anotherFirst

Michael Pearson

11/27/2020

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
library(R.utils)
```

```
## Loading required package: R.oo
## Loading required package: R.methodsS3
## R.methodsS3 v1.8.0 (2020-02-14 07:10:20 UTC) successfully loaded. See ?R.methodsS3 for help.
## R.oo v1.23.0 successfully loaded. See ?R.oo for help.
##
## Attaching package: 'R.oo'
## The following object is masked from 'package:R.methodsS3':
##
##     throw
## The following objects are masked from 'package:methods':
##
##     getClasses, getMethods
## The following objects are masked from 'package:base':
##
##     attach, detach, load, save
## R.utils v2.9.2 successfully loaded. See ?R.utils for help.
##
## Attaching package: 'R.utils'
## The following object is masked from 'package:utils':
##
##     timestamp
## The following objects are masked from 'package:base':
##
##     cat, commandArgs, getOption, inherits, isOpen, nullfile, parse,
##     warnings
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract() masks R.utils::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
library(tidytext)
library(textstem)
```

```
## Loading required package: koRpus.lang.en
```

```
## Loading required package: koRpus
```

```
## Loading required package: sylly
```

```
## For information on available language packages for 'koRpus', run
##
##   available.koRpus.lang()
##
## and see ?install.koRpus.lang()
```

```
##
## Attaching package: 'koRpus'
```

```
## The following object is masked from 'package:readr':
##
##     tokenize
```

Including Plots

You can also embed plots, for example:

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```
set.seed(1126)
samp_per <- 0.6
sam_twit <- tweet_all[sample(1:length(tweet_all),samp_per*length(tweet_all), replace = FALSE)]
sam_test <- tweet_all[-sample(1:length(tweet_all),samp_per*length(tweet_all), replace = FALSE)]
## This is where I changed to a smaller sample size, this can be changed back
sam_twit_test <- sam_test[sample(1:length(sam_test), samp_per*length(sam_test), replace = FALSE)]
write_lines(sam_twit, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/samples/twittersample
write_lines(sam_twit_test, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/test/twittertes
sam_news <- news_all[sample(1:length(news_all),samp_per*length(news_all))]
news_test <- news_all[-sample(1:length(news_all),samp_per*length(news_all))]
## This is where I changed to a smaller sample size, this can be changed back
sam_news_test <- news_test[sample(1:length(news_test), samp_per*length(news_test), replace = FALSE)]
write_lines(sam_news_test, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/test/newstest.t
write_lines(sam_news, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/samples/newssample.t
sam_blog <- blog_all[sample(1:length(blog_all),samp_per*length(blog_all), replace = FALSE)]
blog_test <- blog_all[sample(1:length(blog_all),samp_per*length(blog_all), replace = FALSE)]
## here's where I fuck with the blogs
sam_blog_test <- blog_test [-sample(1:length(blog_test ),samp_per*length(blog_test ), replace = FALSE)]
write_lines(sam_blog, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/samples/blogsample.t
write_lines(sam_blog_test, "/Users/mutecypher/Documents/Coursera/Capstone Project/files/test/blogtest.t
```

```
samp <- "/Users/mutecypher/Documents/Coursera/Capstone Project/files/samples/"
##samplename <- readtext(samp)
##myCorpus <- corpus(samplename)
test_name <- "/Users/mutecypher/Documents/Coursera/Capstone Project/files/test/"
##testname <- readtext(test_name)
##testCorpus <- corpus(testname)kitty <-
```

Prepare the tibbles and then the n-grams

```
tweet_tib <- tibble(line = 1:samp_per*length(tweet_all),text = sam_twit)
news_tib <- tibble(line = 1:samp_per*length(news_all),text = sam_news)
blog_tib <- tibble(line = 1:samp_per*length(blog_all),text = sam_blog)
kitty <- rbind(tweet_tib, news_tib, blog_tib)

## One_grams without stop_words, no lemmatization

start_time <- Sys.time()

one_count_stop_out <- kitty %>% unnest_tokens(word, text) %>% anti_join(stop_words) %>% count(word, sor
```

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

```
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 59.28208 secs
```

```
write_csv(one_count_stop_out, "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/one_gram_
rm(one_count_stop_out)


## One_grams with lemmatization and no stop_words
start_time <- Sys.time()
one_kitty <- kitty %>% unnest_tokens(word, text)
one_kitty$word <- lemmatize_words(one_kitty$word)
one_kitty <- one_kitty %>% count(word, sort = TRUE)
one_kitty <- one_kitty %>% anti_join(stop_words)
```

