Data Science Untuk Pemula

Bakti Siregar S.Si, M.Sc

2020-10-02

Contents

Department of Statistics Faculty of STEM Tangerang, Banten Info: siregarbakti@gmail.com

Kata Pengantar

Buku ini dimulai dari tingkat pengantar, kursus praktik langsung, dirancang untuk `orang-orang pintar` dengan latar belakang dasar matematika atau statistik, ekonometrik, dan ilmu komputer, bahkan tanpa pengalaman pemrograman. Ini akan memperkenalkan Anda pada berbagai perspektif ilmu data, termasuk pemrosesan data, analisis, visualisasi, dan pemodelan. Setiap bab secara praktis akan dibahas dalam RStudio Integrated Development Environment (IDE).

Seperti yang mungkin sudah Anda ketahui, sains data telah menggemparkan dunia. Setiap bidang studi dan bidang bisnis telah terpengaruh, karena orang semakin menyadari nilai dari kuantitas data yang begitu besar telah dihasilkan. Tetapi untuk mengubah nilai dari data tersebut, seseorang perlu dilatih dalam keterampilan sains data yang tepat. Bahasa pemrograman R telah menjadi salah satu yang paling terkenal untuk sains data. Bahasa pemrograman R memiliki fleksibilitas, kekuatan, kecanggihan, dan ekspresifnya telah menjadikannya alat yang tak ternilai bagi ilmuwan data di seluruh dunia.

Sebagian besar materi dalam buku ini diambil dari pengalaman mengajar saya di R dalam struktur data dan algoritme, statistik komputasi, dan sistem basis data dan tentu saja diadopsi secara menyeluruh dari berbagai tahap pengembangan. Beberapa diantaranya di dapatkan dari Coursera, Data Camp, Data Flair, R-tutorial, dll. Di sini, Anda akan belajar dari dasar-dasar pemrograman R, fungsi menulis, manipulasi data, teknik visualisasi, menangani debug, dan mengoptimalkan kode. Dengan pengalaman fundamental, Anda akan memiliki dasar yang kuat untuk membangun perjalanan sains data Anda. Jadi, nikmatilah buku ini, teruslah belajar, dan berlatihlah sebanyak yang Anda inginkan.

Penulis

Saya mulai menggunakan R pada tahun 2012 ketika saya masih menjadi mahasiswa sarjana perguruan tinggi yang saat itu sedang mengerjakan pekerjaan rumah mata kuliah Statistik Komputasi. Versi R yang saya gunakan waktu itu

adalah 2.15.1. Saya adalah seorang mahasiswa jurusan matematika terapan dengan konsentrasi statistik. Setelah kuliah di bulan Oktober 2013, saya mendaftar di Management Trainee (MT) di Asuransi Sinar Mas sebagai Underwriter selama kurang lebih tiga bulan. Kemudian pindah ke proses pelatihan karyawan serupa yang disebut Office Deployment Program (ODP) di MPM Finance sebagai Credit Analyst (sekitar enam bulan). Setelah semua pengalaman ini, saya ingin meningkatkan studi saya ke jenjang berikutnya karena saya menganggap diri saya belum cukup baik untuk sukses dalam karir saya dengan pengetahuan yang terbatas ini. Oleh karena itu, selama mengikuti proses pelatihan, saya selalu berusaha mendaftar ke beberapa universitas luar negeri.

Pada bulan September 2014, saya beruntung karena saya mendapat kesempatan untuk melanjutkan gelar master saya di National Sun Yat-Sen University (Taiwan) dengan program beasiswa. Di Universitas ini, saya mendaftar di jurusan matematika terapan dengan konsentrasi statistik. Saya belajar banyak di kampus ini, mulai dari perspektif kehidupan, budaya, menghargai sesama, selalu tepat waktu, menambah ilmu statistika saya, dan sebagainya. Selama studi saya, saya bekerja dengan profesor saya sebagai asisten pengajar dan saya beruntung mendapatkan tunjangan tambahan. Untuk itu saya fokus mengajar, saya harus menghadiri Lab meeting kami satu atau dua kali seminggu, membantu acara-acara universitas, mengatur beberapa kegiatan trip, dan melakukan penelitian bersama (Prof. Mei-Hui Guo dan Prof. Chung Chang). Mereka menyarankan saya untuk belajar setidaknya dasar-dasar bahasa Mandarin untuk bertahan hidup di Taiwan dan harus meningkatkan kemampuan pemrograman saya dengan R, Python, SAS, dan MATLAB secara konsisten untuk menangani semua pekerjaan rumah saya juga.

Langkah selanjutnya dalam karir saya, saya masuk ke departemen teknik di PT Andalan Furnindo (Samora Group) sebagai seorang analis data. Ini merupakan salah satu industri gula terbaik di Indonesia, dimana saya berada di sini setidaknya selama satu tahun sejak 2017. Kemudian saya merasa bahwa tidak semua pengetahuan saya dapat berguna di perusahaan ini, bahkan tidak dapat mengembangkannya ke tingkat yang lebih baik. Maka dengan tantangan yang besar, saya memutuskan untuk pindah ke tempat baru yaitu Universitas Matana pada tahun 2018, di sini saya mulai mengabdikan diri dengan mengajar statistika bisnis dan terus berkembang dengan ilmu-ilmu baru. Di sini saya banyak belajar dan mengajar tentang sains data, saya fokus mengajar struktur data dan algoritma, sistem basis data, statistik komputasi, ekonometrika, deret waktu, kalkulus, optimasi, metode penelitian, dll. Yang terpenting, saya percaya diri untuk mengatakan bahwa saya memiliki kemampuan expert dengan beberapa bahasa pemrograman seperti R, Python, SQL, dll, dan saya juga memiliki kemampuan yang cukup baik untuk menggunakan Business Intelligence Tools, seperti Tableau dan SAS.

Bakti Siregar, M.Sc Email: siregarbakti@gmail.com atau siregar.bakti@matanauniversity.ac.id Github: https://github.com/Bakti-Siregar LinkedIn: https://www.linkedin.com/in/bakti-siregar-15955480/

Assistant

Saya adalah mahasiswa Universitas Matana jurusan Statistika Bisnis angkatan 2018. Pertama kali saya belajar R adalah ketika saya memasuki perkuliahan di semester ke-3 pada mata kuliah Struktur Data dan Algoritma yang diajarkan oleh Bapak Bakti Siregar, M.Sc.

Juenzy Hodawya Email: hodawya125@gmail.com atau juenzy.hodawya@matanauniversity.ac.id Github: https://github.com/JuenzyHodawya LinkedIn: https://www.linkedin.com/in/juenzy-hodawya-a310ab1a3/

Chapter 1

Pendahuluan

Disini di tuliskan materi

Chapter 2

Dasar-dasar R

Mari kita mulai terlebih dahulu dengan dasar-dasar R. Jika anda sudah memiliki pengalaman dengan R, anda mungkin dapat melewati bagian ini. Saya sangat menyarankan anda untuk bekerja dengan RStudio Integrated Development Environment (IDE). Pastikan juga anda mengerti dua jenis file R:

- File `R` teks `ASCII` yang hanya berisi skrip R.
- File teks `Rmd` dan `ASCII`. Jika dibuka di RStudio dapat dijalankan sebagai R-Notebook atau dikompilasi menggunakan knitr, bookdown, dll.

2.1 Operator Penugasan

Di sini saya merekapitulasi beberapa operator penugasan yang perlu anda ketahui agar anda terbiasa dengan kode R:

- `<-` dikenal sebagai operator penugasan. Artinya, "Buat nama objek di sebelah kiri sama dengan output dari koding di sebelah kanan"
- `&` Berarti AND, dalam logika Boolean
- `|` berarti OR, dalam logika Boolean.
- `!` berarti NOT, dalam logika Boolean.
- Ketika merujuk pada nilai yang dimasukkan sebagai teks, atau tanggal, masukkanlah dengan tanda kutip, seperti ini: "Amerika Serikat", atau "2016-07-26". Angka tidak dikutip.
- Saat memasukkan dua atau lebih nilai sebagai daftar (list), gabungkanlah dengan menggunakan fungsi `c`, dengan nilai yang dipisahkan oleh koma, misalnya: `c ("2020-07-26", "2020-08-04")`.
- Seperti dalam spreadsheet, anda dapat menentukan rentang nilai dengan titik dua, misalnya: `c (1:10)`, fungsi ini menghasilkant daftar bilangan bulat dari satu hingga sepuluh.

Beberapa operator umum:

- `+ -` tambah, kurang;
- – /`kali, bagi; *` > <‘ lebih besar dari, kurang dari;
- `>= < =` lebih besar dari atau sama dengan, kurang dari atau sama dengan;
- `! =` tidak sama dengan.
- Tanda sama dengan bisa sedikit membingungkan, tetapi lihat bagaimana mereka digunakan dalam kode yang kita gunakan hari ini:
- `==` menguji apakah suatu objek sama dengan nilai. Tanda Ini sering digunakan saat memfilter data;
- `=` membuat objek sama dengan nilai; bekerja seperti `<-`, tetapi digunakan di dalam tanda kurung dari sebuah fungsi.
- `$` untuk menentukan kolom individual dengan memisahkan nama frame data dan nama kolom. Catatan Objek dan nama variabel dalam R tidak boleh berisi spasi.

2.2 Kalkulator Sederhana Dalam R

Contoh	Operator S	yntax di R
5+5	Tambah	+
5-5	Kurang	-
5x5	Kali	*
5:5	Bagi	/
5^5	Pangkat	^
√25	Akar kuadrat	Sqrt
Log 5	Logaritma	Log()
Exp 5	Eksponensial	Exp()
(5/5)+5	Tanda kurung	()

2.2.1 Penugasan Variabel

Pertama, ada harus menetapkan nilai ke variabel di konsol R anda:

```
x <- 5          # Memberikan nilai ke ‘x’
y <- 5          # Memberikan nilai ke ‘y’
```

Kemudian, jalankan kode berikut di bawah ini baris demi baris untuk melihat hasilnya:

```
x + y                                   # Tambah
x - y                                   # Pengurangan
x * y                                   # Perkalian
```

```
x / y                                    # Pembagian
x ^ y                                    # Pangkat
sqrt(x * y)                      # Akar kuadrat
log(x)                                    # Logaritma
exp(y)                                    # eksponensial
(x/y) + y                            # Tanda Kurung
```

Anda juga bisa melakukan perhitungan menggunakan nama variabel:

```
total <- x + y                           # menetapkan penambahan `x` dan` y` sebagai variabel
rata_rata <- (x+x+y)/3          # menetapkan rata-rata beberapa variabel
```

2.2.2 Probabilitas

R dapat digunakan sebagai kalkulator probabilitas. Anda mungkin berharap mengetahui ini ketika melakukan kelas Intro To Probability anda.

```
dbinom (x = 3, size = 10, prob = 0.5)             # Fungsi Kepadatan (DF) P (X = 3)
dbinom (3, 10, 0.5)                                                  # hasil yang sama seperti di
pbinom (q = 3, size = 10, prob = 0.5)                  # Fungsi Distribusi Kumulatif
qbinom (p = 0.1718, size = 10, prob = 0.5) # kuantil untuk X ~ B (n = 10, p = 0.5)
rbinom (n = 10, size = 10, prob = 0.5)                  # menghasilkan variabel acak
#? distributions                                                      # Untuk informasi lebih
```

R memiliki banyak distribusi bawaan. Nama mereka mungkin berubah, tetapi awalannya tetap tidak berubah, contoh:

- `d` awalan untuk fungsi kerapatan (density function).
- `p` awalan untuk fungsi distribusi kumulatif (Cummulative distribution function/CDF).
- `q` awalan untuk fungsi kuantil (mis., CDF terbalik atau inverse CDF).
- `r` awalan untuk menghasilkan sampel acak.

Gunakanlah ide-ide dibawah ini dengan menggunakan beberapa distribusi CDF :

- `pbinom()` untuk Binomial CDF.
- `ppois()` untuk Poisson CDF.
- `pnorm()` untuk Gaussian CDF.
- `pexp()` untuk Eksponensial CDF.

