```
tle: "Duomenų analizės įvadas"
btitle: '3.1.. dalis - R programavimas'
thor: "Justas Mundeikis"
stitute: "VU EVAF"
te: "2019-05-08"
tput:
eamer_presentation:
includes:
in_header: header.txt
itor_options:
chunk_output_type: console
```

Turinys

Contents

Natūralūs vs apdoroti duomenys

Intro

Natūralūs duomenys (raw data) -> aprodojimo skriptas -> tvarkingi duomenys -> duomenų analizė -> komunikacija

Natūralūs vs apdoroti duomenys

Natūralūs duomenys:

- Paimti iš duomenų šaltinio
- Ne retai sunkiai pritaikomi analizei
- Duomenų analizė ne retai apima ir duomenų apdorojimą
- Natūralius duomenis gali reikėti apdoroti vieną ar kelis kartus wiki:Raw_data

Apdoroti duomenys:

- Duomenys paruošti analizei
- Apdorojimas gali apimti duomenų apjungimą, dalinimą, transformavimą etc.
- Priklausomai nuo aplinkybių, gali egzistuoti duomenų aprodojimo standartai
- Visi duomenų apdorojimo žingsniai turi būti dokumentuoti

Raw to tidy

1. Natūralūs duomenys (raw data set)
2. Tvarkingi duomenys (tidy data set)
3. *Code book* (meta doumenys) tvarkingiems duomenims
4. R Skriptas (1.-> 2.)

Tvarkingi duomenys

1. Vienas kintamasis - 1 stuleplis
2. Viena observacija - 1 eilutė
3. Vienam kintamajam - 1 lentelė
4. Daug lentelių gali būti sujungtos per vieną stulpelį
5. Pirma eilutė - žmonėms suprantami kintamųjų pavadinimai (Pajamos vs Pj)
6. Viena lentelė - vienas failas

Code book

1. Informacija apie kintamuosus, jų matavimo vienetus (gali nebūti apdorotuose duomenyse)
2. Informacija, kaip surinkti duomenys (apklausos metodai...)
3. Informacija, kaip apdoroti duomenys
4. Pageidautina .txt failas su markdown sintakse

Duomenų apdorojimo skriptas

- Duomenų apdrorojimo skriptas R, Python, Stata, SPSS
- Turi priimti natūralius duomenis
- Grąžinti tvarkingus duomenis
- Skripte neturėtų būti jokių nuo vartotojo priklausomų parametrų / nustatymų
- Jeigu skripte neįmanoma automatiškai atlikti visų veiksmų, būtina detaliai aprašyti (codebook + skripte komentavimo funckija)

Kas būna kai...

Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff

We replicate Reinhart and Rogoff (2010A and 2010B) and find that selective exclusion of available data, coding errors and inappropriate weighting of summary statistics lead to serious miscalculations that inaccurately represent the relationship between public debt and GDP growth among 20 advanced economies. Over 1946–2009, countries with public debt/GDP ratios above 90% averaged 2.2% real annual GDP growth, not 0.1% as published...

Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff

Darbinė direktorija

- `getwd()` ir `setwd()`
- relatyvus adresas `setwd("./data")`, `setwd("../data")`
- absoliutus adresas `setwd(c:/users/studentas/desktop/test/data)`
- nenaudokite absoliučių adresų
- INSTR: pakeisti darbinę direkrotiją į "Desktop"

Darbinė direktorija

- `file.exists("file_name")` testuoja ar egzistuoja failas arba direktorija
- `dir.create("folder_name")` sukuria direktoriją
- jeigu skriptas importuoja duomenis, o duomenys turi būti patalpinti atksiroje direktorijoje:

```
if(!file.exists("data")){
        dir.create("data")
}
```

paste()