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

```
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 1.354229 mins
```

```
write_csv(one_kitty, "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/one_gram_nostop_le
rm(one_kitty)
```

Now for the bigrams

```
## bi_grams

start_time <- Sys.time()
bi_count_stop_out <- kitty %>% unnest_tokens(bigram, text, token = "ngrams",n = 2)  %>% count(bigram, s
end_time <- Sys.time()
```

```
end_time - start_time
```

```
## Time difference of 5.423494 mins
```

```
start_time <- Sys.time()
bigrams_separated <- bi_count_stop_out %>% separate(bigram, c("word1", "word2"), sep = " ")
bi_sep <- bigrams_separated
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 2.333032 mins
```

```
start_time <- Sys.time()
bigrams_separated$word1 <- lemmatize_words(bigrams_separated$word1)
bigrams_separated$word2 <- lemmatize_words(bigrams_separated$word2)
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 9.96521 secs
```

```
start_time <- Sys.time()
bi_count_nostop_lemma <- bigrams_separated %>% filter(!word1 %in% stop_words$word) %>% filter(!word2 %in
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 1.393286 secs
```

```
start_time <- Sys.time()
bi_count_nostop_nolemma <- bi_sep %>% filter(!word1 %in% stop_words$word) %>% filter(!word2 %in% stop_wo
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 2.736506 secs
```

```
write_csv(bi_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/bi_gra
rm(bi_count_nostop_lemma)

write_csv(bi_count_nostop_nolemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/bi_g
rm(bi_count_nostop_lemma)
```

```
## Warning in rm(bi_count_nostop_lemma): object 'bi_count_nostop_lemma' not found
```

tri_grams

```
start_time <- Sys.time()
tri_count_stop_out <- kitty %>% unnest_tokens(trigram, text, token = "ngrams",n = 3)  %>% count(trigram
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 17.01333 mins
```

```
start_time <- Sys.time()
trigrams_separated <- tri_count_stop_out %>% separate(trigram, c("word1", "word2", "word3"), sep = " ")
tri_sep <- trigrams_separated
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 13.58437 mins
```

```
start_time <- Sys.time()
trigrams_separated$word1 <- lemmatize_words(trigrams_separated$word1)
trigrams_separated$word2 <- lemmatize_words(trigrams_separated$word2)
trigrams_separated$word3 <- lemmatize_words(trigrams_separated$word3)
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 2.654228 mins
```

```
start_time <- Sys.time()
tri_count_nostop_lemma <-trigrams_separated %>% filter(!word1 %in% stop_words$word) %>% filter(!word2 %
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 3.640484 secs
```

```
start_time <- Sys.time()
tri_count_nostop_nolemma <- tri_sep %>% filter(!word1 %in% stop_words$word1) %>% filter(!word2 %in% sto
```

```
## Warning: Unknown or uninitialised column: `word1`.
```

```
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 11.34183 secs
```

```
write_csv(tri_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/tri_
rm(tri_count_nostop_lemma)

write_csv(tri_count_nostop_nolemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/tri
rm(tri_count_nostop_nolemma)
```

quad_grams

```
start_time <- Sys.time()
quad_count_stop_out <- kitty %>% unnest_tokens(quadgram, text, token = "ngrams",n = 4)  %>% count(quadg
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 40.94391 mins
```

```
start_time <- Sys.time()
quadgrams_separated <- quad_count_stop_out %>% separate(quadgram, c("word1", "word2", "word3", "word4")
quad_sep <- quadgrams_separated
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 1.984394 hours
```

```
start_time <- Sys.time()
quadgrams_separated$word1 <- lemmatize_words(quadgrams_separated$word1)
quadgrams_separated$word2 <- lemmatize_words(quadgrams_separated$word2)
quadgrams_separated$word3 <- lemmatize_words(quadgrams_separated$word3)
quadgrams_separated$word4 <- lemmatize_words(quadgrams_separated$word4)
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 14.54671 mins
```

```
start_time <- Sys.time()
quad_count_nostop_lemma <-quadgrams_separated %>% filter(!word1 %in% stop_words$word) %>% filter(!word2
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 18.10946 secs
```

```
start_time <- Sys.time()
quad_count_nostop_nolemma <- quad_sep %>% filter(!word1 %in% stop_words$word) %>% filter(!word2 %in% st
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 15.04069 secs
```

```
write_csv(quad_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/quad
rm(quad_count_nostop_lemma)

write_csv(quad_count_nostop_nolemma , "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/qu
rm(quad_count_nostop_nolemma)
```