2.3 Bantuan

Salah satu bagian terpenting dari bekerja dengan bahasa adalah mengetahui di mana mencari bantuan. Selain berbagai sumber daya bantuan dalam ekosistem R, selain itu R juga memiliki beberapa fasilitas in-line. Dapatkanlah bantuan untuk fungsi tertentu.

```
?dbinom
help(dbinom)
```

Jika anda tidak tahu nama fungsi yang anda cari, carilah file bantuan lokal untuk string tertentu:

```
??binom
help.search ('dbinom')
```

Atau muat menu agar anda dapat menavigasi bantuan lokal berbasis web:

```
help.start()
?help
```

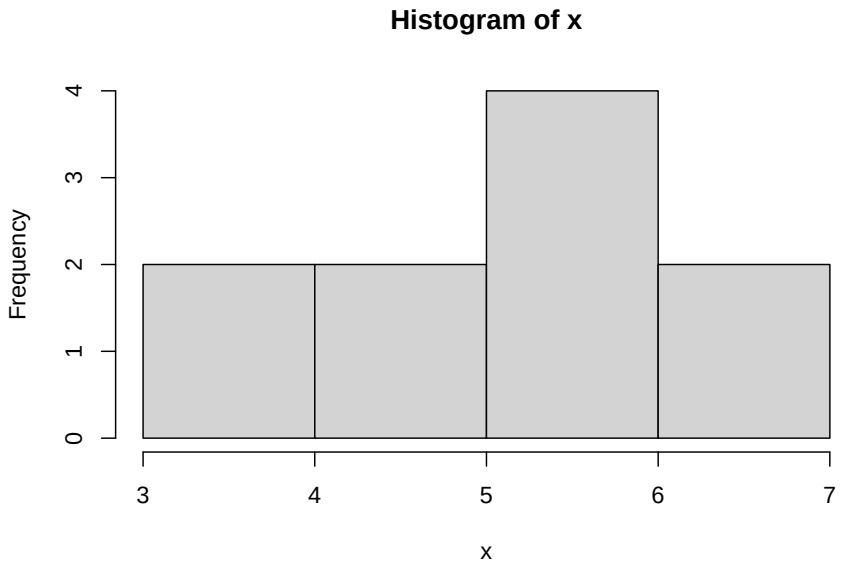
2.4 Statistik Dasar

Mari kita terapkan beberapa output ke objek bernama "x", lalu kita akan melihat beberapa statistik dasar:

```
x = rbinom (n = 10, size = 10, prob = 0.5)            # berfungsi tetapi ini merupaka
x <- rbinom (n = 10, size = 10, prob = 0.5)           # cara terbaik
```

Catatan: Jika Anda terbiasa dengan bahasa pemrograman lain, anda mungkin lebih menggunakan penugasan = daripada penugasan <-. Lihat panduan gaya Hadley untuk lebih lanjut.

```
x                                                      # cetak konten suatu objek
print (x)                                          # mencetak konten suatu objek secara i
(x <- rbinom (n = 10, size = 10, prob = 0.5))      # cara alternatif untuk menetapkan dan
mean (x)                                               # Menghitung rata-rata
var (x)                  # Menghitung variansi
sd (x)                  # Simpangan baku
hist (x)                  # Plot histogram
```



R menyimpan setiap objek yang anda buat dalam RAM. Semua koleksi dari semua objek tersebut adalah ruang kerja yang dapat anda periksa dengan:

```
ls ()                                                    # Koleksi semua objek tersebut
ls (pattern = 'x')                                  # menggunakan ls dengan pola teks
rm (x)                                                   # Menghapus variabel
ls ()                                                   # Memverifikasi
```

Anda mungkin berpikir bahwa jika suatu objek dihapus maka memorinya akan dilepaskan. Ini hampir benar, tergantung pada mekanisme negosiasi antara R dan sistem operasi.

2.5 Nilai yang Hilang

Tidak seperti pemrograman yang biasa, ketika bekerja dengan data di kehidupan nyata, anda mungkin mempunyai nilai yang hilang: pengukuran yang tidak direkam/disimpan/dll. R memiliki mekanisme yang agak canggih untuk menangani nilai-nilai yang hilang. Ini membedakan antara jenis atau tipe berikut:

- `NA`: Entri yang tidak tersedia (nilai `NA` juga memiliki kelas, jadi ada bilangan bulat `NA`, karakter NA, dll.)
- `NaN`: Bukan angka (Nilai NaN juga `NA` tetapi sebaliknya tidak benar)

R mencoba mempertahankan analis dan mengembalikan kesalahan atau `NA` ketika keberadaan nilai yang hilang membuat perhitungan tidak valid:

```
missing.example <- c(10,11,12,NA)                # Membuat vector dengan bilai yang hi
is.na(missing.example)                           # Memeriksa nilai yang hilang
which(is.na(missing.example))                    # Mengidentifikasi `NA` (hanya vector
mean(missing.example)                            # Menghitung rata-rata dengan nilai y
```

Sebagian besar fungsi biasanya memiliki mekanisme internal untuk menghadapi hal ini. Dalam fungsi rata-rata, ada argumen na.rm yang memberi tahu R cara menghapus `NA`.

```
mean(missing.example, na.rm = TRUE)              # Rata-rata tanpa nilai yang hilang
```

Temukan nilai yang hilang di kolom data frame

```
df <- data.frame( x1 = c(NA, 7, 8, 9, 3),        # Variabel numerik dengan satu nilai
                  x2 = c(4, 1, NA, NA, 4),       # Variabel numerik dengan dua nilai
                  x3 = c(1, 4, 2, 9, 6),         # Variabel numerik tanpa nilai hilan
                  x4 = c("Hello", "I am Bakti",
                         NA, "I love R",NA))     # Variabel factor dengan
is.na(df)                                        # Cara klasik untuk memeriksa `NA`
sum(is.na(df))                                   # Menghitung `NA`
apply(is.na(df),2, which)                        # Mengindeks `NA` (hanya df)
which(complete.cases(df))                        # Mengidentifikasi nilai yang diamat
```

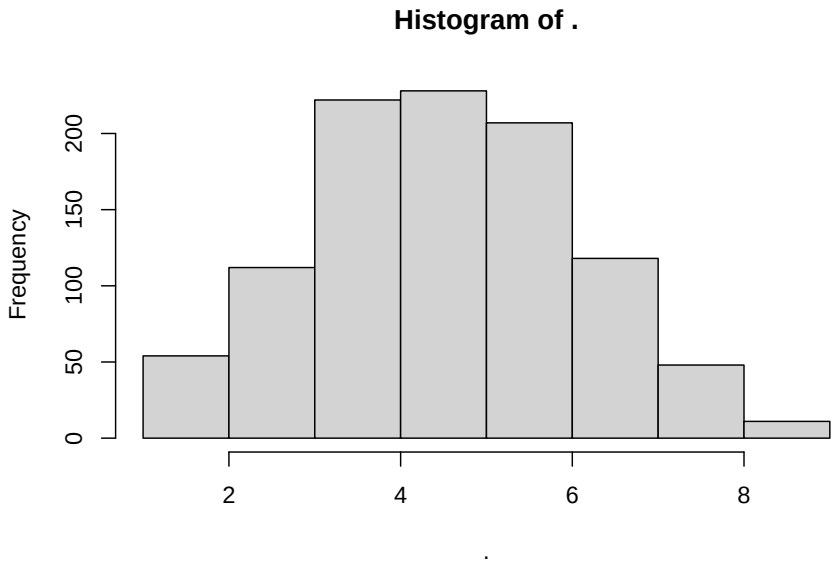
Mekanisme yang lebih umum adalah menghapus ini secara manual:

```
clean.example <- na.omit(missing.example)        # Menghapus vector `NA`
clean.df <- na.omit(df)                          # Menghapus data frame `NA`
```

2.6 Piping

Karena R berasal mewarisi banyak sumber, juga mengadopsi OS seperti Unix dan Linux. `Piping` adalah ide yang diambil dari Linux shell yang memungkinkan untuk menggunakan output dari satu ekspresi sebagai input ke yang lain. Dengan demikian, Piping membuat kode lebih mudah dibaca dan ditulis.

```
# Prasyarat:
library(magrittr)                                # Memuat fungsi piping
# help(package = "magrittr")                     # Memuat informasi `magrittr`
x <- rbinom(n=1000, size=10, prob=0.5)                  # Menghasilkan beberapa samp
# Contoh:
x %>% var()                                      # Bukan `var(x)`
x %>% hist()                                            # Sebagai ganti `hist(x)`
```




```
x %>% mean() %>% round(2) %>% add(10)              # Sebagai ganti beberapa fungsi
```

Contoh berikutnya menunjukkan manfaat dari piping. Dua potongan kode selanjutnya melakukan hal yang sama. Coba uraikan dalam pikiran Anda:

```
# Gaya fungsional Onion style
car_data <-
transform(aggregate(. ~ cyl,
                    data = subset(mtcars, hp > 100),
                    FUN = function(x) round(mean(x, 2))),
                    kpl = mpg*0.4251)
```

```
# Piping (magrittr) style
car_data <- mtcars %>%
  subset(hp > 100) %>%
  aggregate(. ~ cyl, data = ., FUN = . %>% mean %>% round(2)) %>%
  transform(kpl = mpg %>% multiply_by(0.4251)) %>%
  print
```

Tip: RStudio memiliki jalan pintas melalui keyboard untuk %>% operator. Coba `Ctrl+Shift+m`.

2.7 Membuat Vektor

Elemen paling dasar dalam R adalah vektor. Kita sekarang akan melihat cara membuat vector dan mengakses elemen-elemen (yaitu, subset). Berikut adalah

tiga cara untuk membuat vektor yang sama:

```
c(10,11,12,13,14,15,16,17,18,19,20,21)                  # Membuat vektor secara manua
10:21                                                        # Menggunakan operator
seq(from=10, to=21, by=1)                               # fungsi `seq()` dengan "by"
seq(from=10, to=21, len=12)                             # fungsi `seq()` dengan "len"
x <- 10:21                                                   # Mari kita tetapkan ke
x+2                                                          # Operasi biasanya beke
x*2                                                          # Tambahkan 2 untuk set
x^2                                                          # Pangkat 2 untuk setia
sqrt(x)                                                      # Akar kuadrat untuk se
log(x)                                                      # Logaritma untuk setiap
x <- c(0.5, 0.6)                                             # Numerik
x <- c(TRUE, FALSE)                                          # Logis
x <- c(T, F)                                                      # Logis
x <- c("a", "b", "c")                                        # Karakter
x <- 9:29                                                    # Integer
x <- c(1+0i, 2+4i)                                           # Complex
x <- vector("numeric", length = 10)                     # fungsi `vector ()` untuk me
y <- c(1.7, "a")                                            # Karakter
y <- c(TRUE, 2)                                             # Numerik
y <- c("a", TRUE)                                            # Numerik
typeof(y)                                                    # Untuk memeriksa jenis
class(y)                                                     # Untuk memeriksa jenis
```

Catatan: Menurut dokumentasi R untuk `typeof()` dan `class()`, pernyataan tentang "perbedaan utama/ main difference" adalah tidak benar. class adalah atribut dari objek yang dapat ditetapkan terlepas dari mode penyimpanan internalnya, sedangkan `typeof()` menentukan tipe (R internal) atau mode penyimpanan dari objek apa pun. Satu menggambarkan karakteristik logis sedangkan yang lain adalah karakteristik fisik dari suatu objek.

```
x <- 0:10                                          # Mari tetapkan ke objek bernama `x`
class(x)                                           # Periksa kelas `x`
as.numeric(x)                                     # Menetapkan `x` sebagai numerik
as.logical(x)                                      # Menetapkan `x` sebagai logis
as.character(x)                                    # Menetapkan `x` sebagai karakter
as.numeric(c(FALSE,TRUE,TRUE,FALSE))       # Menetapkan vektor logis sebagai angka
```

Terkadang, R tidak dapat menemukan cara untuk memaksa suatu objek dan ini dapat menghasilkan NA.

```
x <- c("a", "b", "c","1")                         # menetapkan nilai `x`
as.numeric(x)                                      # menetapkan `x` sebagai numerik
```

```
## Warning: NAs introduced by coercion
```

```r
as.logical(x)                                   # menetapkan `x` sebagai logis
as.complex(x)                                   # menetapkan `x` sebagai karakter
```

```
## Warning: NAs introduced by coercion
```

Catatan: Saat paksaan tidak masuk akal terjadi, anda biasanya akan mendapat peringatan dari R.

