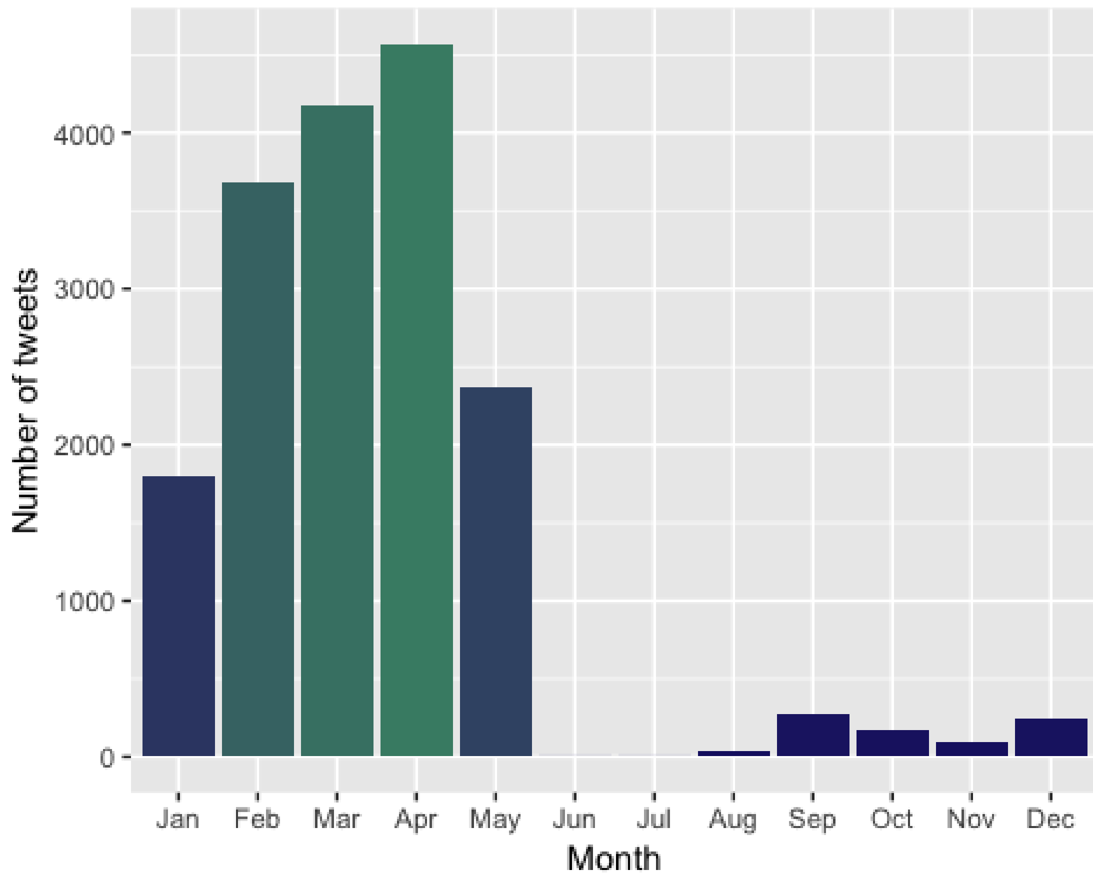
Graph by month

```
ggplot(data = allNewTweets, aes(x = months.Date(created))) +
  geom_bar(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Year by months") + ylab("Number of tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```



Graph show the volume of Tweets by month using ggplot

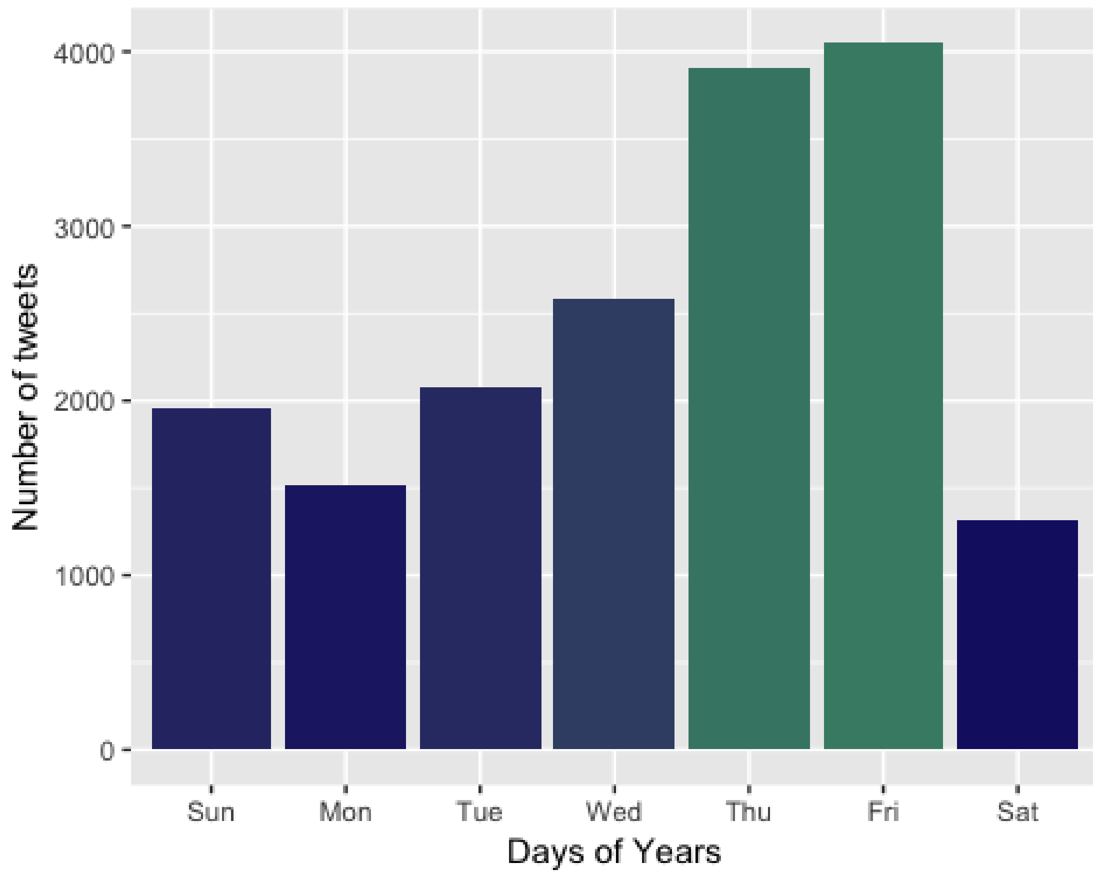
```
ggplot(data = allNewTweets, aes(x = month(created, label = TRUE))) +
  geom_bar(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Month") + ylab("Number of tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```




```
chisq.test(table(month(allNewTweets$created, label = TRUE)))

##
##  Chi-squared test for given probabilities
##
## data:  table(month(allNewTweets$created, label = TRUE))
## X-squared = 24527, df = 11, p-value < 2.2e-16
```

