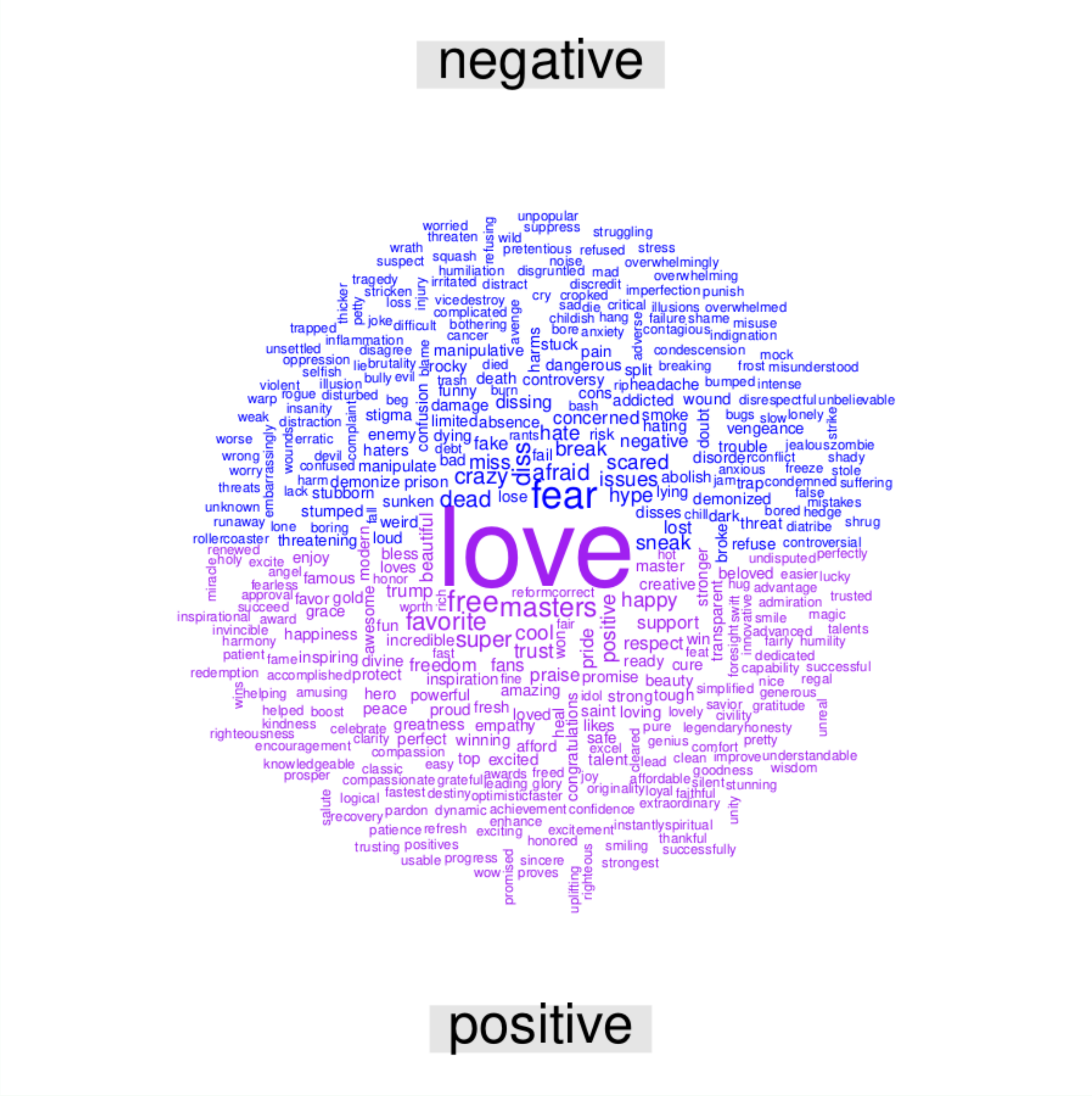
Basic String Manipulation

Spring 2023

April 19 2023

Kanye West tweets analysis



Kanye West Twitter Feed

Tweets that look like this ...

```
# A tibble: 6 × 1
  text
  <chr>
1 "Donda House 🙏👽👽 https://t.co/i4pfZiZmeN"
2 "https://t.co/48B0T5HeRk\n\nWhen we work on our shoes   apparel and accessori…
3 "I'm working on removing the word I ...correction we're working on removing t…
4 "RT @RealCandaceO: “Great minds don't think alike, they think for themselves”…
5 "THESE BRING ME BACK TO MY CHILDHOOD ... GROWING UP ON ANIME ... I NEED A PAI…
6 "RT @KidCudi: Red Table Talk!! https://t.co/bJ4Xbtqwgv"
```

