Analisis Data Eksploratif Dataset Gapminder

Moh. Rosidi

7/8/2020

Contents

Analisis Data Eksploratif

Analisis data eksploratif (*exploratory data analysis* - EDA) merupakan metode eksplorasi data dengan menggunakan teknik aritmatika sederhana dan teknik grafis dalam meringkas data pengamatan.

EDA bukanlah proses formal dengan seperangkat aturan yang ketat. Lebih dari segalanya, EDA adalah *state of mind*. Selama fase awal EDA Anda harus merasa bebas untuk menyelidiki setiap ide yang ada dalam pikiran kita. Beberapa pertanyaan kita selama proses ini dapat membuahkan hasil (menghasilkan *insight*) dan bahkan ada yang gagal atau tidak terjawab. Saat penjelajahan kita berlanjut, kita akan menemukan beberapa area khusus yang pada akhirnya akan kita laporkan dan komunikasikan dengan orang lain.

EDA adalah bagian penting dari setiap analisis data, bahkan jika pertanyaan diserahkan kepada kita di atas kertas, karena kita selalu perlu menyelidiki kualitas data kita Pembersihan data hanyalah salah satu aplikasi EDA: kita mengajukan pertanyaan tentang apakah data kita memenuhi harapan kita atau tidak.

Terdapat 2 pertanyaan utama yang perlu dijawab dalam proses EDA, yaitu:

1. Jenis variasi apa yang terjadi dalam variabel saya?, dan
2. Jenis variasi apa yang terjadi dalam variabel saya?.

Selain 2 pertanyaan tersebut, kita dapat pula menambahkan pertanyaan lain], seperti:

1. Apakah terdapat kolom data yang tidak sesuai?,
2. Apakah terdapat data hilang (*missing value*) pada data kita?,
3. Apakah pada data terdapat observasi yang tidak biasa (*outlier*)?, dll.

Dengan melakukan proses EDA ini diharapkan kita memperoleh gambaran data dan model yang sesuai untuk analisis data tersebut.

Dataset Gapminder

Pada artikel ini, kita akan melakukan analisis eksploratif pada dataset `Gapminder`. `Gapminder` merupakan kutipan dataset tentang usia harapan hidup, PDB per kapita, dan populasi menurut negara dan benua.

Kolom-kolom pada dataset tersebut, antara lain:

- `country` : nama negara
- `continent` : nama benua
- `year` : tahun dengan rentang 1952 sampai dengan 2007 dengan retang pengukuran tiap 5 tahun
- `lifeExp` : angka harapan hidup dalam satuan `tahun`
- `pop` : populasi
- `gdpPercap` : pendapatan domestik bruto per kapita dalam satuan `US$`

Persiapan

Library

```r
if(!require(tidyverse)) install.packages("tidyverse")
if(!require(skimr)) install.packages("skimr")
if(!require(DataExplorer)) install.packages("DataExplorer")

library(tidyverse)
library(skimr)
library(DataExplorer)
```

Terdapat tiga buah *library* yang diperlukan dalam tutorial ini, antara lain:

1. `tidyverse` : koleksi paket R yang dirancang untuk ilmu data. Semua paket berbagi filosofi desain, tata bahasa, dan struktur data yang mendasarinya.
2. `skimr` : menyediakan fungsi untuk membuat ringkasan data yang dapat dibaca secara cepat.
3. `DataExplorer` : menyediakan fungsi yang dapat membantu proses otomasi analisis data eksploratif

Import Dataset

Data yang kita miliki memiliki format `.csv`. Untuk megimport data tersebut, kita dapat menggunakan fungsi `read_csv` dari library `readr`.

```r
gapminder <- read_csv("data/gapminder.csv")
```

Untuk mengecek 10 observasi awal dataset tersebut, jalankan sintaks berikut:

```r
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <chr>       <chr>     <dbl>   <dbl>    <dbl>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

Dasar-Dasar Tidyverse

`Tidyverse` merupakan kumpulan paket yang dikhususkan bagi pengguna `R` yang ingin melakukan analisa data atau aktivitas *data science*. Paket dari `tidyverse` antara lain:

1. **`ggplot2`**: paket yang digunakan untuk membuat visualisasi data yang menarik yang didasarkan pada sistem *Grammar of Graphics*.
2. **`dplyr`**: berisi kumpulan fungsi yang digunakan untuk melakukan manipulasi pada data dengan nama fungsi dan output yang konsisten.

3. **tidyr**: paket yang berisi kumpulan fungsi merapikan data atau membuat *pivot table* dari data.
4. **readr**: paket yang berfungsi untuk membaca file format .csv, .txt, .tsv, dan .fwf.
5. **purrr**: paket yang berguna untuk meningkatkan *fuctional programming* pada R. Fungsi ini telah penulis bahas secara garis besar pada Chapter 1.
6. **tibble**: paket yang digunakan untuk mengubah dataframe menjadi format tibble (bentuk lain dataframe yang lebih konsisten).

Selain paket-paket tersebut, masih terdapat banyak paket lain yang ada seperti **stringr**, **forcats**, dll. Untuk mempelajari *data science* menggunakan paket *tidyverse*, pembaca dapat pergi ke tautan *e-book* R for Data Science.

Seluruh fungsi dalam paket **tidyverse** dapat dikombinasikan dengan penggunaan operator pipa (**%>%**). Operator pipa (**%>%**) sangat berguna untuk merangkai bersama beberapa fungsi **dplyr** dalam suatu urutan operasi. Perhatikan contoh sebelumnya dimana setiap kali kita ingin menerapkan lebih dari satu fungsi, urutannya akan dimulai dalam urutan panggilan fungsi bersarang yang sulit dibaca. Secara ringkas dapat kita tulis sebagai berikut:

```
# cara 1
one <- first(x)
two <- second(one)
three <- third(two)

# cara 2
third(second(first(x)))
```

Jika dituliskan menggunakan operator pipa akan menghasilkan sintak berikut:

```
# cara 3
x %>%
  first() %>%
  second() %>%
  third()
```

Readr

Readr merupakan salah satu paket untuk import data pada paket **tidyverse**. **Readr** secara spesifik digunakan untuk mengimport data yang ada pada file dengan ekstensi **.txt** dan **.csv**. Terdapat beberapa fungsi yang ada pada paket ini, antara lain:

- **read.csv()**: untuk membaca file dengan format *comma separated value*(“**.csv**”).
- **read.csv2()**: varian yang digunakan jika pada file “**.csv**” yang akan dibaca mengandung koma (“,”) sebagai desimal dan semicolon (“;”) sebagai pemisah antar variabel atau kolom.
- **read.delim()**: untuk membaca file dengan format *tab-separated value*(“.txt”).
- **read.delim2()**: membaca file dengan format “.txt” dengan tanda koma (“,”) sebagai penujuk bilangan desimal.

Masing-masing fungsi diatas dapat dituliskan kedalam **R** dengan format sebagai berikut:

```
# Membaca tabular data pada  R
read.table("<LOKASI FILE>", header = FALSE, sep = "", dec = ".")
# Membaca"comma separated value" files (".csv")
read.csv("<LOKASI FILE>", header = TRUE, sep = ",", dec = ".", ...)
```

```r
# atau gunakan read.csv2 jika tanda desimal
# pada data adalah "," dan pemisah kolom adalah ";"
read.csv2("<LOKASI FILE>", header = TRUE, sep = ";", dec = ",", ...)
# MembacaTAB delimited files
read.delim("<LOKASI FILE>", header = TRUE, sep = "\t", dec = ".", ...)
read.delim2("<LOKASI FILE>", header = TRUE, sep = "\t", dec = ",", ...)
```

Implementasi dari paket ini telah dijelaskan pada proses import dataset yang telah dilakukan.

Tidyr

`Tidyr` merupakan sebuah paket yang digunakan untuk mentransformasi dataset dari bentuk *untidy* menjadi *tidy*. Secara umum *tidy* data merupakan data-data yang memegang prinsip-prinsip berikut:

1. Setiap **variabel** harus memiliki **kolomnya** sendiri
2. Setiap **observasi** harus memiliki **barisnya** sendiri
3. Setiap **nilai** harus memiliki **selnya** sendiri

Gambar berikut menggambarkan ketiga aturan tersebut.

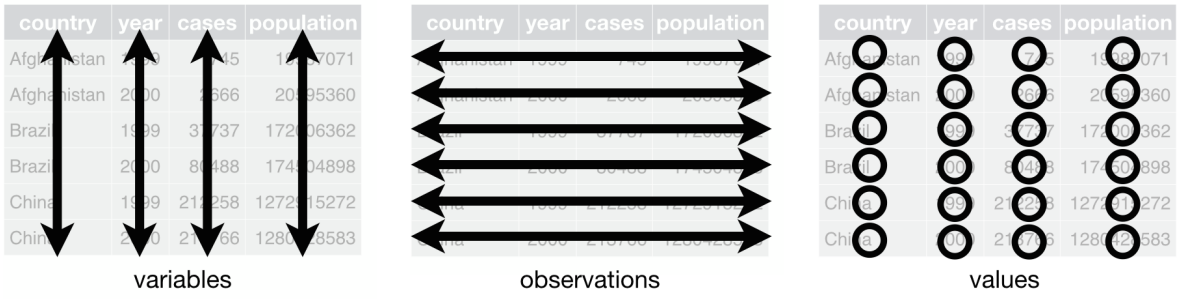


Figure 1: ilustrasi tidy dataset (sumber: Wickham, 2017)

Ketia aturan tersebut merupakan satu-kesatuan yang saling terkait karena tidak mungkin hanya memenuhi satu atau dua dari ketiga aturan tersebut. Sehingga, relasi antara ketiga aturan tersebut dapat diturunkan lagi kedalam dua buah aturan:

1. Letakkan setiap **dataset** ke dalam* **tibble/dataframe**
2. Letakkan setiap **variabel** ke dalam **kolom**

Kenapa kita perlu memastikan data yang kita miliki *tidy*:

1. Secara umum terdapat keuntungan dalam mengambil sebuah bentuk yang konsisten dalam menyimpan data. Jika kita memiliki sebuah bentuk data yang konsisten maka akan lebih mudah bagi kita mempelajari *tools* yang dapat bekerja dengan data tersebut. Sebagian besar *tools* untuk menganalisa data (SPSS, SAS, R, dll) bekerja dengan bentuk data yang seragam
2. Secara spesifik terdapat keuntungan apabila kita meletakkan sebuah variabel pada masing-masing kolom, dimana operasi vektorisasi akan lebih mudah dilakukan. Sebagian besar fungsi di R bekerja dalam bentuk yang tervektorisasi. Hal ini akan mempermudah dan mempercepat operasi pada data.

Terdapat beberapa fungsi yang digunakan untuk untuk melakukan proses transformasi bentuk dataset, antara lain:

- `pivot_wider()` : fungsi untuk merubah struktur memanjang menjadi struktur data melebar (menambah jumlah kolom dan mengurangi jumlah baris).
- `pivot_longer()`: fungsi ini merupakan kebalikan dari fungsi `pivot_longer`, dimana dataset ditransformasi ke dalam bentuk memanjang (memiliki lebih sedikit kolom dan menambah jumlah baris)
- `separate()` : fungsi untuk membuat memecah sebuah kolom menjadi beberapa kolom berdasarkan pemisah di dalam datanya.
- `unite()` : fungsi untuk menggabungkan nilai dari beberapa kolom.

Implementasi dari paket `tidyr` sebagian akan di bahas dalam tutorial kali ini. Jika pembaca ingin mengetahui lebih jauh contoh penerapan fungsi-fungsi tersebut, jalankan sintaks berikut:

```
example(<NAMA FUNGSI>)
```

atau cek dokumentasi fungsi tersebut menggunakan sintaks berikut:

```
vignette("pivot", package = "tidyr")
```

Ggplot

Paket `ggplot2` merupakan implementasi dari *The Grammar of Graphics* yang ditulis oleh **Leland Wilkinson**. `ggplot2` merupakan paket yang dikembangkan oleh **Hadley Wicham** ketika ia sedang menempuh kuliah di **Lowa State Universuty** dan masih dikembangkan hingga sekarang.

Grafik `ggplot2` terdiri dari sejumlah komponen kunci. Berikut adalah sejumlah komponen kunci yang membentuk grafik `ggplot2`.

- **data frame**: menyimpan semua data yang akan ditampilkan di plot.
- **aesthetic mapping**: menggambarkan bagaimana data dipetakan ke warna, ukuran, bentuk, lokasi. Dalam plot diberikan pada fungsi `aes()`
- **geoms**: objek geometris seperti titik, garis, bentuk.
- **facets**: menjelaskan bagaimana plot bersyarat / panel harus dibangun.
- **stats**: transformasi statistik seperti binning, quantiles, smoothing.
- **scales**: skala apa yang digunakan oleh *aesthetic map* (contoh: pria = merah, wanita = biru).
- **coordinate system**: menggambarkan sistem di mana lokasi geom akan digambarkan.

Secara umum fungsi pembuatan grafik menggunakan paket `ggplot` dapat dituliskan sebagai berikut:

```
# Data
ggplot(data = <DATA>) +

  # Aesthetics
  aes(<MAPPINGS>) +

  # Geometrics
  <GEOM_FUNCTION>(
     stat = <STAT>,
     position = <POSITION>
  ) +
```

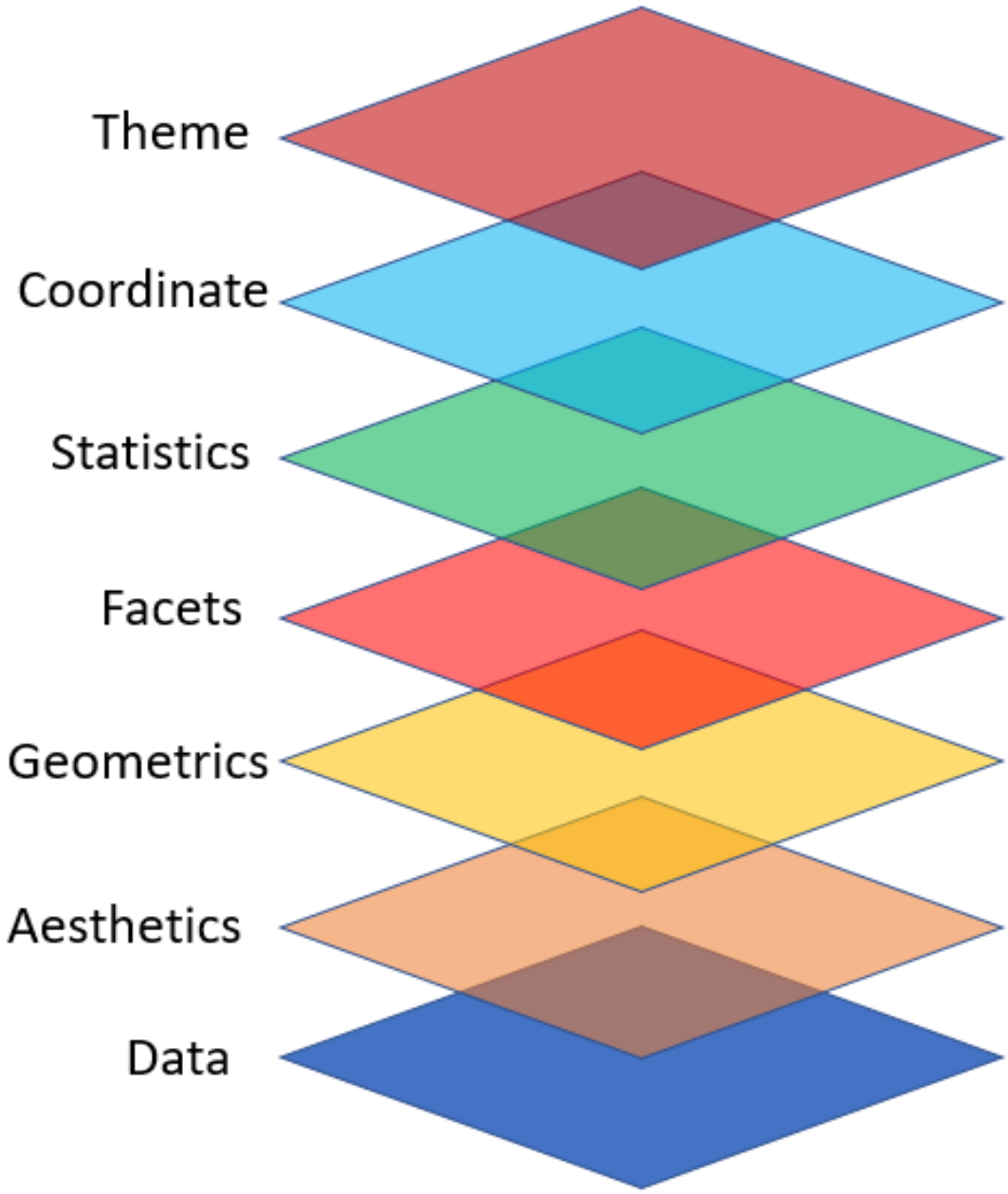


Figure 2: ilustrasi layer grammar of graphics (sumber: google images)

```
# Coordinate
<COORDINATE_FUNCTION> +

# Facet
<FACET_FUNCTION>
```

Dplyr

Data frame merupakan struktur data utama dalam statistik dan dalam `R`. Struktur dasar data frame ialah terdapat satu observasi tiap baris dan setiap kolom mewakili variabel. `R` memiliki implementasi internal data frame yang kemungkinan besar akan kita gunakan paling sering. Namun, ada paket di CRAN yang mengimplementasikan data frame layaknya basis data relasional yang memungkinkan kita untuk beroperasi pada data frame yang sangat besar.

Mengingat pentingnya mengelola data frame, penting bagi kita untuk memiliki alat yang baik untuk melakukannya. `R` memiliki beberapa paket seperti fungsi `subset()` dan penggunaan operator "[" dan "$" untuk mengekstrak himpunan bagian dari frame data. Namun, operasi lain, seperti pemfilteran, pengurutan, dan pengelompokan data, seringkali dapat menjadi operasi yang membosankan di `R` yang sintaksisnya tidak terlalu intuitif. Library `dplyr` dari paket `tidyverse` dirancang untuk mengurangi banyak masalah ini dan menyediakan serangkaian rutinitas yang dioptimalkan secara khusus untuk menangani data frame.

Paket `dplyr` dikembangkan oleh **Hadley Wickham** dari **RStudio** dan merupakan versi yang dioptimalkan dari paket `plyr`-nya. Library `dplyr` tidak menyediakan fungsionalitas baru untuk `R` sendiri, dalam arti bahwa semua yang dilakukan `dplyr` sudah dapat dilakukan dengan fungsi dasar `R`. Paket ini sangat menyederhanakan fungsi dasar yang telah ada di `R`.

Salah satu kontribusi penting dari paket `dplyr` adalah ia menyediakan "*grammar*" (khususnya, kata kerja) untuk manipulasi data dan untuk beroperasi pada data frame. Melalui *grammar* ini, kita dapat berkomunikasi dengan cara yang masuk akal terhadap apa yang akan kita lakukan pada data frame. Melalui cara ini, sintaks yang kit buat akan lebih mudah pula dipahami orang lain (dengan asumsi mereka juga tahu *grammar*-nya). Hal ini berguna karena memberikan abstraksi untuk manipulasi data yang sebelumnya tidak ada. Kontribusi lain yang bermanfaat adalah bahwa fungsi `dplyr` sangat cepat, karena banyak operasi utama dikodekan dalam bahasa C++.

Pada bagian ini pembaca akan belajar **6** fungsi utama yang ada pada paket `dplyr`. Fungsi tersebut antara lain:

1. Mengambil sejumlah observasi berdasarkan nilainya (`filter()`).
2. Mengurutkan kembali baris data frame berdasarkan nilai pada sebuah atau beberapa variabel (`arrange()`).
3. Mengambil atau subset terhadap sebuah atau beberapa variabel berdasarkan nama variabel/kolom (`select()`).
4. Membuat variabel baru atau menambahkan kolom baru (`mutate()`).
5. Membuat ringkasan terhadap data frame (`summarize()`)
6. Mengelompokkan operasi berdasarkan grup data (`group_by()`).

Keseluruhan fungsi tersebut format fungsi yang seragam, yaitu:

1. Argumen pertama adalah data frame.
2. Argumen selanjutnya adalah deskripsi yang akan dilakukan terhadap data frame (filter, pengurutan kembali, membuat ringkasan, dll) menggunakan nama variabel (tanpa tanda kutip).
3. Hasil operasi yang diperoleh adalah data frame baru.

Selain ke-6 fungsi utama tersebut, terdapat fungsi lainnya pada paket `dplyr`. Fungsi lainnya terkait dengan fungsi untuk melakukan *join* atau menggabungkan dua buah dataset. Secara umum, fungsi *join* dapat divisualisasikan berdasarkan gambar berikut:

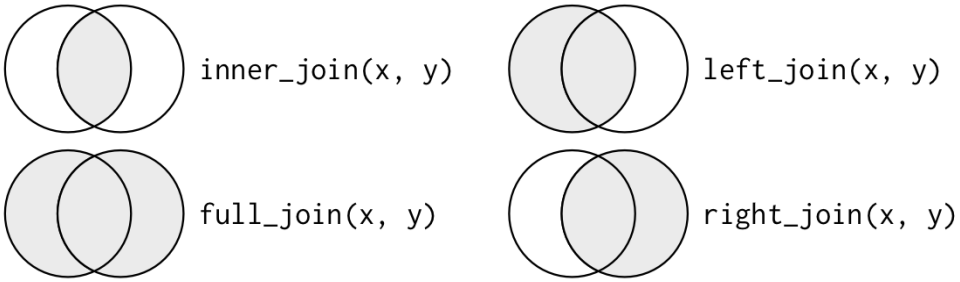


Figure 3: Tipe join pada dua buah data(sumber: Wickham, 2017)

Fungsi join berdasarkan gambar tersebut dapat dijelaskan sebagai berikut:

1. **`inner_join()`** : melakukan join hanya pada observasi dengan elemen kunci yang sama-sama ada pada kedua tabel.
2. **`left_join()`** : menggabungkan seluruh baris pada tabel kiri dan sebagian baris pada tabel kanan yang elemen kuncinya cocok dengan tabel kiri.
3. **`right_join()`** : kebalikan dari fungsi `left_join()`.
4. **`full_join()`** : menggabungkan seluruh observasi pada kedua tabel melalui kolom elemen kunci.

Ringkasan Data

Terdapat beberapa fungsi yang dapat digunakan untuk menampilkan ringkasan data pada dataset, antara lain:

- `glimpse` : fungsi dari library `tibble` untuk menampikan struktur data, seperti: jumlah observasi, jumlah kolom, nama kolom dan jenis datanya, dan contoh data pada masing-masing kolom. `glimpse` merupakan versi transpose dari fungsi `print`, kolom ditampilkan per baris dan data pada tiap kolom ditampilkan secara mendatar disamping nama kolomnya.
- `summary` : fungsi dari library `base` untuk menampilkan ringkasan data pada masing-masing kolom , seperti : `mean`, `median`, `min` dan `max`, `1st Qu.` dan `3rd Qu.`, jumlah baris *missing value* pada masing-masing kolom, dan tabel kontingensi.
- `skim` : fungsi dari library `skimr` merupakan alternatif lain dari fungsi `summary`, dengan cepat memberikan gambaran luas dari kerangka data. Fungsi ini menangani berbagai jenis data, mengirimkan satu set fungsi ringkasan yang berbeda berdasarkan pada jenis kolom dalam dataframe.

Berikut adalah penerapan fungsi `glimpse`:

