Duomenų tvarkymas ir vizualizavimas

1. Įvadas į R. R objektai (vektoriai, matricos, „list“, „data.frame“). Duomenų tipai. Duomenų rinkinių apžvalga. Duomenų importavimas ir eksportavimas.
2. Duomenų apibendrinimas, apjungimas ir valdymas (dplyr, lapply, sapply ir kitos funkcijos). Trūkstamos reikšmės ir išskirtys duomenyse.
3. Požymių/kintamųjų kūrimas ir transformavimas. Duomenų pjūviai ir filtravimas. Atsitiktinių sekų generavimas. Github pažintis.
4. Atsiskaitymas iš – 1-3 temų.
5. Įvadas į duomenų vizualizavimą. Pagrindiniai paketai ir funkcijos (base, ggplot). Skirtingų duomenų tipų atvaizdavimas įvairiais grafikais (skaitinių, faktorinių, datos, laiko eilučių).

6. Tidyverse ekosistema

Figure 1: base R vs tidyverse

https://r4ds.had.co.nz/wrangle-intro.html

6.1 Duomenų analizės procesas

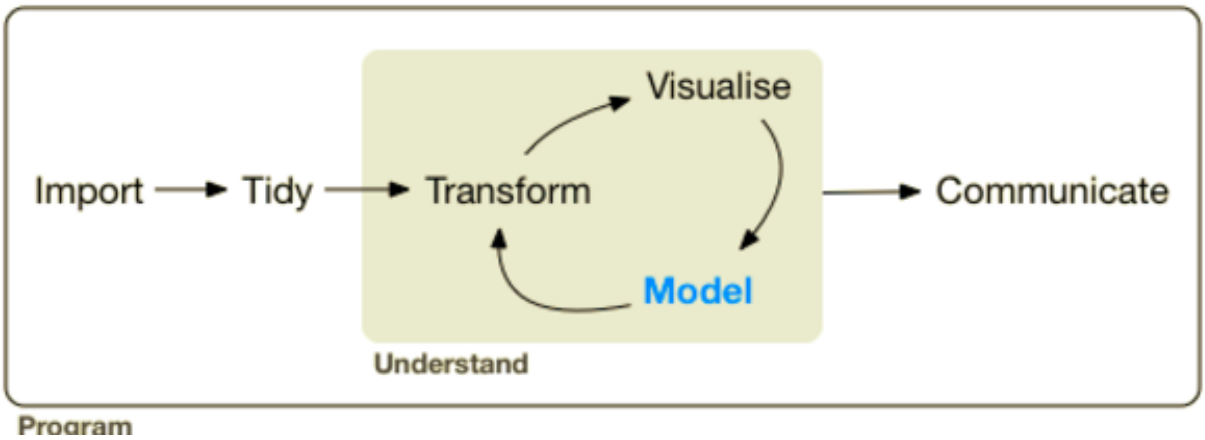


Figure 2: Duomenų analizės procesas - rstudio.com

Kas slepiasi po šia schema?