Kita sudah melihat bahwa elemen dasar dari objek R adalah vektor. Vektor dapat ditetapkan dengan berbagai jenis berikut:

- Karakter (character): di mana setiap elemen adalah string, mis., urutan simbol alfanumerik.
- numeric (numeric) : Dimana setiap elemen adalah bilangan real dalam format floating point presisi ganda (double precision floating point format).
- integer: di mana setiap elemen adalah integer.
- logis: di mana setiap elemen adalah TRUE, FALSE, atau NA3
- complex: di mana setiap elemen adalah bilangan kompleks.

2.8 Matriks

Matriks adalah vektor dengan atribut dimensi. Atribut dimensi itu sendiri merupakan vektor integer dengan panjang 2 (jumlah baris, jumlah kolom)

```r
m <- matrix(nrow = 2, ncol = 3)           # Membuat matriks `NA` sebanyak 2x3
m                                         # Mencetak hasilnya
dim(m)                                    # Memeriksa dimensi
attributes(m)                             # Memeriksa dimensi
```

Matriks dibuat berdasarkan kolom, sehingga entri dapat dianggap dimulai dari sudut "kiri atas" dan mengalir di kolom.

```r
m <- matrix (1: 6, nrow = 2, ncol = 3)    # Membuat sebuah matriks dengan 2x3
```

Matriks juga dapat dibuat langsung dari vektor dengan menambahkan atribut dimensi.

```r
m <- 1:10                                 # Membuat vektor `m`
dim(m) <- c(2, 5)                         # Menetapkan vektor `m` sebagai matriks sebesar 2x5
m                                         # Mencetak hasilnya
```

Matriks dapat dibuat dengan pengikatan kolom atau pengikatan baris dengan fungsi `cbind()` dan `rbind()`.

```r
x <- 1:3                                  # Membuat vektor `x`
y <- 10:12                                # Membuat vektor `y`
cbind(x, y)                               # Menggabungkan vektor `x` dan` y` dengan kolom
rbind(x, y)                               # Menggabungkan vektor `x` dan` y` dengan baris
```

2.9 List

Daftar (List) adalah jenis vektor khusus yang dapat berisi elemen dari kelas yang berbeda. List merupakan tipe data yang sangat penting dalam R dan anda harus mengenalnya dengan baik. List, dalam kombinasi dengan berbagai fungsi "terapkan (apply)" yang akan dibahas nanti, membuat kombinasi yang kuat. List dapat dibuat secara eksplisit menggunakan fungsi `list()`, yang mengambil sejumlah argumen arbitrer.

Secara umum anda dapat menggunakan dua operasi pengindeksan yang berbeda pada List:

- Kurung tunggal untuk mengembalikan daftar elemen yang dipilih `([])`
- Kurung ganda untuk mengembalikan elemen tunggal `([[]])`

```
x <- list(1:5,c("a","b","c"),TRUE,7,5)  # Membuat daftar (list) vektor `x`
x[1]                                    # Kurung tunggal
x[[1]]                                  # Kurung ganda
typeof(x[2])                            # Memeriksa jenis objek menggunakan []
typeof(x[[2]])                          # Memeriksa jenis objek menggunakan [[]]
x[c(1,2)]                               # list vektor pertama dan kedua
```

Kami juga dapat membuat daftar (list) dengan mengganti objek kosong yang ada seperti yang anda lihat dalam kode berikut:

```
x <- vector("list", length = 3)         # Membuat daftar (list) kosong (sesuai kebutuh
name <- c("a","b","c","d")              # Membuat objek sebagai variabel `name`
age  <- c(18, 19, 20, 21)               # Membuat objek sebagai variabel `age`
gender<- c(1, 0, 0, 1)                  # Membuat objek sebagai variabel `gender`
x[[1]] <- name                          # Tambah / ganti objek ke daftar `x`
x[[2]] <- age                           # Tambah / ganti objek ke daftar `x`
x[[3]] <- gender                        # Tambah / ganti objek ke daftar `x`
x                                       # Cetak hasil akhir
```

2.10 Factor

Faktor-faktor digunakan untuk mewakili data kategorikal dan dapat menjadi tidak teratur atau teratur. Orang dapat menganggap faktor sebagai vektor integer di mana setiap integer memiliki label. Faktor-faktor menjadi penting dalam pemodelan statistik dan diperlakukan secara khusus oleh fungsi pemodelan seperti `lm()` dan `glm()`. Menggunakan faktor dengan label lebih baik daripada menggunakan bilangan bulat karena faktor menggambarkan diri sendiri. Memiliki variabel yang memiliki nilai "Pria" dan "Wanita" lebih baik daripada variabel yang memiliki nilai 1 dan 2. Objek objek dapat dibuat dengan fungsi `faktor()`.

```
x <- factor(c("yes","no","yes","no"))  # Membuat objek faktor
x                                      # Cetak hasilnya
```

```r
table(x)                                # Tabel dari `x`
unclass(x)                              # Melihat representasi faktor yang mendasarinya
attr(x,"levels")                        # Melihat representasi faktor yang mendasarinya
```

2.11 Data Frame

Kerangka data (data frame) adalah tabel atau struktur mirip array dua dimensi di mana setiap kolom berisi nilai satu variabel dan setiap baris berisi satu set nilai dari setiap kolom. Berikut ini adalah karakteristik data frame.

- Nama kolom tidak boleh kosong;
- Nama baris harus unik;
- Data yang disimpan dalam data frame bisa dari numerik, faktor atau tipe karakter;
- Setiap kolom harus berisi jumlah item data yang sama.

```r
# Buat data frame pertama.
df1 <- data.frame(id = c (1:5),
                  name = c("Julian","Vanessa","Jeffry","Angel","Nikki"),
                salary = c(623.3,515.2,611.0,729.0,843.25),
            start_date = as.Date(c("2022-01-01", "2022-09-23", "2022-11-15",
                  dept = c("DS","DS","BA","DA","DS"), stringsAsFactors = F)
df1
```

```r
# Buat data frame kedua.
df2 <-data.frame(id = c (6:10),
                 name = c("Ardifo","Irene","Kefas","Sherly","Bakti"),
               salary = c(578.0,722.5,632.8,632.8,NA),
           start_date = as.Date(c("2022-05-21","2022-07-30","2022-06-17",
                                  "2022-07-30","2018-09-03")),
                 dept = c("Actuaries","Actuaries","CA","DE","Lecturer"),stringsAsFactors = F)
df2
```

```r
df3 <- rbind(df1,df2)                   # Gabungkan dua frame data
print(df3)                              # Cetak hasilnya `df3`
head(df3)                               # Cetak enam baris pertama
head(df3,6)                             # Cetak enam baris pertama
#View(df3)                              # Menggunakan RStudio seperti penampil Excel
class(df3)                              # objeknya bertipe data.frame
str(df3)                                # Dapatkan struktur data frame
dim(df3)                                # Periksa dimensi data
```

Data frame biasanya dibuat dengan membaca dalam dataset menggunakan `read.table()` atau `read.csv ()`. Namun, data frame juga dapat dibuat secara eksplisit dengan fungsi `data.frame()` atau mereka dapat dipaksakan dari jenis objek lain seperti list.

2.12 Names

Objek R dapat memiliki nama, yang sangat berguna untuk menulis kode yang dapat dibaca dan menggambarkan objek sendiri. Berikut adalah contoh pemberian nama ke vektor integer.

```
data<-df1                                # Panggil `df1` seperti yang kita gunakan di ba
names(data)                              # Periksa nama variabel data
```

```
## [1] "id"         "name"       "salary"     "start_date" "dept"
```

```
names(data)<-c("no", "nama", "gaji",
               "mulai_bekerja","divisi") # Mengubah nama nama variabel menjadi Indonesia
data                                     # Cetak hasilnya
```

```
##   no    nama   gaji mulai_bekerja divisi
## 1  1  Julian 623.30    2022-01-01     DS
## 2  2 Vanessa 515.20    2022-09-23     DS
## 3  3  Jeffry 611.00    2022-11-15     BA
## 4  4   Angel 729.00    2022-05-11     DA
## 5  5   Nikki 843.25    2022-03-27     DS
```

Matriks dapat memiliki nama kolom dan baris.

```
m <- matrix(1:4, nrow = 2, ncol = 2)
dimnames(m) <- list(c("a", "b"), c("c", "d"))
m
```

```
##   c d
## a 1 3
## b 2 4
```

Nama kolom dan nama baris dapat diatur secara terpisah menggunakan fungsi `colnames()` dan `rownames()`.

```
colnames(m) <- c("h", "f")
rownames(m) <- c("x", "z")
m
```

```
##   h f
## x 1 3
## z 2 4
```

Catatan: Dalam data frame, ada fungsi terpisah untuk mengatur nama baris, fungsi `row.names()`. Juga, data frane tidak memiliki nama kolom, mereka hanya memiliki nama (seperti list). Jadi untuk mengatur nama kolom dari data frame gunakan saja fungsi `names()`. Ya, saya tahu ini membingungkan. Berikut ringkasan singkatnya:

Objek T	etapkan nama kolom	Tetapkan nama baris
data frame	names()	row.names()
matrix	colnames()	rownames()

2.13 Pengeculian (Ekstraksi)

R menyediakan banyak cara untuk mengelompokkan dan mengekstrak elemen dari vektor dan objek lainnya. Dasar-dasarnya cukup sederhana, tetapi dengan tidak memperhatikan "kepribadian" dari setiap mekanisme ekstraksi dapat menyebabkan anda sakit kepala. Sebagai permulaan, ekstraksi dilakukan dengan operator `[]`. Operator dapat mengambil vektor dari banyak jenis.

```
#View(mtcars)                          # Lihat dataset `mtcars` (lingkungan R)
#?mtcars                               # Informasi detail tentang mtcars
typeof(mtcars)                         # Periksa jenis data
class(mtcars)                          # Periksa jenis data
mtcars[1,5]                            # Ekstrak elemen di baris 1 dan kolom 5.
mtcars[1:5,]                           # Ekstrak enam baris pertama mtcars
mtcars[,1:5]                           # Ekstrak enam kolom mtcars pertama
mtcars[,'mpg']                         # Ekstrak kolom khusus
mtcars$mpg                             # Ekstrak kolom khusus
mtcars['Mazda RX4',]                   # Ekstrak baris khusus

subset(mtcars, select=mpg)             # Ekstrak / subset kolom spesifik
subset(mtcars, select=1)               # Ekstrak / subset kolom spesifik
subset(mtcars, select= c(1,2))         # Ekstrak / subset kolom pertama & kedua
subset(mtcars, select= c(1:5))         # Ekstrak / subset kolom spesifik

min(mtcars$mpg )                       # Temukan minimum gallon Miles/(US)
max(mtcars$mpg , na.rm = TRUE)         # Temukan maksimum gallon Miles/(US)
mean(mtcars$mpg , na.rm = TRUE)        # Temukan rata-rata gallon Miles/(US)
var(mtcars$mpg , na.rm = TRUE)         # Temukan varians gallon Miles/(US)
sd(mtcars$mpg , na.rm = TRUE)          # Temukan standar deviasi gallon Miles/(US)
```

Menambahkan variabel ke data.frame dapat dilakukan dengan menetapkan vektor baru. Kekuatan objek data.frame adalah bahwa data.frame menerima hampir semua jenis vektor, yaitu bilangan bulat, numerik, logis, faktor, dan karakter.

```
mtcars$newvar1 <- mtcars$mpg - mtcars$qsec
mtcars$newvar2 <- mtcars$newvar1 > 0
mtcars$newvar3 <- ifelse(mtcars$newvar2, "good", "bad")
mtcars$newvar4 <- factor(mtcars$newvar1>10 & mtcars$newvar3=="good",
                         labels = c("level1","level2"))
```

2.14 Petunjuk Penting

Beberapa petunjuk bermanfaat untuk Rstudio (IDE) meliputi:

Kata Kunci	Perintah	Detail
Ctrl + Return (Enter)	untuk menjalankan baris dari editor	~
Ctrl + Shift + #	untuk fokus pada tab bantuan	kontradiktif
Alt + Shift + k	untuk jalur pintas keyboard RStudio	~
Ctrl + r	untuk menelusuri sejarah perintah	~
Alt + Shift + j	untuk menavigasi antar bagian kode	~
Ctrl + 1	untuk melompat ke editor	tab untuk penyelesaian otomatis
Ctrl + 2	untuk melompat ke konsol	tab untuk penyelesaian otomatis
Ctrl + 8	untuk melompat ke environment list	tab untuk pelengkapan otomatis
Alt + l	Collapse chunk	Code Folding
Alt + Shift + l	Unfold chunk	Code Folding
Alt + o	Collapse all	Code Folding
Alt + Shift + o	Unfold all	Code Folding
Alt + "-"	untuk operator penugasan <-	~
Alt + Shift + c	kode komentar/tanda komentar dalam file	.R kontradiktif

Saat ini, saya sarankan anda untuk menggunakan RStudio di komputer (PC) anda, tetapi di sini saya sarankan beberapa IDE lain:

- Jupyter Lab: IDE yang sangat menjanjikan, awalnya dirancang untuk Python, yang juga mendukung R. Pada saat penulisan ini, tampaknya RStudio lebih nyaman untuk R, tetapi itu jelas merupakan IDE yang harus diikuti. Lihat ulasan Max Woolf.
- Eclipse: Jika anda seorang programmer Java, anda mungkin akrab dengan Eclipse, yang memang memiliki R plugin: StatEt.
- Emacs: Jika anda adalah penggemar Emacs, anda dapat menemukan R plugin: ESS.
- Visual Studio juga mendukung R. Jika anda membutuhkan R untuk tujuan komersial, mungkin ada baiknya mencoba Microsoft's R, daripada R biasa. Lihat di sini untuk petunjuk pemasangan.

Chapter 3

Pemrograman R

We describe our methods in this chapter.

Chapter 4

Interface Data

Dengan R, kita dapat membaca data dari file yang tersimpan di luar Environment R. Kita juga dapat menulis data menjadi bentuk file yang akan disimpan dan diakses oleh sistem operasi. R dapat membaca dan menulis ke beberapa format file seperti `csv, excel, txt, rds, xml, json, dll`.

4.1 Direktori Kerja (Working Directory)

Sebelum kita mulai bekerja menggunakan data (data interface), pertama-tama pastikan working directory Anda berada di koneksi yang tepat. Anda dapat memeriksanya menggunakan fungsi `getwd()`. Anda juga dapat mengeset sebuah working directory baru menggunakan fungsi `setwd()`.

```
print(getwd())                                  # memperoleh dan mencetak working directory
```

```
## [1] "C:/Users/Bakti/Desktop/Website/Data-Science-Untuk-Pemula"
```

```
getwd()                                         # memperoleh dan mencetak working directory
```

```
## [1] "C:/Users/Bakti/Desktop/Website/Data-Science-Untuk-Pemula"
```

```
#setwd("C:/Users/Bakti/Desktop/")               # mengeset working directory Anda
#setwd("C:\\Users\\Bakti\\Desktop\\")           # atau dengan cara ini
```

4.2 Membaca/Menulis CSV

Ini adalah sebuah contoh sederhana dari fungsi `read.csv` untuk membaca sebuah file CSV yang tersedia di working directory Anda saat ini.

```
# csv <- read.csv(file.choose())                    # tampilan file csv tanpa `setwd`
csv1 <- read.csv("Data/csv1.csv",sep = ",")         # ini untuk membaca data dengan pemi
csv2 <- read.csv("Data/csv2.csv",sep = ";")         # ini untuk membaca data dengan pemi
head(csv2,3)                                        # mencetak hasil dari `data1`
```

```
##   id    name salary start_date dept
## 1  1  Julian  623,3 2022-01-01   DS
## 2  2 Vanessa  515,2 2022-09-23   DS
## 3  3  Jeffry    611 2022-11-15   BA
```

R dapat membuat file csv dari data frame yang ada. Fungsi `write.csv()` digunakan untuk membuat file csv. File ini dibuat di working directory.

```
write.csv(csv2,"Data/csv3.csv")
```

4.3 Membaca/Menulis Excel

Microsoft Excel adalah program spreadsheet yang paling banyak digunakan yang menyimpan data dalam format .xls atau .xlsx. R dapat membaca file-file ini secara langsung menggunakan beberapa packages (paket) khusus excel. Kita akan menggunakan package `readxl`.

```
# xlsx <- read_excel(file.choose())                 # tampilan file xlsx tanpa `setwd`
# install.packages("readxl")                        # menginstal package `readxl`
library("readxl")                                   # memuat package `readxl`
xlsx1<-read_excel("Data/xlsx1.xlsx",sheet=1)        # membaca/impor data xlsx dari PC
head(xlsx1,3)
```

```
## # A tibble: 3 x 5
##      id name    salary start_date          dept
##   <dbl> <chr>   <chr>  <dttm>              <chr>
## 1     1 Julian  623    2022-01-01 00:00:00 DS
## 2     2 Vanessa 515    2022-09-23 00:00:00 DS
## 3     3 Jeffry  611    2022-11-15 00:00:00 BA
```

Untuk menulis data frame yang ada ke dalam file excel Anda harus menginstal package `writexl`.

```
# install.packages("writexl")                       # menginstal  package `readxl`
library("writexl")                                  # memuat package `readxl`
writexl::write_xlsx(xlsx1,"Data/xlsx2.xlsx")        # memeriksa output di working direc
```

4.4 Membaca/Menulis TXT dan RDS

Salah satu tugas paling umum yang kita lakukan adalah membaca data dari file CSV dan XLSX. Namun, proses membaca data bisa melambat untuk file CSV atau XLSX yang besar. Salah satu trik yang rapi adalah membaca data

dan menyimpannya sebagai file TXT atau file biner R (RDS). Untuk mengimpor file TXT, kita menggunakan `read.table()` dan untuk mengimpor file RDS kita dapat menggunakan `readRDS()`.

```
# txt1 <- read.table(file.choose())                # tampilan file TXT tanpa `setwd`
txt1 <- read.table("Data/txt1.txt")                # membaca/memuat format TXT (notepad)
txt1 <- source("Data/txt1.Rdmpd")                  # membaca/memuat format TXT (Rdmpd)
rds1 <- readRDS("Data/rds1.rds")                   # membaca/memuat format RDS biner
ascii1 <- readRDS("Data/ascii1.rds")               # membaca/memuat format ASCII biner
```

Untuk menyimpan data sebagai file TXT kita dapat menggunakan fungsi `write.table()`, dan untuk file biner R (RDS) kita dapat menggunakan fungsi `saveRDS()`. Fungsi-fungsi tersebut banyak digunakan oleh R sendiri, sebagai contoh untuk menyimpan data meta untuk suatu package dan untuk menyimpan basis data help.search: ekstensi file `.rds` paling sering digunakan. Format ini dapat berupa biner atau ASCII. Biner lebih padat, sementara ASCII akan lebih efisien dengan sistem kontrol versi seperti Git.

```
data <- read.csv("Data/csv1.csv",sep = ",")        # membaca/mengimpor data dari PC Anda

write.table(data,"Data/txt2.txt")                  # meyimpan dalam format TXT (notepad)
dump("data", "Data/txt2.Rdmpd")                    # meyimpan dalam format TXT (Rdmpd)
saveRDS(data, "Data/rds2.rds")                     # menyimpan suatu objek dalam format RDS biner
saveRDS(data, "ascii2.rds", ascii=TRUE)            # menyimpan suatu objek dalam format RDS ASCII
```

4.5 Membaca/Menulis XML

XML adalah suatu format file yang membagikan format file dan data dari World Wide Web (www), intranet, dan di tempat lain menggunakan teks ASCII standar. XML adalah singkatan dari Extensible Markup Language, lebih lanjut tentang XML. XML berisi tag markup, mirip dengan HTML. Tag markup di xml menggambarkan arti data yang terkandung dalam file tersebut,tetapi tidak seperti HTML di mana tag markup menggambarkan struktur halaman. Anda dapat membaca suatu file xml di R menggunakan package "XML". File xml dibaca oleh R menggunakan fungsi `xmlParse()`. File itu disimpan sebagai daftar (list) di R.

```
library("XML")                                     # memuat package yang dibutuhkan untuk membaca
library("methods")                                 # memuat paket lainnya yang dibutuhkan.
result <- xmlParse(file = "Data/xml1.xml")         # memberikan nama file input ke fungsi
print(result)                                      # Mencetak hasilnya
```

```
## <?xml version="1.0"?>
## <RECORDS>
##   <EMPLOYEE>
##     <id>1</id>
##     <name>Julian</name>
```

```
##     <salary>623.3</salary>
##     <start_date>1/1/2022</start_date>
##     <dept>DS</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>2</id>
##     <name>Vanessa</name>
##     <salary>515.2</salary>
##     <start_date>9/23/2022</start_date>
##     <dept>DS</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>3</id>
##     <name>Jeffry</name>
##     <salary>611</salary>
##     <start_date>11/15/2022</start_date>
##     <dept>BA</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>4</id>
##     <name>Angel</name>
##     <salary>729</salary>
##     <start_date>5/11/2022</start_date>
##     <dept>BA</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>5</id>
##     <name>Nikki</name>
##     <salary>843.25</salary>
##     <start_date>3/27/2022</start_date>
##     <dept>DS</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>6</id>
##     <name>Ardifo</name>
##     <salary>578</salary>
##     <start_date>5/21/2022</start_date>
##     <dept>Actuaries</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>7</id>
##     <name>Irene</name>
##     <salary>722.5</salary>
##     <start_date>7/30/2022</start_date>
##     <dept>Actuaries</dept>
##   </EMPLOYEE>
```

```
##   <EMPLOYEE>
##     <id>8</id>
##     <name>Kefas</name>
##     <salary>632.8</salary>
##     <start_date>6/17/2022</start_date>
##     <dept>CA</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>9</id>
##     <name>Sherly</name>
##     <salary>632.8</salary>
##     <start_date>7/30/2022</start_date>
##     <dept>DE</dept>
##   </EMPLOYEE>
##   <EMPLOYEE>
##     <id>10</id>
##     <name>Bakti</name>
##     <salary>NA</salary>
##     <start_date>9/03/2018</start_date>
##     <dept>Lecturer</dept>
##   </EMPLOYEE>
## </RECORDS>
##
```

Untuk menangani data secara efektif dalam file besar, kita membaca data dalam file `xml` sebagai suatu data frame. Kemudian memproses data frame untuk analisis data.

```
xmldataframe <- xmlToDataFrame(result)              # mengonversi file input `xml` ke bentuk data
head(xmldataframe,3)                                # Mencetak hasilnya
```

```
##   id    name salary start_date dept
## 1  1  Julian  623.3   1/1/2022   DS
## 2  2 Vanessa  515.2  9/23/2022   DS
## 3  3  Jeffry    611 11/15/2022   BA
```

Buat file XML dengan menyalin data ini ke editor teks seperti notepad. Simpan file dengan ekstensi .xml dan pilih jenis file sebagai all files(.).