```
glimpse(gapminder)
```

```
## Rows: 1,704
## Columns: 6
## $ country   <chr> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan...
## $ continent <chr> "Asia", "Asia", "Asia", "Asia", "Asia", "Asia", "Asia", "...
## $ year      <dbl> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 199...
## $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.854, 4...
## $ pop       <dbl> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372,...
## $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.113...
```

Berdasarkan hasil yang diperoleh, dataset `gapminder` memiliki `1704` baris dan `6` kolom. Terdapat dua buah jenis data pada dataset tersebut, yaitu: *character* (`country` dan `continent`) dan `double/numeric`.

Untuk penerapan fungsi `summary`, ditampilkan pada sintaks berikut:

```
summary(gapminder)
```

```
##    country           continent            year         lifeExp
##  Length:1704        Length:1704        Min.   :1952   Min.   :23.60
##  Class :character   Class :character   1st Qu.:1966   1st Qu.:48.20
##  Mode  :character   Mode  :character   Median :1980   Median :60.71
##                                        Mean   :1980   Mean   :59.47
##                                        3rd Qu.:1993   3rd Qu.:70.85
##                                        Max.   :2007   Max.   :82.60
##       pop              gdpPercap
##  Min.   :6.001e+04   Min.   :   241.2
##  1st Qu.:2.794e+06   1st Qu.:  1202.1
##  Median :7.024e+06   Median :  3531.8
##  Mean   :2.960e+07   Mean   :  7215.3
##  3rd Qu.:1.959e+07   3rd Qu.:  9325.5
##  Max.   :1.319e+09   Max.   :113523.1
```

Fungsi `skim` akan menampilkan ringkasan data yang lebih *tidy* dibandingkan fungsi `summary`. Selain itu, pada jenis data *numeric* dan *integer*, fungsi tersebut akan menampilkan hitogram untuk menggambarkan distribusi dari data *numeric* dan *integer*.

```
skim(gapminder)
```

Table 1: Data summary

Name	gapminder
Number of rows	1704
Number of columns	6
Column type frequency:	
character	2
numeric	4
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
country	0	1	4	24	0	142	0
continent	0	1	4	8	0	5	0

Variable type: numeric

skim_variable	missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
year	0	1	1979.50	17.27	1952.00	1965.75	1979.50	1993.25	2007.0	
lifeExp	0	1	59.47	12.92	23.60	48.20	60.71	70.85	82.6	
pop	0	1	29601212.32	106157896.74	60011.00	2793664.00	7023595.50	19585221.75	1318683096.0	
gdpPercap	0	1	7215.33	9857.45	241.17	1202.06	3531.85	9325.46	113523.1	

Berdasarkan output yang dihasilkan, terdapat 4 komponen yang ditampilkan, antara lain:

1. `Data summary`
2. `Column type frequency`
3. `Group variables`
4. Ringkasan data berdasarkan tiap jenis data.

Variasi

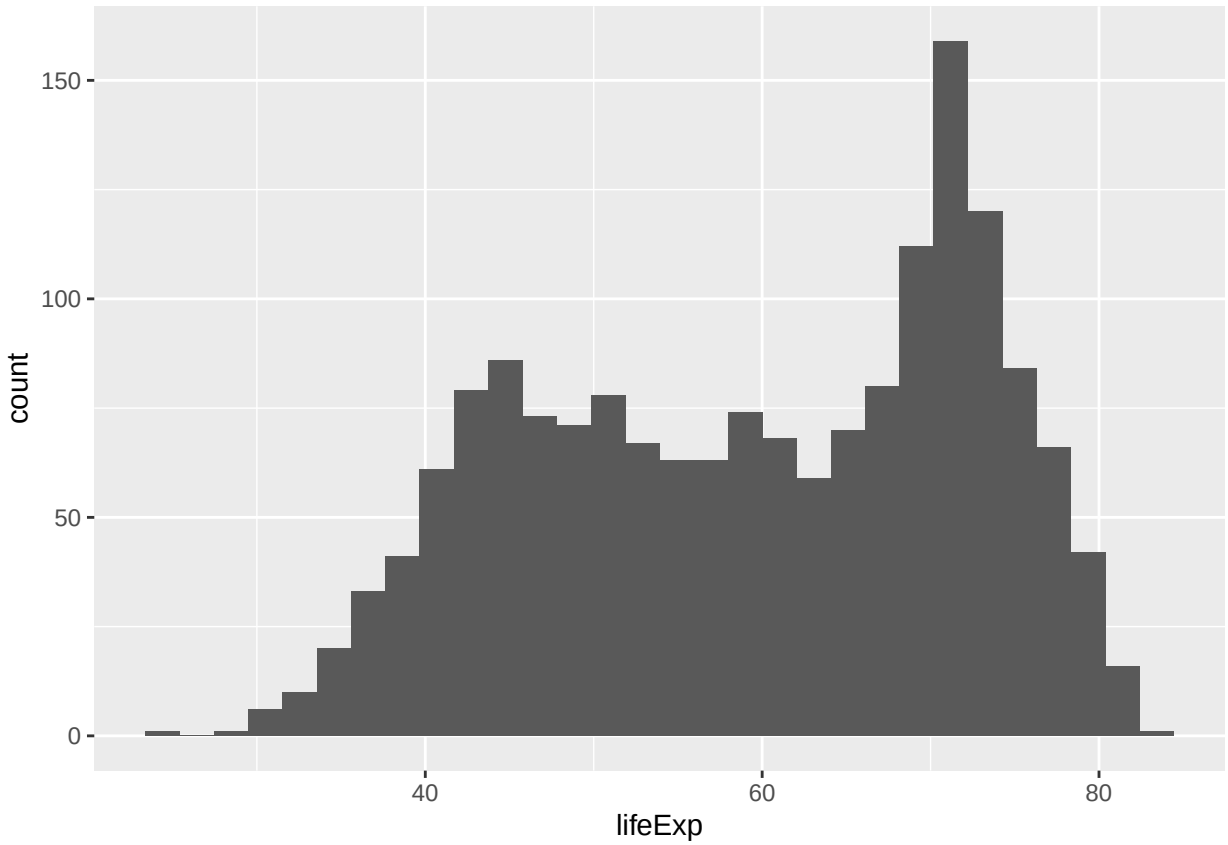
Variasi adalah kecenderungan nilai-nilai variabel berubah dari pengukuran ke pengukuran. Kita dapat melihat variasi dengan mudah dalam kehidupan nyata; jika kita mengukur variabel kontinu dua kali, kita akan mendapatkan dua hasil berbeda. Setiap variabel memiliki pola variasinya sendiri, yang dapat mengungkapkan informasi menarik. Cara terbaik untuk memahami pola itu adalah dengan memvisualisasikan distribusi nilai-nilai variabel.

Data Numerik

Visualisasi yang umum digunakan untuk menggabarkan distribusi data numerik adalah **histogram**. Untuk melakukannya, kita dapat menggunakan fungsi `geom_histogram` dari library `ggplot2`.

```
ggplot(gapminder) +
  geom_histogram(aes(x = lifeExp))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



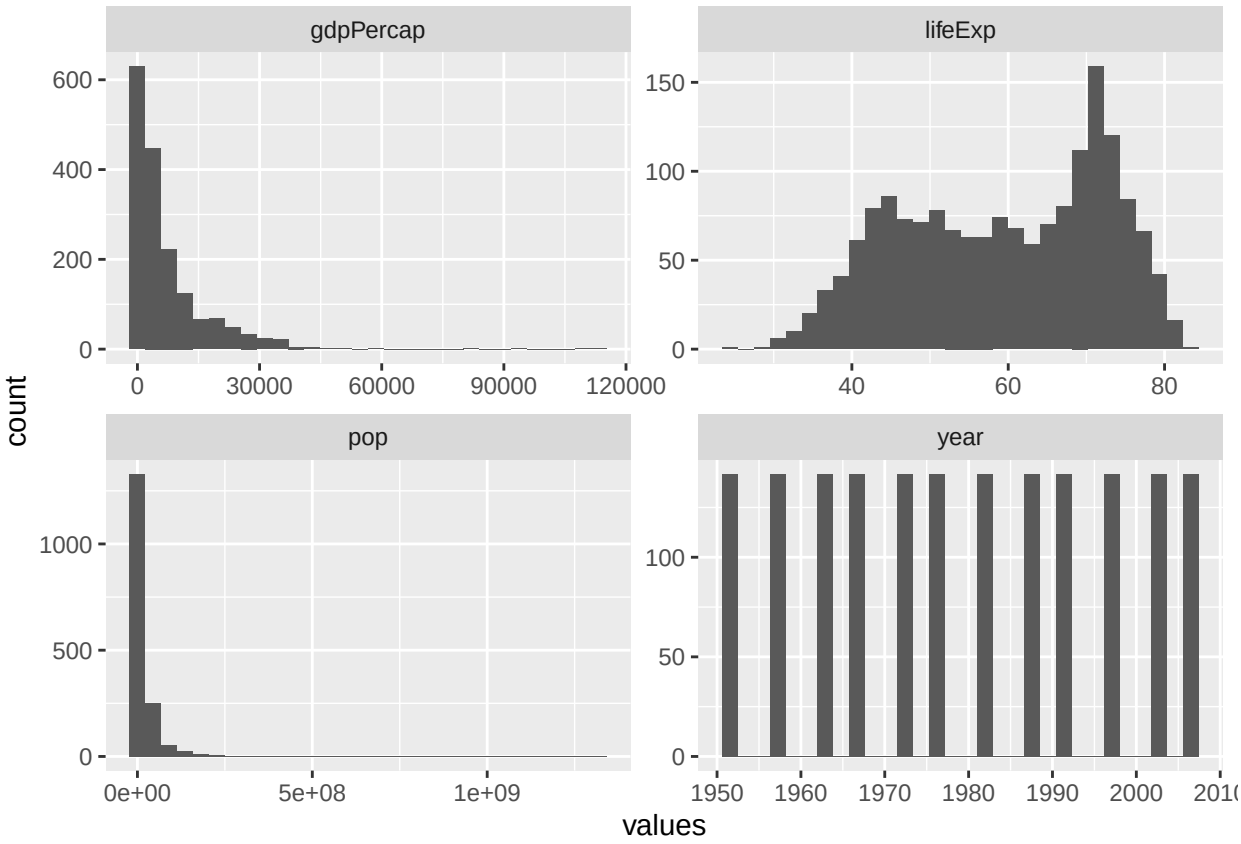
Secara default, data akan dibagi ke dalam 30 buah rentang nilai (*bins*). Berdasarkan visualisai tersebut, dapat kita lihat bahwa distribusi variabel `lifeExp` memiliki dua buah puncak.