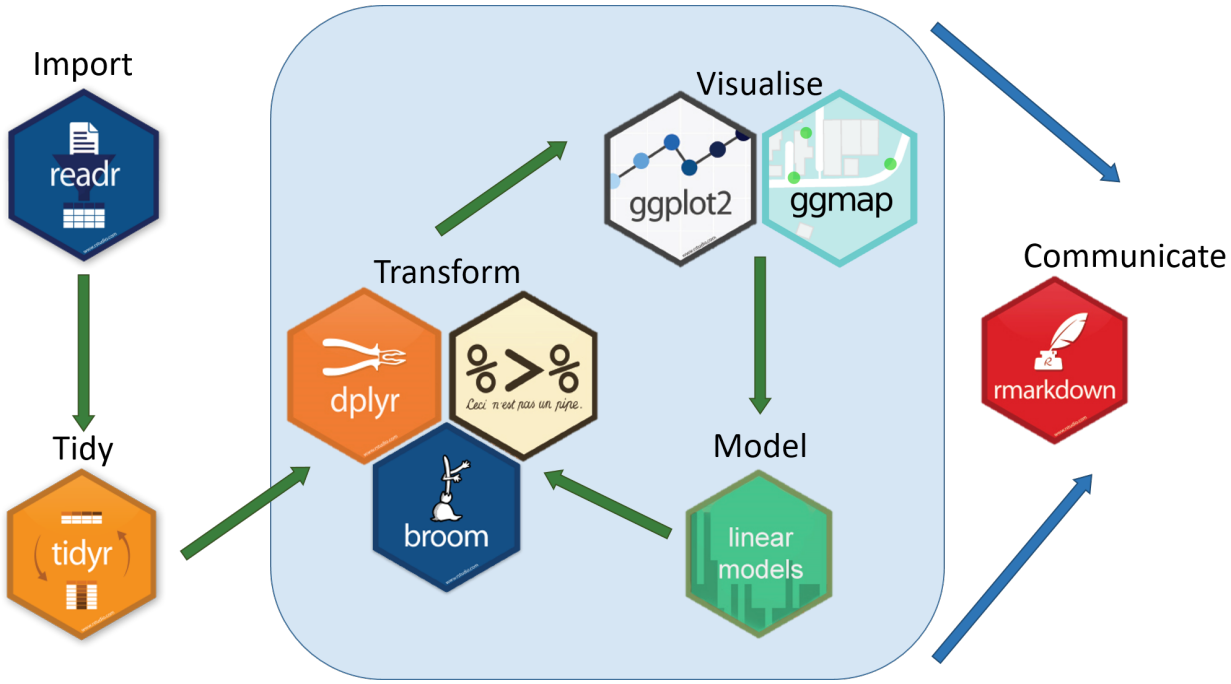


Figure 3: Paketai - teachdatascience.com

Neišsigąskite pranešimų apie konfliktus - tai nesuvienodintos sintaksės R kalboje pasekmė.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------ tidy
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Patarimas: jei norite būti tikri, jog naudojate funkciją iš norimo paketo, rašykite dplyr::filter (bus aktualu ateityje).

