~~~
---
title: "anotherFirst"
author: "Michael Pearson"
date: "12/18/2020"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## 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:

```{r libraries}
library(R.utils)
library(tidyverse)
library(tidytext)
library(textstem)
```

## Including Plots

You can also embed plots, for example:

```{r files_in, echo=FALSE}
newslines <- countLines("/Users/mutecypher/Documents/Coursera/Capstone
Project/files/en_US/en_US.news.txt")
bloglines <- countLines("/Users/mutecypher/Documents/Coursera/Capstone
Project/files/en_US/en_US.blogs.txt")
tweetlines <- countLines("/Users/mutecypher/Documents/Coursera/Capstone
Project/files/en_US/en_US.twitter.txt")
## use that to read the files
tweet_us <- file("/Users/mutecypher/Documents/Coursera/Capstone Project/
files/en_US/en_US.twitter.txt")
tweet_all <- readLines(tweet_us, n= tweetlines, warn = FALSE, encoding =
"UTF=8", skipNul = TRUE)
close(tweet_us)
blog_us <- file("/Users/mutecypher/Documents/Coursera/Capstone Project/
files/en_US/en_US.blogs.txt")
blog_all <- readLines(blog_us, n= bloglines, warn = FALSE, encoding =
"UTF=8", skipNul = TRUE)
close(blog_us)
news_us <- file("/Users/mutecypher/Documents/Coursera/Capstone Project/
files/en_US/en_US.news.txt")
news_all <- readLines(news_us, n = newslines, warn = FALSE, encoding =
"UTF=8", skipNul = TRUE)
~~~

```
close(news_us)
```
```

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

````
```{r makes corpus}
set.seed(1218)
samp_per <- 0.79
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.txt")
write_lines(sam_twit_test, "/Users/mutecypher/Documents/Coursera/Capstone
Project/files/test/twittertest.txt")
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.txt")
write_lines(sam_news, "/Users/mutecypher/Documents/Coursera/Capstone
Project/files/samples/newssample.txt")
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.txt")
write_lines(sam_blog_test, "/Users/mutecypher/Documents/Coursera/Capstone
Project/files/test/blogtest.txt")
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
````

~~~
```{r one_gramsteaks}
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, sort = TRUE)
end_time <- Sys.time()
end_time - start_time
write_csv(one_count_stop_out, "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/one_gram_nostop_nolemma.csv")
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)
end_time <- Sys.time()
end_time - start_time
write_csv(one_kitty, "/Users/mutecypher/Documents/Coursera/Capstone
Project/20sample/one_gram_nostop_lemma.csv")
rm(one_kitty)
~~~
## Now for the bigrams

```{r, bigrams1}


## bi_grams

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

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

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

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

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

write_csv(bi_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/bi_gram_nostop_lemma.csv")
rm(bi_count_nostop_lemma)

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

## tri_grams

``` {r, trigrams1}

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


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

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

start_time <- Sys.time()
tri_count_nostop_lemma <-trigrams_separated %>% filter(!word1 %in%
stop_words$word) %>% filter(!word2 %in% stop_words$word) %>% filter(!word3
%in% stop_words$word)
~~~~

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

start_time <- Sys.time()
tri_count_nostop_nolemma <- tri_sep %>% filter(!word1 %in%
stop_words$word1) %>% filter(!word2 %in% stop_words$word) %>% filter(!word3
%in% stop_words$word)
end_time <- Sys.time()
end_time - start_time

write_csv(tri_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/tri_gram_nostop_lemma.csv")
rm(tri_count_nostop_lemma)

write_csv(tri_count_nostop_nolemma , "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/tri_gram_nostop_nolemma.csv")
rm(tri_count_nostop_nolemma)

```
## quad_grams


``` {r, quadgrams}

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


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

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

start_time <- Sys.time()
quad_count_nostop_lemma <-quadgrams_separated %>% filter(!word1 %in%
stop_words$word) %>% filter(!word2 %in% stop_words$word) %>% filter(!word3
%in% stop_words$word) %>% filter(!word4 %in% stop_words$word)
end_time <- Sys.time()
end_time - start_time

start_time <- Sys.time()
~~~

~~~~
quad_count_nostop_nolemma <- quad_sep %>% filter(!word1 %in%
stop_words$word) %>% filter(!word2 %in% stop_words$word) %>% filter(!word3
%in% stop_words$word) %>% filter(!word4 %in% stop_words$word)
end_time <- Sys.time()
end_time - start_time

write_csv(quad_count_nostop_lemma , "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/quad_gram_nostop_lemma.csv")
rm(quad_count_nostop_lemma)

write_csv(quad_count_nostop_nolemma , "/Users/mutecypher/Documents/
Coursera/Capstone Project/20sample/quad_gram_nostop_nolemma.csv")
rm(quad_count_nostop_nolemma)