Bagaimana cara memvisualisasikan seluruh variabel numerik ke dalam satu visualisasi? Untuk melakukannya, berikut adalah tahapan yang perlu dilakukan:

1. Transformasi seluruh variabel numerik ke dalam dua buah kolom, yaitu: kolom variabel dan nilai. Transformasi akan menggunakan fungsi `pivot_longer`.
2. Visualisasi dengan menggunakan `facet_grid` untuk membuat visualisasi pada tiap kolom data.

```
gapminder %>%
  select(!dplyr::starts_with("co")) %>%
  pivot_longer(cols = year:gdpPercap , names_to = "variables", values_to = "values") %>%
  ggplot() +
  geom_histogram(aes(values)) +
  facet_wrap(~variables, scales = "free")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Berdasarkan hasil visualisasi, ketiga variabel numerik (`gdpPercap`, `lifeExp`, dan `pop`) tidak berdistribusi normal. Transformasi diperlukan untuk variabel tersebut, khususnya variabel `gdpPercap` dan `pop`. Transformasi yang dapat digunakan adalah transformasi logaritmik sebab ketiga variabel tersebut memiliki jenis kemencengan positif (*positif skewness*). Sedangkan pada variabel `year` dapat kita ketahui bahwa jumlah negara yang dicatat atau disurvey sama setiap tahunnya dan pengukuran dilakukan sestiap 5 tahun sekali.

Data Kategorikal

Distribusi variabel kategorikan dapat dipelajari dengan cara membuat tabulasi silang (menghitung jumlah observasi pada masing-masing kategori) atau dengan visualisasi menggunakan barplot. Visualisasi digunakan apabila jumlah kategorinya sedikit (tidak lebih dari 5 atau 10).

```
gapminder %>%
  group_by(country) %>%
  count()
```

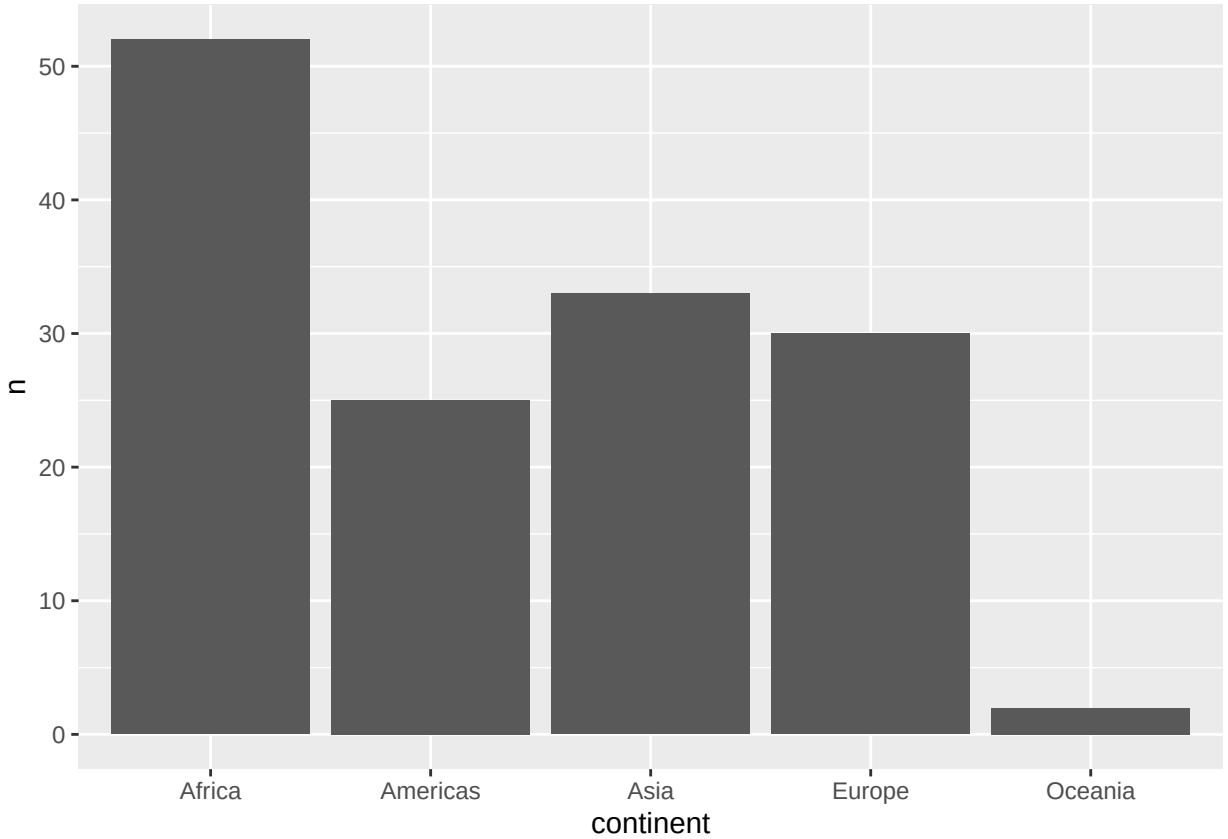
```
## # A tibble: 142 x 2
## # Groups:   country [142]
##    country         n
##    <chr>       <int>
##  1 Afghanistan    12
##  2 Albania        12
##  3 Algeria        12
##  4 Angola         12
##  5 Argentina      12
##  6 Australia      12
```

```
##  7 Austria        12
##  8 Bahrain        12
##  9 Bangladesh     12
## 10 Belgium        12
## # ... with 132 more rows
```

Berdasarkan output yang dihasilkan, masing-masing negara dilakukan survey sebanyak 12 kali setiap 5 tahun dan pada data tersebut, distribusi jumlah survey masing-masing negara sama.

Lebih jauh lagi kita juga dapat memeriksa jumlah negara pada masing-masing benua untuk mengetahui distribusi negara yang disurvey pada masing-masing benua. Distribusi jumlah negara pada masing-masing benua dapat dilakukan visualisasi dengan barplot untuk menggambarkan distribusinya sebab jumlah kategorinya yang tidak sebanyak variabel `country`.