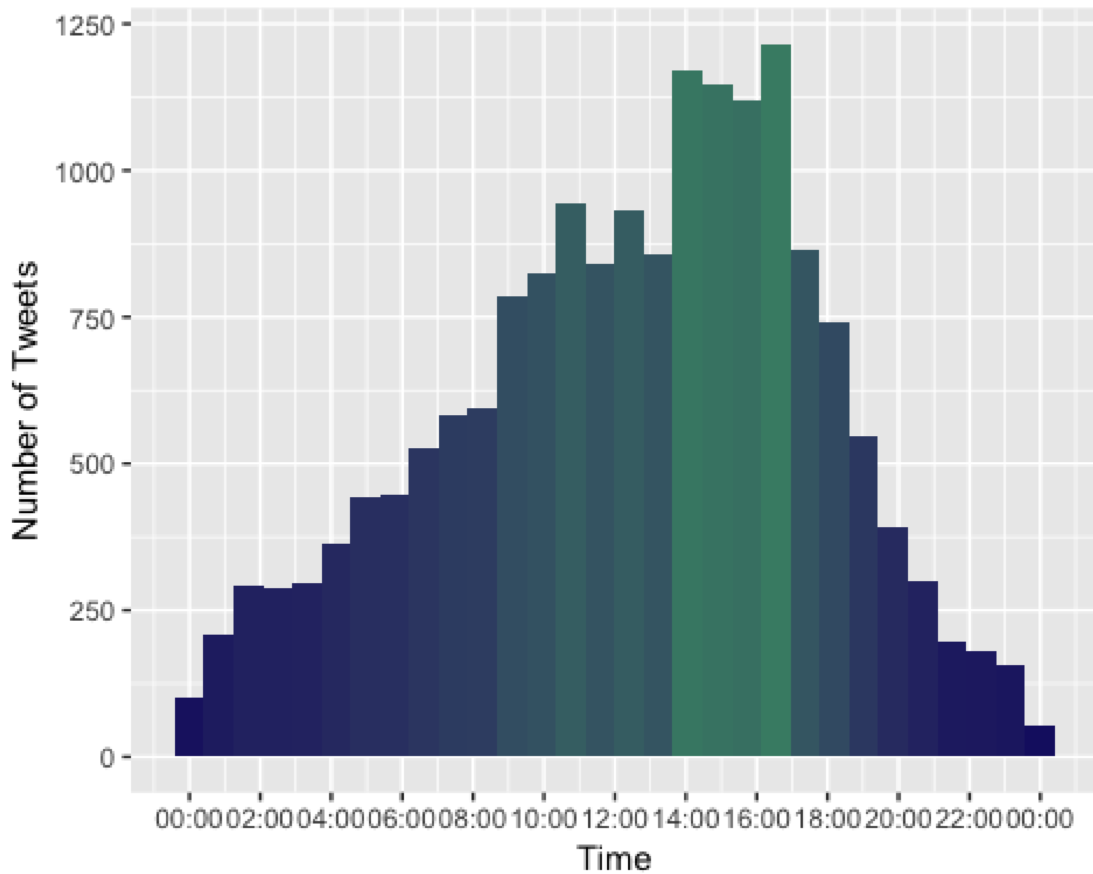
Graph showing by week

```
ggplot(data = allNewTweets, aes(x = wday(created, label = TRUE))) +
  geom_bar(aes(fill = ..count..)) +
  theme(legend.position = "none") +
  xlab("Days of Years") + ylab("Number of tweets") +
  scale_fill_gradient(low = "midnightblue", high = "aquamarine4")
```



Graph show twitter volume by hour

```
allNewTweets$timeonly <- as.numeric(allNewTweets$created -
trunc(allNewTweets$created, "days"))
class(allNewTweets$timeonly) <- "POSIXct"

ggplot(data = allNewTweets, aes(x = timeonly)) +
    geom_histogram(aes(fill = ..count..)) +
    theme(legend.position = "none") +
    xlab("Time") + ylab("Number of Tweets") +
    scale_x_datetime(breaks = date_breaks("2 hours"),
                     label = date_format("%H:00")) +
    scale_fill_gradient(low = "midnightblue", high = "aquamarine4")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Show word cloud for Twitter words

```
#remove special characters, in particular, the symbol @
removeHandles <- str_replace_all(allNewTweets$tweets, "@\\w+", "")
wordCorpus <- Corpus(VectorSource(removeHandles))
wordCorpus <- tm_map(wordCorpus, removePunctuation)
wordCorpus <- tm_map(wordCorpus, content_transformer(tolower))
wordCorpus <- tm_map(wordCorpus, removeWords, stopwords("english"))
wordCorpus <- tm_map(wordCorpus, removeWords, c("amp", "2yo", "3yo", "4yo",
"https"))
wordCorpus <- tm_map(wordCorpus, stripWhitespace)

set.seed(2018)
wordcloud(words = wordCorpus, scale = c(3, 1), max.words = 100, random.order
= FALSE,
          rot.per = 0.35, use.r.layout = FALSE, colors = pal)
```

The word cloud generated displays the word “isis” as most used, which is the topic at hand. Second, the words “killed” and “syria” were used, and are color coded to show that these are the second most used words in the twitter dataset. They are followed by the keywords “islamic”, “aleppo”, “state”, “will”, “iraq”, and “army”.

Produce a word cloud of frequent twitter mentions

Libraries

All the different package used

Function Name	Library	Description
str_extract_all	stringr	Extracts all pieces of a string that matches a pattern.
tm_map	tm	Interface to apply transformation functions (also denoted as mappings) to corpora.

```
tweeterFriends <- str_extract_all(allNewTweets$tweets, "@\\w+")
friendsCorpus <- VCorpus(VectorSource(tweeterFriends))
```

```r
friendsCorpus <- tm_map(friendsCorpus, content_transformer(tolower))
friendsCorpus <- tm_map(friendsCorpus, removeWords, stopwords("english"))
custom_words_to_remove <- c("character")
friendsCorpus <- tm_map(friendsCorpus, removeWords, custom_words_to_remove)
inspect(DocumentTermMatrix(friendsCorpus))
```

```
## <<DocumentTermMatrix (documents: 17410, terms: 3236)>>
## Non-/sparse entries: 12198/56326562
## Sparsity           : 100%
## Maximal term length: 18
## Weighting          : term frequency (tf)
## Sample             :
##        Terms
## Docs    @7layers_ @conflicts @didyouknowvs @maghrebiqm @nidalgazaui
##   12947         0          0             0           0            0
##   15158         0          0             0           0            0
##   15166         0          0             0           0            1
##   16247         0          0             0           0            1
##   16248         1          0             0           0            1
##   17054         1          0             0           0            1
##   39            0          0             0           0            0
##   488           0          0             0           0            0
##   5834          0          0             0           0            0
##   9146          0          0             0           0            0
##        Terms
## Docs    @ramiallolah @scotsmaninfidel @sparksofirhabi3 @uncle_samcoco
##   12947            0                0                0              0
##   15158            1                0                0              0
##   15166            2                0                0              0
##   16247            1                0                0              0
##   16248            0                0                0              0
##   17054            1                0                0              0
##   39               0                0                0              0
##   488              0                0                0              0
##   5834             0                0                0              0
##   9146             0                0                0              0
##        Terms
## Docs    @warreporter1
##   12947             0
##   15158             0
##   15166             0
##   16247             0
##   16248             0
##   17054             0
##   39                0
##   488               0
##   5834              0
##   9146              0
```

```r
tdm <- TermDocumentMatrix(friendsCorpus)

tdm.matrix <- as.matrix(tdm)
tdm.rs <- sort(rowSums(tdm.matrix), decreasing = TRUE)
tdm.df <- data.frame(word = names(tdm.rs), freq = tdm.rs, stringsAsFactors =
FALSE)
as_tibble(tdm.df)

## # A tibble: 3,236 x 2
##    word               freq
##  * <chr>             <dbl>
##  1 @ramiallolah        578
##  2 @nidalgazaui        341
##  3 @warreporter1       256
##  4 @7layers_           116
##  5 @scotsmaninfidel     79
##  6 @sparksofirhabi3     76
##  7 @conflicts           72
##  8 @didyouknowvs        72
##  9 @maghrebiqm          72
## 10 @uncle_samcoco       70
## # ... with 3,226 more rows

set.seed(123)
wordcloud(words = tdm.df$word, freq = tdm.df$freq, min.freq = 10, scale =
c(4, .7),
          max.words = 40, random.order=FALSE, rot.per=0.10, use.r.layout =
FALSE,
          colors=pal)
```

The above word cloud displays the most mentioned twitter users in tweets. This shows that the username "@ramiallolah" was mentioned or re-tweeted the most among user's statements on ISIS, signifying that this user holds great influence on general opinions or invokes responses from twitter. Upon investigation, these users are partners for local news agencies.

Most Mentions	
@ramiallolah	1578
@nidalgazaui	3413
@warreporter1	2564
@7layers_ 1	165
@scotsmaninfidel	796
@sparksofirhabi3	767

@conflicts	728
@didyouknowvs	729
@maghrebiqm	7210

Splited a column into tokens using the tokenizers package, splitting the table into one-token-per-row. This function supports non-standard evaluation through the tidyeval framework.