6.2 Duomenų nuskaitymas

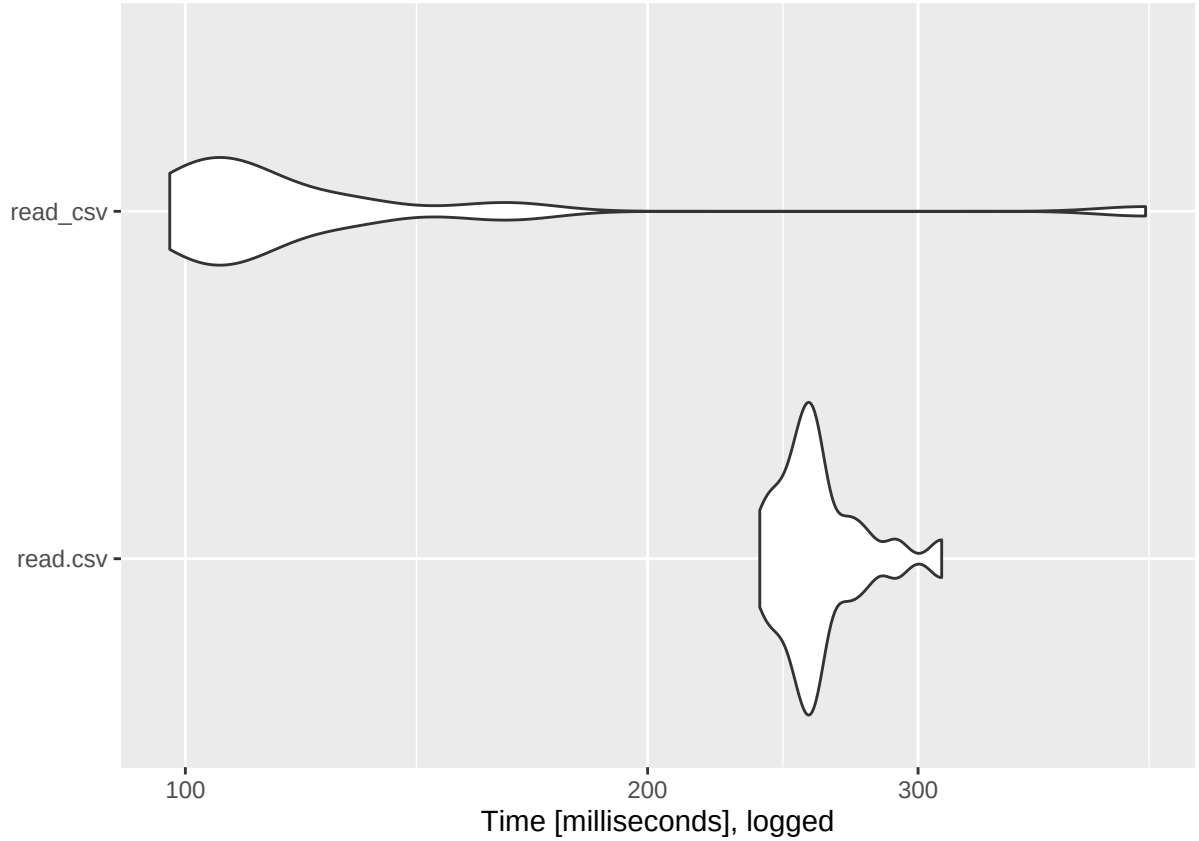
https://r4ds.had.co.nz/data-import.html

base R vs readr

Kodėl verta rinktis readr?

```
library(readr)
library(ggplot2)
library(microbenchmark)
```

```
results_small <- microbenchmark(
  read.csv = read.csv("Effects-of-COVID-19-on-trade-1-February-16-December-2020-provisional.csv"),
  read_csv = read_csv("Effects-of-COVID-19-on-trade-1-February-16-December-2020-provisional.csv"),
  times = 20
)
autoplot(object = results_small) +
  scale_y_log10() +
  labs(y = "Time [milliseconds], logged")
```



Pastaba: naudojant base R visuomet atkreipkite dėmesį į faktoriaus tipą - standartiškai R nori "padėti" vartotojui ir identifikavus kategorinį kintamąjį, jį automatiškai verčia į faktoriaus tipą.

6.2 Pipe operatorius

Pipe operatorius veikia panašiai į linux operatorių "|". R programavimo kalboje jis turi specialų simbolį "%>%". Tai nėra standartinė R funkcija - ji importuojama iš "magrittr" paketo.

Pavyzdys be pipe:

```
head(select(cars, speed), 2)
```

```
##   speed
## 1     4
## 2     4
```

Su pipe:

```
cars %>%
  select(speed) %>%
  head(2)
```

```
##   speed
## 1     4
## 2     4
```

Daugiau informacijos galite rasti čia: https://r4ds.had.co.nz/pipes.html

6.3 Duomenų transformavimas

SQL (angl. Structured Query Language „struktūrizuota užklausų kalba“) – kalba, skirta aprašyti duomenis ir manipuliuoti jais reliacinių duomenų bazių valdymo sistemose.

```
SELECT
    employee.id,
    employee.first_name,
    employee.last_name,
    AVG(DATEDIFF("SECOND", call.start_time, call.end_time)) AS call_duration_avg
FROM call
INNER JOIN employee ON call.employee_id = employee.id
GROUP BY
    employee.id,
    employee.first_name,
    employee.last_name
ORDER BY
    employee.id ASC;
```

6.3.1 dplyr

SQL funkcijos: SELECT, GROUP BY, ORDER, JOIN ir kitos turi atitikmenis R ekosistemoje. Tam naudosime “dplyr” paketą.

```
iris %>%
  select(Species, Sepal.Length) %>%
  group_by(Species) %>%
  summarise(avg_sepal_length = mean(Sepal.Length)) %>%
  arrange(avg_sepal_length)
```

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

```
## # A tibble: 3 x 2
##   Species    avg_sepal_length
##   <fct>                 <dbl>
## 1 setosa                 5.01
## 2 versicolor             5.94
## 3 virginica              6.59
```

Esminės funkcijos:

- select() - pasirinkti reikalingus stulpelius (kintamuosius)
- filter() - filtravimas pagal kintamųjų reikšmes
- mutate() - manipuliavimas reikšmėmis (esamo stulpelio reikšmių redagavimas arba naujo stulpelio sukūrimas)
- group_by() - vienodų reikšmių stulpelyje sugrupavimas
- summarise() - agregavimas (pvz. suma, vidurkis)
- arrange() - rikiavimas
- join() - lentelių apjungimas

Naudojant “dplyr” sintaksę ir pipe operatorių, labai patogu tiek atlikti transformacijas, tiek skaityti kodą. Taip pat galime rezultatą nukreipti į kitą objektą.

```
iris_avg <- iris %>% # naujas R objektas
  select(Species, Sepal.Length) %>% # kokius stulpelius norime pasirinkti analizei
  group_by(Species) %>% # grupuojame pagal kategorinį kintamąjį
  summarise(avg_sepal_length = mean(Sepal.Length)) %>% # ką norime suskaičiuoti (šiuo atveju - vidurkis
  arrange(avg_sepal_length) # rikiuojame didėjimo tvarka
```

Daugiau informacijos čia: https://r4ds.had.co.nz/relational-data.html

Praktinė užduotis: mtcars duomenys.

