Pre-processing data di R

(Pelatihan data sains menggunakan R dan Gephi)

Ujang Fahmi

Pelajaran ke-2

Salam kenal dan selamat datang.
Semoga kita semua bisa saling berbagi pengalaman dan pengetahuan. Saya adalah Ujang Fahmi, Co-founder dan mentor Sadasa Academy.

Jika anda berada dan sedang membaca tutorial ini, maka kemungkinan anda adalah orang yang sedang ingin belajar data sains, atau mungkin ditugaskan untuk mempelajari R oleh institusi atau organisasi anda. Sama seperti saya dulu, dimana tanpa latar belakang enginering saya didiharuskan untuk belajar R, demi menyelesaikan tugas akhir dan akhirnya jadilah seperti saya sekarang ini.

Satu hal yang pasti, ini adalah langkah pertama dari banyak langkah yang harus dilalui, entah melalui lembaga resmi atau belajar secara mandiri. Jadi selamat belajar!!!

Ujang Fahmi,
Yogyakarta, 2021-09-24

Error yang umum 1

Di worksapce kita menulis

```
a + 198
```

Di console menampilkan:

```
Error: object 'a' not found
```

Artinya objek `a` tidak bisa ditemukan sehingga R tidak bisa menyelesaikan perintah yang diberikan

Error yang umum 2

Di workspace kita mencoba untuk memberikan perintah impor data dengan menggunakan fungsi `read_csv`, kita juga udah yakin argumen yang dibutuhkan sudah sesuai.

```
df1 = read_csv("folder1/file.csv")
```

Tapi, di `console` kita masih tetap menerima error berikut:

```
Error in read_csv("folder1/file.csv") :
could not find function "read_csv"
```

Error menunjukkan bahwa fungsi `read_csv` tidak ditemukan. Artinya kita belum memanggil library yang dibutuhkan.

Impor dan Ekspor Data

Di R kita hanya bisa mengolah data yang yang sudah kita impor atau berada di environment *R. Maka ketika ada* `error: object not found` *di console, sebagian besar disebabkan oleh data/objek yang belum ada.*

Impor Data

- Impor data merupakan salah satu langkah pertama dan utama dalam pengolahan data
- impor umumnya menggunakan sintak dengan awalan `read`. Misal `read.csv()` atau `read_csv`, sama-sama digunakan untuk impor data `.csv`

Impor data frame

Data frame atau data flate di R bisa diimpor dengan beberapa sintaks dari beberapa package.
Argumen yang dibutuhkan:

1. namfolder
2. namafile atau link data

```
# dasar
df1 = read.csv(file = "namafolder/namafile.csv", header = TRUE,
               sep = ",")
# readr
library(readr)
df2 = read_csv(file = "namafolder/namafile.csv", trim_ws = TRUE)
# xlsx
library(readxl)
df3 = read_excel(path = "namafolder/namafile.xlsx", sheet = 1)
```

Impor data json

`Json` atau `java script object notation` adalah format data yang umum digunakan untuk ditampilkan di sebuah laman/website.
Contoh:

https://raw.githubusercontent.com/eppofahmi/belajaR/master/upn-surabaya/data/sample.json

```
library(jsonlite)

df4 = fromJSON("namafolder/namafile.json",
               simplifyDataFrame = TRUE)
```

Sama seperi data lainnya, untuk mengimpor data `json` kita perlu menulis nama folder dan nama filenya. Selain data yang ada di folder dalam komputer, kita juga bisa membaca data yang ada di cloud atau website dengan ciri terdapta ekstensi data di url-nya seperti contoh di atas.

