Sentiment Analysis 2

Benjamin Moscona

4/20/2022

#### IPCC Report Twitter

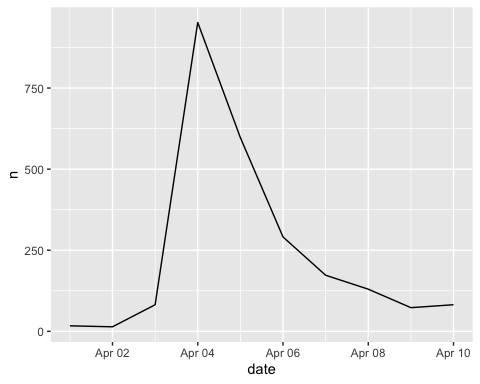
library(quanteda)  
# devtools::install\_github("quanteda/quanteda.sentiment") #not available currently through CRAN  
library(quanteda.sentiment)  
library(quanteda.textstats)  
library(tidyverse)  
library(tidytext)  
library(lubridate)  
library(wordcloud) #visualization of common words in the data set  
library(reshape2)

Last week we used the tidytext approach to sentiment analysis for Nexis Uni .pdf data on coverage of the recent IPCC report. This week we will look at the conversation on Twitter about the same report. We’ll start with the familiar tidy approach, and then introduce the quanteda package later.

raw\_tweets <- read.csv("Data/IPCC\_tweets\_April1-10\_sample.csv", header=TRUE)  
  
dat<- raw\_tweets[,c(4,6)] # Extract Date and Title fields  
  
tweets <- tibble(text = dat$Title,  
 id = seq(1:length(dat$Title)),  
 date = as.Date(dat$Date,'%m/%d/%y'))  
  
  
head(tweets$text, n = 10)

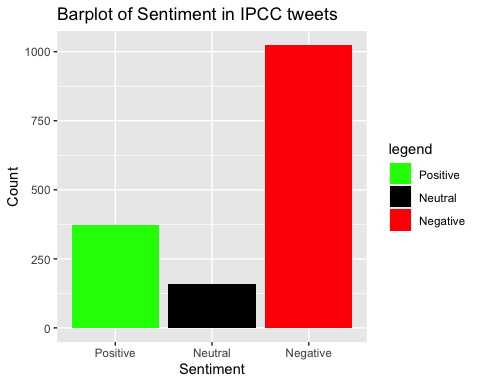
## [1] "thank you, followers, for the great photo suggestions for our upcoming IPCC report - on Monday you will find the lucky one selected for our cover from among your submissions!\n\nwe now need a good picture on #biofuels . any suggestions please for which we can get copyrights fast?"   
## [2] "Greenpeace: The real solution to the climate crisis will require a rapid transition away from fossil fuels. \n\nWhat else we expect from the upcoming #IPCC report on climate solutions, set for publication on Monday, 4 April ⬇️ https://t.co/EC6a25S7tY"   
## [3] "Governments have a responsibility to ensure that #IPCCReport is grounded in rapid phaseout of fossil fuel use and production — not #FalseClimateSolutions. \n\nRead more in our open letter: https://t.co/4larBPgeba https://t.co/Fv1OphPmac"   
## [4] "Next week, the IPCC will publish a new report detailing their new models and policy pathways. \n\nWant to study up before the headlines? Read @bertrandhb's second long read on CCS, explaining how and why IPCC models use so much saviour tech.\n\nhttps://t.co/6yBf0j7UWA"   
## [5] "Live stream of virtual IPCC press conference releasing the report on mitigation of climate change, 9 a.m. GMT o... https://t.co/IqRCvvQxyX"   
## [6] "Attention journalists: The deadline for embargoed materials for the upcoming @IPCC\_CH report on climate mitigation has been extended to TODAY at 5:59 pm EDT. Register here: https://t.co/fLc4eHcOmm https://t.co/0eIlPb21kz"   
## [7] "The IPCC Report and “The Physics of Climate Change” https://t.co/xnxP3fup2a"   
## [8] "With time running short and most of the Summary for Policymakers yet to be approved, #IPCC Working Group III added a fourth plenary to Thursday’s packed schedule in an attempt to make headway.\n\nMore ➡️ https://t.co/CRKNFzykYE\n\n#ClimateChange #AR6 #ClimateReport https://t.co/EoaasmOEZf"  
## [9] "A helpful perspective on how to talk about the scenarios discussed in the forthcoming IPCC report https://t.co/Kpiim9NgNw"   
## [10] "The private sector is an integral component of the water cycle and has much to lose as critical climate and water risks grow. \n\nThis presents an opportunity for collective action, writes Kirsten James of the sustainability nonprofit @CeresNews. \n\nhttps://t.co/pC3kiJ6R1t"

