To be honest, I planned on writing a review of this past weekend’s  
[rstudio::conf 2019](https://resources.rstudio.com/rstudio-conf-2019),  
but several other people have already done a great job of doing  
that—just check out [Karl Broman’s aggregation of reviews at the bottom  
of the page here](https://github.com/kbroman/RStudioConf2019Slides)!  
(More on this in a second.) In short, my thoughts on the whole  
experience are captured perfectly by [Nick  
Strayer](http://nickstrayer.me/)’s tweet the day after the conference  
ended.

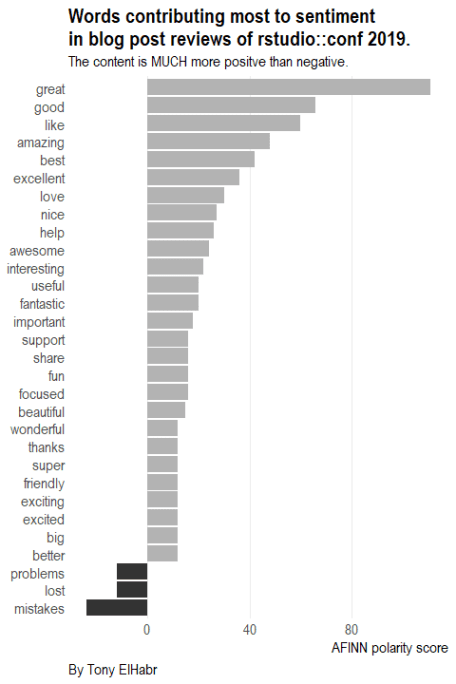
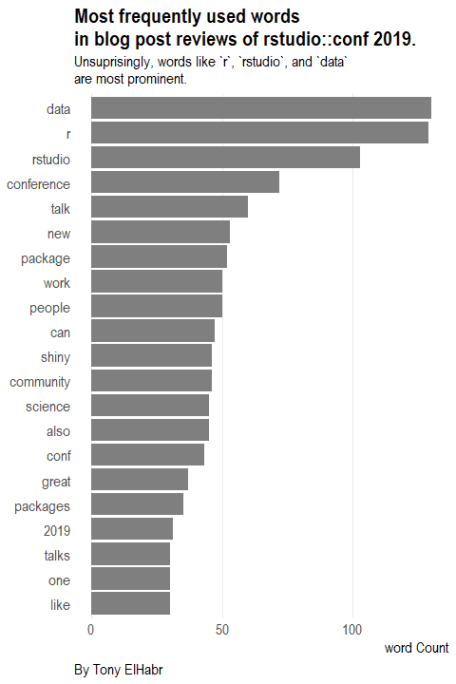
Thing I learned at [#Rstudioconf](https://twitter.com/hashtag/Rstudioconf?src=hash&ref_src=twsrc%5Etfw) that sticks out to me the most: it turns out that this group of people who are so kind and welcoming online are also kind and welcoming in real life. More than any library or api that’s what makes [#rstats](https://twitter.com/hashtag/rstats?src=hash&ref_src=twsrc%5Etfw) great.

— Nick Strayer (@NicholasStrayer) [January 19, 2019](https://twitter.com/NicholasStrayer/status/1086681100787822592?ref_src=twsrc%5Etfw)

Anyways, I figured that this was the perfect opportunity to do some  
[“tidy text analysis”](https://www.tidytextmining.com/)! Why not extract  
the text from the reviews of others—linked in Karl’s repo—and make “my  
own” summary of the event? Plotting word frequencies and sentiments,  
while not exactly “cutting edge” compared to robust  
[NLP](https://en.wikipedia.org/wiki/Neuro-linguistic_programming)  
techniques, is perfect for getting a nice, simple overview of the  
conference. (I have applied some of the techniques described by [David  
Robinson](http://varianceexplained.org/) and [Julia  
Silge](https://juliasilge.com/blog) in their [*Tidy Text Mining with R*  
book](https://www.tidytextmining.com/) [more than  
once](https://tonyelhabr.rbind.io/post/tidy-text-analysis-google-search-history) [in the  
past](https://tonyelhabr.rbind.io/post/tidy-text-analysis-rweekly), but not recently, and not on  
this topic.)

Moreover, after reading [Bob Rudis’s recent  
post](https://rud.is/b/2019/01/21/hrbrthemes-0-6-0-on-cran-other-in-development-package-news/)  
and discovering his work-in-progress “ {curl} +{httr} + {rvest}“  
package called [{reapr}](https://git.sr.ht/~hrbrmstr/reapr), I  
realized that the work of cleaning the HTML and text for each person’s  
blog post would not be so bad. In fact, it turned out to be as easy as  
reapr::reap\_url() %>% reapr::mill() (with only a little bit of extra  
work )!

