More date and time and strings

STAT 220

Bastola

January 28 2022

Reading the Energy Dataset

```
energy <- read_csv("https://raw.githubusercontent.com/deepbas/statdatasets/main/energy.csv",
                   col_type = cols(
                    .default = col_double(),
                     Timestamp = col_datetime(format = ""),
                     dayWeek = col_factor(levels=c("Mon","Tues","Wed","Thurs","Fri","Sat","
                    ))
```

```
energy %>% slice_max(n=2, order_by = Timestamp)
# A tibble: 2 × 90
  Timestamp            year month weekOfYear dayOfMonth dayWeek timeHour
  <dttm>              <dbl> <dbl>      <dbl>      <dbl> <fct>       <dbl>
1 2016-08-31 23:45:00  2016     8         35         31 Wed            23
2 2016-08-31 23:30:00  2016     8         35         31 Wed            23
# … with 83 more variables: timeMinute <dbl>, `100_Nevada_Street` <dbl>,
#   `104_Maple_St.` <dbl>, `106_Winona_St.` <dbl>, Allen_House <dbl>,
#   `Alumni_Guest_House/Johnson_House` <dbl>, Arboretum_Office <dbl>,
#   Art_Studios <dbl>, Benton_House <dbl>, Berg_House <dbl>, Bird_House <dbl>,
#   Boliou_Memorial_Art_Bldg. <dbl>, Burton_Hall <dbl>,
#   `Cassat_Hall_/_James_Hall` <dbl>,
#   `Center_for_Mathematics_&_Computing` <dbl>, Chaney_House <dbl>, …
```

Wide to Long

```
energy_narrow <- energy %>%
  pivot_longer( cols = `100_Nevada_Street`:Wilson_House, names_to = "building", values_to =
```

```
energy_narrow
# A tibble: 2,880,578 × 10
   Timestamp            year month weekOfYear dayOfMonth dayWeek timeHour
   <dttm>              <dbl> <dbl>      <dbl>      <dbl> <fct>       <dbl>
 1 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 2 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 3 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 4 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 5 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 6 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 7 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 8 2015-09-01 00:00:00  2015     9         35          1 Tues            0
 9 2015-09-01 00:00:00  2015     9         35          1 Tues            0
10 2015-09-01 00:00:00  2015     9         35          1 Tues            0
# … with 2,880,568 more rows, and 3 more variables: timeMinute <dbl>,
#   building <chr>, energyKWH <dbl>
```

Dates and times manipulation

```
energy$Timestamp[1]
[1] "2015-09-01 UTC"
## [1] "2015-09-01 UTC"
as.numeric(energy$Timestamp[1])
[1] 1441065600
## [1] 1441065600

# 5th timestamp
( stamp5 <- energy$Timestamp[5] )
[1] "2015-09-01 01:00:00 UTC"
```

```
mdy("1/4/2021")
[1] "2021-01-04"

dmy_hms("01/04/2020-01-30-23")
[1] "2020-04-01 01:30:23 UTC"
```

```
make_datetime(year = 2022,
              month = 1,
              day = 5,
              hour = 10,
              sec = 20)
[1] "2022-01-05 10:00:20 UTC"
```

Duration using lubriate

```
top_dest <- flights %>%
  count(dest) %>%
  slice_max(n, n = 10)

flights %>%
  semi_join(top_dest) %>%
  mutate(sch_datetime = make_datetime(
    year = year, month = month,
    day = day, hour = hour,
    min = minute)
    ) %>%
  select(dest, sch_datetime) %>%
  group_by(dest) %>%
  arrange(sch_datetime) %>%
  mutate(
    diff1 = (lag(sch_datetime) %--%
                    sch_datetime)/minutes(1),
    diff2 = interval(lag(sch_datetime),
                    sch_datetime)/minutes(1))
  summarize(medianMins1 = median(diff1,
                                 na.rm=TRUE),
            medianMins2 = median(diff2,
                                 na.rm=TRUE))
```

```
# A tibble: 10 × 3
   dest  medianMins1 medianMins2
   <chr>       <dbl>       <dbl>
 1 ATL            15          15
 2 BOS            17          17
 3 CLT            18          18
 4 DCA            34          34
 5 FLL            24          24
 6 LAX            19          19
 7 MCO            20          20
 8 MIA            25          25
 9 ORD            15          15
10 SFO            20          20
```