```
tidy_tweets <- isisTweets %>%
  group_by(name,username,tweetid)%>%
  mutate(ln=row_number())%>%
  unnest_tokens(word, tweets)%>%
  ungroup()
head(tidy_tweets, 5)

## # A tibble: 5 x 7
##   name         username tweetid time                 date          ln word
##   <chr>        <chr>      <dbl> <chr>                <date>     <int> <chr>
## 1 Sean Ferigan ferigan  7.52e17 7/11/2016 8:45:3… 2016-11-07     1 isis
## 2 Sean Ferigan ferigan  7.52e17 7/11/2016 8:45:3… 2016-11-07     1 influe…
## 3 Sean Ferigan ferigan  7.52e17 7/11/2016 8:45:3… 2016-11-07     1 on
## 4 Sean Ferigan ferigan  7.52e17 7/11/2016 8:45:3… 2016-11-07     1 the
## 5 Sean Ferigan ferigan  7.52e17 7/11/2016 8:45:3… 2016-11-07     1 decline
```

Count the words

```
isisTweetsWords <- tidy_tweets %>%
  count(word, sort=TRUE)
head(isisTweetsWords, 5)

## # A tibble: 5 x 2
##   word       n
##   <chr>  <int>
## 1 isis  116434
## 2 rt     86467
## 3 the    68001
## 4 in     50096
## 5 to     40554
```

Find the sentiment of the tweets

```
tweets_sentiment <- tidy_tweets%>%
  inner_join(get_sentiments("bing"))
```

```
## Joining, by = "word"
```

```
head(tweets_sentiment)
```

```
## # A tibble: 6 x 8
##   name         username     tweetid time   date          ln word  sentiment
##   <chr>        <chr>          <dbl> <chr> <date>      <int> <chr> <chr>
## 1 Sean Ferigan ferigan      7.52e17 7/11… 2016-11-07      1 decl… negative
## 2 Sean Ferigan ferigan      7.52e17 7/11… 2016-11-07      1 lose  negative
## 3 Sean Ferigan ferigan      7.52e17 7/11… 2016-11-07      1 decl… negative
## 4 m.zakariyya  mzakariyya5  7.52e17 7/11… 2016-11-07      1 igno… negative
## 5 m.zakariyya  mzakariyya5  7.52e17 7/11… 2016-11-07      1 expl… negative
## 6 chutney      plainparatha 7.52e17 7/11… 2016-11-07      1 stra… negative
```

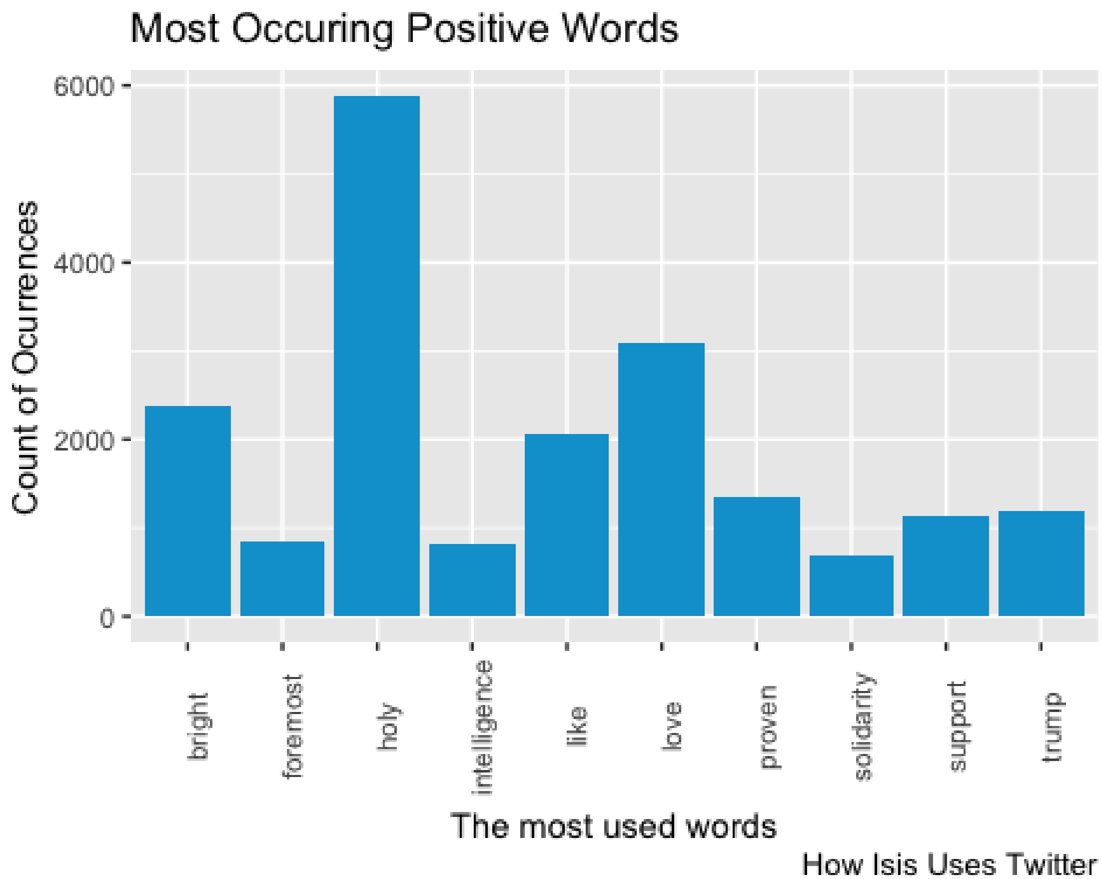
WordCloud of all words in ISIS tweets

```
totalTwiterWordCloud <- tweets_sentiment%>%
  count(word, sort=TRUE)

wordcloud(totalTwiterWordCloud$word,
          totalTwiterWordCloud$n,
          min.freq =100,
          scale=c(4, .8),
          random.order = FALSE,
          random.color = FALSE,
          colors = pal)
```

Finding the sentiment

This word cloud was generated using the tweets on ISIS of the general public from across the world. The word “attack” is most prominent and “suicide”, “terror”, “worst” and “holy” are among the most common words used to describe ISIS. These words display a general hate and dismal attitude towards the topic at hand.