After trying a couple of different approaches (e.g. bigrams, topic  
modeling, etc.) and experimenting with some different visualizations, I  
ended up making the two plots below. (To the interested reader, I’ve  
included all of the code at the bottom of this post.) From the second  
plot—where positive sentiment heavily outweighs negative sentiment—one  
thing is clear: the R community is **super supportive and positive**,  
just as Nick alluded to in his tweet.



I’ve said it before and I’ll happily said it again: thanks again to  
David Robinson and Julia Silge for their great [*Tidy Text Mining with  
R* book](https://www.tidytextmining.com/) and everything else that  
they’ve done for the community! The techniques that they’ve documented  
and shared are super helpful for doing a quick exploration just like  
this.

# Reference for adding appendix: https://yihui.name/en/2018/09/code-appendix/

DIR\_POST <- "content/post/rstudio-conf-2019-summary"

PATH\_SANS\_EXT <- "index-to-render"

PATH\_RMD <- file.path(DIR\_POST, paste0(PATH\_SANS\_EXT, ".Rmd"))

PATH\_OUTPUT <- file.path(DIR\_POST, paste0(PATH\_SANS\_EXT, ".md"))

# PATH\_R <- paste0(PATH\_SANS\_EXT, ".R")

# rmarkdown::render(knitr::spin(PATH\_R, knit = FALSE))

# # Spell-checking.

# spelling::spell\_check\_files(PATH\_RMD)

# # Convert from Rmd to output specified in YAML.

# rmarkdown::render(PATH\_RMD, output\_file = PATH\_OUTPUT, output\_dir = DIR\_POST, knit\_root\_dir = DIR\_POST, intermediates\_dir = DIR\_POST)

# rmarkdown::render(PATH\_RMD)

# rmarkdown::render(PATH\_RMD, output\_file = PATH\_OUTPUT)

knitr::opts\_knit$set(root.dir = here::here(DIR\_POST))

knitr::opts\_chunk$set(

echo = FALSE,

cache = FALSE,

include = FALSE,

fig.show = "hide",

fig.align = "center",

fig.width = 4.5,

fig.asp = 1.5,

warning = FALSE,

message = FALSE

)

library("tidyverse")

library("teplot")

# library("hrbrthemes")

# theme\_set(teplot::theme\_te())

url <- "https://github.com/kbroman/RStudioConf2019Slides/blob/master/ReadMe.md"

page <- url %>% reapr::reap\_url()

nodes\_p <- page$parsed\_html %>% rvest::html\_nodes("p")

# nodes\_p %>% rev() %>% magrittr::extract(1:20)

extract\_nodes\_after <- function(x, pattern) {

stopifnot(class(x) == "xml\_nodeset")

text <- rvest::html\_text(x)

mask <- str\_detect(text, pattern)

mask <- dplyr::cumany(mask)

x[mask]

}

nodes\_blogposts <-

nodes\_p %>%

extract\_nodes\_after("^Jacqueline Nolis")

str\_subset\_inv <- function(string, pattern, invert = TRUE) {

setdiff(string, str\_subset(string, pattern))

}

extract\_nodes\_without <- function(x, pattern) {

stopifnot(class(x) == "xml\_nodeset")

text <- rvest::html\_text(x)

res <- str\_subset\_inv(text, pattern)

res

}

# nodes\_blogposts %>% rvest::html\_text()

# nodes\_blogposts %>% rvest::html\_nodes("a") %>% rvest::html\_text()

links <-

nodes\_blogposts %>%

rvest::html\_nodes("a") %>%

rvest::html\_attr("href") %>%

str\_subset\_inv("twitter|reources")

mdtext <- nodes\_blogposts %>% rvest::html\_text()

authors <- mdtext %>% str\_extract("(^[^,]+)")