4.6 Membaca/Menulis JSON

File JSON menyimpan data sebagai teks dalam format yang dapat dibaca oleh manusia. Json adalah singkatan dari JavaScript Object Notation. R dapat membaca file JSON menggunakan package rjson. Lebih lanjut tentang JSON.

```
library("rjson")                                    # memuat package untuk membaca file JSON
json1 <- fromJSON(file= "Data/json1.json")          # memberikan nama file input ke fungsi
```

```
print(json1)                                    # Mencetak hasilnya
```

```
## $id
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
##
## $name
##  [1] "Julian"  "Vanessa" "Jeffry"  "Angel"   "Nikki"   "Ardifo"  "Irene"
##  [8] "Kefas"   "Sherly"  "Bakti"
##
## $salary
##  [1] "623.3"  "515.2"  "611"    "729"    "843.25" "578"    "722.5"  "632.8"
##  [9] "632.8"  "NA"
##
## $start_date
##  [1] "1/1/2022"   "9/23/2022"  "11/15/2022" "5/11/2022"  "3/27/2022"
##  [6] "5/21/2022"  "7/30/2022"  "6/17/2022"  "7/30/2022"  "9/3/2018"
##
## $dept
##  [1] "DS"         "DS"         "BA"         "DA"         "DS"         "Actuaries"
##  [7] "Actuaries" "CA"         "DE"         "Lecturer"
```

Kita dapat mengonversi data yang diekstraks di atas ke bentuk data frame R untuk analisis lebih lanjut menggunakan fungsi as.data.frame().

```
json_data_frame <- as.data.frame(json1)             # mengonversi file JSON ke bentuk d
head(json_data_frame,3)                             # Mencetak hasilnya
```

```
##   id    name salary start_date dept
## 1  1  Julian  623.3   1/1/2022   DS
## 2  2 Vanessa  515.2  9/23/2022   DS
## 3  3  Jeffry    611 11/15/2022   BA
```

4.7 Membaca Data dari Web

Banyak situs web yang menyediakan data untuk digunakan oleh penggunanya. Dengan menggunakan program R, kita dapat mengekstrak data tertentu dari situs web tersebut secara terprogram. Di bagian ini, saya memberikan contoh cara mengimpor data dari repositori github, tetapi Anda dapat melakukan hal yang serupa pada situs web atau repositori lainnya.

4.7.1 CSV:

```
web_csv <- read.csv("https://github.com/Bakti-Siregar/dataset/raw/master/Bookdown-Data-
head(web_csv,3)
```

```
##   id    name salary start_date dept
```

```
## 1  1  Julian  623,3   1/1/2022   DS
## 2  2 Vanessa  515,2  9/23/2022   DS
## 3  3  Jeffry    611 11/15/2022   BA
```

4.7.2 XLSX:

```
library(rio)                                          # package ini untuk mengimpor data dari github
install_formats()                                     # mungkin Anda perlu menginstal packages yang
```

```
## [1] TRUE
```

```
web_xlsx <-rio::import("https://github.com/Bakti-Siregar/dataset/blob/master/Bookdown-Data-Scienc
head(web_xlsx,3)
```

```
##   id    name salary start_date dept
## 1  1  Julian    623 2022-01-01   DS
## 2  2 Vanessa    515 2022-09-23   DS
## 3  3  Jeffry    611 2022-11-15   BA
```

4.7.3 LAINNYA:

```
web_txt <- read.table("type URL/Web.txt here")      # membaca/memuat format TXT (notepad) dari web
web_rds <- readRDS("type URL/Web.rds here")         # membaca/memuat format RDS dari web
web_ascii <- readRDS("type URL/Web.ascii here")     # membaca/memuat format ASCII dari web

web_xml<- xmlParse("type URL/Web.xml here")         # membaca/memuat format XML dari web
xmlToDataFrame(web_xml)                             # mengonversi file xml input ke bentuk data fr

web_json <- fromJSON("type URL/Web.json here")      # membaca/memuat format JSON dari web
as.data.frame(web_json)                             # mengonversi file JSON ke bentuk data frame
```

4.8 Sistem Basis Data dengan R

Data tersebut adalah sistem basis data relasional yang disimpan dalam format yang dinormalisasi. Jadi, untuk melakukan komputasi statistik kita akan memerlukan query Sql yang lebih lanjut dan sangat kompleks. Tetapi R dapat terhubung ke banyak basis data relasional dengan mudah seperti MySql, Oracle, server Sql, dll serta mengambil catatan sebagai data frame dari basis data tersebut. Setelah data itu tersedia di lingkungan (environment) R, maka akan menjadi kumpulan data R normal dan dapat dimanipulasi atau dianalisis menggunakan semua package dan fungsi yang kuat.

Catatan: Kita akan mempelajari bagian ini lebih mendalam di “Sistem Basis Data dengan R”.

Chapter 5

Manipulasi Data dengan R

Salah satu keterampilan paling mendasar yang harus dimiliki seorang Data Scientist adalah memanipulasi data. Untuk menjadi seseorang yang sangat efektif, Anda harus ahli dalam memanipulasi data penting. Hal ini perlu diperhatikan karena sebagian besar pekerjaan Anda akan melibatkan pengambilan dan pembersihan data.

Pada bagian ini, Anda akan mempelajari bagaimana memanipulasi data dengan mudah menggunakan R. Kami akan membahas fungsi-fungsi manipulasi data mendasar yang sebagian besar akan Anda gunakan untuk memanipulasi data Anda.

- `read_csv()` Mengimpor data (Anda bisa menggunakan fungsi lainnya)
- `str()` Struktur Data
- `apply()` Untuk mengecek dan mengganti data yang hilang.
- `select()` Memilih kolom yang akan disertakan.
- `filter()` Memilih subset yang ada di dalam data.
- `arrange()` Mengurutkan data, berdasarkan ukuran dari variabel kontinu, berdasarkan tanggal, atau menurut abjad.
- `rename()` Mengganti nama kolom.
- `mutate()` Membuat kolom baru di dalam data, atau mengganti kolom yang sudah ada.
- `bind_rows()` Menggabungkan dua data frame menjadi satu, menggabungkan data dari kolom-kolom dengan nama yang sama.
- `group_by()` Mengelompokkan data berdasarkan variabel kategorikal.
- `summarize()` Meringkas, atau mengagregat (untuk setiap kelompok jika mengikuti `group_by`). Sering digunakan bersama fungsi sebagai berikut:
  - `mean()` Menghitung rata-rata.
  - `median()` Menghitung median.
  - `max()` Mencari nilai maksimum.

  - `min()` Mencari nilai minimum.
  - `sum()` Menambahkan semua nilai secara bersamaan.
  - `n()` Menghitung jumlah record. Saya sarankan Anda untuk menginstal package `tidyverse`. Karena inti dari tidyverse mencakup packages yang cendrung Anda gunakan dalam analisis data sehari-hari.

```
install.packages("tidyverse")
```

Kita sebagian besar akan bekerja dengan dua package yang sangat berguna yang dikembangkan oleh Hardley Wickham, kepala scientist di RStudio:

- `readr` Untuk membaca dan menulis CSV dan file teks lainnya.
- `dplyr` Untuk memproses dan memanipulasi data.

5.1 Impor Data

Data yang akan kita gunakan pada bagian ini adalah `pfizer.csv` dan `fda.csv`, silakan unduh dan tempatkan di desktop Anda. Sebagai opsional, Anda dapat memuat data ke sesi R saat ini dengan memilih `Import Dataset>From Teks File...` di tab Environment. Tapi, dalam hal ini kita akan menggunakan fungsi `read_csv()` dari package `readr`. Salinlah kode berikut ke skrip Anda dan jalankan:

```
suppressPackageStartupMessages(library(tidyverse))# memuat tidyverse
#setwd("C:/Users/Bakti/Desktop/")                 # ingatlah untuk mengatur working di
pfizer <- read_csv("Data/pfizer.csv")             # memuat data `pfizer`
```

```
fda <- read_csv("Data/fda.csv")
```

5.2 Struktur Data

Perhatikan bahwa Anda akan membutuhkan pemahaman yang kuat mengenai tipe-tipe data mendasar dan struktur data dan bagaimana cara mengoprasikannya. Fungsi `str()` akan memberi tahu lebih banyak mengenai kolom dalam data Anda, termasuk tipe datanya. Salinlah kode berikut ke skrip Anda dan jalankan:

```
str(pfizer)                                       # melihat struktur dari data `pfizer
str(fda)                                          # melihat struktur dari data `fda`
```

Sangat penting untuk dipahami karena ini adalah objek yang akan Anda manipulasi di R setiap saat. Jika Anda perlu mengubah tipe data untuk kolom apa pun, gunakan fungsi-fungsi di bawah ini: * `as.character()` mengubah ke teks string. * `as.numeric()` mengubah ke angka. * `as.factor()` mengubah ke variabel kategorikal. * `as.integer()` mengubah ke bilangan bulat. * `as.Date()` mengubah ke tanggal. * `as.POSIXct()` mengubah ke tanggal dan waktu penuh.

Misalnya, tambahkan kode berikut ke skrip Anda untuk mengubah total konversi dalam data pfizer ke variabel numerik (yang memungkinkannya menyimpan nilai desimal, jika ada).

```
pfizer$total <- as.numeric(pfizer$total)          # konversi total ke variabel numerik
str(pfizer$total)                                 # mari periksa struktur datanya lagi
```

5.3 Missing Value

Tidak seperti pemrograman biasa, ketika bekerja dengan data sesungguhnya, Anda mungkin menemukan nilai yang hilang: pengukuran yang tidak terekam/tersimpan/dll. R memiliki mekanisme yang cukup canggih untuk menangani nilai-nilai yang hilang. Ini membedakan antara jenis berikut:

- `NA` : Not Available (Nilai NA juga memiliki kelas, ada bilangan bulat NA, karakter NA, dll).
- `NaN` : Not a Number (Nilai NaN juga merupakan NA tetapi NA bukan merupakan NaN)

Temukan nilai yang hilang di kolom dataframe `pfizer`

```
is.na(pfizer)                                     # cara untuk mengecek NA
sum(is.na(pfizer))                                # menghitung jumlah NA
apply(is.na(pfizer),2, which)                     # indeks NA (hanya df)
which(complete.cases(pfizer))                     # mengidentifikasi nilai lengkap yang diamati
```

Mekanisme yang lebih umum adalah menghapusnya secara manual:

```
clean.vector<- na.omit(pfizer$first_name)         # bersihkan/hapus NA di vektor
clean.df <- na.omit(pfizer)                       # berishkan/hapus NA di dataframe
apply(is.na(clean.df),2, which)                   # pastikan jika ada nilai yang hilang
```

5.4 Mengganti Nilai yang Hilang

Kita juga dapat mengganti nilai yang hilang dengan rata-rata (median). Praktik yang baik adalah dengan membuat dua variabel terpisah untuk mean. Setelah dibuat, kita dapat mengganti nilai yang hilang dengan variabel yang baru dibentuk. Mari unggah dan memeriksa data yang hilang.

```
PATH <- "https://raw.githubusercontent.com/Bakti-Siregar/dataset/master/Bookdown-Data-Science-for
titanic <- read.csv(PATH, sep = ",")
list_na <- colnames(titanic)[ apply(titanic, 2, anyNA) ]
list_na
```

Dalam hal ini, kita tidak menghapus semua nilai yang hilang, tetapi kita menggunakan metode `apply()` untuk menghitung rata-rata kolom dengan `NA`. Pertama, kita perlu menghitung rata-rata dengan argumen `na.rm = TRUE`. Argu-

men ini wajib karena kolom memiliki data yang hilang dan ini memberi tahu R untuk mengabaikannya.

```
average_missing <- apply(titanic[,colnames(titanic) %in% list_na],
                         2,
                         mean,
                         na.rm =  TRUE)
average_missing
```

Catatan: Terdapat 4 argumen di metode apply yang kita jalankan.

- `df` titanic[,colnames(titanic) %in% list_na]. Kode ini akan mengembalikan nama kolom dari objek list_na (mis. "age" dan "fare").
- `2` Menghitung fungsi pada kolom.
- `mean` Menghitung rata-rata.
- `na.rm = TRUE` Menolak nilai yang hilang.

Selanjutnya, kita dapat mengganti nilai-nilai `NA`. Fungsi mutate dari library `dplyr` berguna untuk membuat variabel baru. Kita tidak perlu mengubah kolom asli, jadi kita dapat membuat sebuah variabel baru tanpa `NA`. Fungsi mutate mudah digunakan, kita hanya perlu memilih nama variabel dan menentukan bagaimana cara membuat variabel tersebut. Berikut ini kode lengkapnya:

```
titanic_replace <- titanic %>%
mutate(age  = ifelse(is.na(Age), average_missing[1], Age),
fare = ifelse(is.na(Fare), average_missing[2], Fare))
sum(is.na(titanic_replace$Age))
sum(is.na(titanic_replace$Fare))
```

Kolom usia asli memiliki 86 nilai yang hilang sementara variabel yang baru dibuat telah mengganti nilai yang hilang dengan rata-rata usia variabel. Anda dapat mencoba sendiri mengganti penelitian yang hilang dengan nilai median juga.