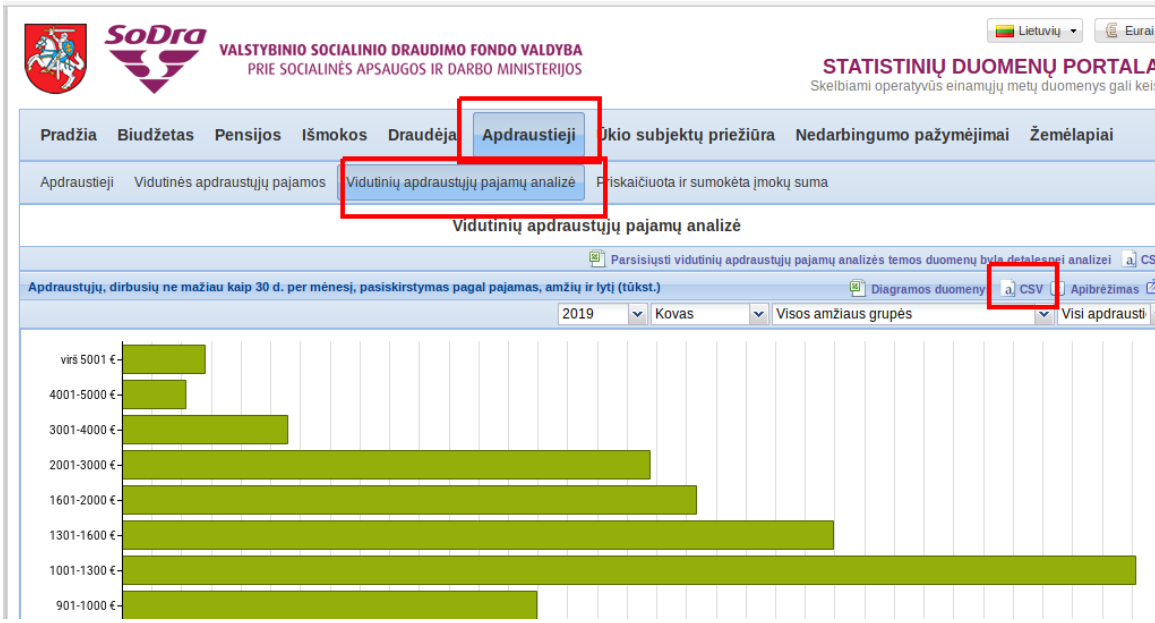
- `paste()` funkcija sujungia stringus / vectorius į vieną *character* string

```
times <-"Times"; x <- 3
paste("Hello", "World", x, times, sep=" ")
## [1] "Hello World 3 Times"
paste("Hello", "World", x, times , sep = "_")
## [1] "Hello_World_3_Times"
a <- c(1,2,3,4); b <- c("a", "b", "c")
paste(a,b, sep="&", collapse="%")
## [1] "1&a%2&b%3&c%4&a"
```

Duomenys iš interneto

Duomenys iš interneto

- `download.file(url, destfile, method, quiet = FALSE, mode = "w", cacheOK = TRUE, extra = getOption("download.file.extra"))`
- tinka duomenų failų parsiuntimui (.txt, .csv, etc)
- Keliaujam į http://atvira.sodra.lt/lt-eur/ - Apdraustieji - Vidutinių apdraustųjų pajamų analizė - CSV
- Nusikopijuojam url



Duomenys iš interneto

```
URL <- "http://atvira.sodra.lt/csv/lt-eur/apdraustieji_3_1.csv"
Sys.time()
```

```
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
filename <- paste("./data/apdraustieji_3_1_",
                          DownloadDate,
                          ".csv",
                          sep = "")
filename
# pastaba: method="curl" arba "wininet" kartais veikia geriau, svarbu išsibandyti
download.file(
        url = URL,
        destfile = filename,
        method = "auto")
```

