Advanced Text Mining

Stat 220

Bastola

February 02 2022

Some more regexes

```
aboutMe <- c("My phone number is 236-748-4508.")
```

```
str_view_all(aboutMe, "\\.") # literal period "."
```

```
My phone number is 236-748-4508.
```

```
str_view_all(aboutMe, "[^(\\d)(\\s)(\\-)(\\.)]") # everything except
```

```
My phone number is 236-748-4508.
```

Alternates: OR

```
aboutMe <- c("My phone number is 236-748-4508.")
```

```
str_view(aboutMe,"8|6-")
```

```
My phone number is 236-748-4508.
```

```
str_view(aboutMe,"(8|6-)")
```

```
My phone number is 236-748-4508.
```

```
str_view_all(aboutMe,"(8|6)-")
```

```
My phone number is 236-748-4508.
```

More Duplicating Groups

```
foo <- c("addidas", "missim")
```

```
# anything then repeat anything
str_view(foo, "(.)\\1")
```

addidas

missim

```
# strings like `xyzzyx`
str_view(foo, "(.)(.)(.)\\3\\2\\1")
```

addidas

missim

```
str_view(foo, "(.)(.)\\1")
```

addidas

missim

Finding patterns

```
# find the last word in a sentence
str_view_all("it's a goat.",
             "[a-z]+\\.")
```

```
it's a goat.
```

```
# find word with  `'s`
str_view_all("it's a goat.",
             "[a-z]+\\'\\w")
```

```
it's a goat.
```

```
# find a single letter word separated by spaces
str_view_all("it's a goat.",
             "(\\s)(\\w)\\s")
```

```
it's a goat.
```

Your Turn 1

Please git clone the repository on advanced string manipulations to your local folder.

```
x <- "My SSN is 593-29-9502 and my age is 55"
y <- "My phone number is 612-643-1539"
z <- "My old SSN number is 39532 9423."
out <- str_flatten(c(x,y,z), collapse = ". ")
```

Please complete the assigned tasks.

04:00

Look ahead example

Positive look ahead operator `x(?=[y])` will find `x` when it comes before `y`

Negative version is `x(?![y])` (`x` when it comes before something that isn't `y`)

```
str_view_all("it's a goat.", "t(?=[\\.])") # t before a period
```

```
it's a goat.
```

Look ahead example

Positive look ahead operator `x(?=[y])` will find `x` when it comes before `y`

Negative version is `x(?![y])` (`x` when it comes before something that isn't `y`)

```
str_view_all("it's a goat.","[a-z]+(?=[\\.])") # 1+ letters before a period
```

```
it's a goat.
```

Look behind example

Positive look behind operator `(?<=[x])y` will find y when it follows x

Negative version is `(?<![x])y` (y when it does not follow x)

```
str_view_all("that is a top cat.","(?<=[a-z])t+")
```

```
that is a top cat.
```

Look behind example

Positive look behind operator `(?<=[x])y` will find y when it follows x

Negative version is `(?<![x])y` (y when it does not follow x)

```
str_view_all("that is a top cat.","(?<![a-z])t[a-z]+")
```

`that` is a `top` cat.

Your Turn 2

Use **negative look behind** `?<!` and **negative look ahead** `?!` operator to correct this!

```
ssn <- "([0-8]\\d{2})[-\\s]?(\\d{2})[-\\s]?(\\d{4})"
test <- c("123-45-67890","1123 45 6789")
str_view_all(test, ssn)
```

```
123-45-67890
1123 45 6789
```

04:00

Analyzing Trump tweets

What proportion of tweets (text) mention “Hillary” or “Clinton”?

```
tweets %>%
  summarize(prop = mean(str_detect(str_to_lower(text),"hillary|clinton")))
# A tibble: 1 × 1
   prop
  <dbl>
1 0.174
```

- About 17.4% of these tweets mention Hillary or Clinton.