Impor data lain

Selain jenis-jenis format data seperti yang sudah dibahas. Di R kita juga bisa membaca jenis lain, seperti:

1. Data output dari SPSS dan Stata
2. Data `.nc`
3. Data list
4. etc.

TIPS:
Jika ingin mengimpor data dengan ekstensi tertentu, kita bisa mencari tutorialnya dengan mengetikan di google search. Misalnya untuk membaca file stata, kita bisa menggunakan kata kunci berikut.

read stata data in r

Your Turn 1

1. Buatlah data xlsx dan csv dengan menggunakan excel
2. Letakan di folder data
3. Impor data tersebut menggunakan sintak-sintak yang sudah dipelajari
4. Impor data json dari url yang ada dicontoh

Ekspor Data

Setelah bisa mengimpor, dan setelah melakukan pemrosesan di R, selanjutnya kita mungkin perlu menyimpan data dihardisk.

Ekspor data frame

Jika untuk mengimpor sebagian besar fungsinya diawali dengan `read*` maka untuk mengekspor data sebagian besar fungsinya diawali dengan `write`.
Contoh:

```
write_csv(namaobjek, "namafolder/namafile.csv")
```

Ekspor list

List atau bisa juga disebut sebagai data bertingkat juga dapat disimpan atau diekspor dari R. Tipe data ini biasanya digunakan untuk menyimpan data hasil pengolahan dalam satu file. Untuk mengekspor list di R umumnya dilakukan menggunakan tipe data `.rds`. Tipe data ini hanya spesifik untuk R saja.

Jika Menghendaki format yang lebih umum, maka sebaiknya list disimpan dalam format data yang juga umum. Misalnya json.

```
library(readr)

write_rds(namaobjek, "namafolder/namafile.rds")
```

Ekspor json

Untuk menyimpan json kita juga bisa menggunakan pola sintak yang sama seperti yang sudah dipelajari sebelumnya, yaitu:

```
jsonlite::write_json(x = namaobjek, path = "namafolder/namafile.json")
```

Pengecekan Data

Cobalah untuk mengimpor data ke R, dan perhatikan apa yang ada dibagian environment.

```
nama_objek1 = read_csv("folder/file.csv")
```

Gantilah nama folder dan file yang sesuai dengan yang dimiliki di laptop masing-masing

Jumlah observasi dan variabel

- Jumlah observasi menunjukkan jumlah baris, sementara jumlah variabel menunjukkan jumlah kolom
- Selain itu, sebagai analis kita juga harus tahu masing-masing tipe variabel yang dimiliki
- Untuk mengetahuinya kita bisa menggunakan fungsi glimpse()

Contoh:

```
nama_objek2 = mtcars
glimpse(nama_objek2)
```

```
## Rows: 32
## Columns: 11
## $ mpg  <dbl> 21.0, 21.0, 22.8, 21
## $ cyl  <dbl> 6, 6, 4, 6, 8, 6, 8,
## $ disp <dbl> 160.0, 160.0, 108.0,
## $ hp   <dbl> 110, 110, 93, 110, 1
## $ drat <dbl> 3.90, 3.90, 3.85, 3.
## $ wt   <dbl> 2.620, 2.875, 2.320,
## $ qsec <dbl> 16.46, 17.02, 18.61,
## $ vs   <dbl> 0, 0, 1, 1, 0, 1, 0,
## $ am   <dbl> 1, 1, 1, 0, 0, 0, 0,
## $ gear <dbl> 4, 4, 4, 3, 3, 3, 3,
## $ carb <dbl> 4, 4, 1, 1, 2, 1, 4,
```

Rangkuman data

- Untuk mendapatkan rangkuman dari data kita bisa menggunakan fungsi `summary()` atau `skim()` dari package `skimr`
- Rangkuman biasanya digunakan untuk mendapatkan informasi umum terkait dengan data yang akan diolah
- Rangkuman secara umum jika angka akan berupa *central tendency* statistik
- Rangkuman data teks biasanya berupa jenis datanya saja