```
pos_neg <- tweets_sentiment%>%
  count(word, sentiment, sort=TRUE)

pos_neg %>%
  filter(sentiment=='positive')%>%
  head(10) %>%
  ggplot(aes(x=word, y=n))+
  geom_bar(stat="identity", fill="#00a0d3")+
  theme(axis.text.x=element_text(angle=90))+
  labs(title="Most Occuring Positive Words",
       y="Count of Ocurrences",
       x="The most used words",
       caption="How Isis Uses Twitter")
```



List all positives words

```
positiveWord <- pos_neg %>%
  filter(sentiment=='positive')
head(positiveWord)

## # A tibble: 6 x 3
##   word   sentiment     n
##   <chr>  <chr>     <int>
## 1 holy   positive   5879
## 2 love   positive   3080
## 3 bright positive   2386
## 4 like   positive   2056
## 5 proven positive   1339
## 6 trump  positive   1186
```

create just positive cloud words

```
wordcloud(positiveWord$word,
          positiveWord$n,
          min.freq =100,
          scale=c(4, .8),
          random.order = FALSE,
```

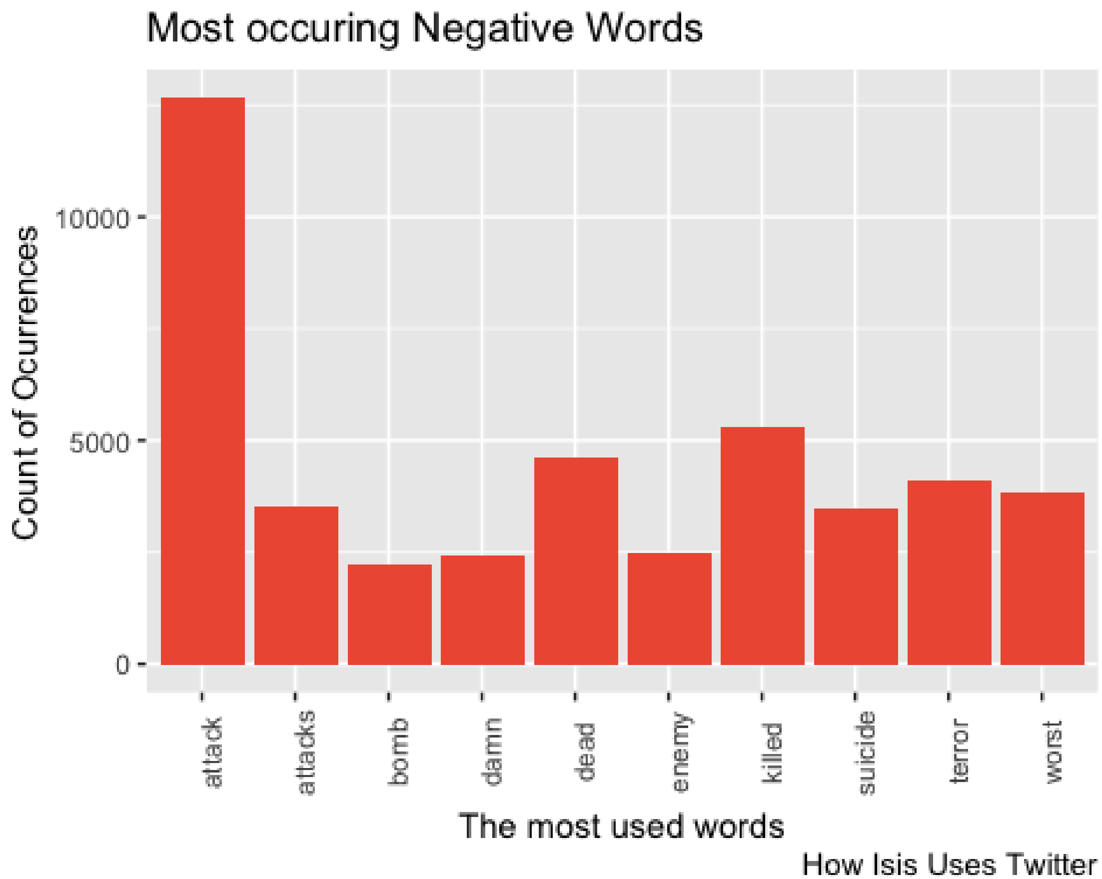
```
            random.color = FALSE,
            colors = pal)
```

The words in the above word cloud were the most "positive" words included in tweets. However, given that the topic is about "ISIS", these words were most likely typed as a show of support for those who have suffered. For example, the word "holy" is the most used positive word to describe ISIS, but upon further analysis, we can see that the overall attitude in the tweet containing this word is most likely negative. One example includes, "RT @xeni: Dude ISIS is bombing Muslim people in Muslim communities during the Muslim holy month of Ramadan how is ISIS Muslim no they're…."

Find all negative words to make a word cloud

```
#Find all negative words to make a word cloud

pos_neg %>%
  filter(sentiment=='negative')%>%
  head(10) %>%
  ggplot(aes(x=word,y=n))+geom_bar(stat="identity",fill="tomato2")+
  theme(axis.text.x=element_text(angle=90))+
  labs(title="Most occuring Negative Words",
       y="Count of Ocurrences",
```

```
       x="The most used words",
       caption="How Isis Uses Twitter")
```



Wordcloud for all negative words

List all negative words

```
negativeWord <- pos_neg %>%
  filter(sentiment=='negative')
head(negativeWord)

## # A tibble: 6 x 3
##   word    sentiment     n
##   <chr>   <chr>     <int>
## 1 attack  negative  12658
## 2 killed  negative   5324
## 3 dead    negative   4602
## 4 terror  negative   4113
## 5 worst   negative   3842
## 6 attacks negative   3506
```

Create just negative cloud words

```
wordcloud(negativeWord$word,
          negativeWord$n,
```

```
           min.freq =100,
           scale=c(4, .8),
           random.order = FALSE,
           random.color = FALSE,
           colors =pal)
```

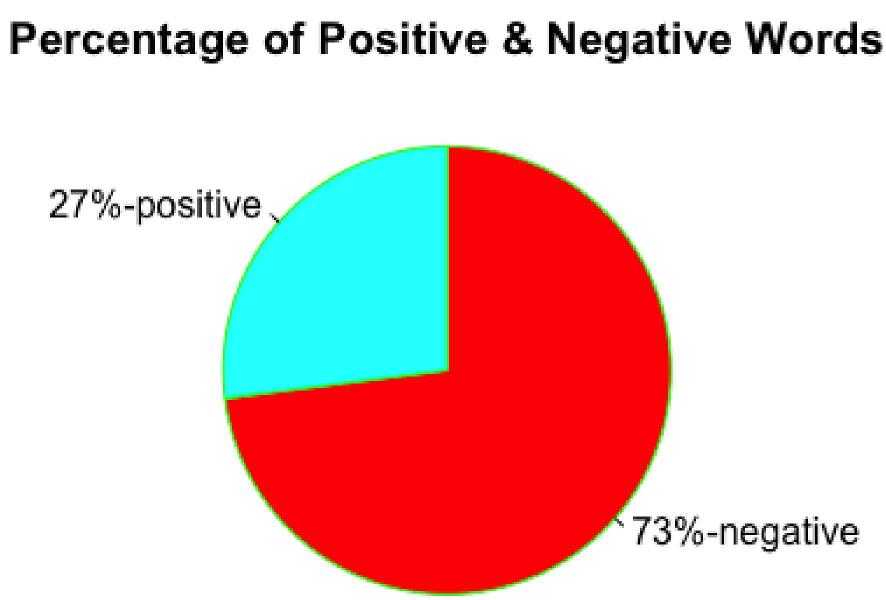
Find the percentage of the Positive vs Negative

```
perc <- tweets_sentiment%>%
  count(sentiment)%>%
    mutate(total=sum(n))%>%
      group_by(sentiment)%>%
        mutate(percent=round(n/total,2)*100)%>%
          ungroup()

label <-c(paste(perc$percent[1],'%','-',perc$sentiment[1],sep=''),
          paste(perc$percent[2],'%','-',perc$sentiment[2],sep=''))

head(perc)