... are changed to a cleaner version

```
# A tibble: 6 × 1
  text_clean
  <chr>
1 donda house
2 when we work on our shoes apparel and accessories i just think if this thing …
3 I am working on removing the word i correction we are working on removing the…
4 great minds do not think alike they think for themselves sakiyah
5 these bring me back to my childhood growing up on anime i need a pair
6 red table talk exclamationmark exclamationmark
```

Let's Define Strings

- *A string is any sequence of characters*
- *Define a string by surrounding text with either single quotes or double quotes.*

```
s <- "Hello!"    # double quotes define a string
s <- 'Hello!'    # single quotes define a string
```

The cat() or writeLines() function displays a string as it is represented inside R.

```
cat(s)
Hello!
```

```
writeLines(s)
Hello!
```

```
s <- `Hello`    # backquotes do not define a string
s <- "10""    # error - unclosed quotes
```

String Parsing

pulling apart some text or string to do something with it

- **The most common tasks in string processing include:**
  - **extracting numbers from strings, e.g. "12%"**
  - **removing unwanted characters from text, e.g. "New Jersey_* "**
  - **finding and replacing characters, e.g. "2,150"**
  - **extracting specific parts of strings, e.g. "Learning #datascience is fun!"**
  - **splitting strings into multiple values, e.g. "123 Main St, Springfield, MA, 01101"**

Regular expressions: Regex

`Regular expressions are a language for expressing patterns in strings`

- ***Regex can include special characters unlike a regular string***
- ***To use regex in R, you need to use the `stringr` package***

`stringr` package

- detecting, locating, extracting and replacing elements of strings.
- begin with `str_` and take the string as the first argument

Special characters

- The "escape" backslash \ is used to escape the special use of certain characters

```
writeLines("\"")
"
writeLines("\\")
\
writeLines("Math\\Stats")
Math\Stats
```

- To include both single and double quotes in string, escape with \

```
s <- '5\'10"'    # outer single quote
writeLines(s)
5'10"
```

```
s <- "5'10\""    # outer double quote
writeLines(s)
5'10"
```

Combining strings

```
str_c("iron", "wine")
[1] "ironwine"
str_flatten(c("iron", "wine"), collapse = " and ")
[1] "iron and wine"
```

```
a <- c("a", "b", "c")
b <- c("A", "B", "C")
str_c(a, b)
[1] "aA" "bB" "cC"
```

```
building <- "CMC"
room <- "102"
begin_time <- "12:30 a.m."
end_time <- "1:40 p.m."
days <- "MWF"
class <- "STAT 220"
```

```
str_c(class, "meets from", begin_time, "to", end_time,
      days, "in", building, room, sep = " ")
[1] "STAT 220 meets from 12:30 a.m. to 1:40 p.m. MWF in CMC 102"
```

str_length()

tells you how many characters are in each entry of a character vector

```
gapminder %>% names()
[1] "country"   "continent" "year"      "lifeExp"   "pop"       "gdpPercap"
```

```
# length of each column names
gapminder %>% names() %>% str_length()
[1] 7 9 4 7 3 9
```

str_count()

counts the number of non-overlapping matches of a pattern in each entry of a character vector

```
gapminder %>% names()
[1] "country"   "continent" "year"      "lifeExp"   "pop"       "gdpPercap"
```

```
# count number of vowels in each column name
vowels_pattern <- "[aeiouAEIOU]"
gapminder %>% names() %>% str_count(vowels_pattern)
[1] 2 3 2 3 1 2
```

str_glue()

allows one to interpolate strings and values that have been assigned to names in R

```
y <- Sys.Date() # current date
str_glue("today is {y}")
today is 2023-04-19
```

```
# base R equivalent
paste0("today is ", y)
[1] "today is 2023-04-19"
```

```
nm <- "Alex"
str_glue("Hi, my name is {nm}.")
Hi, my name is Alex.
```

```
a <- 5
str_glue("a = {a}")
a = 5
```

str_sub()