```
gapminder %>%
  filter(year == 2007) %>%
  select(dplyr::starts_with("co")) %>%
  group_by(continent) %>%
  count() %>%
  ggplot() +
  geom_bar(aes(x = continent, y = n), stat = "identity")
```



Berdasarkan hasil visualisasi dapat diketahui bahwa distribusi jumlah negara yang disurvey atau tercatat tidak seimbang pada masing-masing benua, dimana benua afrika memiliki jumlah negara yang tercatat lebih dari 50 sedangkan oceania hanya 2 negara saja. Kondisi tersebut akan berpengaruh pada performa model yang akan dibentuk dimana benua afrika akan memberikan efek dominan pada model, sedangkan benua oceania memberikan efek paling kecil atau bisa jadi efek yang dihasilkan tidak tertangkap oleh model. Jika

model yang dibuat ingin menangkap seluruh efek benua secara merata, proses `over` atau `under sampling` perlu dilakukan berdasarkan variabel `continent`.

Kovarian

Jika variasi menggambarkan perilaku di dalam suatu variabel, kovariasi menggambarkan perilaku di antara variabel. Kovarian adalah kecenderungan untuk nilai-nilai dari dua atau lebih variabel bervariasi bersama dalam cara yang terkait. Cara terbaik untuk menemukan kovarisasi adalah memvisualisasikan hubungan antara dua variabel atau lebih. Bagaimana kita melakukannya harus kembali tergantung pada jenis variabel yang terlibat.

Kategorikal dan Kontinu

Cara paling mudah untuk memvisualisasikan hubungan antara sebuah variabel kategorikal dengan variabel kontinu adalah dengan menggunakan boxplot. Boxplot merupakan sejenis ringkasan visual untuk distribusi nilai-nilai yang populer di kalangan ahli statistik. Setiap boxplot terdiri dari:

- Sebuah kotak yang membentang dari persentil ke-25 dari distribusi ke persentil ke-75, jarak yang dikenal sebagai rentang interkuartil (IQR). Di tengah kotak adalah garis yang menampilkan median, yaitu persentil ke-50, dari distribusi. Ketiga baris ini memberi kita gambaran tentang distribusi dan apakah distribusinya simetris tentang median atau condong ke satu sisi.
- Poin visual yang menampilkan pengamatan yang jatuh lebih dari 1,5 kali IQR dari kedua sisi kotak. Titik-titik *outlier* ini merupakan titik yang tidak biasa sehingga diplot secara individual.
- Baris (atau kumis) yang memanjang dari setiap ujung kotak dan menuju ke titik non-outlier terjauh dalam distribusi.

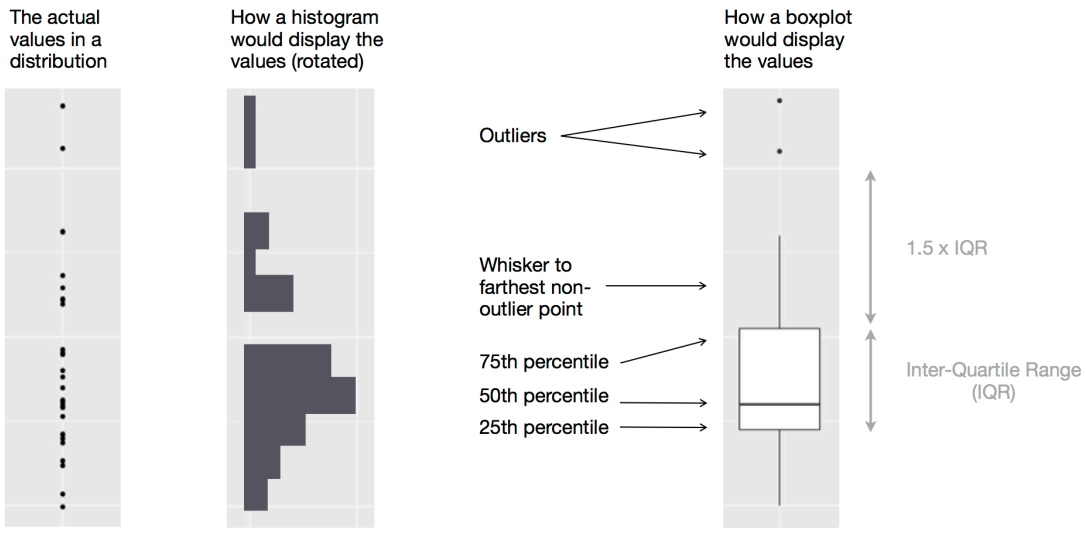
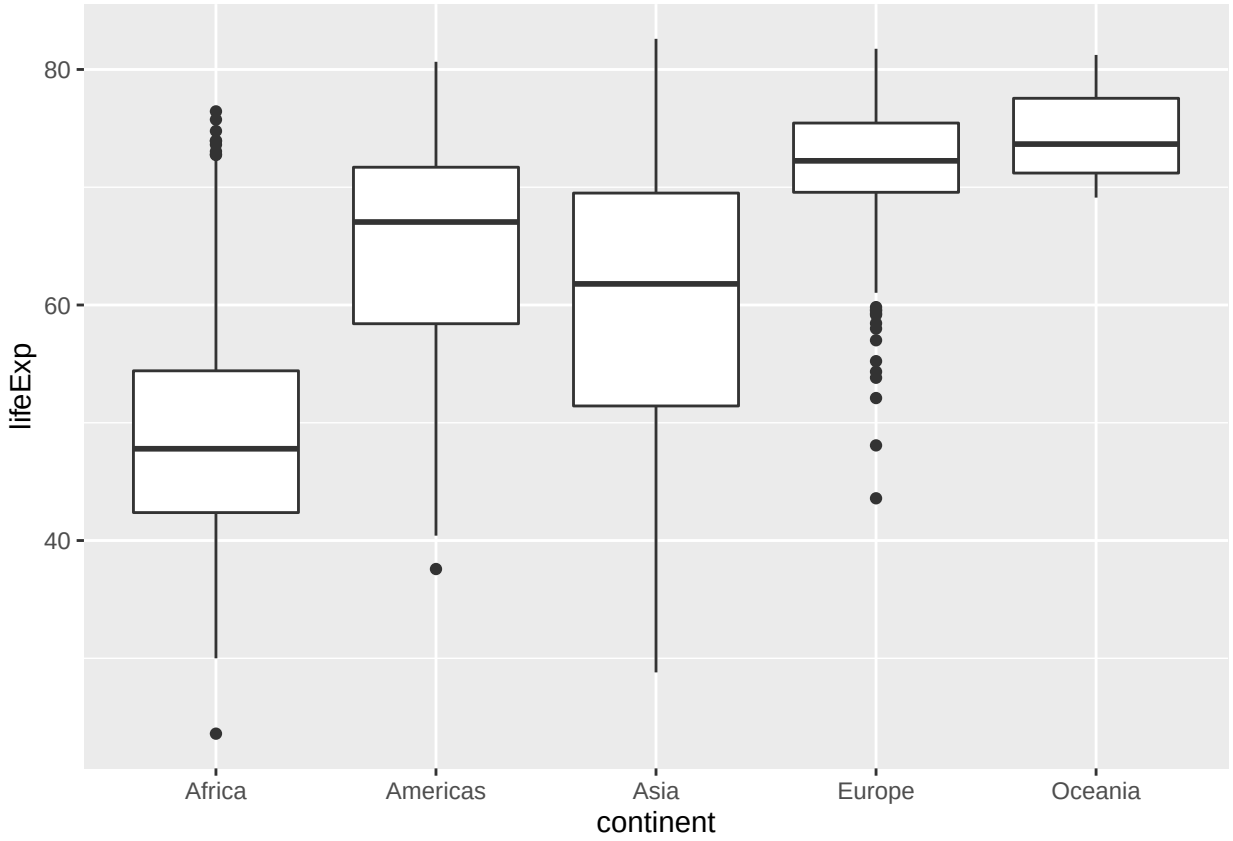


Figure 4: sumber: Hadley, 2017

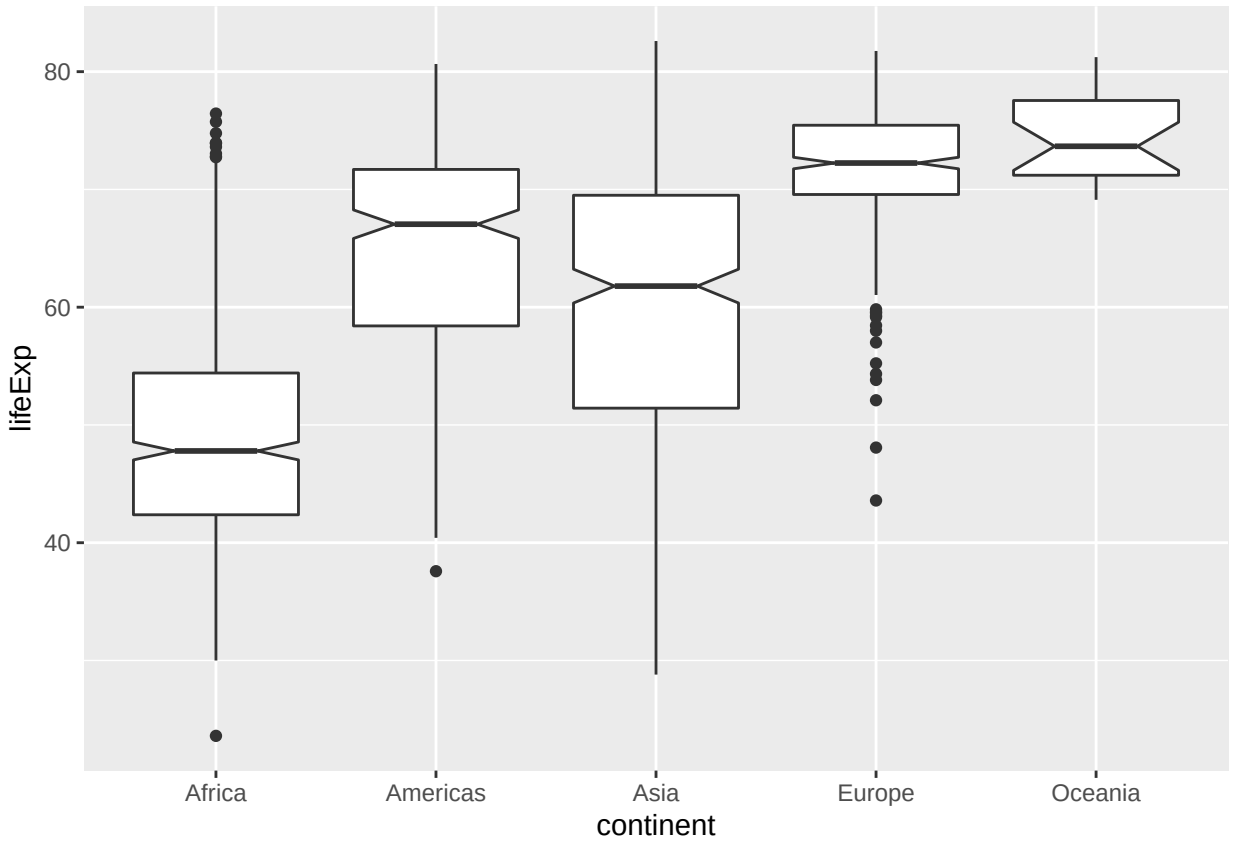
Mari kita coba visualisasikan hubungan antara variabel kategorikal (`continent`) dengan variabel kontinu (`lifeExp`).

```
ggplot(gapminder) +
  geom_boxplot(aes(x = continent, y = lifeExp))
```



Berdasarkan visualisasi yang ditampilkan dapat dilihat bahwa benua afrika memiliki median `lifeExp` yang paling rendah dibandingkan benua lainnya. Sedangkan median `lifeExp` pada benua eropa dan oceania relatif sama. Untuk mengetahui apakah nilai media antara satu benua dengan benua lainnya berbeda secara signifikan, kita dapat menambahkan *notch* pada boxplot yang menunjukkan nilai retang keyakinan dari median.

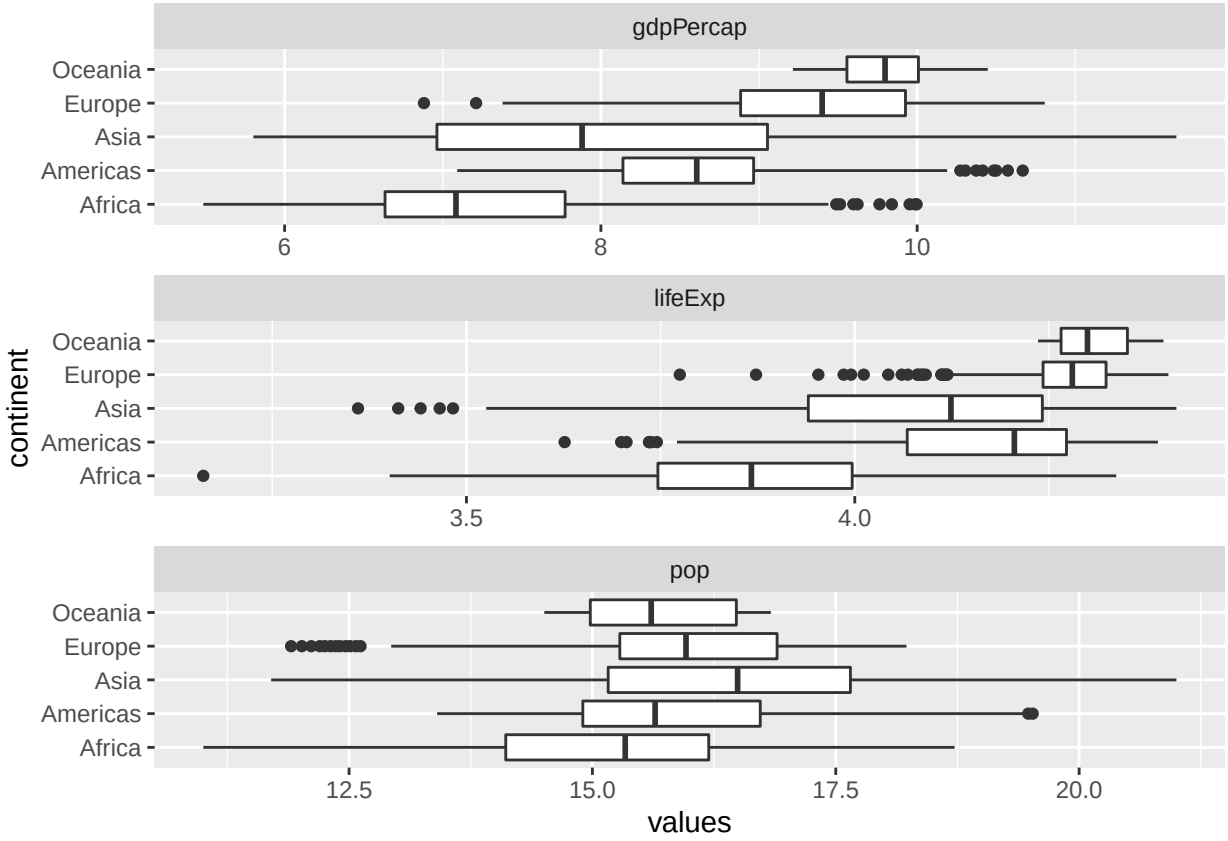
```
ggplot(gapminder) +
  geom_boxplot(aes(x = continent, y = lifeExp), notch = TRUE)
```



Dapat kita lihat, median `lifeExp` benua eropa dan oceania tidak berbeda secara signifikan yang ditunjukkan dengan rentang median yang saling overlap.

Untuk memvisualisasikan beberapa variabel kontinu dengan sebuah variabel kategorikal dapat dilakukan dengan menggunakan metode yang telah dijelaskan pada subbab visualisasi distribusi data numerik.