## # A tibble: 2 x 4
##   sentiment      n  total percent
##   <chr>      <int>  <int>   <dbl>
```

```
## 1 negative  110892 151727       73
## 2 positive   40835 151727       27
```

Pie Chart from data frame with Appended Sample Sizes

```
pie(perc$percent, labels=label,
    col = rainbow(length(perc$percent)),
    border = "green",
    clockwise = TRUE,
    main="Percentage of Positive & Negative Words",
    radius = 1)
```



Compare Positive Negative

Positive

```
pos<-pos_neg %>% filter(sentiment=='positive')
head(pos)
```

```
## # A tibble: 6 x 3
##   word  sentiment     n
##   <chr> <chr>     <int>
## 1 holy  positive   5879
## 2 love  positive   3080
```

```
## 3 bright positive   2386
## 4 like   positive   2056
## 5 proven positive   1339
## 6 trump  positive   1186
```

Negative

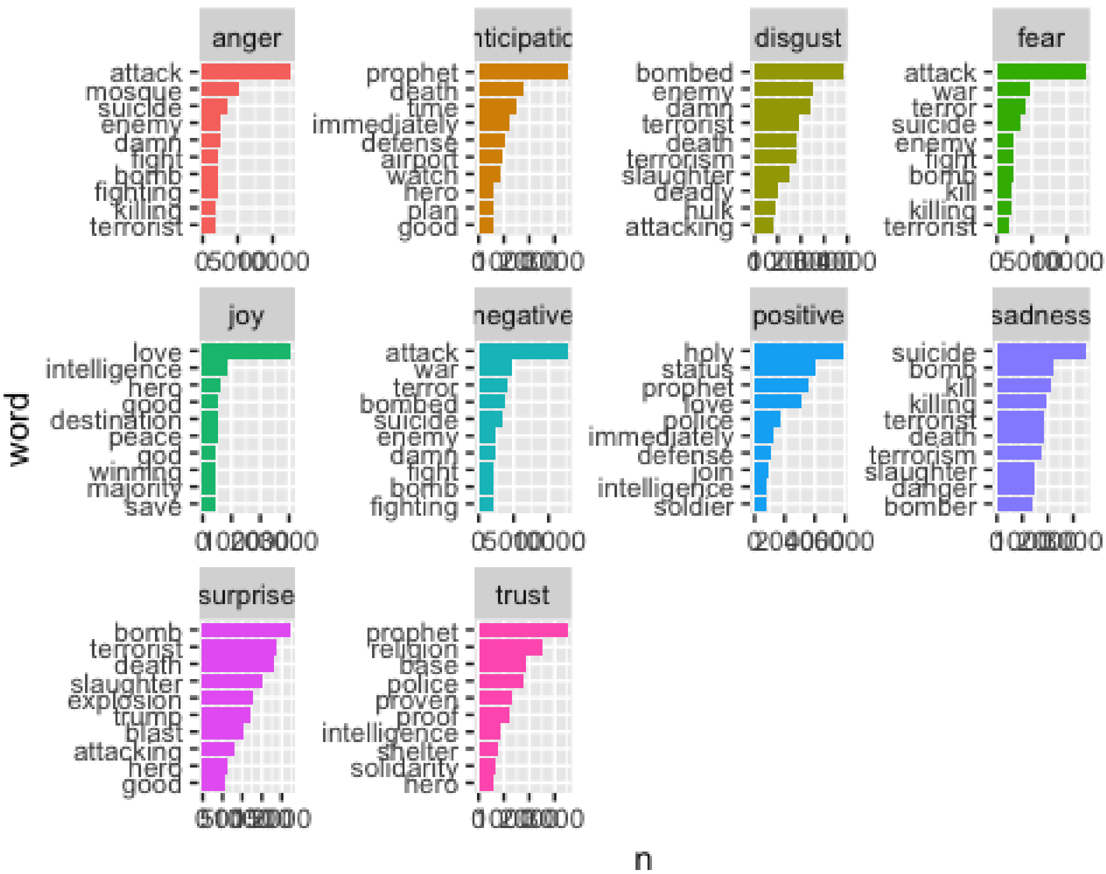
```
neg<-pos_neg %>% filter(sentiment=='negative')
head(neg)
```

```
## # A tibble: 6 x 3
##   word    sentiment     n
##   <chr>   <chr>     <int>
## 1 attack  negative  12658
## 2 killed  negative   5324
## 3 dead    negative   4602
## 4 terror  negative   4113
## 5 worst   negative   3842
## 6 attacks negative   3506
```

Get the sentimen using nrc

```
tidy_tweets%>%
  inner_join(get_sentiments("nrc")) %>%
    count(word,sentiment) %>%
      group_by(sentiment)%>%
        top_n(10)%>%
          ungroup() %>%
            mutate(word=reorder(word,n))%>%
              ggplot(aes(x=word,y=n,fill=sentiment)) +
              geom_col(show.legend = FALSE) +
              facet_wrap(~ sentiment, scales = "free") +
              coord_flip()
```

```
## Joining, by = "word"

## Selecting by n
```



Libraries

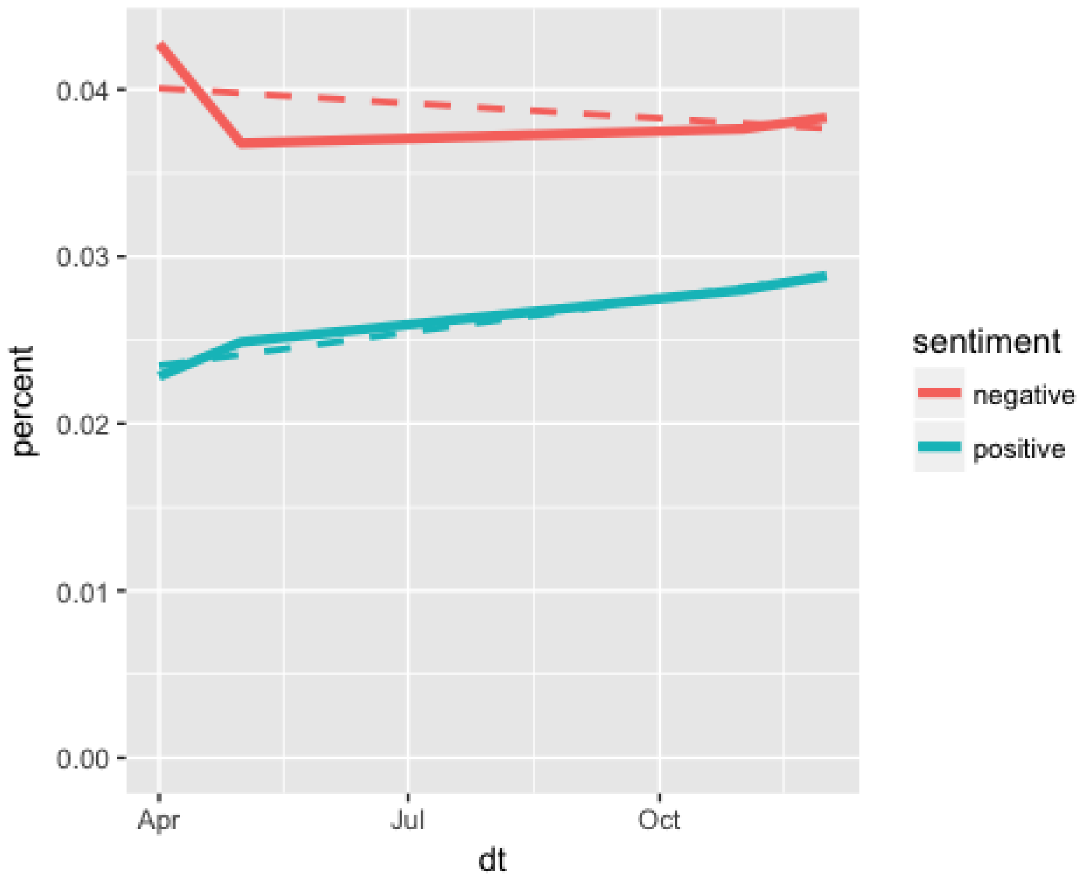
All the different package used

Function Name	Library	Description
get_sentiments	tidytext	Get specific sentiment lexicons in a tidy format, with one row per word, in a form that can be joined with a one-word-per-row dataset.

Positive & Negative Words over time