String Parsing

- Powerful tool useful for overcoming many data wrangling challenges
- The most common tasks in string processing include:
  - extracting numbers from strings
  - removing unwanted characters from text
  - finding and replacing characters
  - extracting specific parts of strings
  - converting free form text to more uniform formats
  - splitting strings into multiple values

`stringr` package

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

Regular expressions: Regex

- A way to describe a specific pattern of characters of text.
- To use regex in R, you need to use the stringr package
- stringr functions can take a regex as a pattern.
- Main difference between a regex and a regular string is that a regex can include special characters.

Defining 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
```

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
cat(s)
5'10"
```

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

More Special Characters

- The | symbol inside a regex means `"or"`.
- Use `\\s` to match white space characters (spaces, tabs, and newlines)
- Use `\\w` to match alphanumeric characters (letters and numbers)
- Use `\\d` to represent digits (numbers)

More Special Characters

- `^` = start of a string
- `$` = end of a string
- `.` = any character
- `[:alpha:]` = any letter
- `[:digit:]` = any digit

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

Combining strings

```
str_c("fire", "man")
[1] "fireman"
```

```
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 p.m."
end_time <- "01:40 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 p.m. to 01:40 p.m. MWF in CMC 102"
```

str_detect()

- `str_detect()` indicates whether a pattern is present in a string.

```
head(murders_raw)
# A tibble: 6 × 4
  state      population total murder_rate
  <chr>      <chr>      <chr>        <dbl>
1 Alabama    4,853,875  348            7.2
2 Alaska     737,709    59             8
3 Arizona    6,817,565  309            4.5
4 Arkansas   2,977,853  181            6.1
5 California 38,993,940 1,861          4.8
6 Colorado   5,448,819  176            3.2
```

```
# detect whether a comma is present

pattern <- ","
str_detect(murders_raw$population, pattern)
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[46] TRUE TRUE TRUE TRUE TRUE TRUE
```

str_detect()

```
days <- rep(c("Monday", "Tuesday", "Wednesday",
              "Thursday", "Friday", "Saturday", "Sunday"),2)
```

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

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

```
days %>%

  str_which("^T")  #indices of matching entries
[1]  2  4  9 11
```

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_extract() and 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"
```

- 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, "[:alpha:]*")
[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_54"), pattern = "_") # returns a list
[[1]]
[1] "a" "3"

[[2]]
[1] "d"  "54"
```

```
str_split(c("a_3", "d_54"), pattern = "_", simplify = TRUE) # returns a matrix
     [,1] [,2]
[1,] "a"  "3"
[2,] "d"  "54"
```

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"
```

Your turn 1

Please git clone the repository on basic string manipulation to your local folder.

```
cell_info
 [1] "Specimen1_RNA_SARS5_2022-03-13"  "Specimen2_RNA_H5N1_2022-01-26"
 [3] "Specimen3_RNA_SARS5_2022-01-07"  "Specimen4_DNA_COV19_2022-05-17"
 [5] "Specimen5_RNA_H5N1_2022-03-19"   "Specimen6_DNA_COV19_2022-03-22"
 [7] "Specimen7_DNA_COV19_2022-02-12"  "Specimen8_DNA_H5N1_2022-04-13"
 [9] "Specimen9_RNA_H5N1_2022-04-27"   "Specimen10_RNA_H5N1_2022-03-17"
[11] "Specimen11_DNA_H5N1_2022-02-01"  "Specimen12_DNA_COV19_2022-04-19"
```

- Complete the required tasks using `str_subset`, `str_extract`, `str_split` and `str_replace`.

07:30