Duomenys iš interneto

- Dabar visą zip failą “apdraustuju_pajamu_analize.zip”

```
URL <- "http://atvira.sodra.lt/downloads/lt-eur/apdraustuju_pajamu_analize.zip"
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
filename <- paste("./data/apdraustuju_pajamu_analize_",
                          DownloadDate,
                          ".zip",
                          sep = "")
filename
download.file(
        url = URL,
        destfile = filename,
        method = "auto")

# jeigu daugiau nei vienas failas, nurodyti tikslų failo pavadinimą
unzip(filename,
      exdir="data",
      files = "apdraustuju_pajamu_analize.csv")
```

Duomenys iš interneto

- sukuriamas `temp()` failas
- į jį nudownloadinamas failas
- iš temp failo išsiekstrahuojame reikiamą turinį
- kai nebereikia `unlink(...)` panaikina `temp()` failą

```
URL <- "http://atvira.sodra.lt/downloads/lt-eur/apdraustuju_pajamu_analize.zip"
temp <- tempfile()
download.file(url = URL, temp, method = "auto")
GYV_PAJ <- read.csv(unzip(temp,"apdraustuju_pajamu_analize.csv"),
                 header=TRUE,
                 sep=";",
                 stringsAsFactors = FALSE)
unlink(temp)
```

Duomenys iš interneto

```
median(GYV_PAJ$Mėnesio.pajamos)
list(lower_bound = 0.5*median(GYV_PAJ$Mėnesio.pajamos),
median= median(GYV_PAJ$Mėnesio.pajamos),
upper_bound = 2*median(GYV_PAJ$Mėnesio.pajamos))

maximum <- 4000
hist(GYV_PAJ$Mėnesio.pajamos[GYV_PAJ$Mėnesio.pajamos<=maximum],
     main="Histogram of declared labor income",
     xlab="Income grouped by 100",
     breaks=seq(0,maximum,100),
     xaxt="n")
axis(side=1, at=seq(0,maximum, 100), labels=seq(0,maximum,100))
```

Duomenys iš interneto

- 100% tikrumo, koks failo `encoding` neduos niekas
- Geras būdas, pabandyti atsidaryti su LibreOffice Calc, Sublime
- Mažiau tiksli alternatyva `guess_encoding` iš paketo `readr`
- google

```
#install.packages("readr")
library(readr)
guess_encoding("./data/apdraustuju_pajamu_analize.csv", n_max = 1000)
## # A tibble: 3 x 2
##   encoding   confidence
##   <chr>           <dbl>
## 1 ISO-8859-1       0.63
## 2 ISO-8859-2       0.41
## 3 ISO-8859-9       0.22
```

Duomenu nuskaitymas

- `read.table`
- `read.csv` , `read.csv2`, etc
- `readr` importavimo tool'sas (generuoja R kodą)

```
df <- read.table("./data/apdraustuju_pajamu_analize.csv",
                 header=TRUE,
                 sep=";",
                 fileEncoding = "ISO-8859-13",
                 stringsAsFactors = FALSE)
```

Galvos skausmas Excel formatai

- iš Sodros
- parsisiunčiam `.xlsx` failą

```
URL <-
"http://atvira.sodra.lt/downloads/lt-eur/apdraustuju_pajamu_analize.xlsx"
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
```

```
download.file(url = URL,
              destfile = paste("./data/apdraustuju_pajamu_analize_",
                               DownloadDate,
                               ".xlsx",
                               sep = ""),
              method = "auto")
```

Galvos skausmas Excel formatai

- `readxl` paketas
- `xlsx` paketas
- alternatyva atsidaryti su Excel, išsaugoti kaip `.csv`, importuoti kaip `.csv`

```
#install.packages("readxl")
library(readxl)
# su Excel pasitikriname, kurį sheet importuosime
df2 <- read_excel(".data/apdraustuju_pajamu_analize_2019_05_08.xlsx",
                  sheet = "Duomenys")
str(df2)
```