```
sentiment_by_time <- tidy_tweets %>%
  mutate(dt = floor_date(date, unit = "month")) %>%
  group_by(dt) %>%
  mutate(total_words = n()) %>%
  ungroup() %>%
  inner_join(get_sentiments("nrc"))

## Joining, by = "word"
```

```
sentiment_by_time %>%
  filter(sentiment %in% c('positive','negative')) %>%
  count(dt,sentiment,total_words) %>%
  ungroup() %>%
  mutate(percent = n / total_words) %>%
  ggplot(aes(x=dt,y=percent,col=sentiment,group=sentiment)) +
  geom_line(size = 1.5) +
  geom_smooth(method = "lm", se = FALSE, lty = 2) +
  expand_limits(y = 0)
```



Word association for all the Tweets with AFINN

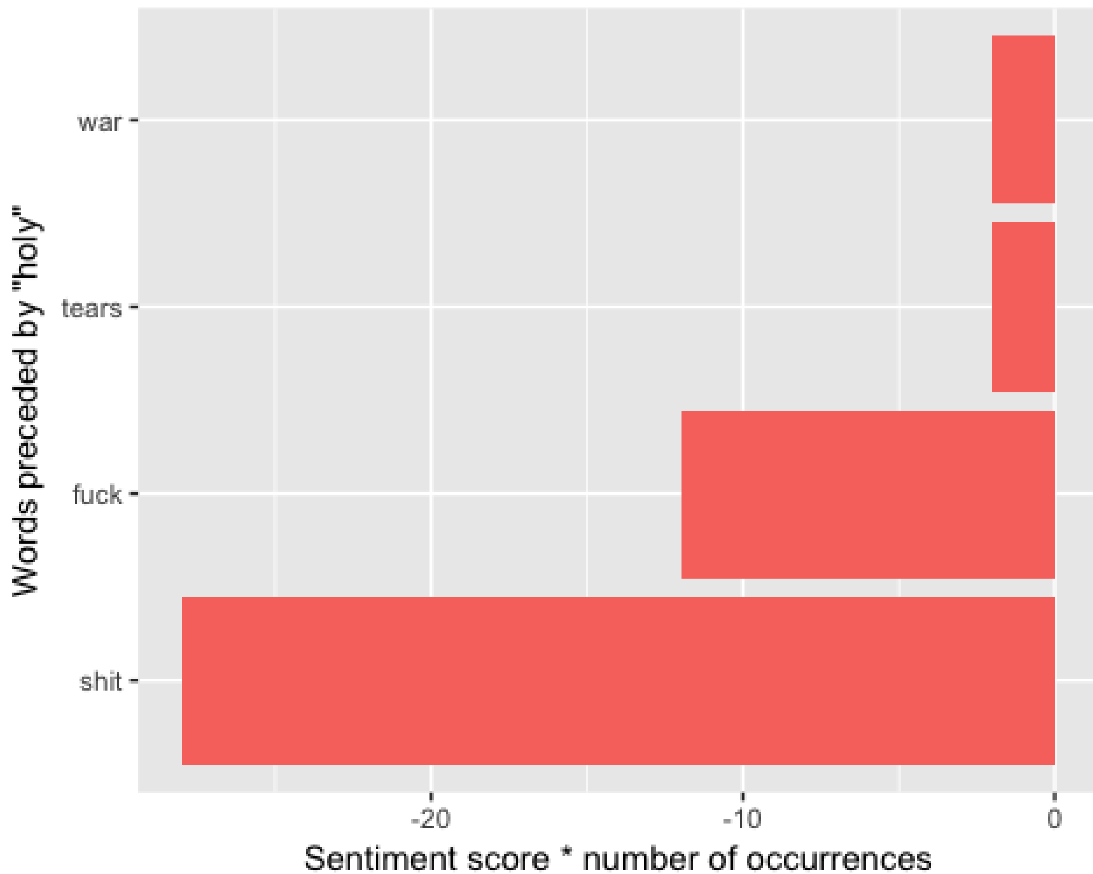
```
AFINN <- get_sentiments("afinn")
demo_bigrams <- unnest_tokens(isisTweets,
                              input = tweets,
                              output = bigram,
                              token = "ngrams",
                              n=2)
demo_bigrams %>%
  count(bigram, sort = TRUE)