R Markdown

Do the combi thing for samples

```
tri_nostop_lemma <- read.csv(file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/tri
tri_nostop_lemma <- data.table(tri_nostop_lemma )
combi_tri_nostop_lemma <- unite(tri_nostop_lemma, bigrams, c("word1", "word2"), sep = " ")
rm(tri_nostop_lemma)
write.csv(combi_tri_nostop_lemma,file = "~/Documents/Coursera/Capstone Project/20sample/combi_tri_nostop
rm(combi_tri_nostop_lemma)
tri_nostop_nolemma <- read.csv(file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/tr
tri_nostop_nolemma <- data.table(tri_nostop_nolemma)
combi_tri_nostop_nolemma <- unite(tri_nostop_nolemma, bigrams, c("word1", "word2"), sep = " ")
rm(tri_nostop_nolemma)
write.csv(combi_tri_nostop_nolemma,file = "~/Documents/Coursera/Capstone Project/20sample/combi_tri_nost
rm(combi_tri_nostop_nolemma)
```

quadgrams

```
quad_nostop_lemma  <- read.csv(file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/qu
quad_nostop_lemma  <- data.table(quad_nostop_lemma )
combi_quad_nostop_lemma  <- unite(quad_nostop_lemma , trigrams, c("word1", "word2", "word3"), sep = " "
rm(quad_nostop_lemma )
write.csv(combi_quad_nostop_lemma,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample
rm(combi_quad_nostop_lemma)
quad_nostop_nolemma <- read.csv(file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/c
quad_nostop_nolemma  <- data.table(quad_nostop_nolemma)
combi_quad_nostop_nolemma <- unite(quad_nostop_nolemma, trigrams, c("word1", "word2", "word3"), sep = "
rm(quad_nostop_nolemma)
write.csv(combi_quad_nostop_nolemma ,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sam
rm(combi_quad_nostop_nolemma)


## Including Stuff at the end
```

Remove the one-offs

now let's process the ones with multiple bigrams

```
blocky <- function(trap, tim, ful_tri) {
a <- floor(nrow(tim)/100)
b <- 101
c <- a
d <- 1
 full_tri <- data.table()
 for (j in 1:b)
  {
    mid_tri <- data.table()
    if(nrow(tim) - a >= c )
      {
setkey(trixy,word1)
      for (i in d:a)
{
##setkey(trixy,bigrams)
tardis <- trixy[as.character(aggy$word1[i])]
tardis$prob <- tardis$bi_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
##trixy <- trixy[bigrams != aggy$bigrams[i],]
##print(paste("i is ",i))
##print(paste("number of rows in trixy is ",nrow(trixy)))
}
    d <- a + 1
    a <- a + c
      }
    else {
      a <- nrow(tim)
      d <- 100*floor(nrow(tim)/100) + 1
    for (i in d:a)
{
tardis <- trixy[word1 == aggy$word1[i],]
tardis$prob <- tardis$bi_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    }
   full_tri <- rbind(full_tri, mid_tri)
 }
return(full_tri)
}
combi_bi_ns_ns <- read.csv("/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/bi_gram_noste
combi_bi_ns_ns <- data.table(combi_bi_ns_ns)
trixy <- combi_bi_ns_ns[combi_bi_ns_ns$n >= 2,]
##trixy <- data.table(combi_bi_ns_ns)
aggy <- trixy[,.(sum = sum(n)), by = word1]
aggy <- aggy[aggy$sum >= 70]
aggy <- data.table(aggy)
blah <- blocky(trixy, aggy, full_tri)
write.csv(blah,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/nosingles_bi_ns_ns
rm(trixy)
rm(aggy)
```

```
rm(combi_bi_ns_ns)
rm(blah)
##print(traa)
```