#simple plot of tweets per day  
tweets %>%  
 count(date) %>%  
 ggplot(aes(x = date, y = n))+  
 geom\_line()

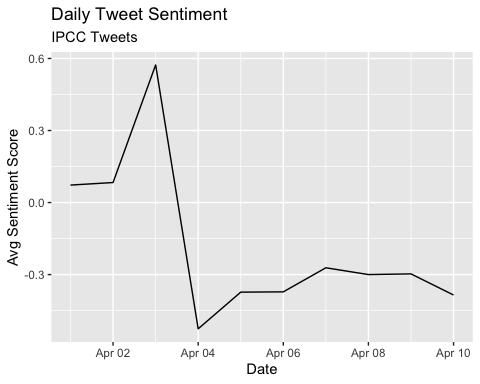


#let's clean up the URLs from the tweets  
tweets$text <- gsub("http[^[:space:]]\*", "",tweets$text)  
tweets$text <- str\_to\_lower(tweets$text)  
tweets$text <- iconv(tweets$text, "latin1", "ASCII", sub="")  
  
#load sentiment lexicons  
bing\_sent <- get\_sentiments('bing')  
nrc\_sent <- get\_sentiments('nrc')  
  
#tokenize tweets to individual words  
words <- tweets %>%  
 select(id, date, text) %>%  
 unnest\_tokens(output = word, input = text, token = "words") %>%  
 anti\_join(stop\_words, by = "word") %>%  
 left\_join(bing\_sent, by = "word") %>%  
 left\_join(  
 tribble(  
 ~sentiment, ~sent\_score,  
 "positive", 1,  
 "negative", -1),  
 by = "sentiment")

#take average sentiment score by tweet  
tweets\_sent <- tweets %>%  
 left\_join(  
 words %>%  
 group\_by(id) %>%  
 summarize(  
 sent\_score = mean(sent\_score, na.rm = T)),  
 by = "id")  
  
neutral <- length(which(tweets\_sent$sent\_score == 0))  
positive <- length(which(tweets\_sent$sent\_score > 0))  
negative <- length(which(tweets\_sent$sent\_score < 0))  
  
Sentiment <- c("Positive","Neutral","Negative")  
Count <- c(positive,neutral,negative)  
output <- data.frame(Sentiment,Count)  
output$Sentiment<-factor(output$Sentiment,levels=Sentiment)  
ggplot(output, aes(x=Sentiment,y=Count))+  
 geom\_bar(stat = "identity", aes(fill = Sentiment))+  
 scale\_fill\_manual("legend", values = c("Positive" = "green", "Neutral" = "black", "Negative" = "red"))+  
 ggtitle("Barplot of Sentiment in IPCC tweets")



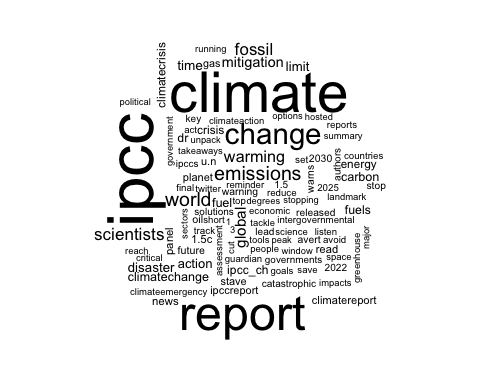
# tally sentiment score per day  
daily\_sent <- tweets\_sent %>%  
 group\_by(date) %>%  
 summarize(sent\_score = mean(sent\_score, na.rm = T))  
  
daily\_sent %>%  
 ggplot( aes(x = date, y = sent\_score)) +  
 geom\_line() +  
 labs(x = "Date",  
 y = "Avg Sentiment Score",  
 title = "Daily Tweet Sentiment",  
 subtitle = "IPCC Tweets")