5.5 Memilih Data

Pada bagian ini, Anda akan mempelajari bagaimana cara memilih atau mengelompokkan kolom dataframe berdasarkan nama dan posisi menggunakan fungsi R yaitu `select()` dalam package `dplyr`. Anda akan belajar cara menggunakan fungsi berikut ini:

- `pull()` mengekstrak nilai kolom sebagai sebuah vektor. Kolom bunga dapat ditentukan berdasarkan nama atau indeks.
- `select()` mengekstrak satu atau beberapa kolom sebagai tabel data. Fungsi ini juga dapat menghapus kolom dari data frame.
- `select_if()` Memilih kolom berdasarkan kondisi tertentu. Seseorang dapat menggunakan fungsi ini, misalnya untuk memilih kolom jika numerik.

- Fungsi Pembantu: `starts_with()`, `ends_with()`, `contains()`, `matches()`: Pilihlah kolom/variabel berdasarkan namanya.

```
library(tidyverse)                          # muat `tidyverse`, yang termasuk dalam `dplyr`
pfizer %>% pull(state) %>% head()           # ekstrak nilai kolom `state` sebagai vektor
pfizer %>% select(1:3)                      # memilih kolom 1 sampai 3
pfizer %>% select(1,3)                      # memilih kolom 1 dan 3, tidak termasuk 2
pfizer %>% select(state:total)              # memilih semua kolom dari `state` sampai `tota
pfizer %>% select(state,total)              # memilih kolom berdasarkan nama variabel
pfizer %>% select_if(is.numeric)            # hanya memilih kolom numerik
pfizer %>% select_if(is.character)          # hanya memilih kolom karakter
pfizer %>% select(starts_with("first"))     # hanya memilih kolom yang dimulai dengan `firs
pfizer %>% select(ends_with("name"))        # hanya memilih kolom yang diakhiri dengan `nam
pfizer %>% select(contains("rst"))          # memilih kolom yang namanya mengandung `rst`
pfizer %>% select(matches("_"))             # memilih kolom yang namanya cocok dengan regul
pfizer %>% select(-(state:total))           # menghapus semua kolom dari `state` sampai `to
pfizer %>% select(-state, -total)           # menghapus kolom `state` dan `total`
```

5.6 Menyaring dan Mengurutkan Data

Sekarang kita akan `filter()` dan `arrange()` data dengan cara tertentu. Untuk setiap contoh berikut, salin kode berikut ke dalam skrip Anda, dan lihat hasilnya. Perhatikan bagaimana kita membuat objek baru untuk menampung data yang diproses.

Contoh 1: Temukan dokter di California yang dibayar $10.000 atau lebih oleh Pfizer untuk menjalankan "Professional Advising"!

```
ca_expert_10000 <- pfizer %>%               # memuat semua `pfizer`
  filter(state == "CA" &                    # memuat semua `pfizer` disaring berdasarkan `s
         total >= 10000 &                   # dan juga disaring berdasarkan `total` lebih b
         category == "Professional Advising")  # kemudian disaring berdasarkan `category`
ca_expert_10000                             # cetak hasilnya
```

Contoh 2: Sekarang tambahkan daftar urutan secara menurun pembayaran yang diterima oleh dokter di bagian akhir kode!

```
ca_expert_10000 <- pfizer %>%               # memuat semua `pfizer`
  filter(state == "CA" &                    # disaring berdasarkan `state`di California
         total >= 10000 &                   # dan juga disaring berdasarkan `total` lebih b
         category == "Professional Advising")%>%  # kemudian disaring berdasarkan `category`
  arrange(desc(total))                      # mengurutkan secara menurun pembayaran yang di
ca_expert_10000                             # cetak hasilnya

# arrange((total))                          # mengurutkan secara menaik telah ditetapkan se
```

Contoh 3: Temukan dokter di California atau New York yang dibayar $10.000

atau lebih oelh Pfizer untuk menjalankan "Professional Advising"!

Perhatikan bahwa, dalam kasus ini kita menggunakan | operator Boolean, dan tanda kurung di sekitar bagian kueri tersebut. Ini memastikan bahwa bagian kueri ini dijalankan terlebih dahulu. Lihat apa yang terjadi jika Anda mengecualikan mereka.

```
ca_ny_expert_10000 <- pfizer %>%                    # memuat semua `pfizer`
  filter((state == "CA" | state == "NY")  &         # disaring berdasarkan `state` Calif
         total >= 10000 &                           # dan juga disaring berdasarkan `tot
         category == "Professional Advising")%>%    # kemudian disaring berdasarkan `cat
  arrange(desc(total))                              # mengurutkan secara menurun pembaya
ca_ny_expert_10000                                  # cetak hasilnya
```

Contoh 4: Temukan dokter di negara bagian selain California yang dibayar $10.000 atau lebih oleh Pfizer untuk menjalankan "Professional Advising"!

```
not_ca_expert_10000 <- pfizer %>%                   # memuat semua `pfizer`
  filter(state != "CA" &                            # disaring berdasarkan `state` selai
         total >= 10000 &                           # dan juga disaring berdasarkan `tot
         category == "Professional Advising")%>%    # kemudian disaring berdasarkan `cat
  arrange(desc(total))                              # mengurutkan secara menurun pembaya
not_ca_expert_10000                                 # cetak hasilnya
```

Contoh 5: Temukan 20 dokter di empat negara bagian terbesar (CA, TX, FL, NY) yang dibayar paling tinggi untuk "Expert-Led Forums"!

```
ca_ny_tx_fl_prof_top20 <- pfizer %>%
  filter((state == "CA" |
          state == "NY" |
          state == "TX" |
          state == "FL") &
          category == "Expert-Led Forums") %>%
  arrange(desc(total)) %>%
  head(20)
ca_ny_tx_fl_prof_top20
```

Contoh 6: Saring data `pfizer` untuk semua pembayaran untuk menjalankan "Expert-Led Forums" atau untuk "Professional Advising", dan urutkan nama dokter berdasarkan abjad (nama belakang, kemudian nama depan)

```
expert_professional_advice <- pfizer %>%
  filter(category == "Expert-Led Forums" |
         category == "Professional Advising") %>%
  arrange(last_name, first_name)
expert_professional_advice
```

5.7 Mengganti Nama dan Mengubah (Mutate)

Di bagian ini, Anda akan belajar bagaimana mengganti nama kolom dari sebuah data frame di R. Kemudian, Anda akan belajar bagaimana cara menghitung dan menambah variabel baru ke data frame di R. Anda akan belajar fungsi-fungsi R berikut ini dari package R yaitu dplyr:

- `rename()` Kode ini digunakan untuk mengganti nama kolom dari sebuah data frame di R.
- `mutate()` Menghitung dan menambah variabel baru ke dalam sebuah tabel data. Hal ini tidak menghilangkan variabel yang ada.
- `transmute()` Menghitung kolom baru tetapi menghilangkan variabel yang ada.

```
# Mengganti nama kolom dari data `pfizer` dengan fungsi-fungsi dasar R:
names(pfizer)[names(pfizer) == "org_indiv"] <- "rename1"
names(pfizer)[1] <- "rename2"
names(pfizer)[names(pfizer)==names(pfizer)] <- c("rename3",
                                                 "rename4",
                                                 "first_name",
                                                 "last_name",
                                                 "city",
                                                 "state",
                                                 "category",
                                                 "cash",
                                                 "other",
                                                 "total")
# Mengganti nama kolom dari data `pfizer` dengan `dplyr::rename()`:
pfizer %>%
  rename(
    org_indiv = rename3 ,
    first_plus = rename4
    )
# Menambahkan kolom baru (year*) dengan mempertahankan data `fda` yang ada:
letters_year <- fda %>%
  mutate(year = format(issued, "%Y")) %>%
  group_by(year)
letters_year
# Menambahkan kolom baru (year*) dan (last_name*) dengan menghilangkan data `fda` yang ada:
fda %>%
  transmute(
    year = format(issued, "%Y"),
    last_name = name_last
    )
```

5.8 Menggabungkan Data

Ada juga beberapa fungsi gabungan di dplyr untuk menggabungkan data dari dua data frame. Berikut ini adalah fungsi yang paling berguna:

- `inner_join()` Mengembalikan nilai dari kedua tabel hanya jika ada kecocokan.
- `left_join()` Mengembalikan semua nilai dari tabel yang pertama disebutkan, ditambah nilai dari tabel kedua yang cocok.
- `semi_join()` Menyaring tabel yang pertama disebutkan untuk mendapatkan nilai yang memiliki kecocokan dengan tabel kedua.
- `anti_join()` Menyaring tabel yang pertama disebutkan untuk mendapatkan nilai yang tidak memiliki kecocokan dengan table kedua.

Sebagai ilustrasinya, gabungan ini akan menemukan dokter yang dibayar oleh Pfizer untuk menjalankan Expertp-Led Forums yang juga menerima surat peringatan dari `fda`:

```
expert_warned_inner <- inner_join(pfizer, fda,
                                  by=c("first_name" = "name_first",
                                       "last_name" = "name_last")) %>%
                                  filter(category=="Expert-Led Forums")

expert_warned_semi <- semi_join(pfizer, fda,
                                by=c("first_name" = "name_first",
                                     "last_name" = "name_last")) %>%
                                filter(category=="Expert-Led Forums")
```

Kode dalam `by=c()` menentukan bagaimana gabungan harus dibuat. Jika instruksi tentang bagaimana cara menggabungkan tabel tidak disediakan, `dplyr` akan mencari kolom dengan nama yang cocok, dan melakukan penggabungan berdasarkan hal tersebut. Perbedaan antara dua gabungan di atas adalah yang pertama berisi semua kolom dari kedua data frame, sedangkan yang kedua hanya memberikan kolom dari data frame pfizer.

Dalam praktiknya, mungkin Anda ingin dan kemudian menggunakan fungsi select dari `dplyr` untuk memilih kolom yang ingin Anda pertahankan, misalnya:

```
expert_warned <- inner_join(pfizer, fda,
                            by=c("first_name" = "name_first",
                                 "last_name" = "name_last")) %>%
                            filter(category=="Expert-Led Forums") %>%
                            select(last_name,
                                   city,
                                   state,
                                   total,
                                   issued)
```

```
expert_warned <- inner_join(pfizer, fda,
                            by=c("first_name" = "name_first",
                                 "last_name" = "name_last")) %>%
                            filter(category=="Expert-Led Forums") %>%
                            select(2:5,10,12)
```

5.9 Kelompok dan Ringkasan

Bagian ini memperkenalkan cara menghitung ringkasan statistik dengan mudah di R menggunakan package `dplyr`. Anda akan belajar, bagaimana:

- Menghitung ringkasan statistik untuk data yang tidak dikelompokkan, serta untuk data yang dikelompokan menurut satu atau beberapa variabel. Fungsi R: `summarise()` dan `group_by()`.
- Meringkas beberapa kolom variabel. Fungsi R:
- `summarise_all()` Menerapkan fungsi ringkasan ke setiap kolom dalam data frame.
- `summarise_at()` Menerapkan fungsi ringkasan ke kolom tertentu yang dipilih dengan vektor karakter.
- `summarise_if()` Menerapkan fungsi ringkasan ke kolom yang dipilih dengan fungsi yang menampilkan TRUE

Contoh 7: Hitunglah total pembayaran data `pfizer`, dengan urutan negara bagian secara menurun!

```
state_sum <- pfizer %>%
  group_by(state) %>%
  summarize(sum = sum(total)) %>%
  arrange(desc(sum))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
state_sum
```

Contoh 8: Hitunglah beberapa ringkasan statistik tambahan dari data `pfizer`, dengan urutan negara bagian secara menurun!

```
state_summary <- pfizer %>%
  group_by(state) %>%
  summarize(sum = sum(total),
        average = mean(total),
         median = median(total),
            min = min(total),
            max = max(total),
            count = n()) %>%
  arrange(desc(sum))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
state_summary
```

Contoh 9: Kelompokkan dan rangkum data `pfizer` untuk beberapa kategori naik!

```
state_summary <- pfizer %>%
  group_by(state, category) %>%
  summarize(sum = sum(total),
        average = mean(total),
         median = median(total),
            min = min(total),
            max = max(total),
            count = n()) %>%
  arrange(state, category)
```

```
## `summarise()` regrouping output by 'state' (override with `.groups` argument)
```

```
state_summary
```

Contoh 10: Saring data `fda` untuk surat-surat yang telah dikirim dari awal tahun 2006 dan seterusnya dan rangkum!

```
year_summary <- fda %>%
  filter(issued >= "2005-01-01") %>%
  arrange(issued) %>%
  mutate(year = format(issued, "%Y")) %>%
  group_by(year) %>%
  summarize(letters=n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
year_summary
```

Catatan: Menggunakan kembali variabel dapat menyebabkan hasil yang tidak dapat diharapkan, tetapi jangan khawatir. R akan memberikan peringatan seperti `summarise() ungrouping output (override with .groups argument)`.