6.3.2 tidyr

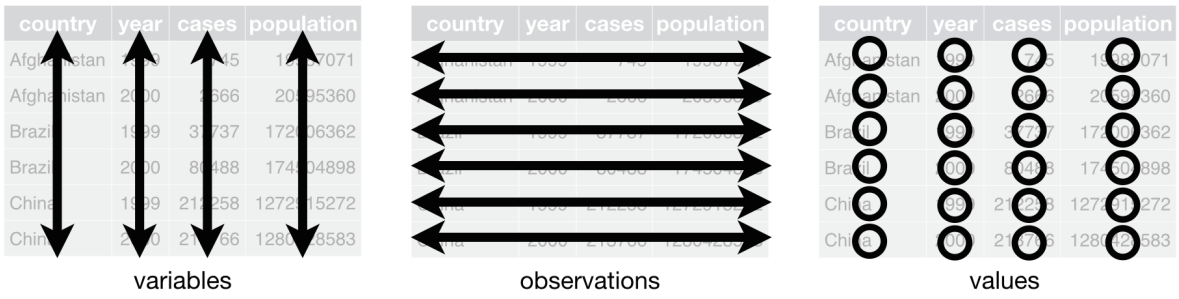


Figure 4: Tvarkingi duomenys

```
billboard
```

```
## # A tibble: 317 x 79
##    artist track date.entered   wk1   wk2   wk3   wk4   wk5   wk6   wk7   wk8
##    <chr>  <chr> <date>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 2 Pac  Baby~ 2000-02-26      87    82    72    77    87    94    99    NA
##  2 2Ge+h~ The ~ 2000-09-02      91    87    92    NA    NA    NA    NA    NA
##  3 3 Doo~ Kryp~ 2000-04-08      81    70    68    67    66    57    54    53
##  4 3 Doo~ Loser 2000-10-21      76    76    72    69    67    65    55    59
##  5 504 B~ Wobb~ 2000-04-15      57    34    25    17    17    31    36    49
##  6 98^0   Give~ 2000-08-19      51    39    34    26    26    19     2     2
##  7 A*Tee~ Danc~ 2000-07-08      97    97    96    95   100    NA    NA    NA
##  8 Aaliy~ I Do~ 2000-01-29      84    62    51    41    38    35    35    38
##  9 Aaliy~ Try ~ 2000-03-18      59    53    38    28    21    18    16    14
## 10 Adams~ Open~ 2000-08-26      76    76    74    69    68    67    61    58
## # ... with 307 more rows, and 68 more variables: wk9 <dbl>, wk10 <dbl>,
## #   wk11 <dbl>, wk12 <dbl>, wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>,
## #   wk17 <dbl>, wk18 <dbl>, wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>,
## #   wk23 <dbl>, wk24 <dbl>, wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>,
## #   wk29 <dbl>, wk30 <dbl>, wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>,
## #   wk35 <dbl>, wk36 <dbl>, wk37 <dbl>, wk38 <dbl>, wk39 <dbl>, wk40 <dbl>,
## #   wk41 <dbl>, wk42 <dbl>, wk43 <dbl>, wk44 <dbl>, wk45 <dbl>, wk46 <dbl>,
## #   wk47 <dbl>, wk48 <dbl>, wk49 <dbl>, wk50 <dbl>, wk51 <dbl>, wk52 <dbl>,
## #   wk53 <dbl>, wk54 <dbl>, wk55 <dbl>, wk56 <dbl>, wk57 <dbl>, wk58 <dbl>,
## #   wk59 <dbl>, wk60 <dbl>, wk61 <dbl>, wk62 <dbl>, wk63 <dbl>, wk64 <dbl>,
```

```
## #   wk65 <dbl>, wk66 <lgl>, wk67 <lgl>, wk68 <lgl>, wk69 <lgl>, wk70 <lgl>,
## #   wk71 <lgl>, wk72 <lgl>, wk73 <lgl>, wk74 <lgl>, wk75 <lgl>, wk76 <lgl>
```

```
billboard %>%
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank",
    values_drop_na = TRUE
  ) %>%
  head()
```

```
## # A tibble: 6 x 5
##   artist track                   date.entered week   rank
##   <chr>  <chr>                   <date>       <chr> <dbl>
## 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk1      87
## 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk2      82
## 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk3      72
## 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk4      77
## 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk5      87
## 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk6      94
```

```
fish_encounters %>%
  head()
```

```