How are the hashtags used?

```
tweets %>%
  mutate(ct = str_count(text, "#")) %>%
  select(ct, text) %>%
  summarize(prop = mean(ct > 0))
```

```
# A tibble: 1 × 1
   prop
  <dbl>
1 0.283
```

Finding URLs

URLs in tweets start with https://t.co/ followed by a string of letters or numbers

```
link <- "https://t.co/[A-Za-z\\d]+"
tweets$text[992]
[1] "I LOVE NEW YORK! #NewYorkValues \r\nhttps://t.co/dbTDhYAX1v"
```

```
str_view(tweets$text[992], link)
```

I LOVE NEW YORK! #NewYorkValues `https://t.co/dbTDhYAX1v`

What proportion of tweets have links?

```
tweets %>%
  summarize(prop = mean(str_detect(text, link)))
# A tibble: 1 × 1
   prop
  <dbl>
1 0.342
```

- about 34.2% of tweets have a link.

Removing links from tweets

```
tw_noLink <- tweets %>%
  mutate(textNoLink = str_replace_all(text, link, ""))
```

```
tw_noLink$text[992]
[1] "I LOVE NEW YORK! #NewYorkValues \r\nhttps://t.co/dbTDhYAX1v"
tw_noLink$textNoLink[992]
[1] "I LOVE NEW YORK! #NewYorkValues \r\n"
```

Get the tweets with links

```
tweets %>%
  filter(str_detect(text, link)) %>%
  select(text)
# A tibble: 517 × 1
   text
   <chr>
 1 "Join me in Fayetteville, North Carolina tomorrow evening at 6pm. Tickets no…
 2 "#ICYMI: \"Will Media Apologize to Trump?\" https://t.co/ia7rKBmioA"
 3 "Thank you Windham, New Hampshire! #TrumpPence16 #MAGA https://t.co/ZL4Q01Q4…
 4 ".@Larry_Kudlow - 'Donald Trump Is the middle-class growth candidate'\r\nhtt…
 5 "#CrookedHillary is not fit to be our next president! #TrumpPence16 \r\nhttp…
 6 "Good luck #TeamUSA\r\n#OpeningCeremony #Rio2016 https://t.co/mS8qsQpJPh"
 7 "'Trump is right about violent crime: It\x92s on the rise in major cities'\r…
 8 "Thank you Green Bay, Wisconsin! Governor @Mike_Pence and I will be back soo…
 9 "DON'T LET HILLARY CLINTON DO IT AGAIN!\r\n#TrumpPence16\r\nhttps://t.co/1mG…
10 "Thank you Des Moines, Iowa! Governor @Mike_Pence and I appreciate your supp…
# … with 507 more rows
```

Extract all tweets with links

```
tweets %>% select(text) %>%
  str_extract_all(link)
[[1]]
  [1] "https://t.co/Z80d4MYIg8" "https://t.co/ia7rKBmioA"
  [3] "https://t.co/ZL4Q01Q49s" "https://t.co/YbqkhWNm0g"
  [5] "https://t.co/I0zJO2sZKk" "https://t.co/mS8qsQpJPh"
  [7] "https://t.co/XbnZ5vktGk" "https://t.co/qsYbyrm3UR"
  [9] "https://t.co/1mGkPNZPKF" "https://t.co/gr6tGqqmcm"
 [11] "https://t.co/5yuLKyh8Q6" "https://t.co/3EzG620fpT"
 [13] "https://t.co/jsAMGO3s4P" "https://t.co/3Hcnzj0Slx"
 [15] "https://t.co/sEwLWkn1Sz" "https://t.co/UODSMp0oTo"
 [17] "https://t.co/oVfF28rWL5" "https://t.co/RhblAXkNPw"
 [19] "https://t.co/hr4O8Xgq2R" "https://t.co/Iui1F2z9ca"
 [21] "https://t.co/3Hcnzj0Slx" "https://t.co/sEwLWkn1Sz"
 [23] "https://t.co/0Ei3EdQdXB" "https://t.co/xrTQjt9WOC"
 [25] "https://t.co/VSnBoQYoZs" "https://t.co/Al5bZlRFYk"
 [27] "https://t.co/QoxJf4Xzbc" "https://t.co/IAcLfXe463"
```