```
gapminder %>%
  select(continent, lifeExp:gdpPercap) %>%
  # lakukan transformasi logaritmik pada seluruh variabel numerik
  mutate_if(is.numeric, log) %>%
  pivot_longer(cols = lifeExp:gdpPercap, names_to = "variables", values_to = "values") %>%
  ggplot() +
  geom_boxplot(aes(x = continent, y = values)) +
  coord_flip() +
  facet_wrap(~variables, scales = "free", nrow = 3)
```



Kategorikal dan Kategorikal

Untuk melihat kovarian antara dua variabel kategorikal, kita dapat membuat tabulasi silang.

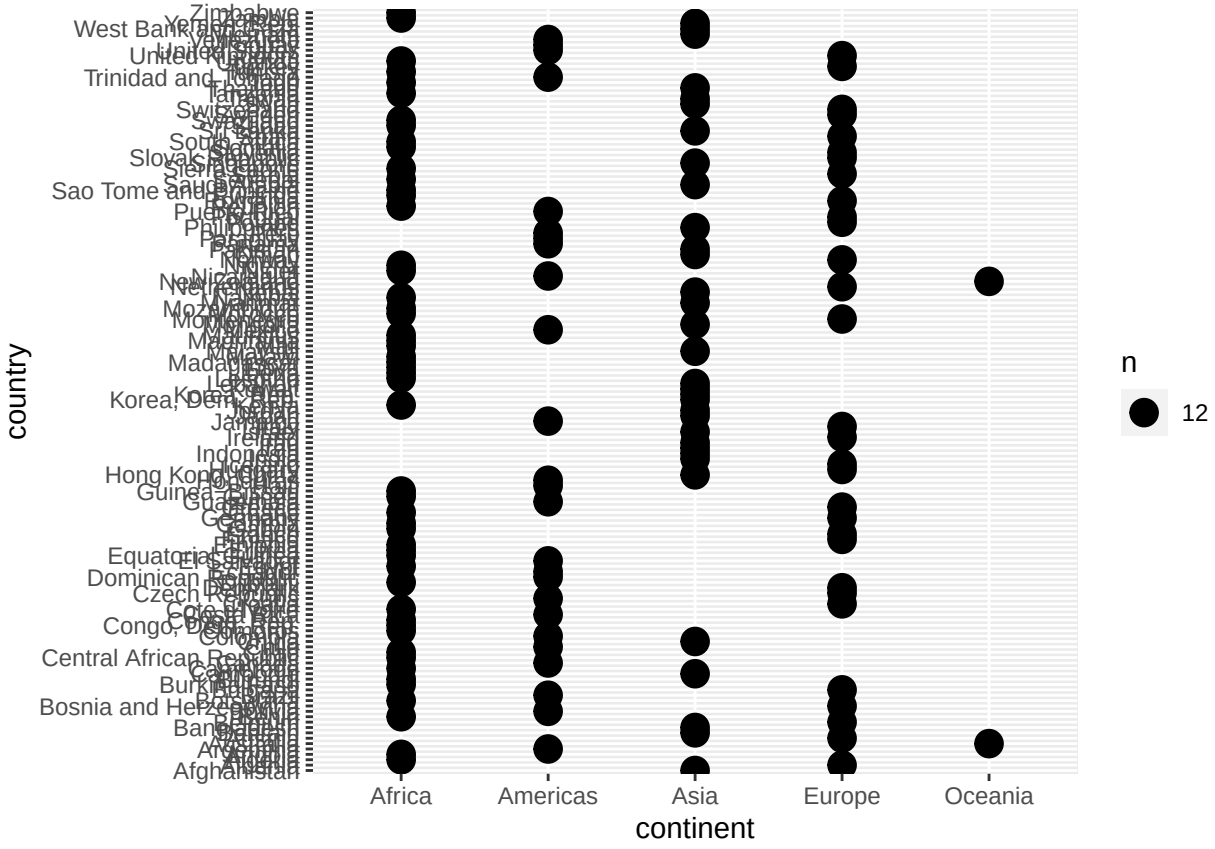
```
gapminder %>%
  group_by(country, continent) %>%
  count()
```

```
## # A tibble: 142 x 3
## # Groups:   country, continent [142]
##    country     continent     n
##    <chr>       <chr>     <int>
##  1 Afghanistan Asia         12
##  2 Albania     Europe       12
##  3 Algeria     Africa       12
##  4 Angola      Africa       12
##  5 Argentina   Americas     12
##  6 Australia   Oceania      12
##  7 Austria     Europe       12
##  8 Bahrain     Asia         12
##  9 Bangladesh  Asia         12
## 10 Belgium     Europe       12
## # ... with 132 more rows
```

Pada umumnya visualisasi hubungan antara 2 variabel kategroikal dapat dilakukan dengan menggunakan

fungsi `geom_count` pada `ggplot`> Perlu dicatat bahwa variabel kategorikal yang divisualisasikan sebaikanya tidak lebih dari 5 atau 10.

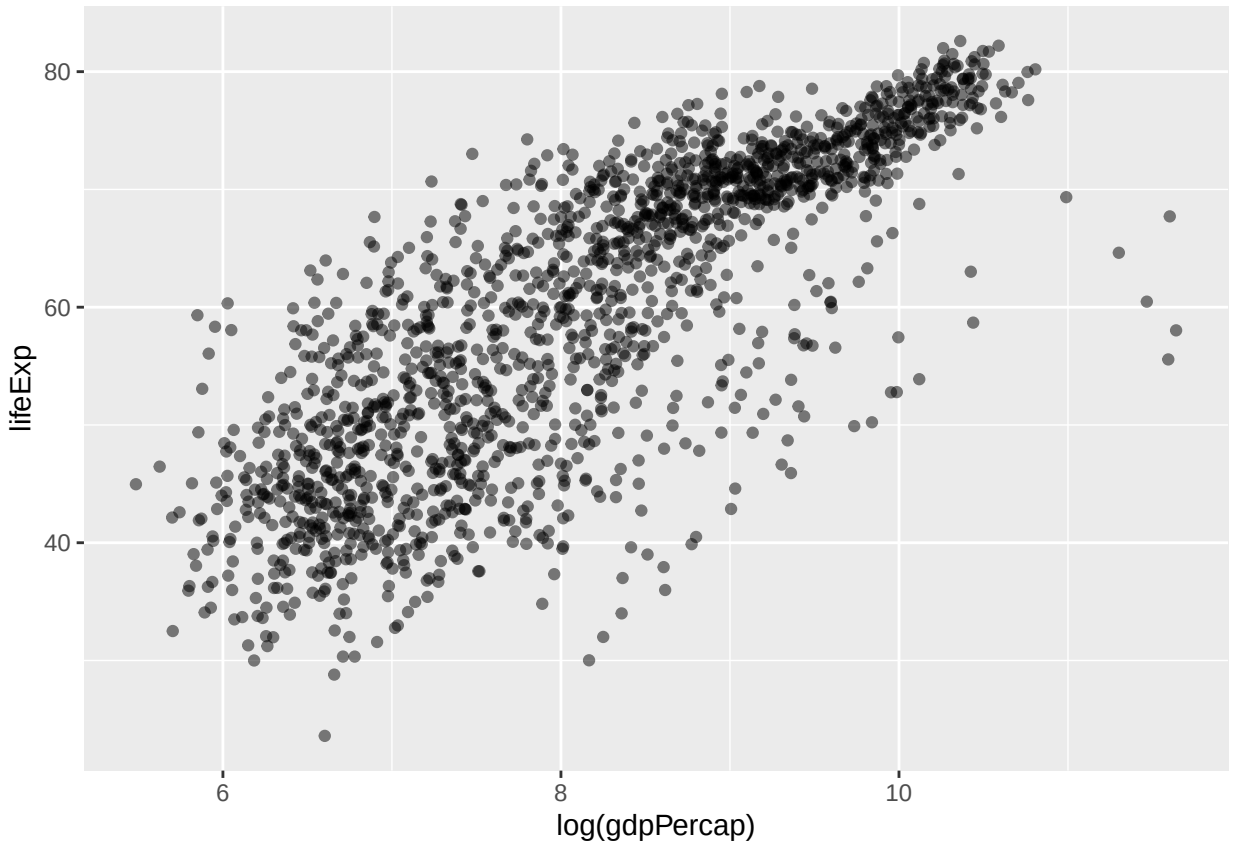
```
ggplot(gapminder) +
  geom_count(aes(x = continent, y = country))
```



Kontinu dan Kontinu

Kovarian antara dua variabel kontinu dapat divisualisasikan dengan menggunakan scatterplot. Berikut adalah contoh visualisasi antara variabel `lifeExp` dan log `gdpPercap`.

```
ggplot(gapminder) +
  geom_point(aes(x = log(gdpPercap), y = lifeExp), alpha = 0.5)
```



Berdasarkan visualisasi tersebut dapat kita cermati bahwa negara dengan `gdpPercap` tinggi cenderung memiliki `lifeExp` yang tinggi pula.

Alternatif lainnya yang dapat digunakan untuk mevisualisasikan kovarian antara dua variabel numerik adalah dengan menggunakan heatmap. Berbeda dengan scatterplot, hetamap mevisualisasikan nilai koefisien korelasi anatar dua variabel.

Analisis Data Exploratif Menggunakan `DataExplorer`

`DataExplorer` merupakan library alternatif yang dapat kita gunakan untuk melakukan otomasi pada proses EDA. Pada `DataExplorer` terdapat sejumlah fungsi yang dapat mereduksi waktu dan jumlah sintaks yang ditulis untuk melihat pola pada data. Berikut adalah contoh sintaks yang dapat kita gunakan:

Laporan

Untuk memperoleh laporan terkait dataset `gapminder`, kita dapat menjalanakan funsgi `create_report`.

```
create_report(gapminder)
```

Fungsi tersebut akan memberikan kita laporan dalam bentuk format `html` terkait ringkasan data dan visualisasi dari dataset `gapminder`.

Visualisasi

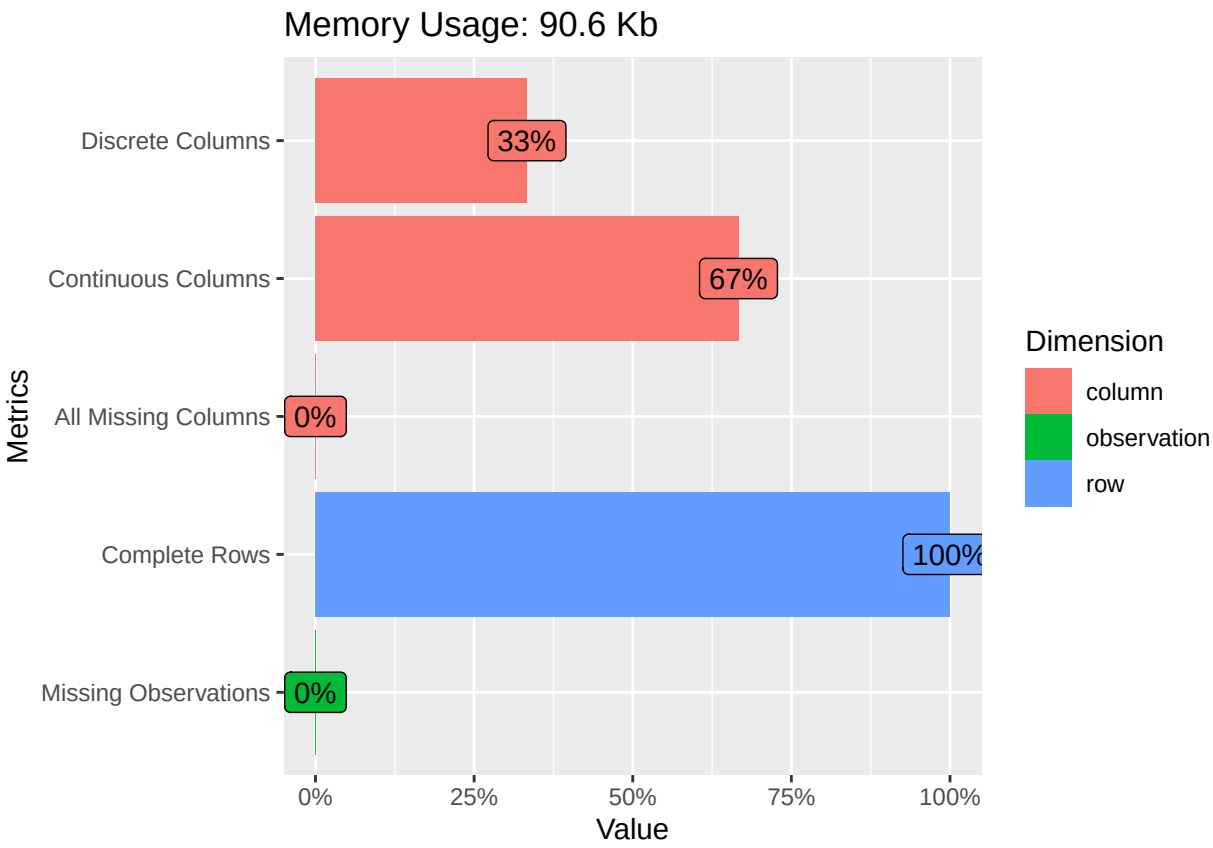
Dibanding menjalankan fungsi `create_report`, kita dapat menggunakan fungsi lainnya secara satu-persatu untuk menghasilkan output yang kita inginkan. Ini berguna ketika kita ingin meletakkan deskripsi pada temuan yang akan kita buat.