## # A tibble: 6 x 3
##   fish  station  seen
##   <fct> <fct>   <int>
## 1 4842  Release     1
## 2 4842  I80_1       1
## 3 4842  Lisbon      1
## 4 4842  Rstr        1
## 5 4842  Base_TD     1
## 6 4842  BCE         1
```

```
fish_encounters %>%
  pivot_wider(names_from = station, values_from = seen)
```

```
## # A tibble: 19 x 12
##    fish  Release I80_1 Lisbon  Rstr Base_TD   BCE   BCW  BCE2  BCW2   MAE   MAW
##    <fct>   <int> <int>  <int> <int>   <int> <int> <int> <int> <int> <int> <int>
##  1 4842        1     1      1     1       1     1     1     1     1     1     1
##  2 4843        1     1      1     1       1     1     1     1     1     1     1
##  3 4844        1     1      1     1       1     1     1     1     1     1     1
##  4 4845        1     1      1     1       1    NA    NA    NA    NA    NA    NA
##  5 4847        1     1      1    NA      NA    NA    NA    NA    NA    NA    NA
##  6 4848        1     1      1     1      NA    NA    NA    NA    NA    NA    NA
##  7 4849        1     1     NA    NA      NA    NA    NA    NA    NA    NA    NA
##  8 4850        1     1     NA     1       1     1     1    NA    NA    NA    NA
##  9 4851        1     1     NA    NA      NA    NA    NA    NA    NA    NA    NA
## 10 4854        1     1     NA    NA      NA    NA    NA    NA    NA    NA    NA
## 11 4855        1     1      1     1       1    NA    NA    NA    NA    NA    NA
```

```
## 12 4857        1     1      1     1       1     1     1     1     1    NA    NA
## 13 4858        1     1      1     1       1     1     1     1     1     1     1
## 14 4859        1     1      1     1       1    NA    NA    NA    NA    NA    NA
## 15 4861        1     1      1     1       1     1     1     1     1     1     1
## 16 4862        1     1      1     1       1     1     1     1     1    NA    NA
## 17 4863        1     1     NA    NA      NA    NA    NA    NA    NA    NA    NA
## 18 4864        1     1     NA    NA      NA    NA    NA    NA    NA    NA    NA
## 19 4865        1     1      1    NA      NA    NA    NA    NA    NA    NA    NA
```

Daugiau informacijos čia: https://tidyr.tidyverse.org/articles/pivot.html

6.4 Laiko formatai su lubridate

Daugiau informacijos čia: https://r4ds.had.co.nz/dates-and-times.html

6.5 Darbas su tekstu/simboliais

Daugiau informacijos čia: https://r4ds.had.co.nz/strings.html

6.6 ggplot2

```
covid <- read_csv("Effects-of-COVID-19-on-trade-1-February-16-December-2020-provisional.csv")
```

```
## Parsed with column specification:
## cols(
##   Direction = col_character(),
##   Year = col_double(),
##   Date = col_character(),
##   Weekday = col_character(),
##   Current_Match = col_character(),
##   Country = col_character(),
##   Commodity = col_character(),
##   Transport_Mode = col_character(),
##   Measure = col_character(),
##   Value = col_double(),
##   Cumulative = col_double()
## )
```

```
covid %>%
  group_by(Country) %>%
  summarise(stebejimu_suma = n()) %>%
  arrange(desc(stebejimu_suma))
```