## # A tibble: 352,459 x 2
##    bigram                n
##    <chr>             <int>
```

```
##   1 islamic state     10939
##   2 2016 07           10569
##   3 isis is            9510
##   4 https t.co         8364
##   5 during the         6530
##   6 in the             5410
##   7 https twitter.com  5150
##   8 of ramadan         5004
##   9 is the             4956
##  10 holy month         4941
## # ... with 352,449 more rows
```

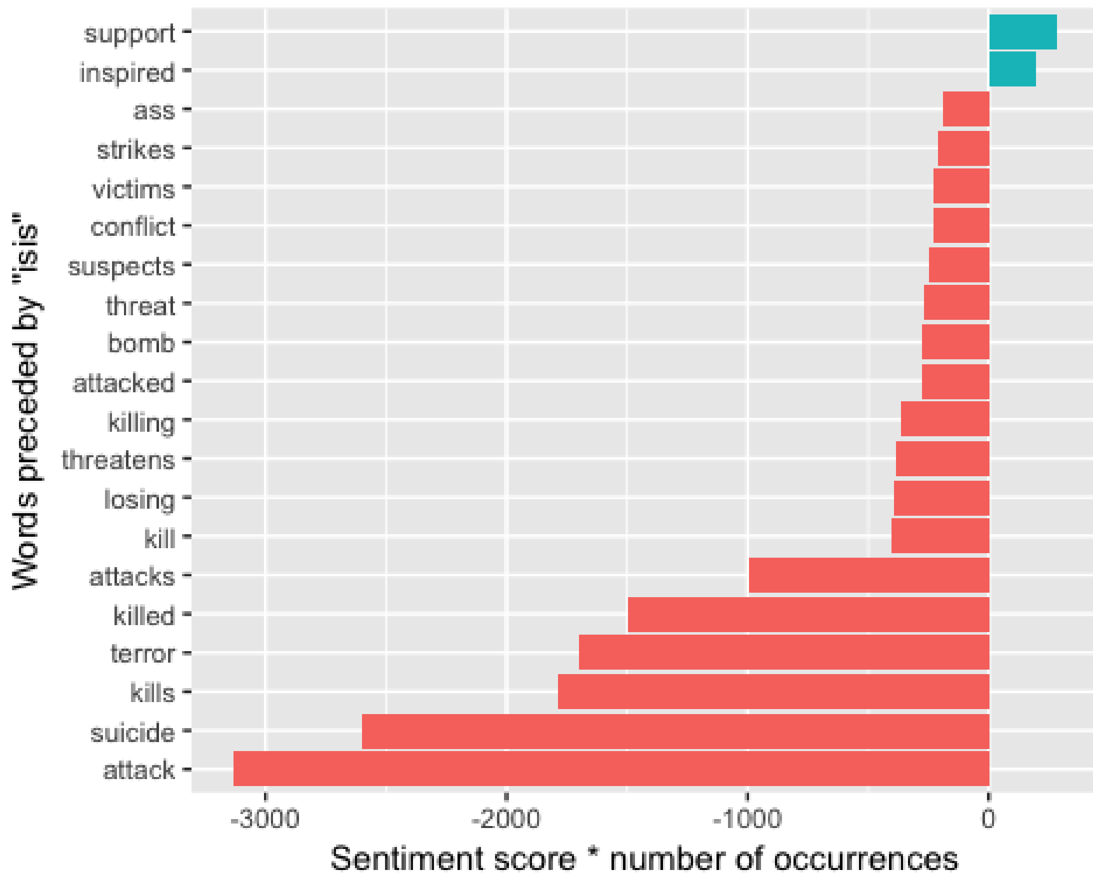
```
bigrams_separated <- demo_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")
```

From words and association

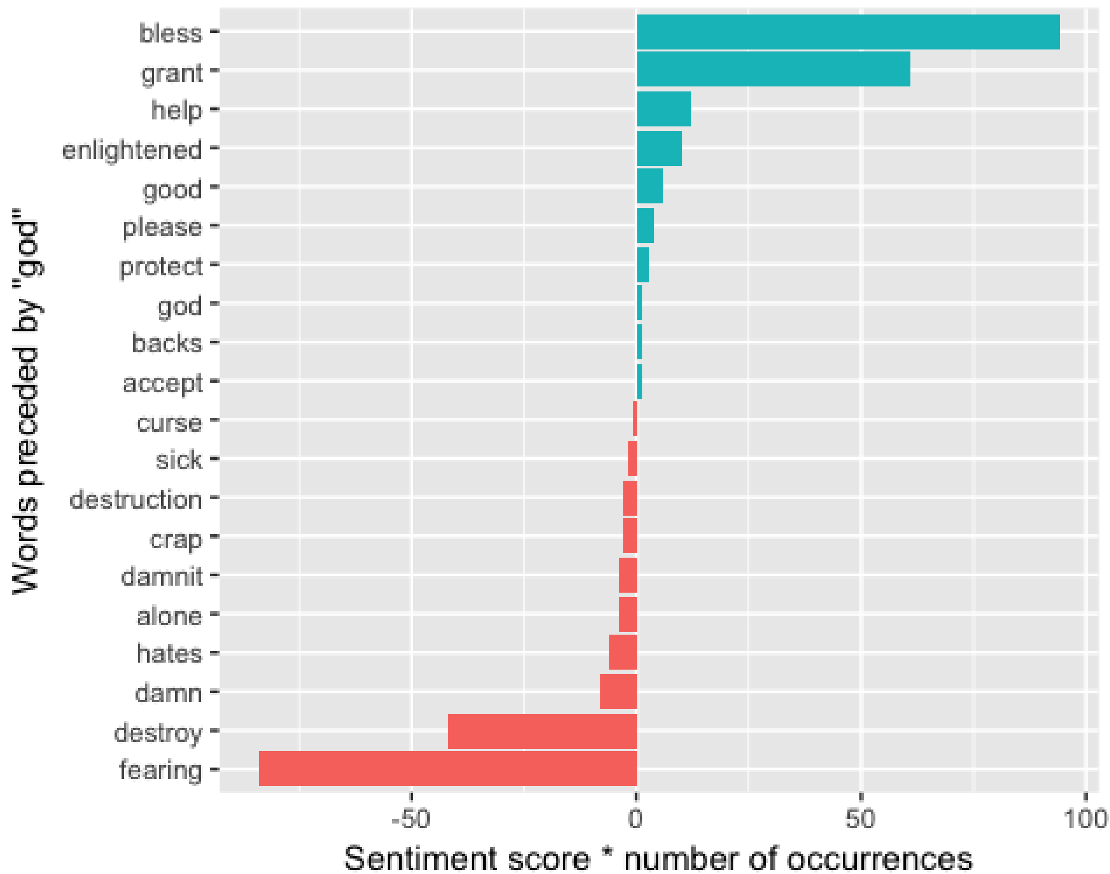
```
not_words <- bigrams_separated %>%
  filter(word1 == "holy") %>%
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word2, score, sort = TRUE) %>%
  ungroup()

not_words %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(20) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("Words preceded by \"holy\"") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip()
```




```
not_words <- bigrams_separated %>%
  filter(word1 == "isis") %>%
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word2, score, sort = TRUE) %>%
  ungroup()

not_words %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(20) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("Words preceded by \"isis\"") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip()
```




```
not_words <- bigrams_separated %>%
  filter(word1 == "god") %>%
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word2, score, sort = TRUE) %>%
  ungroup()

not_words %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(20) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("Words preceded by \"god\"") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip()
```



Demo bigrams

Visualizing a network of bigrams with ggraph We may be interested in visualizing all of the relationships among words simultaneously, rather than just the top few at a time. As one common visualization, we can arrange the words into a network, or "graph." Here we'll be referring to a "graph" not in the sense of a visualization, but as a combination of connected nodes. A graph can be constructed from a tidy object since it has three variables:

from: the node an edge is coming from

to: the node an edge is going towards

weight: A numeric value associated with each edge

The igraph package has many powerful functions for manipulating and analyzing networks. One way to create an igraph object from tidy data is the graph_from_data_frame() function, which takes a data frame of edges with columns for "from", "to", and edge attributes (in this case n):

```
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)
```

```r
# new bigram counts:
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)


bigram_counts
```

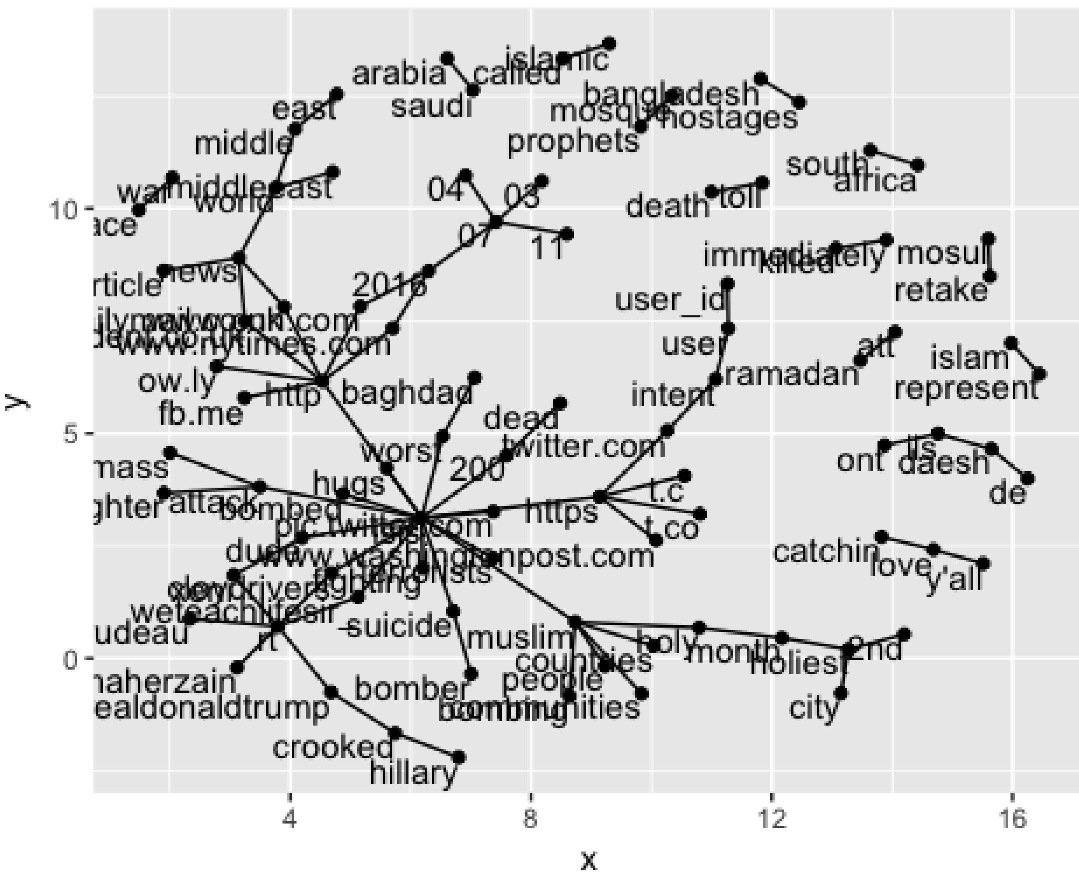
```
## # A tibble: 224,397 x 3
##    word1   word2               n
##    <chr>   <chr>           <int>
##  1 2016    07              10569
##  2 https   t.co             8364
##  3 https   twitter.com      5150
##  4 holy    month            4941
##  5 rt      realdonaldtrump  3909
##  6 isis    muslim           3645
##  7 muslim  holy             3614
##  8 muslim  people           3580
##  9 bombing muslim           3573
## 10 muslim  communities      3549
## # ... with 224,387 more rows
```