Now let’s try a new type of text visualization: the wordcloud.

words %>%  
 anti\_join(stop\_words) %>%  
 count(word) %>%  
 with(wordcloud(word, n, max.words = 100))

## Joining, by = "word"



words %>%  
inner\_join(get\_sentiments("bing")) %>%  
count(word, sentiment, sort = TRUE) %>%  
acast(word ~ sentiment, value.var = "n", fill = 0) %>%  
comparison.cloud(colors = c("gray20", "gray80"),  
 max.words = 100)

## Joining, by = c("word", "sentiment")

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## sustainability could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## catastrophe could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## healthy could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## comprehensive could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## effectively could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## exceed could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## powerful could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## resilient could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## slashing could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("gray20", "gray80"), max.words = 100):  
## suffering could not be fit on page. It will not be plotted.



#### The quanteda package

quanteda is a package (actually a family of packages) full of tools for conducting text analysis. quanteda.sentiment (not yet on CRAN, download from github) is the quanteda modular package for conducting sentiment analysis.

quanteda has its own built in functions for cleaning text data. Let’s take a look at some. First we have to clean the messy tweet data:

corpus <- corpus(dat$Title) #enter quanteda  
summary(corpus)

## Corpus consisting of 2411 documents, showing 100 documents:  
##   
## Text Types Tokens Sentences  
## text1 43 53 2  
## text2 37 42 2  
## text3 31 32 2  
## text4 42 49 3  
## text5 21 25 2  
## text6 30 33 1  
## text7 10 12 1  
## text8 40 42 2  
## text9 16 17 1  
## text10 36 42 2  
## text11 16 16 1  
## text12 34 44 6  
## text13 35 46 3  
## text14 46 52 2  
## text15 42 51 1  
## text16 7 7 1  
## text17 42 48 2  
## text18 17 17 2  
## text19 43 60 1  
## text20 27 34 3  
## text21 40 43 3  
## text22 44 50 3  
## text23 28 30 2  
## text24 35 38 3  
## text25 36 41 3  
## text26 37 43 4  
## text27 21 23 1  
## text28 29 31 1  
## text29 12 13 1  
## text30 45 47 2  
## text31 38 42 1  
## text32 31 36 1  
## text33 14 14 1  
## text34 41 49 1  
## text35 7 7 1  
## text36 44 54 2  
## text37 26 28 1  
## text38 13 13 1  
## text39 13 13 1  
## text40 31 37 2  
## text41 47 54 4  
## text42 38 46 1  
## text43 42 46 2  
## text44 22 24 2  
## text45 38 46 1  
## text46 16 16 1  
## text47 30 32 1  
## text48 17 17 1  
## text49 13 13 1  
## text50 23 23 1  
## text51 23 25 1  
## text52 25 27 1  
## text53 13 13 1  
## text54 34 35 3  
## text55 38 46 1  
## text56 38 46 1  
## text57 38 46 1  
## text58 38 46 1  
## text59 38 46 1  
## text60 38 46 1  
## text61 19 19 2  
## text62 17 18 1  
## text63 11 11 1  
## text64 13 13 1  
## text65 14 16 1  
## text66 12 12 2  
## text67 18 18 1  
## text68 38 46 1  
## text69 15 16 1  
## text70 12 13 1  
## text71 30 35 2  
## text72 22 23 1  
## text73 38 46 1  
## text74 39 46 1  
## text75 13 13 1  
## text76 32 35 1  
## text77 38 46 1  
## text78 39 45 2  
## text79 38 46 1  
## text80 36 41 1  
## text81 33 33 2  
## text82 18 19 1  
## text83 38 46 1  
## text84 38 46 1  
## text85 38 46 1  
## text86 39 43 2  
## text87 13 13 1  
## text88 13 13 1  
## text89 38 46 1  
## text90 38 46 1  
## text91 38 46 1  
## text92 40 43 1  
## text93 11 11 1  
## text94 41 49 1  
## text95 38 46 1  
## text96 15 15 1  
## text97 29 31 1  
## text98 11 11 1  
## text99 13 13 1  
## text100 38 46 1

tokens <- tokens(corpus) #tokenize the text so each doc (page, in this case) is a list of tokens (words)  
  