Galvos skausmas Excel formatai

- `write_excel_csv()` su `readr` paketu
- `write.xlsx` su `openxlsx` paketu
- SVARBU! Excel neatidaro daugiau nei
- 1 048 57 eilučių ir 16 384 stulpelių

```
install.packages("openxlsx")
library(openxlsx)
l<-list(iris=iris, mtcars=mtcars, quakes=quakes)
write.xlsx(l, file = "./data/datasets.xlsx")
```

LSD

- LSD sukelia daug galvos skausmų, tačiau LSD turi API
- "RESTful žiniatinklio paslaugos" *`rsdmx` paketas sutvarko LSD failus

```
#install.packages("rsdmx")
library(rsdmx)
```

LSD

Pavyzdžiai: * kai norima gauti duomenų rinkinių sąrašą * https://osp-rs.stat.gov.lt/rest_xml/dataflow/ * kai norima gauti konkretaus duomenų rinkinio apibrėžimą * https://osp-rs.stat.gov.lt/rest_xml/dataflow/LSD/S3R629_M3010217 * kai norima gauti duomenų struktūros apibrėžimą * https://osp-rs.stat.gov.lt/rest_xml/datastructure/lsd/M3010217/

```
#metadata
url_meta <- "https://osp-rs.stat.gov.lt/rest_xml/dataflow/"

meta <- readSDMX(url_meta)
```

```
meta <- as.data.frame(meta)

write.csv(meta, "./data/meta.csv")
write.xlsx(meta, file = "./data/meta.xlsx")
```

LSD

- išsirinkus kokio kintamojo reikia

```
# S3R838 - Deaths by cause of death
# Age (5 year groups) | Causes of death (27) | Sex (2000 - 2016)
S3R838_M3010608<- readSDMX(providerId = "LSD",
                           resource = "data",
                           flowRef = "S3R838_M3010608",
                           dsd = TRUE)
## -> Fetching 'https://osp-rs.stat.gov.lt/rest_xml/data/S3R838_M3010608/all/'
## -> DSD ref identified in dataset = 'M3010608'
## -> Attempt to fetch & bind DSD to dataset
## -> Fetching 'https://osp-rs.stat.gov.lt/rest_xml/datastructure/all/M3010608/latest/?references=child
## -> DSD fetched and associated to dataset!
S3R838_M3010608 <- as.data.frame(S3R838_M3010608 ,
                                 labels = TRUE)
```

RSDMX

- padeda imporuoti ir duomenis iš OECD, Eurostat ir kitų šaltinių, kuriuose duomenys pateikiami XML formatu
- žr. https://github.com/opensdmx/rsdmx

Eurostat

- `eurostat` paketas
- google "eurostat cheatsheet"
- einam į Eurostat database išsirinkt duomenų...

```
#install.packages("eurostat")
library(eurostat)
nama_10_gdp <- get_eurostat("nama_10_gdp", stringsAsFactors = FALSE)
```

Duomenų inspektavimas

Inspection

```
head(df,2)
##   Metai Menuo Amžius   Lytis Menesio.pajamos
## 1  2019     3     39 Moteris         4124.80
## 2  2019     3     55 Moteris          861.56
tail(df,2)
##         Metai Menuo Amžius   Lytis Menesio.pajamos
```

```
## 1108113  2019     3     36   Vyras         1528.95
## 1108114  2019     3     65 Moteris          555.00
```

Inspection

```
str(df)
## 'data.frame':    1108114 obs. of  5 variables:
##  $ Metai          : num  2019 2019 2019 2019 2019 ...
##  $ Menuo          : num  3 3 3 3 3 3 3 3 3 3 ...
##  $ Amžius         : num  39 55 34 68 53 30 57 39 56 69 ...
##  $ Lytis          : chr  "Moteris" "Moteris" "Vyras" "Vyras" ...
##  $ Menesio.pajamos: num  4125 862 1463 1290 561 ...
```