# titles <- mdtext %>% str\_replace("(^.\*)\\,[\\s\\n]+(.\*$)", "\\2")

blogposts <-

tibble(

idx\_blog = seq.int(1L, length(authors)),

author = authors,

link = links

)

blogposts

# Exclude Nolis and Lopp due to difficulty with Medium articles.

# Exclude Cortina because it is an "analysis" type of article.

# Exclude Nantz because it is just a description of a podcast.

blogposts\_filt <-

blogposts %>%

filter(!str\_detect(author, "Nolis|Lopp|Cortina|Nantz"))

blogposts\_content <-

blogposts\_filt %>%

mutate(content = purrr::map(link, ~reapr::reap\_url(.x) %>% reapr::mill()))

blogposts\_content

# content1 <-

# blogposts %>%

# slice(2) %>%

# pull(link) %>%

# reapr::reap\_url() %>%

# reapr::mill()

#

# tokens1 <-

# content1 %>%

# str\_split("\\n") %>%

# enframe(name = "line", value = "text") %>%

# unnest() %>%

# # Still some empty lines to remove.

# filter(text != "") %>%

# # tidytext::unnest\_tokens(output = "word", text)

# tidytext::unnest\_tokens(output = "ngram", input = text, token = "ngrams", n = 3)

# tokens1

tidy\_1post <-

function(text) {

text %>%

str\_split("\\n") %>%

enframe(name = "line", value = "text") %>%

unnest() %>%

# Still some empty lines to remove.

filter(text != "") %>%

# For unigrams.

tidytext::unnest\_tokens(output = "word", input = text)

# For bigrams.

# tidytext::unnest\_tokens(output = "ngram", input = text, token = "ngrams", n = 2)

}

posts\_tokenized <-

blogposts\_content %>%

select(-link) %>%

mutate(tokens = purrr::map(content, tidy\_1post))

posts\_tokenized

tokens <-

posts\_tokenized %>%

select(author, tokens) %>%

unnest(tokens)

tokens

stopwords\_base <- tidytext::get\_stopwords()

# stopwords\_base <- tidytext::get\_stopwords(source = "smart")

# Remove the numbers used for ordered lists as well as html-related things

# (mostly from Medium posts?).

stopwords\_custom <-

as.character(0:9) %>%

# c(

# "https",

# "url",

# "href",

# "httpstatus",

# "alts",

# "anchortype",

# "rel",

# "github.com",

# "200",

# "markups",

# "name",

# "text",

# "title",

# "type",

# "start",

# "end",

# "originalwidth",

# "originalheight",

# "true",

# "false",

# "id"

# ) %>%

tibble(word = .)

stopwords <-

bind\_rows(stopwords\_base, stopwords\_custom)

tokens %>%

count(word, sort = TRUE)

tokens %>%

anti\_join(stopwords) %>%

count(author, word, sort = TRUE)

# # If `ngram` is a bigram, then...

# tokens\_filt <-

# tokens %>%

# separate(ngram, into = c(paste0("word", 1:2)), remove = FALSE) %>%

# anti\_join(stopwords, by = c("word1" = "word")) %>%

# anti\_join(stopwords, by = c("word2" = "word"))

# tokens\_filt

tokens\_filt <-

tokens %>%

anti\_join(stopwords)

tokens\_filt

theme\_custom <- function() {

teplot::theme\_te(base\_size = 12) +

theme(panel.grid.major.y = element\_blank())

}

lab\_title\_suffix <- "in blog post reviews of rstudio::conf 2019."

lab\_caption <- "By Tony ElHabr"

tokens\_filt\_n <-

tokens\_filt %>%

count(word, sort = TRUE)

viz\_top\_n <-

tokens\_filt\_n %>%

top\_n(20, n) %>%

mutate(word = reorder(word, n)) %>%

ggplot(aes(x = word, y = n)) +

geom\_col(fill = "grey50") +

coord\_flip() +

theme\_custom() +

labs(

x = NULL,

y = "word Count",

title = sprintf("Most frequently used words\n%s", lab\_title\_suffix),

subtitle = "Unsuprisingly, words like `r`, `rstudio`, and `data`\nare most prominent.",

caption = lab\_caption

)

viz\_top\_n

# teproj::export\_ext\_png(

# viz\_top\_n,

# # dir = DIR\_POST,

# units = "in",

# height = 8,

# width = 4.5

# )

# sentiments\_bing <- tidytext::get\_sentiments("bing")

sentiments\_afinn <- tidytext::get\_sentiments("afinn")