#examine the uncleaned version  
tokens

## Tokens consisting of 2,411 documents.  
## text1 :  
## [1] "thank" "you" "," "followers" ","   
## [6] "for" "the" "great" "photo" "suggestions"  
## [11] "for" "our"   
## [ ... and 41 more ]  
##   
## text2 :  
## [1] "Greenpeace" ":" "The" "real" "solution"   
## [6] "to" "the" "climate" "crisis" "will"   
## [11] "require" "a"   
## [ ... and 30 more ]  
##   
## text3 :  
## [1] "Governments" "have" "a" "responsibility"  
## [5] "to" "ensure" "that" "#IPCCReport"   
## [9] "is" "grounded" "in" "rapid"   
## [ ... and 20 more ]  
##   
## text4 :  
## [1] "Next" "week" "," "the" "IPCC" "will"   
## [7] "publish" "a" "new" "report" "detailing" "their"   
## [ ... and 37 more ]  
##   
## text5 :  
## [1] "Live" "stream" "of" "virtual" "IPCC"   
## [6] "press" "conference" "releasing" "the" "report"   
## [11] "on" "mitigation"  
## [ ... and 13 more ]  
##   
## text6 :  
## [1] "Attention" "journalists" ":" "The" "deadline"   
## [6] "for" "embargoed" "materials" "for" "the"   
## [11] "upcoming" "@IPCC\_CH"   
## [ ... and 21 more ]  
##   
## [ reached max\_ndoc ... 2,405 more documents ]

#clean it up  
tokens <- tokens(tokens, remove\_punct = TRUE,  
 remove\_numbers = TRUE)  
  
tokens <- tokens\_select(tokens, stopwords('english'),selection='remove') #stopwords lexicon built in to quanteda  
  
#tokens <- tokens\_wordstem(tokens) #stem words down to their base form for comparisons across tense and quantity  
  
tokens <- tokens\_tolower(tokens)

We can use the kwic function (keywords-in-context) to briefly examine the context in which certain words or patterns appear.

head(kwic(tokens, pattern = "climate", window = 3))

## Keyword-in-context with 6 matches.   
## [text2, 4] greenpeace real solution | climate |  
## [text2, 17] upcoming#ipcc report | climate |  
## [text5, 10] releasing report mitigation | climate |  
## [text6, 9] upcoming@ipcc\_ch report | climate |  
## [text7, 4] ipcc report physics | climate |  
## [text10, 10] much lose critical | climate |  
##   
## crisis require rapid   
## solutions set publication   
## change a.m gmt   
## mitigation extended today   
## change https://t.co/xnxp3fup2a  
## water risks grow

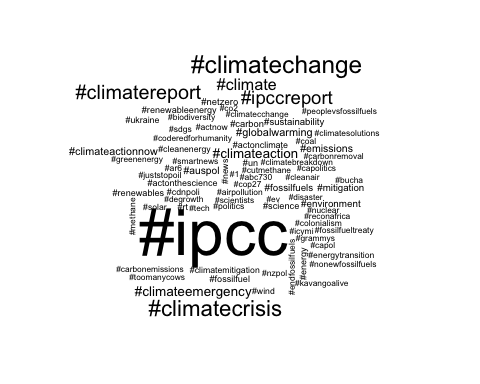
head(kwic(tokens, pattern = phrase("climate change"), window = 3))

## Keyword-in-context with 6 matches.   
## [text5, 10:11] releasing report mitigation | climate change |  
## [text7, 4:5] ipcc report physics | climate change |  
## [text14, 15:16] avert worst effects | climate change |  
## [text15, 10:11] s#climatereport emissions | climate change |  
## [text20, 1:2] | climate change |  
## [text24, 6:7] report revealed threat | climate change |  
##   
## a.m gmt o   
## https://t.co/xnxp3fup2a   
## anyone think revolution   
## meenakshi raman@sahabatalammsia  
## want learn 100s   
## team weighed findings

hash\_tweets <- tokens(corpus, remove\_punct = TRUE) %>%   
 tokens\_keep(pattern = "#\*")  
  
dfm\_hash<- dfm(hash\_tweets)  
  
tstat\_freq <- textstat\_frequency(dfm\_hash, n = 100)  
head(tstat\_freq, 10)