Inspection

```
summary(df)
##      Metai          Menuo       Amžius           Lytis
##  Min.   :2019   Min.   :3   Min.   :   1.00   Length:1108114
##  1st Qu.:2019   1st Qu.:3   1st Qu.:  33.00   Class :character
##  Median :2019   Median :3   Median :  45.00   Mode  :character
##  Mean   :2019   Mean   :3   Mean   :  44.23
##  3rd Qu.:2019   3rd Qu.:3   3rd Qu.:  55.00
##  Max.   :2019   Max.   :3   Max.   :1824.00
##  Menesio.pajamos
##  Min.   :     0.16
##  1st Qu.:   645.13
##  Median :   965.82
##  Mean   :  1250.22
##  3rd Qu.:  1487.60
##  Max.   :178142.36
```

Inspection

```
quantile(df$Mėnesio.pajamos)
##        0%       25%       50%       75%      100%
##      0.16    645.13    965.82   1487.60 178142.36
quantile(df$Mėnesio.pajamos, probs = seq(0,1,0.1))
##         0%        10%        20%        30%        40%        50%
##      0.160    535.620    585.000    710.400    807.730    965.820
##        60%        70%        80%        90%       100%
##   1145.000   1354.221   1655.000   2208.571 178142.360
table(df$Lytis)
##
## Moteris   Vyras
##  573129  534985

sum(is.na(df$Mėnesio.pajamos))
## [1] 0
```

Inspection

```
table(df$Lytis, df$Amžius)
```

Subsetting

- užsiloadinam dataframe "mtcars"

```
df <- mtcars
df[c(1:3),]
df[c(1:3),c("mpg", "cyl", "hp")]
```

Subsetting

- užsiloadinam dataframe "mtcars"

```
df <- mtcars
df[(df$mpg>=10 &
       df$mpg<=20& df$hp>=250 | df$qsec <=16), ]
##                  mpg cyl disp  hp drat   wt  qsec vs am gear carb
## Duster 360      14.3   8  360 245 3.21 3.57 15.84  0  0    3    4
## Camaro Z28      13.3   8  350 245 3.73 3.84 15.41  0  0    3    4
## Ford Pantera L 15.8   8  351 264 4.22 3.17 14.50  0  1    5    4
## Ferrari Dino    19.7   6  145 175 3.62 2.77 15.50  0  1    5    6
## Maserati Bora   15.0   8  301 335 3.54 3.57 14.60  0  1    5    8
```

Subsetting

- `which()` grąžina skaitinų vektorių priimdamas loginį vektorių

```
a <- c(1,NA,20,NA,40)
which(is.na(a)) #
## [1] 2 4
which(!is.na(a))
## [1] 1 3 5
a[which(!is.na(a))]
## [1]  1 20 40

which(df$mpg<=14)
## [1] 15 16 24
df[which(df$mpg<=14),1:3]
##                      mpg cyl disp
## Cadillac Fleetwood  10.4   8  472
## Lincoln Continental 10.4   8  460
## Camaro Z28          13.3   8  350
```

Sorting

- `sort()` sortiruoja vektorius arba df
- tačiau veikia tik su vienu kintamuoju
- 2+ kintamųjų -> `order()`

```
a <- c(1,NA,20,NA,40)
sort(a)
## [1]  1 20 40
sort(a, decreasing = TRUE)
## [1] 40 20  1
sort(a, decreasing = TRUE, na.last = TRUE)
## [1] 40 20  1 NA NA
head(df[order(df$mpg,df$hp),],3)
##                      mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Cadillac Fleetwood  10.4   8  472 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8  460 215 3.00 5.424 17.82  0  0    3    4
## Camaro Z28          13.3   8  350 245 3.73 3.840 15.41  0  0    3    4
```