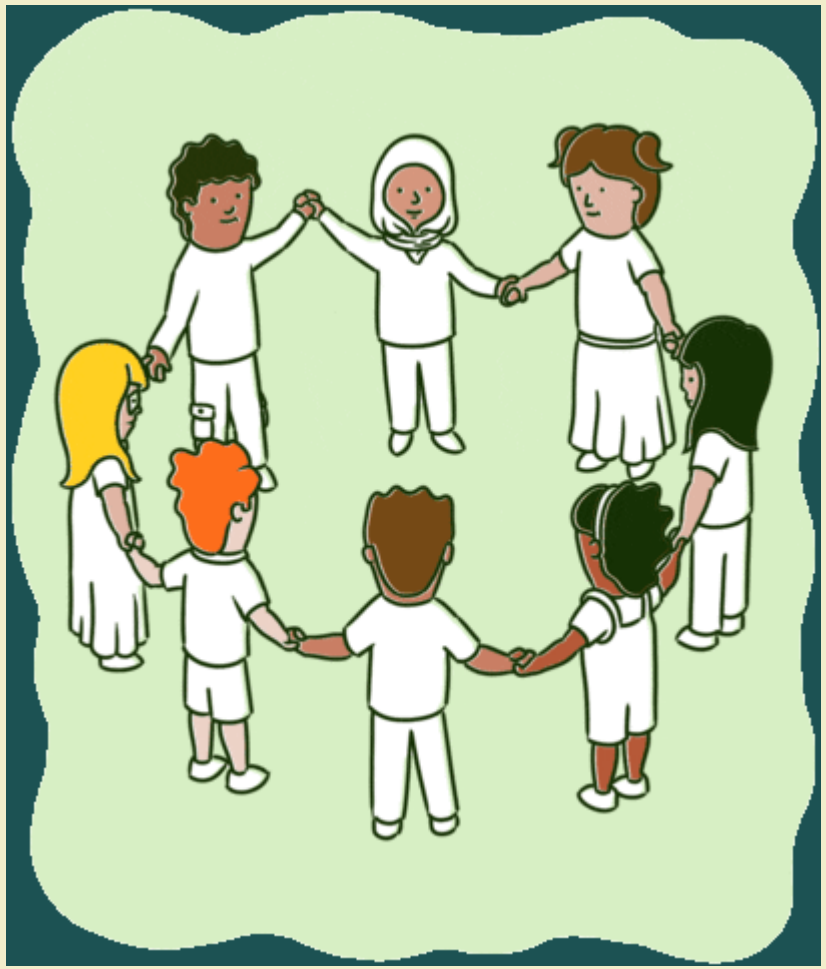
Extract and replace substrings from a character vector

```
phrase <- "cellar door"
str_sub(phrase, start = 1, end = 6)
[1] "cellar"
```

```
str_sub(phrase, start = c(1,8), end = c(6,11))
[1] "cellar" "door"
```

GROUP ACTIVITY 1

05:00



- *Let's go over to maize server/ local Rstudio and our class moodle*
- *Get the class activity 11.Rmd file*
- *Work on problem 1*

More Special Characters

- The `|` symbol inside a regex means `"or"`.
- Use `\n` to matche a newline character
- Use `\s` to match white space characters (spaces, tabs, and newlines)
- Use `\w` to match alphanumeric characters (letters and numbers)
  - can also use `[:alpha:]`
- Use `\d` to represent digits (numbers)
  - can also use `[:digit:]`

Click here for extensive lists

More Special Characters

- `^` = start of a string
- `$` = end of a string
- `.` = any character

Quantifiers

- `*` = matches the preceding character any number of times
- `+` = matches the preceding character once
- `?` = matches the preceding character at most once (i.e. optionally)
- `{n}` = matches the preceding character exactly n times

Try more regexes here

str_detect()

```
days <- c("Monday", "Tuesday", "Wednesday",
              "Thursday", "Friday", "Saturday", "Sunday")
```

```
str_detect(days, "^[Ss]un.*")
[1] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

```
days %>%
  str_which("^T")  # indices of matching entries
[1] 2 4
```

str_subset()

returns all values in a vector which match a pattern

```
gapminder$country %>%
  unique() %>%
  str_subset("^[CU].*a$")
[1] "Cambodia"   "Canada"     "China"      "Colombia"   "Costa Rica"
[6] "Croatia"    "Cuba"       "Uganda"
```

```
# columns with names starting with "c"
gapminder %>%
  names() %>%
  str_subset("^c")
[1] "country"   "continent"
```

str_sub()

extracts parts of strings based on their position with the start and end arguments

```
gapminder %>%
  names() %>%
  str_sub(start = 1, end = 6) # return the 1st 6 characters of each column name
[1] "countr" "contin" "year"   "lifeEx" "pop"    "gdpPer"
```

str_extract()

extract just the part of the string matching the specified regex instead of the entire entry