## feature frequency rank docfreq group  
## 1 #ipcc 464 1 460 all  
## 2 #climatechange 137 2 135 all  
## 3 #climatecrisis 118 3 117 all  
## 4 #climatereport 97 4 97 all  
## 5 #ipccreport 87 5 87 all  
## 6 #climate 68 6 67 all  
## 7 #climateemergency 45 7 45 all  
## 8 #climateaction 44 8 44 all  
## 9 #globalwarming 24 9 24 all  
## 10 #climateactionnow 23 10 23 all

#tidytext gives us tools to convert to tidy from non-tidy formats  
hash\_tib<- tidy(dfm\_hash)  
  
hash\_tib %>%  
 count(term) %>%  
 with(wordcloud(term, n, max.words = 100))



Create the sparse matrix representation known as the document-feature matrix. quanteda’s textstat\_polarity function has multiple ways to combine polarity to a single score. The sent\_logit value to fun argument is the log of (pos/neg) counts.

dfm <- dfm(tokens)  
  
topfeatures(dfm, 12)

## climate ipcc report change now #ipcc world   
## 1396 1243 1225 651 505 464 346   
## emissions never new latest scientists   
## 333 291 279 279 274

dfm.sentiment <- dfm\_lookup(dfm, dictionary = data\_dictionary\_LSD2015)  
  
head(textstat\_polarity(tokens, data\_dictionary\_LSD2015, fun = sent\_logit))

## doc\_id sentiment  
## 1 text1 2.197225  
## 2 text2 -1.098612  
## 3 text3 1.945910  
## 4 text4 0.000000  
## 5 text5 1.098612  
## 6 text6 1.098612

### Assignment

You will use the tweet data from class today for each part of the following assignment.

1. Think about how to further clean a twitter data set. Let’s assume that the mentions of twitter accounts is not useful to us. Remove them from the text field of the tweets tibble.

tweets$text <- str\_remove(tweets$text, "@.\*")

1. Compare the ten most common terms in the tweets per day. Do you notice anything interesting?

#tokenize tweets to individual words  
words <- tweets %>%  
 select(id, date, text) %>%  
 unnest\_tokens(output = word, input = text, token = "words") %>%  
 anti\_join(stop\_words, by = "word") %>%  
 left\_join(bing\_sent, by = "word") %>%  
 left\_join(  
 tribble(  
 ~sentiment, ~sent\_score,  
 "positive", 1,  
 "negative", -1),  
 by = "sentiment")  
  
words %>%   
 group\_by(word) %>%   
 summarize(n = n()) %>%   
 arrange(desc(n))

## # A tibble: 5,261 × 2  
## word n  
## <chr> <int>  
## 1 ipcc 1577  
## 2 climate 1378  
## 3 report 1084  
## 4 change 621  
## 5 world 332  
## 6 emissions 326  
## 7 scientists 269  
## 8 fossil 255  
## 9 warming 233  
## 10 global 199  
## # … with 5,251 more rows

All 10 words are related directly to the IPCC report or climate change. “ipcc” is mentioned most, followed by “climate” and “report”. None of this is particularly surprising to me.

1. Adjust the wordcloud in the “wordcloud” chunk by coloring the positive and negative words so they are identifiable.

words %>%  
 anti\_join(stop\_words) %>%   
 inner\_join(get\_sentiments("bing")) %>%   
 count(word, sentiment, sort = TRUE) %>%  
 acast(word ~ sentiment, value.var = "n", fill = 0) %>%  
 comparison.cloud(colors = c("red", "green"),  
 max.words = 100)

## Joining, by = "word"

## Joining, by = c("word", "sentiment")

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## sustainability could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## catastrophe could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## revolutionary could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## helping could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## powerful could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## supports could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## approval could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## coherent could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## confidence could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## courage could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## effectively could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## encourage could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## equitable could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## feasible could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## hopeful could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## relief could not be fit on page. It will not be plotted.

## Warning in comparison.cloud(., colors = c("red", "green"), max.words = 100):  
## surpass could not be fit on page. It will not be plotted.