```



---
title: "Secondprocess"
author: "Michael Pearson"
date: "11/19/2020"
output:
  pdf_document: default
  word_document: default
  html_document: default
---

```{r part_2, include=FALSE}

library(tidyr)
library(data.table, quietly = TRUE)
```

## R Markdown

Do the combi thing for samples



``` {r trigrams except ns_ns, eval = TRUE}
tri_nostop_lemma <- read.csv(file = "/Users/mutecypher/Documents/Coursera/
Capstone Project/20sample/tri_gram_nostop_lemma.csv" ,colClasses = c( NA,
NA, NA, NA) )
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_lemma.csv" )
rm(combi_tri_nostop_lemma)
tri_nostop_nolemma <- read.csv(file = "/Users/mutecypher/Documents/
Coursera/Capstone Project/20sample/tri_gram_nostop_nolemma.csv" ,colClasses
= c(NA, NA, NA, NA) )
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_nostop_nolemma.csv" )
rm(combi_tri_nostop_nolemma)

```


## quadgrams


``` {r quadgrams1 except ns_ns, eval = TRUE}
quad_nostop_lemma  <- read.csv(file = "/Users/mutecypher/Documents/
Coursera/Capstone Project/20sample/quad_gram_nostop_lemma.csv",colClasses =
c( NA, NA, NA, NA,NA) )
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/combi_quad_nostop_lemma.csv" )
rm(combi_quad_nostop_lemma)
quad_nostop_nolemma <- read.csv(file = "/Users/mutecypher/Documents/
Coursera/Capstone Project/20sample/quad_gram_nostop_nolemma.csv",colClasses
= c( NA, NA, NA, NA,NA) )
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/20sample/combi_quad_nostop_nolemma.csv" )
rm(combi_quad_nostop_nolemma)


## Including Stuff at the end



```



---
title: "Thirdprocess"
author: "Michael Pearson"
date: "11/19/2020"
~~~

~~~
output:
  pdf_document: default
  word_document: default
  html_document: default
---

```{r part3, include=FALSE}

library(dplyr, quietly = TRUE)
library(readr, quietly = TRUE)
#library(R.utils, quietly = TRUE)
#library(SnowballC, quietly = TRUE)
library(tidyr, quietly = TRUE)
library(data.table, quietly = TRUE)
#library(quanteda)
library(stringr)
#library(tinytex)
```

## Remove the one-offs


## now let's process the ones with multiple bigrams
```{r bigrams2, eval = TRUE}
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_nostop_lemma.csv", colClasses = c( NA, NA, NA) )
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.csv" )
rm(trixy)
rm(aggy)
rm(combi_bi_ns_ns)
rm(blah)
##print(traa)
```
## Now the Trigrams

``` {r trigrams2, eval = TRUE}
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$bigrams[i],]
}
    }
   full_tri <- rbind(full_tri, mid_tri)
 }
return(full_tri)
}
combi_tri_ns_ns <- read.csv("/Users/mutecypher/Documents/Coursera/Capstone
Project/20sample/combi_tri_nostop_lemma.csv", colClasses = c("NULL", NA,
NA, NA) )
combi_tri_ns_ns <- data.table(combi_tri_ns_ns)
trixy <- combi_tri_ns_ns[combi_tri_ns_ns$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_ns.csv" )
##rm(trixy)
##rm(aggy)
##rm(combi_tri_ns_ns)
##rm(blah)
print(traa)
```

## should run first


``` {r quadgrams2, eval = TRUE}
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_nostop_lemma.csv", colClasses = c("NULL", NA,
NA, NA) )
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_ns.csv" )
rm(trixy)
rm(aggy)
rm(combi_quad_ns_ns)
rm(blah)
```
Now the Quin-grams

```{r quingrams, eval = FALSE}
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_ns_ns.csv", colClasses = c("NULL", NA, NA, NA)
)
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_ns.csv" )
rm(trixy)
rm(aggy)
rm(combi_quin_ns_ns)
rm(blah)
```
~~~