```
name_phone <- c("Moly: 250-999-8878",
       "Ali: 416-908-2044",
       "Eli: 204-192-9829",
       "May: 250-209-7047")
str_extract(name_phone,  "\\w+")
[1] "Moly" "Ali"  "Eli"  "May"
```

str_split()

splits a string into a list or matrix of pieces based on a supplied pattern

```
str_split(c("a_3", "d_4"), pattern = "_") # returns a list
[[1]]
[1] "a" "3"

[[2]]
[1] "d" "4"
```

```
str_split(c("a_3", "d_4"), pattern = "_", simplify = TRUE) # returns a matrix
     [,1] [,2]
[1,] "a"  "3"
[2,] "d"  "4"
```

str_replace()

str_replace() replaces the first instance of the detected pattern with a specified string.

```
gap_names <- gapminder %>% names

str_replace(gap_names,
            pattern = "^.{3}", # match the 1st 3 characters of each string in the vector
            replacement = "X_")
[1] "X_ntry"   "X_tinent" "X_r"      "X_eExp"   "X_"       "X_Percap"
```

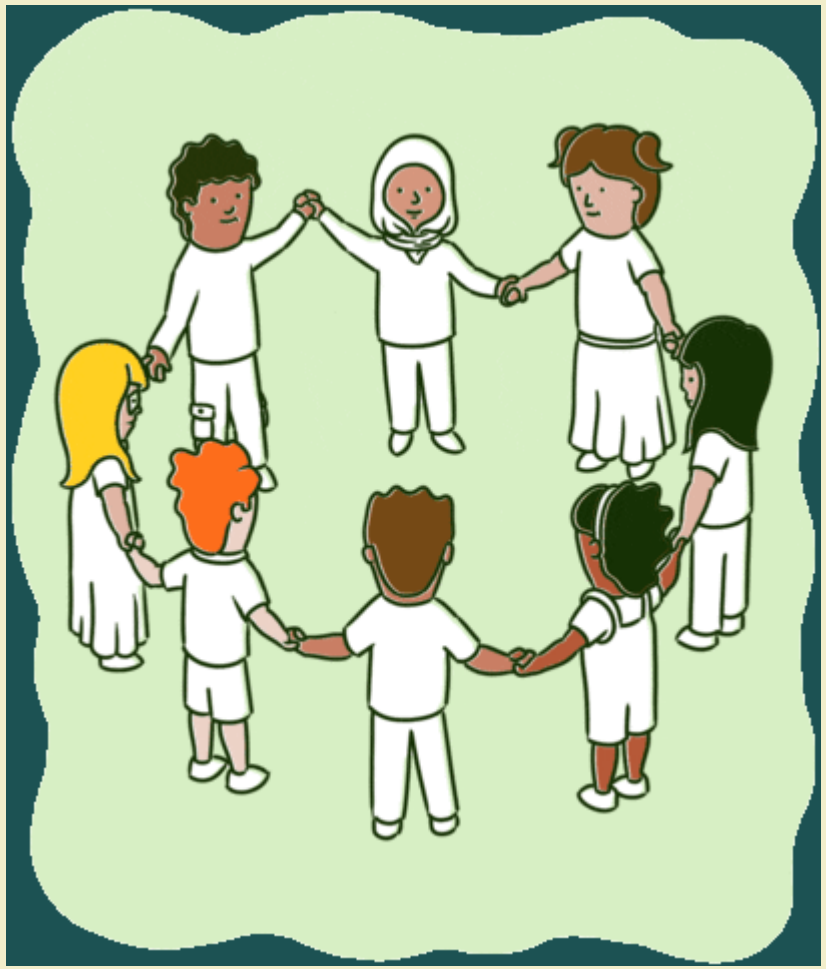
str_replace_all()

```
murders %>%
  mutate(population = str_replace_all(population, "
         total = str_replace_all(total, ",", "")) %
  mutate_at(vars(2:3), as.double)
```

```
# A tibble: 51 × 4
   state                population total murder_rat
   <chr>                     <dbl> <dbl>       <dbl
 1 Alabama                 4853875   348          7.
 2 Alaska                   737709    59          8
 3 Arizona                 6817565   309          4.
 4 Arkansas                2977853   181          6.
 5 California             38993940  1861          4.
 6 Colorado                5448819   176          3.
 7 Connecticut             3584730   117          3.
 8 Delaware                 944076    63          6.
 9 District of Columbia     670377   162         24.
10 Florida                20244914  1041          5.
# … with 41 more rows
```

GROUP ACTIVITY 2

10:00



- Go back to the activity file
- Work on problems 2-4