```
# melihat deskripsi dasar dari dataset gapminder
introduce(gapminder)
```

```
## # A tibble: 1 x 9
##    rows columns discrete_columns continuous_colu~ all_missing_col~
##   <int>   <int>            <int>            <int>            <int>
## 1  1704       6                2                4                0
## # ... with 4 more variables: total_missing_values <int>, complete_rows <int>,
## #   total_observations <int>, memory_usage <dbl>
```

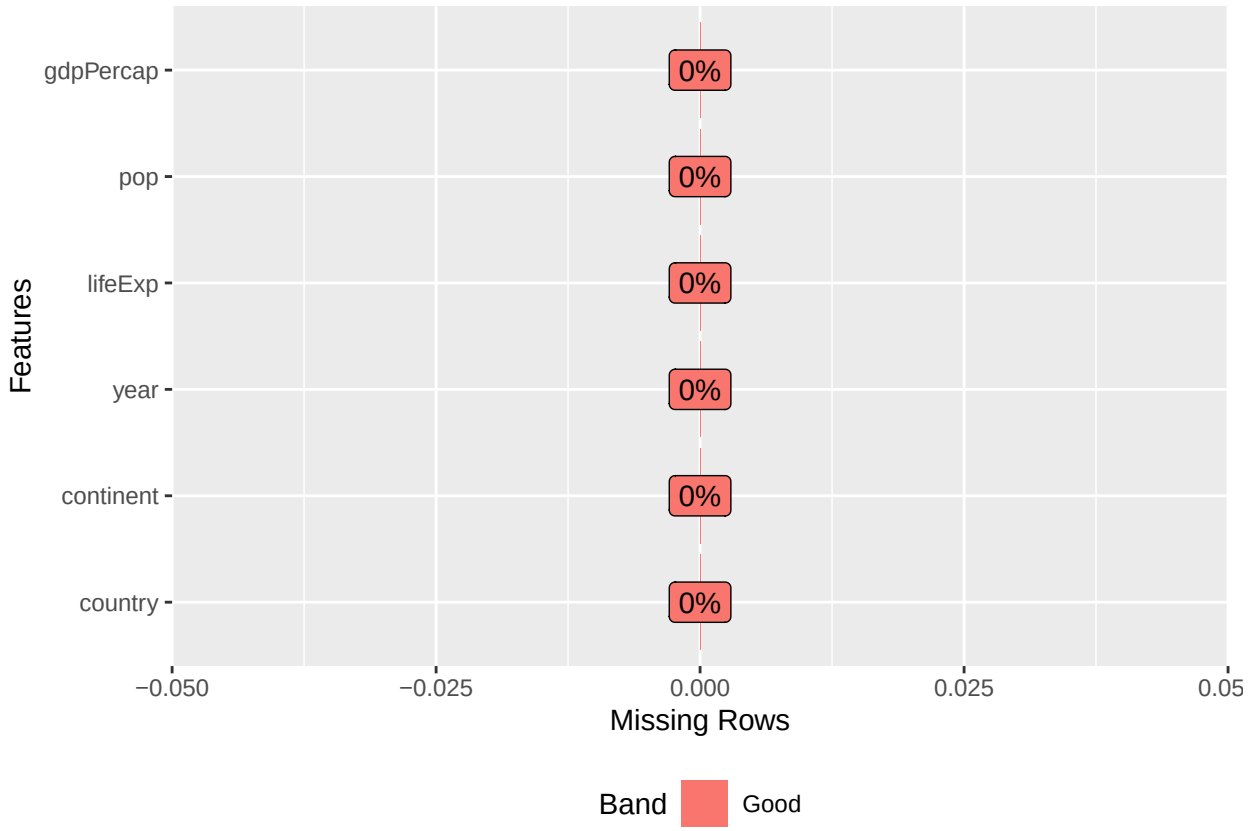
Fungsi berikut akan memberikan ringkasan terkait penggunaaan memori pada dataset `gapminder`.

```
# visualisasi deskripsi dasar dataset `gapminder`
plot_intro(gapminder)
```



Untuk melihat jumlah *missing value* pada data, jalankan fungsi `plot_missing`.

```
plot_missing(gapminder)
```

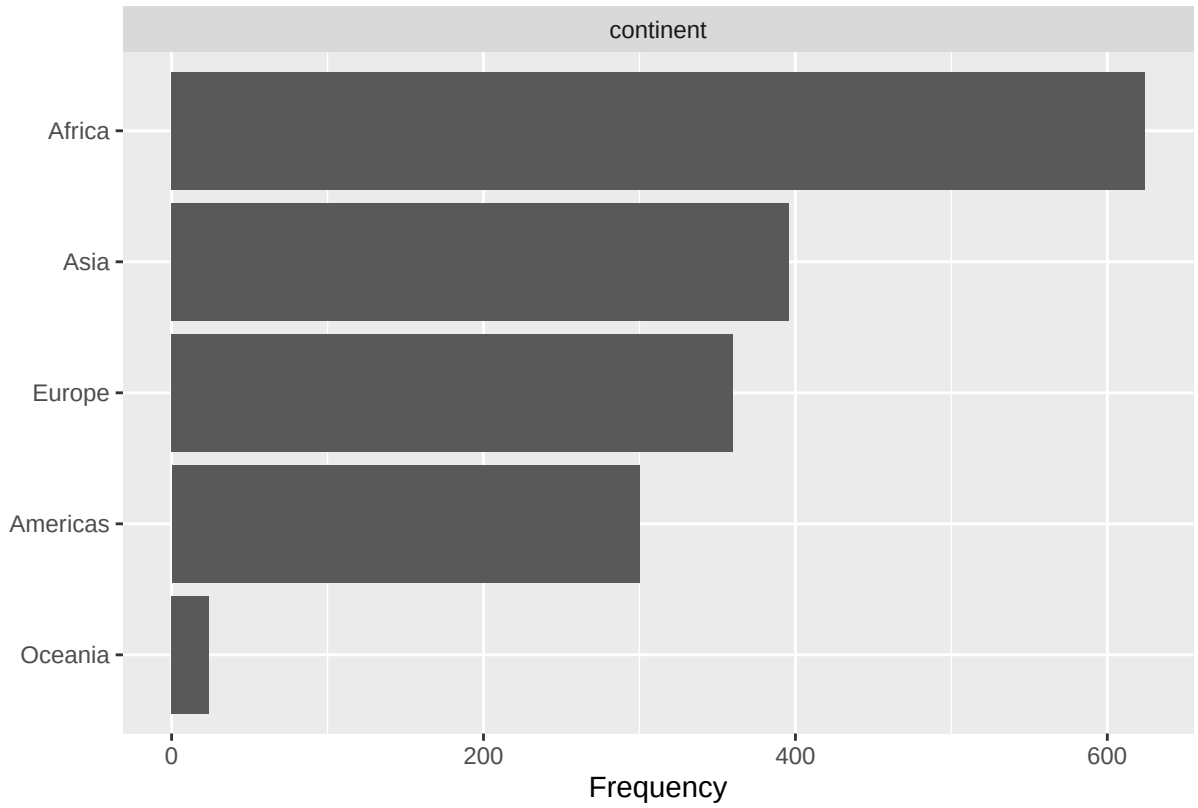


Berdasarkan output yang dihasilkan, pada dataset `gapminder` tidak terdapat *missing value*.

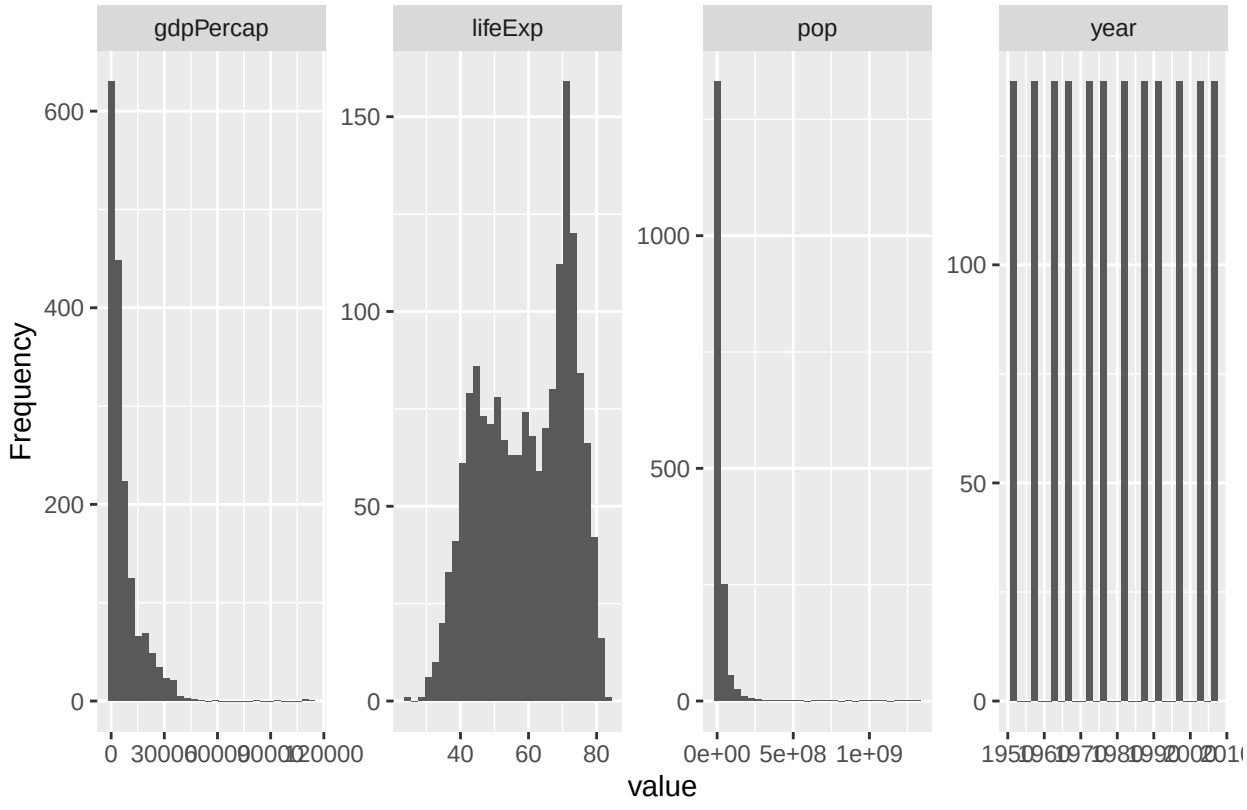
Untuk memvisualisasikan variasi pada dataset, kita dapat menggunakan fungsi `plot_bar` dan `plot_histogram`.