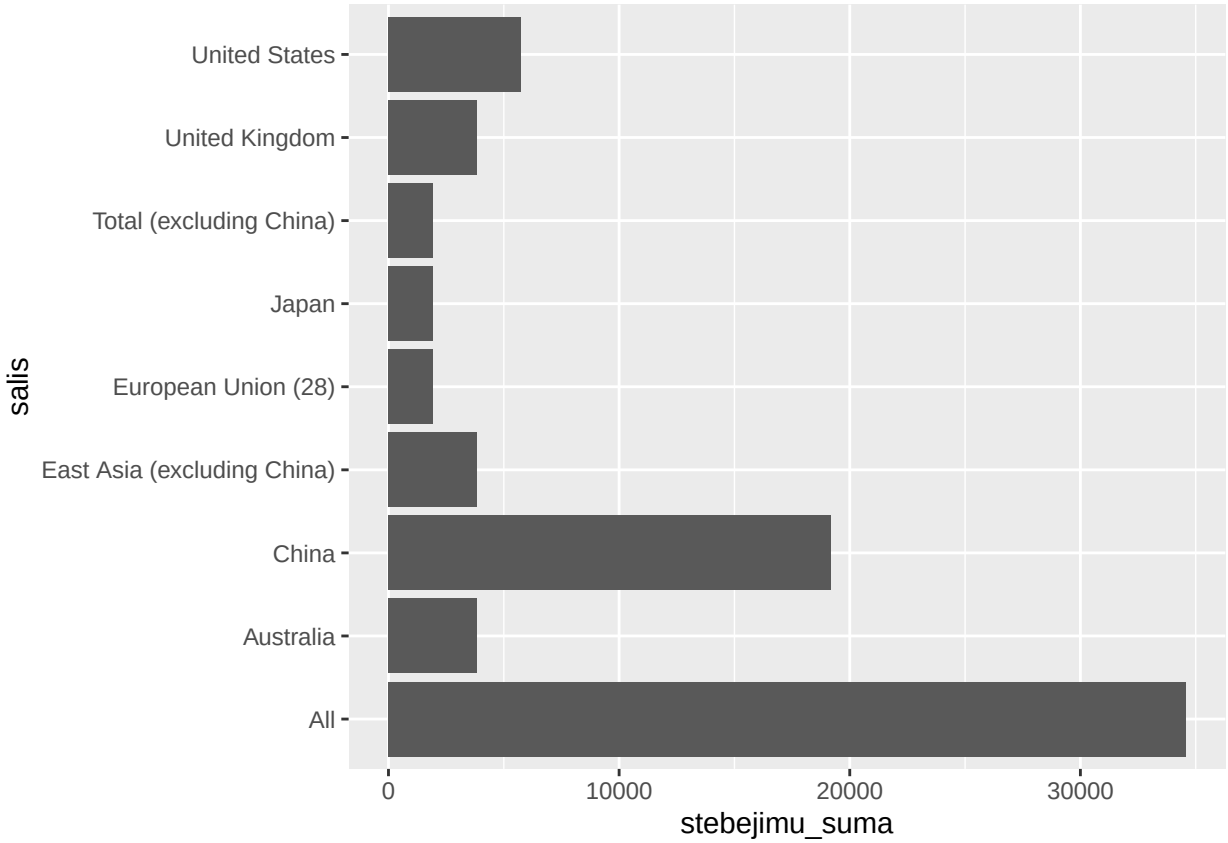
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 9 x 2
##   Country                     stebejimu_suma
##   <chr>                                <int>
## 1 All                                  34558
## 2 China                                19168
## 3 United States                         5760
## 4 Australia                             3840
## 5 East Asia (excluding China)           3840
## 6 United Kingdom                        3840
## 7 European Union (28)                   1920
## 8 Japan                                 1920
## 9 Total (excluding China)               1920
```

Galime sujungti duomenų apdorojimą ir grafiko atvaizdavimą į vieną ilgą išraišką, kurią patogu skaityti:

```
covid %>%
  group_by(Country) %>%
  summarise(stebejimu_suma = n()) %>%
  rename(salis = Country) %>%
  ggplot() +
    geom_bar(aes(y = salis, x = stebejimu_suma), stat  = "identity")
```

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



https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf

6.6 Praktika

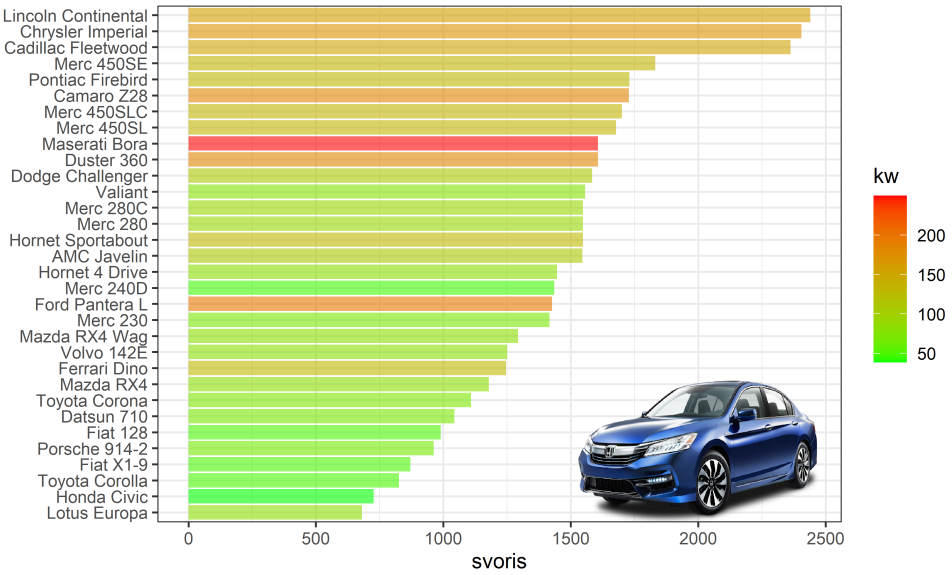


Figure 5: Su R galima piešti

7. Pirmasis Jūsų duomenų produktas - WEB aplikacija

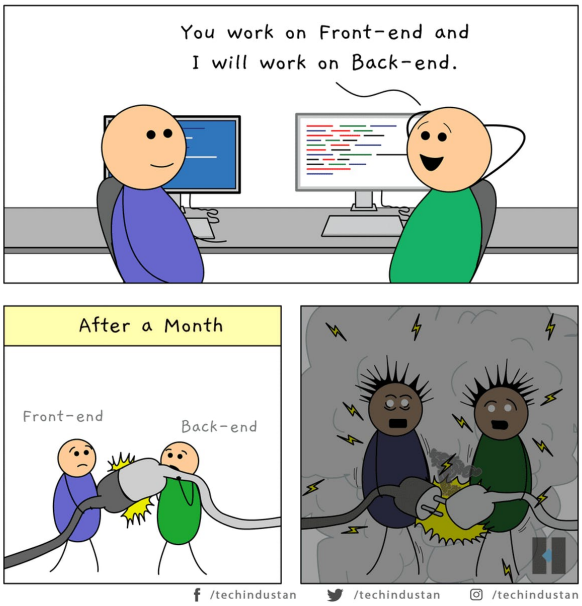


Figure 6: Tipinis WEB aplikacijos kūrimo procesas

7.1 Reaktyvaus programavimo paradigma

https://shiny.rstudio.com/articles/reactivity-overview.html

7.2 Įžanga į shiny

https://mastering-shiny.org/basic-app.html

```
library(shiny)
ui <- fluidPage(
  "Hello, world!"
)
server <- function(input, output, session) {
}
shinyApp(ui, server)
```

7.3 Hello world v2 aplikacija

```
#
# This is a Shiny web application. You can run the application by clicking
# the 'Run App' button above.
#
# Find out more about building applications with Shiny here:
#
#    http://shiny.rstudio.com/
#

library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30)
        ),

        # Show a plot of the generated distribution
        mainPanel(
           plotOutput("distPlot")
        )
    )
)

# Define server logic required to draw a histogram
server <- function(input, output) {

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}

# Run the application
shinyApp(ui = ui, server = server)
```

7.4 WEB aplikacijos publikavimas

1. Lokalus PC
2. Shiny server
3. https://www.shinyapps.io/
4. Konteineriai*

7.5 Idėjos Jūsų WEB aplikacijai

- https://www.r-graph-gallery.com/
- https://shiny.rstudio.com/gallery/
- https://dreamrs.shinyapps.io/memory-hex/
- https://nz-stefan.shinyapps.io/blog-explorer/#project
- https://sparktuga.shinyapps.io/ShinyDecisions/
- https://kneijenhuijs.shinyapps.io/
- https://rcharlie.shinyapps.io/sentify/

Papildoma literatūra

- https://r4ds.had.co.nz/
- https://ggplot2-book.org/
- https://ggplot2.tidyverse.org/
- https://rkabacoff.github.io/datavis
- https://mastering-shiny.org/