5.10 Membagi Data

Seperti yang sudah saya sampaikan pada bab Pemrograman R tentang menulis fungsi untuk membagi data latihan dan data percobaan. Di sini, kita akan mempelajari lebih lanjut tentang bagaimana menggunakan beberapa packages untuk membagi data. Karena bagian ini sangat penting sebagai seorang data scientist khususnya ketika Anda menerapkan tugas Machine Learning untuk menganalisis data. Pada bagian ini biasanya kita perlu membagi dataset antara set latihan dan set percobaan. Set latihan memungkinkan algoritme untuk belajar dari data. untuk menguji kinerja model kita, kita dapat menggunakan

set percobaan untuk mengembalikan ukuran kinerja. Jadi, mari kita lihat beberapa packages yang dapat Anda gunakan untuk membagi data:

5.10.1 dplyr

Anda dapat menggunakan `dplyr` untuk ini, membuat jadi sangat sederhana. itu memang memerlukan variabel id dalam dataset Anda, yang mana ini merupakan ide yang bagus, tidak hanya untuk membuat set-set tetapi juga untuk keterlacakan selama proyek Anda. Tambahkan jika belum ada.

```
library(dplyr)                                  # memuat package `dplyr`
data(mtcars)                                    # menggunakan data dari environment R
set.seed(123)                                   # untuk memastikan kita menghasilkan data yang
mtcars$id <- 1:nrow(mtcars)                     # menambahkan `id` jika belum ada
train<-mtcars %>% dplyr::sample_frac(.75)       # beri nama untuk set latihan
test<-dplyr::anti_join(mtcars, train, by = 'id')  # beri nama untuk set percobaan
dim(train)                                      # mengecek dimensi set latihan
dim(test)                                       # mengecek dimensi set percobaan
```

Catatan: Dataset ini hanya sebagai contoh, ini tidak cocok untuk tugas machine learning karena datanya terlalu sedikit.

5.10.2 caTools

Ada banyak pendekatan untuk mencapai partisi data. Untuk pendekatan yang lebih lengkap, lihat fungsi `createDataPartition` dalam package `caTools`.

```
library(caTools)                                # memuat package `caTools`
data(mtcars)                                    # menggunakan data dari environment R
set.seed(123)                                   # untuk memastikan kita mendapatkan data yang s
smp_size <- floor(0.75 * nrow(mtcars))          # 75% dari ukuran sampel
train_ind <- sample(seq_len(nrow(mtcars)),
                    size = smp_size)
train <- mtcars[train_ind, ]                    # memberi nama untuk set latihan
test <- mtcars[-train_ind, ]                    # memberi nama untuk set percobaan
dim(train)                                      # mengecek dimensi set latihan
dim(test)                                       # mengecek dimensi set percobaan
```

5.10.3 caret

Packages hebat lainnya yang dapat Anda gunakan untuk membagi dataset adalah `caret`.

```
library(caret)                                  # memuat package `caret`

data(mtcars)                                    # menggunakan data dari environment R
set.seed(123)                                   # untuk memastikan kita mendapatkan data yang s
```

```
intrain<-createDataPartition(mtcars$mpg,
                             p=0.75,list=FALSE)
train<-mtcars[intrain,]
test<-mtcars[-intrain,]
dim(train)                                    # mengecek dimensi set latihan
dim(test)                                     # mengecek dimensi set percobaan
```

Chapter 6

Eksplorasi Data Analisis

Analisis Data Eksplorasi (ADE) adalah pendekatan analisis data untuk meringkas dan memvisualisasikan karakteristik penting dari data. ADE dapat dianggap sebagai asumsi bebas, biasanya dilakukan dalam perilaku analisis data. ADE juga dikenal sebagai analitik visual atau statistik deskriptif. ADE juga merupakan praktik pengamatan, dan mengeksplorasi data, sebelum Anda menekankan beberapa hipotesis, menyesuaikan prediktor, dan tujuan lainnya pada statistik inferensial. ADE biasanya mencakup penghitungan ringkasan statistika sederhana yang menangkap beberapa properti yang diminati dalam data dan visualisasi.

Seseorang yang bekerja dengan data, salah satu aktivitasnya yang paling menarik adalah mengeksplorasi dataset yang baru. Anda harus memehami variabel apa saja yang Anda miliki, berapa banyak records yang ada di dalam dataset, berapa banyak data yang hilang, apa struktur variabelnya, apa hubungan variabelnya, dan masih banyak lagi. Oleh karena itu, pada bagian ini akan kita akan menampilkan beberapa poin penting dalam ADE dasar sebagai berikut:

- Bagaimana mengeksplorasi: dengan statistik ringkasan, atau secara visual?
- Berapa banyak variabel yang dianalisis secara bersamaan: univariat, bivariat, atau multivariat?
- Apa tipe dari variabel: kategorikal atau kontinu?

Pada bagian ini, kita menggunakan dataset Students, download dan simpan di komputer (PC) Anda .

6.1 Kualitatif

Variabel kualitatif (kategorikal), yang disebut data `faktor` atau `karakter` di R, menghadirkan tantangan-tantangan unik dalam perselisihan data (data wrangling). Perhatikan bahwa: data kategorikal tidak cocok untuk operasi matematika apapun, jadi jangan menerimanya. Kita tidak dapat menggunakan sum, atau bahkan sort, kita hanya dapat menghitungnya. Dengan demikian, ringkasan dari variabel-variabel kategorikal akan selalu dimulai dengan menghitung frekuensi setiap kategori.

Variabel kategorikal dapat dibagi menjadi:

- **Kategorikal Nominal:** data yang levelnya adalah label atau deskripsi, dan yang tidak dapat diurutkan. data yang levelnya adalah label atau deskripsi, dan yang tidak dapat dipesan. Contoh variabel nominal adalah jenis kelamin, sekolah, dan pertanyaan dengan jawaban ya/tidak. Biasanya juga disebut variabel "kategorikal nominal" atau "kualitatif", dan tingkat variabel kadang-kadang disebut "kelas" atau "kelompok".
- **Kategorikal Ordinal:** dapat disusun atau diberi peringkat dalam urutan yang logis, tetapi interval antara level variabel belum tentu diketahui. Pengukuran subyektif seringkali merupakan variabel ordinal. Salah satu contoh adalah dengan meminta orang memberikan empat urutan peringkat ke suatu item berdasarkan prioritas. Contoh lainnya adalah meminta orang untuk menilai beberapa item berdasarkan skala peringkat Likert:"Pada skala satu sampai lima, apakah Anda setuju atau tidak setuju dengan pernyataan ini?". Contoh ketiga adalah tingkat pendidikan untuk orang dewasa, misalnya dengan mempertimbangkan "dibawah sekolah menengah", "sekolah menengah", "gelar asosiasi/sarjana", dll.

6.1.1 Kategorikal Univariat

Analisis univariat mengeksplorasi variabel (atribut) satu per satu. Variabel dapat berupa kategorikal atau numerik. Ada perbedaan teknik statistik dan visualisasi dari penyelidikan untuk setiap jenis variabel. Variabel numerik dapat diubah menjadi bagian kategorikal dengan proses yang disebut binning atau diskritisasi. Dimungkinkan juga untuk mengubah variabel kategorikal menjadi bagian numerik dengan proses yang disebut pengkodean. Terakhir, penanganan yang tepat atas nilai yang hilang merupakan masalah penting dalam penambahan data. Sekarang, mari kita lihat cara mengeksplor data kategorikal univariat:

```
library(readr)                                  # antarmuka dataset
#setwd("C:/Users/Bakti/Desktop/")               # mengatur working directory Anda
df= read_csv("Data/students.csv")               # mengimpor dataset
#str(df)                                        # memeriksa struktur datanya jika d
apply(is.na(df),2, which)                       # memeriksa NA dalam data frame
df<-na.omit(df)                                 # menghapus data yang hilang
head(df,3)                                      # untuk melihat 3 baris pertama dat
```

```
Cat1 <- table(df$Gender)                          # menghitung frekuensinya
Cat1
```

Jika seandainya Anda menginginkan proporsi dari jumlah level, Anda dapat menggunakan `prop.table`.

```
prop.table(table(df$Gender))
```

6.1.2 Kategorikal Bivariat

Contoh data bivariat dapat diberikan untuk membantu Anda memahami hubungan antara dua variabel dan untuk memahami ide di balik definisi dan makna analisis data bivariat. Perlu diperhatikan bahwa, dalam kasus ini, kita fokus pada variabel kategorikal.

```
library(readr)                                    # antarmuka dataset
library(dplyr)                                    # untuk memanipulasi data
library(magrittr)                                 # untuk memanipulasi data mirip dengan dplyr
Cat2<- df %>%                                     # memuat data
select(Gender, Horoscope) %>%                     # memilih vektor ke dalam matriks dan periksa
table()                                           # menghitung frekuensi kombinasi bivariat
#prop.table()                                     # menggunakan tabel proporsi jika diinginkan
Cat2                                              # mencetak hasilnya
```

6.1.3 Kategorikal Multivariat

Anda mungkin bertanya-tanya bagaimana R menangani tabel dengan lebih dari dua dimensi. Memang tidak mudah untuk menyampaikan hal ini dengan cara yang bisa dibaca oleh manusia. R menawarkan beberapa solusi: `tabel` agar lebih mudah dihitung, dan `ftable` dapat dibaca manusia.

```
Cat3 <- df %>%                                    # memuat data
select(Gender, Horoscope, Subject) %>%            # memilih vector ke dalam matriks dan periksal
#table()                                          # tabel yang dapat dibaca mesin
#prop.table()                                     # menggunakan tabel proporsi jika diinginkan
ftable()                                          # tabel yang bisa dibaca manusia
Cat3                                              # mencetak hasilnya
```

6.2 Kuantitatif

Variabel kuantitatif (Kontinu) adalah nilai-nilai dasar dalam R. Variabel-variabel tersebut disimpan sebagai numerik atau integer. Variabel kontinu menerima lebih banyak perlakuan daripada kategorikal. Kita dapat menghitung jumlah, rata-rata, median, quantiles, dan ringkasan statistik lainnya.

6.2.1 Univariat Kontinu

Ukuran Tendensi Sentral

Ukuran tendensi sentral adalah nilai tunggal yang mencoba menggambarkan sekumpulan data dengan mengidentifikasi posisi sentral dalam kumpulan data itu. Dengan demikian, ukuran tendensi sentral terkadang disebut pengukuran lokasi sentral. Ukuran tendensi sentral juga digolongkan sebagai ringkasan statistik. Mean (rata-rata) kemungkinan besar adalah ukuran tendensi sentral yang paling Anda ketahui, tetapi ada juga yang lainnya, seperti kuantil, median, mode, dan Alpha Trimmed Mean.