Naujų stulpelių sukūrimas

- `ifelse(condition, value_true, value_false)`
- priima vektorius! `if(){}else{}` vertina tik pirmą elementą

```
mean(df$mpg)
## [1] 20.09062
df$loginis <- ifelse(df$mpg>=mean(df$mpg),"daugiau","mažiau")
head(df,3)
##                mpg cyl disp  hp drat    wt  qsec vs am gear carb loginis
## Mazda RX4     21.0   6  160 110 3.90 2.620 16.46  0  1    4    4 daugiau
## Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4 daugiau
## Datsun 710    22.8   4  108  93 3.85 2.320 18.61  1  1    4    1 daugiau

# alternatyva su cbind
# df <- cbind(df, loginis=ifelse(...)
```

Cross tabs

- Nusiskaitom kitą `df` ir `df2` priskiriame "susiaurintą" df versiją
- Linuxe ir MAC

```
df <- read.csv("data/apdraustieji_3_1_2019_05_08.csv",
                header=TRUE,
                sep=";",
                stringsAsFactors = FALSE)

df2 <- df[(df$Metai==2018 &
                  df$Mėnesio.pajamos=="401-450 €"&
                  df$Amžius=="Visos amžiaus grupės"&
                  df$Lytis!="Visi apdraustieji"), ]
```

Cross tabs

- Create a contingency table (optionally a sparse matrix) from cross-classifying factors, usually contained in a data frame, using a formula interface.

```
table <- xtabs(data=df2,
               Apdraustųjų.skaičius ~ Lytis + Mėnuo,
```

```
                drop.unused.levels = TRUE)
table
```

Cross tabs

- Create a contingency table (optionally a sparse matrix) from cross-classifying factors, usually contained in a data frame, using a formula interface.

```
table <- xtabs(data=df2,
                Apdraustųjų.skaičius ~ Lytis + Mėnuo,
                drop.unused.levels = TRUE)
barplot(table)
```

Cross tabs

```
men <- c(Sausis=1,Vasaris=2,Kovas=3,
         Balandis=4,Gegužė=5,Birželis=6,
         Liepa=7,Rugpjūtis=8,Rugsėjis=9,
         Spalis=10,Lapkritis=11,Gruodis=12)
df2$men <- men[df2$Mėnuo]
table <- xtabs(data=df2,
                Apdraustųjų.skaičius ~ Lytis + men,
                drop.unused.levels = TRUE)
```

Cross tabs

```
barplot(table)
```

Cut

- `cut()` - cut divides the range of x into intervals and codes the values in x according to which interval they fall.
- `cut()` - sukuria faktorius

```
# cut(GYV_PAJ$Mėnesio.pajamos)
table(cut(GYV_PAJ$Mėnesio.pajamos, breaks=quantile(GYV_PAJ$Mėnesio.pajamos)))

#install.packages("Hmisc")
library(Hmisc)
table(cut2(GYV_PAJ$Mėnesio.pajamos, g=4))
```

Cut

- `cut2()` iš `Hmisc`
- sukuriame naują stulelį su kvantiliodydžio
- bei kitą stulpelį su kvantilio numer išnauydojant `as.numeric`

```
library(Hmisc)
GYV_PAJ$fact <-cut2(GYV_PAJ$Mėnesio.pajamos, g=4)
GYV_PAJ$fact_num <-as.numeric(cut2(GYV_PAJ$Mėnesio.pajamos,, g=4))
```

Kitos bazinės funkcijos

- `abs(x)` absoliuti vertė
- `sqrt(q)` šaknis
- `ceiling(x)` suapvalinimas į viršų
- `floor(x)` suapvalinimas žemyn
- `round(x, digits=n)` suapvalinimas iki n ženklų
- `sin(x)`, `cos(x)`, `tan(x)`
- `log(x)`, `log2(x)`, `log10(x)`
- `exp(x)` e^x

reshape2 - melt

```
library(reshape2)

mtcars$car_name <- rownames(mtcars)
mtcars_melt <- melt(mtcars, id=c("car_name", "gear", "cyl"), measure.vars = c("mpg", "hp"))
```

reshape2 - melt

```
head(mtcars_melt,3)
##        car_name gear cyl variable value
## 1     Mazda RX4    4   6      mpg  21.0
## 2 Mazda RX4 Wag    4   6      mpg  21.0
## 3    Datsun 710    4   4      mpg  22.8
tail(mtcars_melt,3)
##         car_name gear cyl variable value
## 62  Ferrari Dino    5   6       hp   175
## 63 Maserati Bora    5   8       hp   335
## 64    Volvo 142E    4   4       hp   109
```