Now the Trigrams

```
blocky <- function(trap, tim, ful_tri) {
a <- floor(nrow(tim)/1000)
b <- 1001
c <- a
d <- 1
 full_tri <- data.table()
 for (j in 1:b)
  {
    mid_tri <- data.table()
   if(nrow(tim) - a >= c )
      {
setkey(trixy,bigrams)
      for (i in d:a)
{
tardis <- trixy[as.character(aggy$bigrams[i])]
tardis$prob <- tardis$tri_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    d <- a + 1
    a <- a + c
      }
    else {
      a <- nrow(tim)
      d <- 1000*floor(nrow(tim)/1000) + 1
    for (i in d:a)
{
tardis <- trixy[bigrams == aggy$bigrams[i],]
tardis$prob <- tardis$tri_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
##trixy <- trixy[bigrams != aggy$bigrams[i],]
}
    }
   full_tri <- rbind(full_tri, mid_tri)
 }
return(full_tri)
}
combi_tri_ns_ns <- read.csv("/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/combi_tri_n
combi_tri_ns_ns <- data.table(combi_tri_ns_ns)
trixy <- combi_tri_ns_ns[combi_tri_ns_ns$n >= 2,]
##trixy <- data.table(combi_tri_ns_ns)
aggy <- trixy[,.(sum = sum(n)), by = bigrams]
aggy <- aggy[aggy$sum >= 50]
aggy <- data.table(aggy)
traa <- system.time(blocky(trixy, aggy, full_tri))
blah <- blocky(trixy, aggy, full_tri)
write.csv(blah,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/nosingles_tri_ns_
##rm(trixy)
```

```
##rm(aggy)
##rm(combi_tri_ns_ns)
##rm(blah)
print(traa)
```

```
##    user  system elapsed
##   1.112   0.191   1.503
```

should run first

```
blocky <- function(trap, tim, ful_tri) {
a <- floor(nrow(tim)/100)
b <- 101
c <- a
d <- 1
 full_tri <- data.table()
 for (j in 1:b)
  {
    mid_tri <- data.table()
    if(nrow(tim) - a >= c )
      {
setkey(trixy,trigrams)
      for (i in d:a)
{
tardis <- trixy[as.character(aggy$trigrams[i])]
tardis$prob <- tardis$quad_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    d <- a + 1
    a <- a + c
      }
    else {
      a <- nrow(tim)
      d <- 100*floor(nrow(tim)/100) + 1
    for (i in d:a)
{
tardis <- trixy[as.character(aggy$trigrams[i])]
tardis$prob <- tardis$tri_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    }
   full_tri <- rbind(full_tri, mid_tri)
 }
return(full_tri)
}
combi_quad_ns_ns <- read.csv("/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/combi_quad
combi_quad_ns_ns <- data.table(combi_quad_ns_ns)
trixy <- combi_quad_ns_ns[combi_quad_ns_ns$n >= 2,]
##trixy <- data.table(combi_quad_ns_ns)
aggy <- trixy[,.(sum = sum(n)), by = trigrams]
aggy <- aggy[aggy$sum >= 6]
aggy <- data.table(aggy)
blah <- blocky(trixy, aggy, full_tri)
```

```
write.csv(blah,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/nosingles_quad_ns
rm(trixy)
rm(aggy)
rm(combi_quad_ns_ns)
rm(blah)
```

Now the Quin-grams

```
blocky <- function(trap, tim, ful_tri) {
a <- floor(nrow(tim)/100)
b <- 101
c <- a
d <- 1
 full_tri <- data.table()
 for (j in 1:b)
  {
    mid_tri <- data.table()
    if(nrow(tim) - a >= c )
      {
setkey(trixy,quadgrams)
      for (i in d:a)
{
tardis <- trixy[as.character(aggy$quadgrams[i])]
tardis$prob <- tardis$quin_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    d <- a + 1
    a <- a + c
      }
    else {
      a <- nrow(tim)
      d <-  d <- 100*floor(nrow(tim)/100) + 1
    for (i in d:a)
{
tardis <- trixy[as.character(aggy$trigrams[i])]
tardis$prob <- tardis$quad_gram_ns_ns/aggy$sum[i]
mid_tri <- rbind(mid_tri, tardis)
}
    }
   full_tri <- rbind(full_tri, mid_tri)
 }
return(full_tri)
}
combi_quin_ns_ns <- read.csv("/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/combi_quin
combi_quin_ns_ns <- data.table(combi_quin_ns_ns)
trixy <- combi_quin_ns_ns[combi_quin_ns_ns$quin_gram_ns_ns >= 1,]
##trixy <- data.table(combi_quin_ns_ns)
aggy <- trixy[,.(sum = sum(quin_gram_ns_ns)), by = quadgrams]
aggy <- aggy[aggy$sum >= 3]
aggy <- data.table(aggy)
blah <- blocky(trixy, aggy, full_tri)
write.csv(blah,file = "/Users/mutecypher/Documents/Coursera/Capstone Project/20sample/nosingles_quin_ns
rm(trixy)
rm(aggy)
```

```r
rm(combi_quin_ns_ns)
rm(blah)
```