# Reference: http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html#Lollipop%20Chart

# pal2 <- c("positive" = "#00ba38", "negative" = "#f8766d")

# # If `ngram` is a bigram...

# tokens\_filt %>%

# left\_join(sentiments\_afinn %>% rename\_all(funs(paste0(., "1")))) %>%

# left\_join(sentiments\_afinn %>% rename\_all(funs(paste0(., "2")))) %>%

# mutate\_at(vars(matches("score")), funs(coalesce(., 0L, .))) %>%

# mutate(n = score1 + score2) %>%

# top\_n(20, n) %>%

# ggplot(aes(x = ngram, y = n)) + ...

viz\_sentiments <-

tokens\_filt %>%

# inner\_join(sentiments\_bing) %>%

# count(word, sentiment, sort = TRUE) %>%

# top\_n(20, n) %>%

# mutate\_at(vars(word), funs(reorder(., n))) %>%

# ggplot(aes(x = word, y = n, fill = sentiment)) +

inner\_join(sentiments\_afinn) %>%

group\_by(word) %>%

summarise(.n = n(), .sum = sum(score)) %>%

ungroup() %>%

mutate(sentiment = if\_else(.sum >= 0, "positive", "negative")) %>%

top\_n(30, abs(.sum)) %>%

mutate\_at(vars(word), funs(reorder(., .sum))) %>%

ggplot(aes(x = word, y = .sum, fill = sentiment)) +

scale\_fill\_manual(values = c("negative" = "grey20", "positive" = "grey70")) +

geom\_col(show.legend = FALSE) +

coord\_flip() +

theme\_custom() +

labs(

x = NULL,

y = "AFINN polarity score",

title = sprintf("Words contributing most to sentiment\n%s", lab\_title\_suffix),

subtitle = "The content is MUCH more positve than negative.",

caption = lab\_caption

)

viz\_sentiments

# teproj::export\_ext\_png(

# viz\_sentiments,

# # dir = DIR\_POST,

# units = "in",

# height = 8,

# width = 4.5

# )

# stopwords\_custom\_topics <-

# c(

# "r",

# "rstudio",

# "data",

# "science",

# "conference",

# "conf",

# "talk",

# "talks",

# "2019"

# ) %>%

# tibble(word = .)

#

# dtms <-

# tokens\_filt %>%

# anti\_join(stopwords\_custom\_topics) %>%

# count(author, word) %>%

# tidytext::cast\_dfm(author, word, n)

#

# topics <- dtms %>% stm::stm(K = 4, verbose = FALSE, init.type = "Spectral")

# topics

# topics\_betas <- topics %>% broom::tidy()

# topics\_betas %>%

# group\_by(topic) %>%

# top\_n(2, beta) %>%

# arrange(desc(beta)) %>%

# head(30)

#

# topics\_betas\_top\_n <-

# topics\_betas %>%

# group\_by(topic) %>%

# top\_n(100, beta)

# topics\_betas\_top\_n %>%

# spread(topic, beta) %>%

# filter\_all(all\_vars(!is.na(.)))

#

# viz\_topics <-

# topics\_betas %>%

# group\_by(topic) %>%

# top\_n(10, beta) %>%

# ungroup() %>%

# mutate(

# topic = paste0("Topic ", topic),

# term = tidytext::reorder\_within(term, beta, topic)

# ) %>%

# ggplot(aes(term, beta, fill = as.factor(topic))) +

# geom\_col(show.legend = FALSE) +

# facet\_wrap(~ topic, scales = "free\_y") +

# coord\_flip() +

# tidytext::scale\_x\_reordered() +

# theme\_custom() +

# theme(axis.text.x = element\_blank()) +

# labs(

# x = NULL,

# y = "Beta probability of ngram belonging to topic",

# title = sprintf("Major \"latent\" topics and associated words discussed\n%s", lab\_title\_suffix),

# subtitle = "Although the topics themselves aren't clear, it's clear that\n`shiny`, `production`, `community`, and `packages` were major focal points.",

# caption = lab\_caption

# )

# viz\_topics

# Just to help myself out... DEFINITELY NOT a "best practice".

DIR\_VIZ <- file.path(DIR\_POST, "index-to-render\_files", "figure-markdown\_strict")

..file\_copy <-

function(file\_from,

file\_to = file\_from,

dir\_from = DIR\_VIZ,

dir\_to = DIR\_POST) {

invisible(file.copy(

from = file.path(dir\_from, file\_from),

to = file.path(dir\_to, file\_to),

overwrite = TRUE

))

}

.file\_copy <- function(file\_from, ...) {

..file\_copy(file\_from = file\_from, ...)

}

.file\_copy\_rename <-

function(file\_from, file\_to = "featured.jpg", dir\_from = DIR\_POST, ...) {

..file\_copy(file\_from = file\_from, file\_to = file\_to, dir\_from = dir\_from, ...)

}

.file\_copy("viz\_top\_n-1.png")

.file\_copy("viz\_sentiments-1.png")

.file\_copy\_rename("viz\_sentiments-1.png")