Contoh:

```
summary(nama_objek2)
```

```
##       mpg             cyl
##  Min.   :10.40   Min.   :4.000
##  1st Qu.:15.43   1st Qu.:4.000
##  Median :19.20   Median :6.000
##  Mean   :20.09   Mean   :6.188
##  3rd Qu.:22.80   3rd Qu.:8.000
##  Max.   :33.90   Max.   :8.000
##       drat             wt
##  Min.   :2.760   Min.   :1.513
##  1st Qu.:3.080   1st Qu.:2.581
##  Median :3.695   Median :3.325
##  Mean   :3.597   Mean   :3.217
##  3rd Qu.:3.920   3rd Qu.:3.610
##  Max.   :4.930   Max.   :5.424
```

Penataan Data *(Data Wrangling)*

Penataan atau juga bisa disebut wrangling data juga merupakan salah satu bagian penting dalam tahap pra-pemrosesan data. Hal ini bisanya dilakukan mulai dari menstandarisasi nama kolom hingga memutuskan metode pengisian data yang kosong atau imputasi.

Data wrangling berfungsi agar analis bisa fokus pada metode dan hasil pada saat melakukan analisis. Tidak lagi bingung atau harus menata data yang tidak terstruktur yang tidak mudah dipahami.

Standar nama kolom

Nama kolom/variabel di R sebaiknya dibuat berdasarkan aturan yang sama untuk membuat objek R, yaitu:

- Tidak diawali dengan angka (`kolom1`, bukan `1kolom`)
- Tidak menggunakan spasi (`kolom_ke1` bukan `kolom ke 1`)
- Sebaiknya menggunakan huruf yang seragam (huruf kecil atau besar semua)

Contoh:

```
library(janitor)
# impor data
df1 <- read_delim("data/sample.csv",
    escape_double = FALSE)
glimpse(df1)

# nama kolom standar
df1 = janitor::clean_names(df1)
glimpse(df1)
```

Memilih Kolom dan Menyusun nama kolom

Untuk memilih nama kolom kita bisa menggunakan fungsi `select` dari `dplyr`. Namun sebelum menggunakannya, cobalah untuk menjalankan dan membaca dokumentasinya dengan menjalankan fungsi berikut:

```
library(dplyr)

?dplyr::select
```

Untuk menata atau mengurutkan kolom/variabel sesuai kebutuhan kita juga bisa mengkombinasikan fungsi `select()` dengan nomor indeks atau anam kolomnya. Misalnya:

```
nama_objek2 %>%
   select(c(1:4, 10))
```

Kita memilih kolom 1 sampai 4 dan kolom ke 10 dari data `nama_objek2`

Memilih Baris

- Untuk memilih baris kita bisa menggunakan `filter()` dari dplyr
- `filter()` digunakan untuk memilih baris dari data berdasarkan kondisi yang ditetapkan pada satu atau lebih kolom

```
nama_objek2 %>%
   filter(disp > 250)
```

Skrip diatas digunakan untuk memfilter (memilih baris) yang nilai dikolom `disp` nya lebih dari 250.

Nama baris menjadi kolom

```
df3 = mtcars
df3 %>%
   select(1:2) %>%
   head(5)
```

```
##                    mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Datsun 710        22.8   4
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
```

```
df3 = rownames_to_column(df3)
df3 %>%
   select(1:2) %>%
   head(5)
```

```
##              rowname  mpg
## 1          Mazda RX4 21.0
## 2      Mazda RX4 Wag 21.0
## 3         Datsun 710 22.8
## 4     Hornet 4 Drive 21.4
## 5 Hornet Sportabout 18.7
```

Memisahkan nilai kolom

Untuk memisahkan nilai kolom kita bisa menggunakan fungsi `separate()`

```
df4 = msleep
df4 %>%
   select(1:3) %>%
   head(5)
```

Menjadi...

```
df4 = msleep
df4 = df4 %>%
   separate(name, into = c("nama_depan",
                           "nama_tengah",
                           "nama_belakang"),
            sep = " ") %>%
   select(1:3) %>%
   head(5)
```

Menggabungkan nilai kolom

Untuk mengabungkan nilai beberapa kolom menjadi berada dalam satu kolom kita bisa menggunakan fungsi `unite()`

```
df4 %>%
   unite(col = "nama_lengkap", sep = " ", remove = FALSE)
```