```
plot_bar(gapminder)
```

```
## 1 columns ignored with more than 50 categories.
## country: 142 categories
```




```
plot_histogram(gapminder)
```

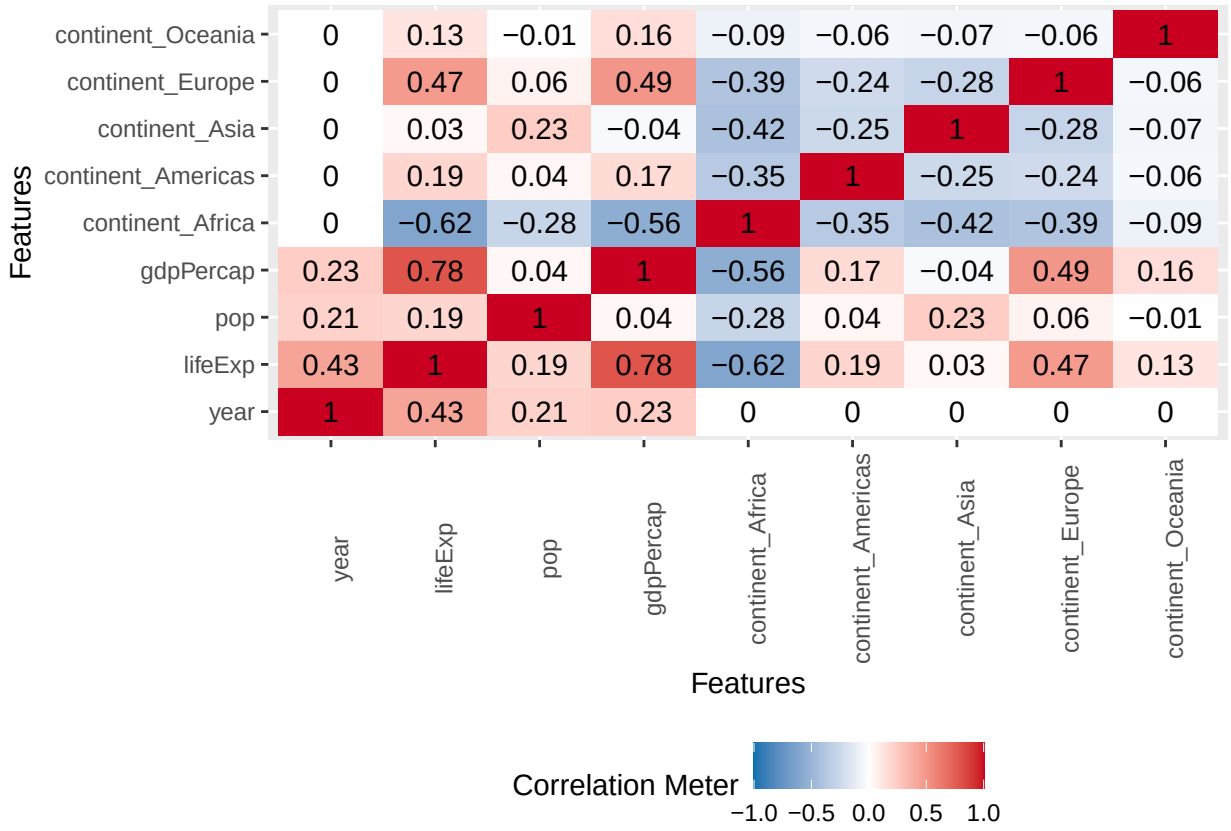


Berdasrakan output yang dihasilkan, variabel `country` tidak divisualisasikan karena memiliki jumlah kategori lebih besar dari 50.

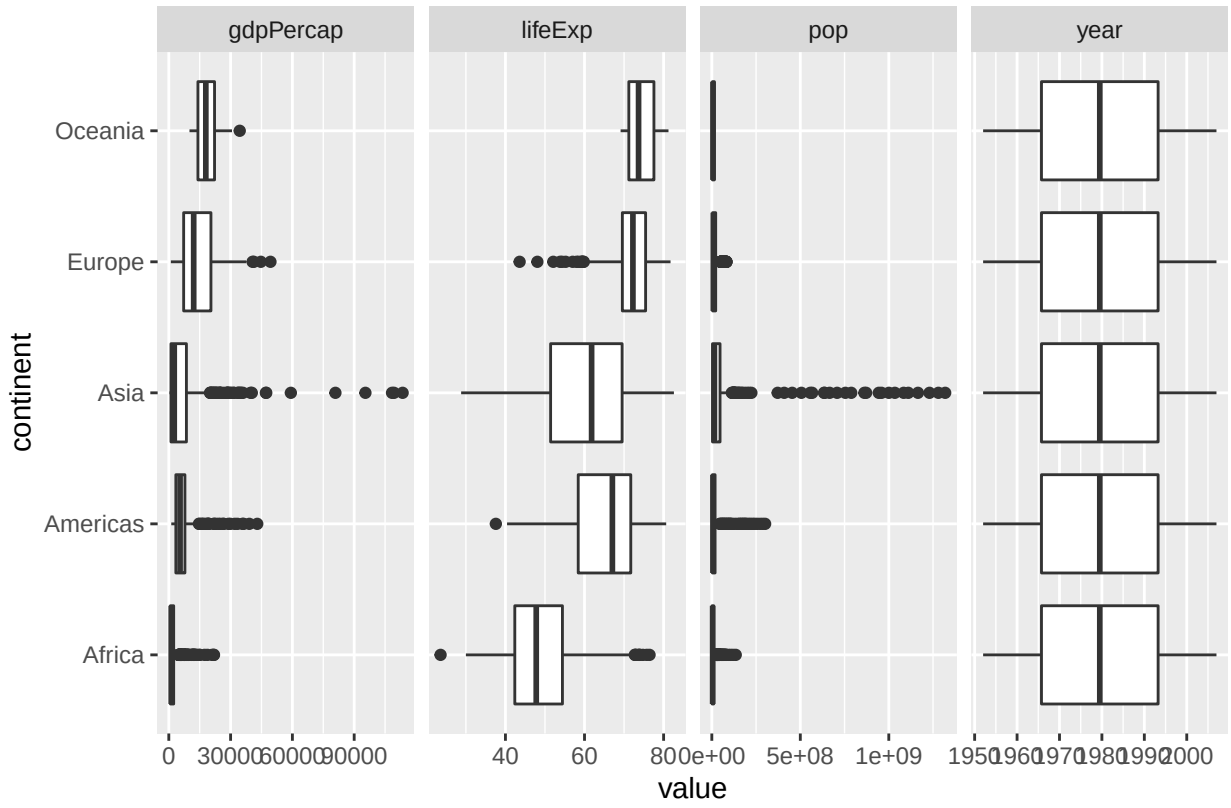
Untuk mwmvisualisikan kovarian antara dua variabel, kita dapat menggunakan fungsi `plot_correlation`, `plot_boxplot`, dan `plot_scatterplot`.

```
# heatmat korelasi
gapminder %>%
  mutate_if(is.numeric, log) %>%
  plot_correlation()
```

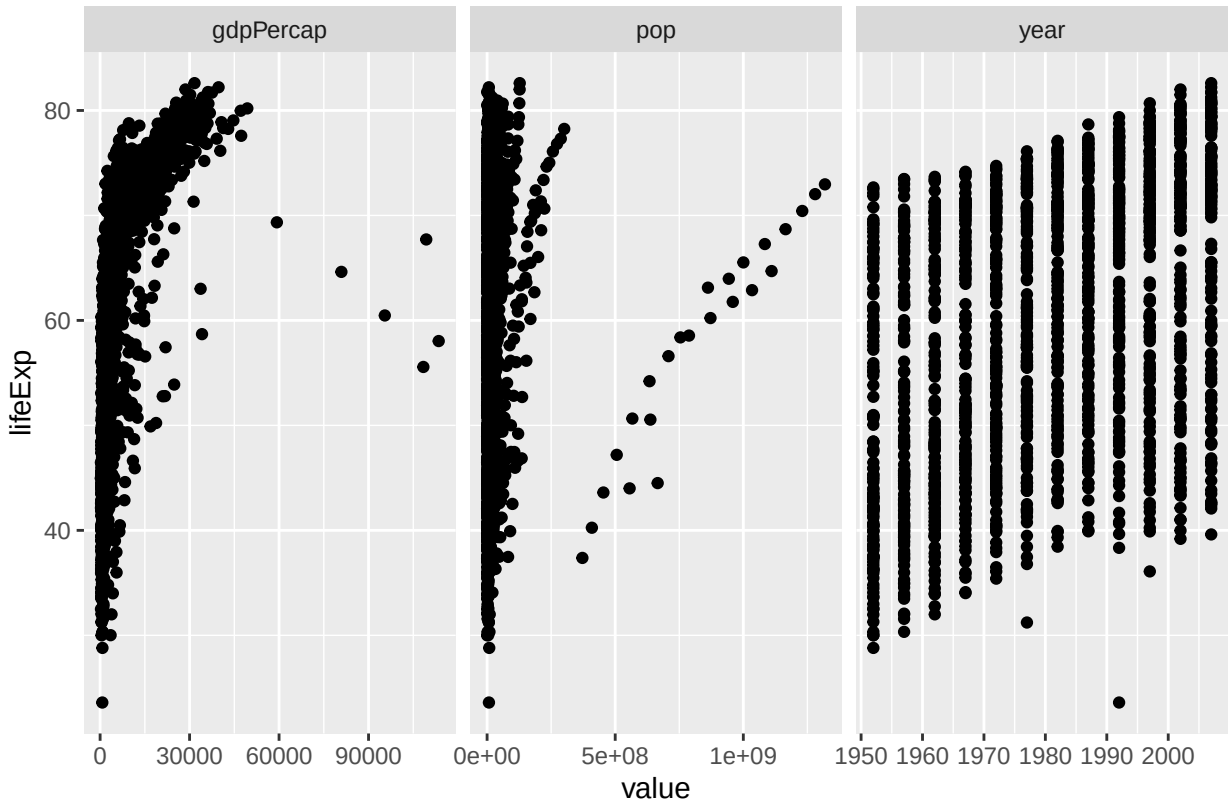
```
## 1 features with more than 20 categories ignored!
## country: 142 categories
```




```
# kontinu vs kategorikal
plot_boxplot(gapminder, by = "continent")
```




```
# kontinu vs kontinu
gapminder %>%
  select(year:gdpPercap) %>%
  plot_scatterplot(by = "lifeExp")
```



Referensi

- R For Data Science
- DataExplorer
- Tutorial R
- Skimr