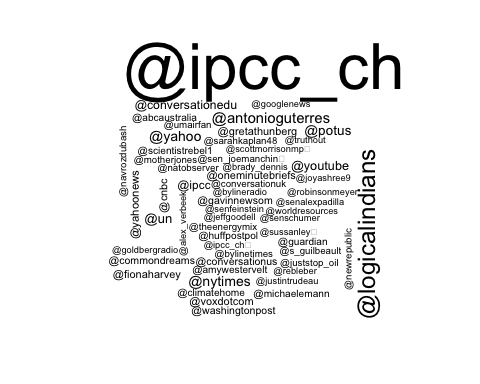


1. Let’s say we are interested in the most prominent entities in the Twitter discussion. Which are the top 10 most tagged accounts in the data set. Hint: the “explore\_hashtags” chunk is a good starting point.

hash\_tweets <- tokens(corpus, remove\_punct = TRUE) %>%   
 tokens\_keep(pattern = "@\*")  
  
dfm\_hash<- dfm(hash\_tweets)  
  
tstat\_freq <- textstat\_frequency(dfm\_hash, n = 100)  
head(tstat\_freq, 10)

## feature frequency rank docfreq group  
## 1 @ipcc\_ch 131 1 131 all  
## 2 @logicalindians 38 2 38 all  
## 3 @antonioguterres 16 3 16 all  
## 4 @nytimes 14 4 14 all  
## 5 @yahoo 14 4 14 all  
## 6 @potus 13 6 13 all  
## 7 @un 12 7 12 all  
## 8 @youtube 11 8 11 all  
## 9 @conversationedu 10 9 10 all  
## 10 @ipcc 9 10 9 all

#tidytext gives us tools to convert to tidy from non-tidy formats  
hash\_tib<- tidy(dfm\_hash)  
  
hash\_tib %>%  
 count(term) %>%  
 with(wordcloud(term, n, max.words = 100))

 5. The Twitter data download comes with a variable called “Sentiment” that must be calculated by Brandwatch. Use your own method to assign each tweet a polarity score (Positive, Negative, Neutral) and compare your classification to Brandwatch’s (hint: you’ll need to revisit the “raw\_tweets” data frame).

dat<- raw\_tweets[,c(4,6, 10)] # Extract Date and Title fields  
  
tweets <- tibble(text = dat$Title,  
 id = seq(1:length(dat$Title)),  
 date = as.Date(dat$Date,'%m/%d/%y'),  
 sentiment\_brandwatch = dat$Sentiment)  
  
#let's clean up the URLs from the tweets  
tweets$text <- gsub("http[^[:space:]]\*", "",tweets$text)  
tweets$text <- iconv(tweets$text, "latin1", "ASCII", sub="")  
tweets$text <- str\_to\_lower(tweets$text)  
  
#load sentiment lexicons  
bing\_sent <- get\_sentiments('bing')  
nrc\_sent <- get\_sentiments('nrc')  
  
#tokenize tweets to individual words  
words <- tweets %>%  
 select(id, date, text, sentiment\_brandwatch) %>%  
 unnest\_tokens(output = word, input = text, token = "words") %>%  
 anti\_join(stop\_words, by = "word") %>%  
 left\_join(bing\_sent, by = "word") %>%  
 left\_join(  
 tribble(  
 ~sentiment, ~sent\_score,  
 "positive", 1,  
 "negative", -1),  
 by = "sentiment")  
  
compare\_sents <- words %>%   
 mutate(sent\_score = replace\_na(sent\_score, 0)) %>%   
 group\_by(id) %>%   
 summarize(sentiment = mean(sent\_score),  
 sentiment\_brandwatch = first(sentiment\_brandwatch)) %>%   
 mutate(sentiment = case\_when(sentiment >= 0.2 ~ "positive",  
 sentiment < 0.2 & sentiment > -0.2 ~ "neutral",  
 sentiment <= -0.2 ~ "negative"))  
   
table(compare\_sents$sentiment, compare\_sents$sentiment\_brandwatch)

##   
## negative neutral positive  
## negative 70 112 0  
## neutral 180 2009 18  
## positive 0 15 1

My method of taking the mean after assigning neutral words a zero and then splitting at -0.2 to 0.2 for neutral returns fairly similar results to the Brandwatch sentiment score. While both models are fairly similar for neutral and negative tweets, they disagree on which ones should be labeled positive vs. neutral. There aren’t many particularly positive tweets in this set which is why there may be less agreement there.