- **Mean(Rata-rata):** adalah ukuran tendensi sentral yang paling populer dan terkenal. mean dapat digunakan dengan data diskrit maupun kontinu, meskipun penggunaannya paling sering dengan data kontinu (lihat panduan Jenis Variabel kami untuk tipe data). Rata-rata sama dengan jumlah semua nilai dalam data set dibagi dengan seberapa banyak jumlah unit nilai dalam data set itu sendiri. Jadi, jika kita memiliki nilai di sebuah data set dan nilainya ..., mean sampel, biasanya dinotasikan dengan (dilafalkan "x bar"), adalah:

$$\bar{x} = \frac{(x_1 + x_2 + ... + x_n)}{n}, \text{atau}$$
$$\bar{x} = \frac{\Sigma x}{n}, \text{Sample Mean}$$
$$\mu = \frac{\Sigma x}{N}, \text{Population Mean}$$

- **Kuantil:** kuantil α dari sampel x, dilambangkan dengan x_α, didefinisikan (tidak unik) sebagai nilai di atas $100_\alpha\%$ dari sampel, dan di bawah $100(1-\alpha)\%$. Kami menekankan bahwa kuantil sampel tidak ditentukan secara unik. Perhatikan `?quantile` untuk definisi berbeda `9(!)` yang diberikan oleh R.

- **Median:** Median dari sampel x, dilambangkan dengan $x_{0.5}$ dimana $\alpha = 0.5$ kuantil sampel.

- **Mode:** Mode adalah nilai frekuensi yang paling sering di dataset kami. Pada histogram, mode mewakilkan batang tertinggi dalam diagram batang atau histogram. Oleh karena itu, Anda terkadang dapat menganggap mode sebagai opsi paling populer. Biasanya, mode digunakan untuk data kategorikal dimana kita ingin mengetahui kategori mana yang paling umum.

- **Alpha Trimmed Mean:** Pemangkasan rata-rata α dari sampel x, dilambangkan $\bar{x}_\alpha$ adalah rata rata dari sampel setelah menghilangkan pengamatan proporsi terbesar α dan pengamatan proporsi terkecil α. Rata-rata dan median seerhana adalah contoh dari pemangkasan rata-rata α: masing-masing $\bar{x}_0$ dan $\bar{x}_{0.5}$.

Berikut ini adalah implementasi R:

```
Quan <- df %>%
select_if(is.numeric)                          # hanya memilih kolom numerik
names(Quan)                                    # memeriksa nama-nama variabel Kuantitatif
mean(Quan$Grade)                               # rata-rata dari `Grade`
quantile(Quan$Grade)                           # kuantil dari `Grade`
median(Quan$Grade)                             # median dari `Grade`
mode(Quan$Grade)                               # Mode dari `Grade`
summary(Quan)                                  # ringkasan statistik sederhana dalam satu fu
```

Skala

Skala/Variabilitas data, terkadang dikenal sebagai penyebaran, dapat dianggap sebagai variabilitas sebagai berikut:

- **Varians:** Varians adalah ukuran numerik tentang bagaimana nilai data tersebar di sekitar mean. Secara khusus, varians sampel didefinisikan sebagai s^2. Demikian pula, varians populasi σ^2 didefinisikan dalam hal rata-rata populasi μ dan ukuran populasi N:

  $s^2 = \frac{1}{n-1}\Sigma_{i=1}^{n}(x_i - \bar{x})^2$

  $\sigma^2 = \frac{1}{N-1}\Sigma_{i=1}^{n}(x_i - \mu)^2$

- **Deviasi Standar:** Deviasi standar dari sebuah variabel penelitian adalah akar kuadrat dari variansinya, dilambangkan dengan σ dan didefinisikan sebagai:

  $\sigma = \sqrt{\frac{1}{N-1}\Sigma_{i=1}^{n}(x_i - \mu)^2}$

- **MAD:** Median Absolute Deviation (Deviasi Absolut Median) dari median, dilambangkan dengan `MAD(x)`, didefinisikan sebagai: $MAD(x) := c|x - x_{0.5}|_{0.5}$ di mana c adalah konstanta, biasanya ditentukan $c = 1.4826$ sehingga MAD dan σ memiliki batas sampel yang sama besar.

- **IQR:** Inter Quantile Range (Antar Rentang Kuantil) dari sampel x, dilambangkan dengan $IQR(x)$, didefinisikan sebagai $IQR(x) := x_{0.75} - x_{0.25}$.

Berikut ini adalah implementasi R:

```
var(Quan$Grade)                                # Varian dari `Grade`
sd(Quan$Grade)                                 # Deviasi standar dari `Grade`
mad(Quan$Grade)                                # Deviasi Absolut Median dari `Grade`
IQR(Quan$Grade)                                # Antar Rentang Kuantile dari `Grade`
```

Kecondongan (Skewness)

Kecondongan atau Asimetri mengacu pada distorsi atau asimetri dalam kurva lonceng simetris, atau distribusi normal, dalam dataset. Jika kurva bergeser ke

kiri atau ke kanan, maka dikatakan condong. Kecondongan dapat dikuantifikasi sebagai representasi dari sejauh mana suatu distribusi bervariasi dari distribusi normal. Distribusi normal memiliki kemiringan nol, sedangkan distribusi normal log, sebagai contoh, akan menunjukkan beberapa derajat kemiringan ke kanan.

```
library(e1071)                                    # memuat e1071
skewness(Quan$Grade)                              # menerapkan fungsi kecondongan `sk
```

Kecondongan variabel `Quan$Grade` adalah 0.2625807. Itu menunjukkan bahwa distribusi variabel `Quan$Grade` adalah condong ke arah kanan.

Kurtosis

Kurtosis dari distribusi normal univariat adalah 3. Membandingkan distribusi kurtosis dengan nilai ini (3) adalah umum. Distribusi dengan kurtosis kurang dari 3 disebut platykurtic, meskipun ini tidak berarti distribusinya "flat-topped" seperti yang kadang-kadang dinyatakan. Sebaliknya, ini berarti distribusi menghassilkan outlier yang lebih sedikit dan tidak terlalu ekstrim daripada distribusi normal.

```
kurtosis(Quan$Grade)                              # menerapkan fungsi `kurtosis`
```

Kelebihan kurtosis dari variabel `Quan$Grade` adalah -1.048464, yang menandakan bahwa distribusi variabel `Quan$Grade` adalah platykurtic. Hal ini sesuai dengan fakta bahwa histogramnya tidak berbentuk lonceng.

6.2.2 Bivariat Kontinu

Saat berhadapan dengan variabel kontinu bivariat, atau multivariat, kita tentu dapat menghitung ringkasan univariat untuk setiap variabel secara terpisah. Ini bukan topik dari bagian ini, di mana kami ingin meringkas hubungan antara variabel, dan bukan di dalamnya.

Kovariansi

Kovariansi antara dua contoh, x dan y, dengan panjang yang sama n, didefinisikan sebagai

$Cov(x, y) := \frac{1}{(n-1)}\Sigma(x_i - \bar{x})(y_i - \bar{y})$ Kami menekankan ini bukanlah kovariansi yang Anda sudah pelajari di kelas probabilitas, karena ini bukan kovariansi antara dua variabel acak, melainkan antara dua sampel. Untuk alasan ini, beberapa penulis menyebutnya kovarians empiris, atau kovarians sampel.

Koefisien Korelasi Pearson

Koefisien korelasi Pearson, yang dikenal sebagai korelasi produk momen Pearson, atau sederhananya, korelasi, dilambangkan dengan $\rho(x, y)$, didefinisikan sebagai

$\rho(x,y) := \frac{Cov(x,y)}{\sigma_x \sigma_y}$ Jika Anda menganggap definisi ini membingungkan, anggap saja korelasi sebagai kovariansi antara x dan y setelah mengubahnya menjadi skala `skor-z` tanpa unit.

Skor-Z

Skor-z dari sampel x didefinisikan sebagai observasi yang dinormalisasi dengan skala tengah-tengah:

$z_i(x) := \frac{x_i - \bar{x}}{\sigma_x}$ Jadi kita punya, $\rho(x,y) := Cov(z(x), z(y))$

```
cov(Quan$Grade,Quan$Sleep)                    # menerapkan fungsi `cov()`
```

```
cor(Quan$Grade,Quan$Sleep)                    # menerapkan fungsi `corr()`
```

```
zscore=(Quan$Grade-mean(Quan$Grade))/sd(Quan$Grade) # manual skor-z
```

6.2.3 Multivariat Kontinu

Kovariansi adalah ringkasan sederhana dari hubungan antara dua variabel, tetapi tentu saja ini mungkin tidak mencakup keseluruhan "cerita" saat menangani lebih dari dua variabel. Ringkasan paling umum dari relasi multivariat adalah matriks kovarians, tetapi kami memperingatkan bahwa hanya relasi multivariat paling sederhana yang diringkas sepenuhnya oleh matriks ini.

Matriks Sampel Kovarians

Diberikan pengamatan n pada variabel p, dilambangkan dengan $x_{i,j}$ pengamatan ke-i dari variabel ke-j. Matriks sampel kovarians, dilambangkan dengan $\hat{\Sigma}$ didefinisikan sebagai

$\hat{\Sigma}_{k,l} := \frac{1}{(n-1)} \Sigma [(x_{i,k} - \bar{x}_k)(y_{i,l} - \bar{y}_l)]$ di mana $\bar{x}_k := \frac{1}{n} \Sigma_i x_{i,k}$. Dengan kata lain, entri ke-k,l dalam $\hat{\Sigma}$ adalah kovarians sampel antara variabel k dan l.

Matriks Sampel Korelasi

Matriks korelasi mengacu pada deretan angka simetris. Matriks korelasi: R

$R = []$ ———-(Saya belum tau cara membuat matriksnya, Pak)———-

$Y_{j,k} = \frac{S_{j,k}}{S_j S_k} = \frac{\Sigma_{i=1}^{n}(x_{i,j} - \bar{x}_j)(x_{i,k} - \bar{x}_k)}{\sqrt{\Sigma_{i=1}^{n}(x_{i,j} - \bar{x}_j)^2} \sqrt{\Sigma_{i=1}^{n}(x_{i,k} - \bar{x}_k)^2}}$ $Y_{j,k}$ = koefisien korelasi x_j dan x_k

Berikut ini adalah implementasi R:

```
cov(Quan)                                     # menerapkan fungsi `cov()`
```

```
cor(Quan)                                     # menerapkan fungsi `corr()`
```

6.3 ADE dengan cara Lazy-way

Jalankan semua fungsi di posting ini satu kali dengan fungsi berikut:

```
library(funModeling)
library(tidyverse)
library(Hmisc)
library(skimr)
basic_eda <- function(data)
{
  glimpse(data)
  skim(data)
  df_status(data)
  freq(data)
  profiling_num(data)
  plot_num(data)
  describe(data)
}
basic_eda(df)
```

6.3.1 Fungsi Glimpse

Fungsi sekilas dari package `dplyr`. Ini akan menampilkan pratinjau vertikal dari dataset. Ini memungkinkan kita untuk melihat jenis data dan sampel data dengan mudah.

6.3.2 Fungsi Skim

Fungsi `skim` dari package `skimr`. Fungsi skim adalah tambahan yang baik untuk fungsi ringkasan. Ini menampilkan sebagian besar atribut numerik dari ringkasan, tetapi juga menampilkan data yang hilang, lebih banyak informasi kuantil, dan histogram sebaris untuk setiap variabel!

6.3.3 Fungsi Lain

Gunakan kode berikut untuk mengetahui lebih lanjut tentang fungsi:

```
?df_status
?freq
?profiling_num
?plot_num
?describe
```

6.4 Laporan ADE

Dan akhirnya pièce de résistance, daya tarik utama dan alasan saya menulis blog ini; fungsi create_report dalam package DataExplorer. Fungsi satu baris

yang menarik mengagumkan ini akan menarik profil data lengkap dari data frame Anda. Ini akan menghasilkan file HTML dengan statistik dasar, struktur, data yang hilang, visualisasi distribusi, matriks korelasi, dan analisis komponen utama dari data frame Anda! Saya baru-baru ini mengetahui tentang fungsi ini dalam sebuah lokakarya yang diberikan oleh Stephe Locke yang dipandu oleh R Ladies Austin. Fungsi ini berdampak besar!

```
library(DataExplorer)
DataExplorer::create_report(df)
```

Chapter 7

Visualisasi Data

Disini di tuliskan materi

Chapter 8

Vusualisasi Data Lanjut

Disini di tuliskan materi