Unlist the list entries

```
tweets %>% select(text) %>%
  str_extract_all(link) %>%
  unlist()              # unlist and coerce into a vector
  [1] "https://t.co/Z80d4MYIg8" "https://t.co/ia7rKBmioA"
  [3] "https://t.co/ZL4Q01Q49s" "https://t.co/YbqkhWNm0g"
  [5] "https://t.co/I0zJO2sZKk" "https://t.co/mS8qsQpJPh"
  [7] "https://t.co/XbnZ5vktGk" "https://t.co/qsYbyrm3UR"
  [9] "https://t.co/1mGkPNZPKF" "https://t.co/gr6tGqqmcm"
 [11] "https://t.co/5yuLKyh8Q6" "https://t.co/3EzG620fpT"
 [13] "https://t.co/jsAMGO3s4P" "https://t.co/3Hcnzj0Slx"
 [15] "https://t.co/sEwLWkn1Sz" "https://t.co/UODSMp0oTo"
 [17] "https://t.co/oVfF28rWL5" "https://t.co/RhblAXkNPw"
 [19] "https://t.co/hr4O8Xgq2R" "https://t.co/Iui1F2z9ca"
 [21] "https://t.co/3Hcnzj0Slx" "https://t.co/sEwLWkn1Sz"
 [23] "https://t.co/0Ei3EdQdXB" "https://t.co/xrTQjt9WOC"
 [25] "https://t.co/VSnBoQYoZs" "https://t.co/Al5bZlRFYk"
 [27] "https://t.co/QoxJf4Xzbc" "https://t.co/IAcLfXe463"
```

Your Turn 3

```
tweets<- read_csv("https://raw.githubusercontent.com/deepbas/statdatasets/main/Tr
```

Use `tweets` dataset to answer the following:

a. What proportion of tweets (text) mention "America"?

b. What proportion of these tweets in **a.** include "great"?

c. What proportion of the tweets mention "@"?

d. Remove the tweets having mentions "@".

04:00

Tidy Text

- tidy data principles
- works with existing data manipulation tools
- streamlined integration with other text mining libraries

Tidy Text Data

- Each variable is a column
- Each observation is a row
- Each type of observational unit is a table
- tidy text format is a **table with one-token-per-row**

Tokenization

```
text <- c("US opposition politicians and aid agencies have questioned
          a decision by President Donald Trump to cut off aid to three
          Central American states --- or so the story reports!")
```

```
text_data <- tibble(text = text)
text_data
# A tibble: 1 × 1
  text
  <chr>
1 "US opposition politicians and aid agencies have questioned  \n          a de…
```

Tokenization of words

```
text_data %>%
  unnest_tokens(word,text,token = "words") # Words are Default
# A tibble: 28 × 1
   word
   <chr>
 1 us
 2 opposition
 3 politicians
 4 and
 5 aid
 6 agencies
 7 have
 8 questioned
 9 a
10 decision
# … with 18 more rows
```

Tokenization of tweets

```
tibble(text = "Hey @professor, this assignment is very challenging") %>%
  unnest_tokens(word, text, token = "tweets") %>% head(3)
# A tibble: 3 × 1
  word
  <chr>
1 hey
2 @professor
3 this
```

```
tibble(text = "Hey @professor, this assignment is very challenging") %>%
  unnest_tokens(word, text, token = "words") %>% head(3)
# A tibble: 3 × 1
  word
  <chr>
1 hey
2 professor
3 this
```

Counting words

```
text_data %>%
  unnest_tokens(word,text) %>%
  count(word, sort = TRUE)
# A tibble: 26 × 2
   word          n
   <chr>     <int>
 1 aid           2
 2 to            2
 3 a             1
 4 agencies      1
 5 american      1
 6 and           1
 7 by            1
 8 central       1
 9 cut           1
10 decision      1
# … with 16 more rows
```

Stopwords

- `tidytext` comes with a database of common stop words
- carry little to no unique information, and need to be removed

```
stop_words %>% sample_n(10)
# A tibble: 10 × 2
   word       lexicon
   <chr>      <chr>
 1 j          SMART
 2 welcome    SMART
 3 always     SMART
 4 u          SMART
 5 presumably SMART
 6 as         snowball
 7 comes      SMART
 8 seemed     onix
 9 around     SMART
10 once       onix
```

Your Turn 4

```
reg <- "([^A-Za-z\\d#@']|'(?![A-Za-z\\d#@]))"
Links <- "https://t.co/[A-Za-z\\d]+"
```

Using the `tweets` dataset again, perform the following tasks:

- `filter` out words that start with "
- `replace` all instances of url links
- split the text column into tokens, flattening the table into **one-token-per-row**.
- `filter` the `stop_words` and words starting with numbers out

05:00