Membuat kolom baru dan Merangkum data

Untuk membuat kolom baru kita bisa menggunakan fungsi `mutate()`, di sini kita akan menggunakan dengan mengisi jumlah observasi dari kolom lain.

```
fauna = msleep
rangkuman = fauna %>%
   group_by(vore) %>%
   count(order)
rangkuman

mobil = mtcars
mobil %>%
   group_by(cyl) %>%
   mutate(mean = mean(disp)) %>%
   select(cyl, disp, mean)
```

Mengatasasi Nilai Hilang

- Terkadang kita memiliki data yang tidak semua barisnya terisi. Kondisi tersebut disebut value not available, missinf value atau `NA`.
- Keputusan untuk tetap menggunakan data yang terdapat nilai kosong dalam salah satu variabelnya merupakan keputusan teoretis
- Secara teknis, ada beberapa upaya yang bisa dilakukan. Upaya ini disebut imputasi.

Anggap saja kita memiliki data berikut, dimana tidak ada nilai yang kosong di dalamnya.

```
library(missForest)
data(iris)
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

Membuat nilai kosong untuk simulasi

```
set.seed(1234)
iris.mis <- prodNA(iris, noNA = 0.2)
summary(iris.mis)
```

```
##    Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##   Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##   1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.525   1st Qu.:0.300
##   Median :5.700   Median :3.000   Median :4.400   Median :1.300
##   Mean   :5.774   Mean   :3.072   Mean   :3.756   Mean   :1.186
##   3rd Qu.:6.400   3rd Qu.:3.400   3rd Qu.:5.100   3rd Qu.:1.800
##   Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##   NA's   :25      NA's   :29      NA's   :28      NA's   :31
##         Species
##   setosa     :32
##   versicolor:39
##   virginica :42
##   NA's       :37
##
```

Mengisi Nilai kosong dengan rerata

Dari rangkuman diketahui bahwa ada 25 bari yang kosong pada kolom `Sepal.Length`

```
# mendapatakan rearata kolom Sepal.Length
# mean(iris.mis$Sepal.Length, na.rm = TRUE)
iris.mis = iris.mis %>%
   mutate(Sepal.Length = case_when(
      is.na(Sepal.Length) ~ mean(iris.mis$Sepal.Length, na.rm = TRUE),
      TRUE ~ Sepal.Length
   ))
summary(iris.mis)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.200   1st Qu.:2.800   1st Qu.:1.525   1st Qu.:0.300
##  Median :5.774   Median :3.000   Median :4.400   Median :1.300
##  Mean   :5.774   Mean   :3.072   Mean   :3.756   Mean   :1.186
##  3rd Qu.:6.300   3rd Qu.:3.400   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##                  NA's   :29      NA's   :28      NA's   :31
```

Menggunakan package

Untuk melakukan *imputation* kita bisa menggunakan fungsing `missForest()` seperti contoh skrip berikut.

```
iris.imp <- missForest(iris.mis, xtrue = iris, verbose = TRUE)

iris.imp$OOBerror
iris.imp$error
result0 <- iris.imp$ximp

summary(result0)
```

Paket lain untuk *imputation*

Selain dengan menggunakan paket `missForest`, melakukan *imputation* juga dapat dilakukan dengan beberapa paket lain dengan metode dan atau pendekatan yang berbeda. Seperti paket `Hmisc` yang juga cukup populer dikalangan pengguna R menawarkan metode *imputation* dengan beberapa algoritme.

Misalnya, dalam paket `Hmisc` terdapat fungsi `impute()` di mana pengguna dapat melakukan *imputation* dengan metode yang didefinisikan sendiri (contoh: *mean, max, median,* dll). Paket `mi` juga bisa digunakan untuk melakukan *imputation* dengan menggunakan metode *Predictive Mean Matching Method*.

Table of Contents