```r
bigram_graph <- bigram_counts %>%
  filter(n > 1000) %>%
  graph_from_data_frame()

bigram_graph
```

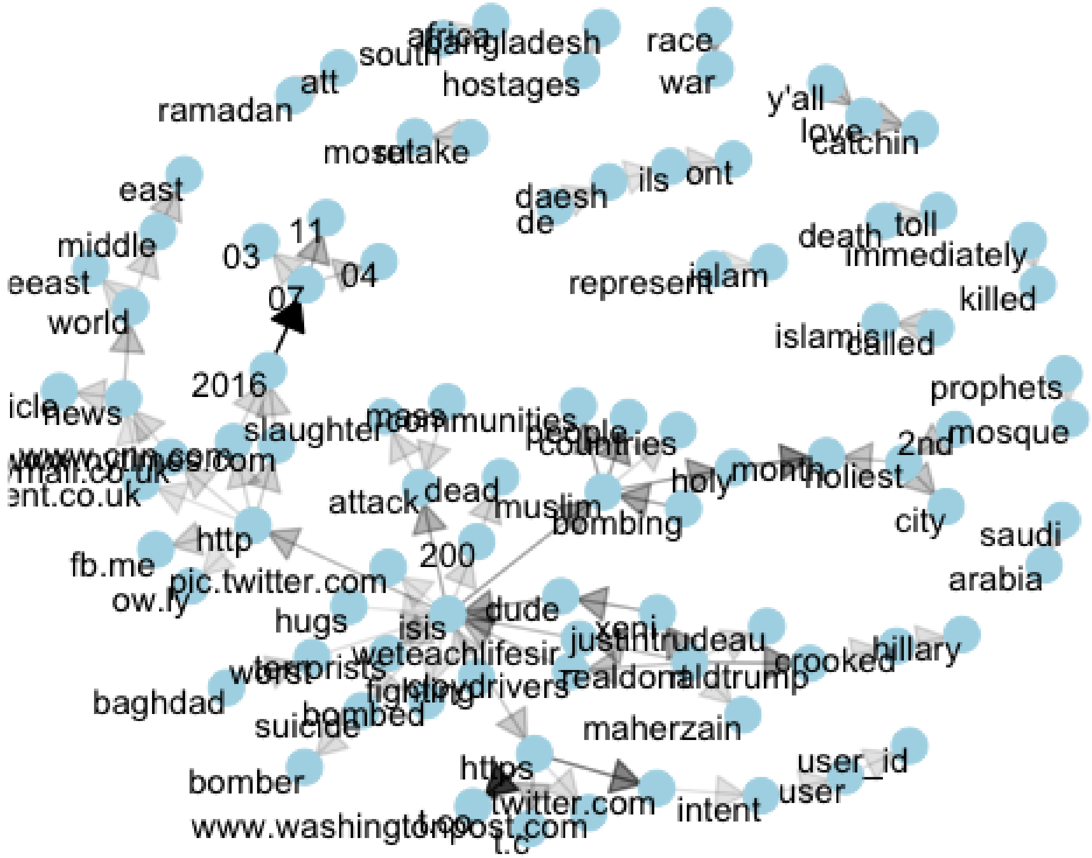
```
## IGRAPH d3cba4f DN-- 90 80 --
## + attr: name (v/c), n (e/n)
## + edges from d3cba4f (vertex names):
##  [1] 2016   ->07              https  ->t.co
##  [3] https  ->twitter.com     holy   ->month
##  [5] rt     ->realdonaldtrump isis   ->muslim
##  [7] muslim ->holy            muslim ->people
##  [9] bombing->muslim          muslim ->communities
## [11] dude   ->isis            rt     ->xeni
## [13] xeni   ->dude            isis   ->attack
## [15] 07     ->11              isis   ->http
## + ... omitted several edges
```

Display Bigrams

```r
set.seed(123)


ggraph(bigram_graph, layout = "fr")+
  geom_edge_link() +
  geom_node_point() +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1)
```




```r
set.seed(2016)

a <- grid::arrow(type = "closed", length = unit(.15, "inches"))

ggraph(bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                 arrow = a, end_cap = circle(.07, 'inches')) +
  geom_node_point(color = "lightblue", size = 5) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
  theme_void()
```



Conclusion

It was interesting to analyze data from across the world on topics surrounding ISIS. The overall attitude towards this terrorist group was dismal, as expected, however a dataset containing pro-isis tweets were thrown into the mix and interesting opinions and points of view displayed a trend and was graphed The word cloud and charts derived from the data from both groups concluded that similar words (such as attack, war, kill, solidarity) and expressions were used to describe ISIS, however due to the nature of natural language, words such as “love” and “holy” that were found in analysis as having a positive meaning, were in fact used in a negative way in some instances. Sentiment analysis is a great way to learn the opinions by various groups of people around the world and this information would be beneficial to organizations such as the UN, UNICEF, governments, etc. in making important world decisions.

Member Contribution:

- Sonya Hidar - Contributed to write-up of final report, added chart descriptions, as well as introduction, conclusion and helped with analysis write-up.
- David Guardia - Contribute with the research, code implementation, learning how to process the information most relevant for the project, learning how to use rstudio and how to create RMarkdown and generate the report from within the IDE. Create the word , html, document
- Sucharita Das - Contributed to the part of the code and description of the various functions and the libraries used.
- Zhoujun Cai - Contributed to the slides, added the descriptions.

Bibliography

1. https://www.tidytextmining.com/sentiment.html
2. https://www.quora.com/How-do-I-perfeorm-sentiment-analysis-on-Twitter-data-using-hashtags-in-R
3. https://colinpriest.com/2015/07/04/tutorial-using-r-and-twitter-to-analyse-consumer-sentiment/
4. https://www.youtube.com/watch?v=0xsM0MbRPGE
5. https://www.youtube.com/watch?v=otoXeVPhT7Q
6. https://rpubs.com/williamsurles/316682
7. https://rstudio-pubs-static.s3.amazonaws.com/66739_c4422a1761bd4ee0b0bb8821d7780e12.html