Tidyverse tutorial 2 - More advanced operations

Ariane Ducellier

10/05/2023

Load R packages.

```
library(httr)
library(jsonlite)
library(mice)
```

```
##
## Attaching package: ’mice’

## The following object is masked from ’package:stats’:
##
##     filter

## The following objects are masked from ’package:base’:
##
##     cbind, rbind
```

```
library(rvest)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()         masks mice::filter(), stats::filter()
## x purrr::flatten()        masks jsonlite::flatten()
## x readr::guess_encoding() masks rvest::guess_encoding()
## x dplyr::lag()            masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

1. Dealing with missing data

```
header <- c("age", "workclass", "fnlwgt", "education",
  "education_num", "marital_status", "occupation",
  "relationship", "race", "sex", "capital_gain",
  "capital_loss", "hours_per_week", "native_country", "target")
df <- read_csv("https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data",
  col_names=header, trim_ws=TRUE)
```

```
## Rows: 32561 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (9): workclass, education, marital_status, occupation, relationship, rac...
## dbl (6): age, fnlwgt, education_num, capital_gain, capital_loss, hours_per_week
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df <-df %>%
  mutate(workclass = na_if(workclass, "?"),
         occupation = na_if(occupation, "?"),
         native_country = na_if(native_country, "?"))
```

1.1 Filling values with previous value

```
df_fill1 <- df %>%
  fill(workclass, occupation, native_country, .direction="down")
```

1.2 Filling values with most frequent value

For categorical variables.

```
m_freq_workcls <- names(table(df$workclass))[which.max(table(df$workclass))]
m_freq_occup <- names(table(df$occupation))[which.max(table(df$occupation))]
df_fill2 <- df %>%
  replace_na(list(workclass = m_freq_workcls,
                  occupation = m_freq_occup))
```

1.3 Dropping rows with missing values

Dropping rows with at least one missing value.

```
df_no_na <- df %>% na.omit()
```

Dropping rows with missing values for specific columns.

```
df_native <- df %>%
  drop_na(native_country)
```

1.4 Imputing with mice

```
data("txhousing")
txhousing$date <- date_decimal(txhousing$date, tz="GMT")
txhousing$city <- as.factor(txhousing$city)
```

```
idx <- which(rowSums(is.na(txhousing)) == 5)
txhousing <- txhousing[-idx,]
```

Impute median value for sales, volume and median.

```
txhousing$sales[is.na(txhousing$sales)] <- median(txhousing$sales, na.rm=TRUE)
txhousing$volume[is.na(txhousing$volume)] <- median(txhousing$volume, na.rm=TRUE)
txhousing$median[is.na(txhousing$median)] <- median(txhousing$median, na.rm=TRUE)
```

Use mice to impute listings and inventory.

```
impute <- mice(data.frame(txhousing[,7:8]), seed=123)
```

```
## 
##  iter imp variable
##   1   1  listings  inventory
##   1   2  listings  inventory
##   1   3  listings  inventory
##   1   4  listings  inventory
##   1   5  listings  inventory
##   2   1  listings  inventory
##   2   2  listings  inventory
##   2   3  listings  inventory
##   2   4  listings  inventory
##   2   5  listings  inventory
##   3   1  listings  inventory
##   3   2  listings  inventory
##   3   3  listings  inventory
##   3   4  listings  inventory
##   3   5  listings  inventory
##   4   1  listings  inventory
##   4   2  listings  inventory
##   4   3  listings  inventory
##   4   4  listings  inventory
##   4   5  listings  inventory
##   5   1  listings  inventory
##   5   2  listings  inventory
##   5   3  listings  inventory
##   5   4  listings  inventory
##   5   5  listings  inventory
```

```
impute_data <- complete(impute, 1)
txhousing_clean <- txhousing %>%
  mutate(listings = impute_data[,1],
         inventory = impute_data[,2])
```

2. Getting data from the web

- Go to the Wiki page.
- Right-click and select Inspect.
- Find the piece of code that highlights the table.
- Right-click and select Copy > XPath.

```
page <- "https://en.wikipedia.org/wiki/List_of_countries_by_GDP_(nominal)"
gdp <- rvest::read_html(page)
```

Get the first paragraph.

```
p1 <- gdp %>%
  html_elements("p") %>%
  html_text()
p1[3]
```

```
## [1] "Gross domestic product (GDP) is the market value of all final goods and services from a nation i
```

Get the table.

```
gdp_df <- gdp %>%
  html_elements(xpath = '//*[@id="mw-content-text"]/div[1]/table[2]') %>%
  html_table() %>%
  .[[1]]
```

3. Getting data from an API

The base URL is: https://api.fiscaldata.treasury.gov/services/api/fiscal_service

Th end point is: /v1/accounting/mts/mts_table_1

Gathering both gives you data in the JSON format.

```
url <- "https://api.fiscaldata.treasury.gov/services/api/fiscal_service/v1/accounting/mts/mts_table_1"
treasury_api <- GET(url)
```

```
result <- content(treasury_api, "text", encoding="UTF-8")
df_json <- fromJSON(result, flatten=TRUE)
df <- as.data.frame(df_json$data)
```

4. Miscellaneous functions