Cast

- Use `acast` or `dcast` depending on whether you want vector/matrix/array output or data frame output. Data frames can have at most two dimensions.

```
# length = count
dcast(mtcars_melt, cyl~variable)
## Aggregation function missing: defaulting to length
##   cyl mpg hp
## 1   4  11 11
## 2   6   7  7
## 3   8  14 14
acast(mtcars_melt, cyl~variable)
## Aggregation function missing: defaulting to length
```

```
##   mpg hp
## 4  11 11
## 6   7  7
## 8  14 14
```

Cast

```
dcast(mtcars_melt, cyl~variable, mean)
##   cyl      mpg        hp
## 1   4 26.66364  82.63636
## 2   6 19.74286 122.28571
## 3   8 15.10000 209.21429
dcast(mtcars_melt, gear~variable, mean)
##   gear      mpg       hp
## 1    3 16.10667 176.1333
## 2    4 24.53333  89.5000
## 3    5 21.38000 195.6000
```

tapply

```
tapply(mtcars$mpg,mtcars$cyl,mean)
##        4        6        8
## 26.66364 19.74286 15.10000
tapply(mtcars_melt$value, mtcars_melt$variable, mean)
##       mpg        hp
##  20.09062 146.68750
```

merge

```
set.seed(101)
a <- data.frame(id=sample(1:10), name=sample(letters[1:10],10), val1=sample(3,10,replace = T))
b <- data.frame(id=sample(1:10), name=sample(letters[1:10],10), val2=sample(3,10,replace = T))
a
b
```

merge

```
names(a)
names(b)
intersect(names(a), names(b))
merge(a,b)
merge(a,b,all = TRUE)
merge(a,b,by.x="id", by.y = "id")
```

dplyr

dplyr

- Sukurtas Hadley Wickham @Rstudio
- pagerinta `plyr` paketo versija
- pagerina naudojimąsi R
- veikia labai greitai, nes DF aprodojimas perkoduotas į `C++`

dplyr

- select
- filter
- arrange
- rename
- mutate
- summarize
- pipe `%>%`

select

```
library(dplyr)

head(nama_10_gdp)
head(select(nama_10_gdp, 1:3))
head(select(nama_10_gdp, unit, geo, values))
select(nama_10_gdp, -(1:3))
```

filter

```
filter(nama_10_gdp, geo=="LT")
df <- filter(nama_10_gdp, geo=="LT" &
               na_item=="B1G" &
               unit=="CLV10_MEUR"&
               time>="2000-01-01")
plot(df$values, type="l")
```

arrange

```
df <- arrange(df, time)
plot(df$values, type="l")

#df <- arrange(df, desc(time))
#plot(df$values, type="l")
```

rename

- new_name=old_name

```
df <- rename(df, rodiklis=na_item)
head(df)
```

mutate

```
df <- mutate(df, mean=mean(df$values))
head(df)
plot(df$values, type="l")
lines(df$mean)
```

summarize

```
df <- filter(nama_10_gdp,
             na_item=="B1G" &
             unit=="CLV10_MEUR"&
             time>="2000-01-01")

df_g <- group_by(df, geo)

summarise(df_g, mean=mean(values), median=median(values))
```

piping

- %>%

```
df <- nama_10_gdp %>%
       filter(geo=="LT" &
                na_item=="B1G" &
                unit=="CLV10_MEUR"&
                time>="2000-01-01")%>%
        select(time, values) %>%
        mutate(mean_LT=mean(values))%>%
        arrange(time)

plot(df$values, col="red", type="l")
lines(df$mean_LT, col="blue")
```