Duomenų analizės įvadas

3.1.. dalis - R programavimas

Justas Mundeikis

VU EVAF

2019-05-01

Turinys

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

Darbinė direktorija

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

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")
}
```

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

```
URL <- "http://atvira.sodra.lt/csv/lt-eur/apdraustieji_3_1.csv"
Sys.time()
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
download.file(
        url = URL,
        destfile = paste("./data/apdraustieji_3_1_",
                         DownloadDate,
                         ".csv",
                         sep = ""),
        method = "curl")
```

Duomenys iš interneto

- Dabar visą zip failą “apdraustuju_pajamu_analize.zip”

```
URL <- "http://atvira.sodra.lt/downloads/lt-eur/apdraustuju_pajamu_analize.
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
download.file(
        url = URL,
        destfile = paste("./data/apdraustuju_pajamu_analize_",
                         DownloadDate,
                         ".zip",
                         sep = ""),
        method = "curl")
unzip("./data/apdraustuju_pajamu_analize_2019_05_01.zip",
      exdir="./data/",
      files = "apdraustuju_pajamu_analize.csv") #jeigu daugiau nei vienas f
```

Duomenys iš interneto

- sukuriamas `temp()` failas
- į jį nudownloadinamas failas, kai nebereikia `unlink(...)`

```
URL <- "http://atvira.sodra.lt/downloads/lt-eur/apdraustuju_pajamu_analize.
temp <- tempfile()
download.file(url = URL, temp, method = "curl")
GYV_PAJ <- read.csv(unzip(temp,"apdraustuju_pajamu_analize.csv"),
                    header=TRUE,
                    sep=";",
                    fileEncoding = "ISO-8859-13",
                    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
- google

```
#install.packages("readr")
library(readr)
guess_encoding("./data/apdraustieji_3_1_2019_05_01.csv", n_max = 1000)
## # A tibble: 2 x 2
##   encoding     confidence
##   <chr>             <dbl>
## 1 windows-1252       0.26
## 2 windows-1250       0.21
guess_encoding("./data/data-table.csv", n_max = 1000)
## # A tibble: 3 x 2
##   encoding     confidence
##   <chr>             <dbl>
## 1 UTF-8                1
## 2 windows-1252         0.32
```

Duomenu nuskaitymas

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

```
?read.table
df <- read.table("./data/apdraustieji_3_1_2019_05_01.csv",
                 header=TRUE,
                 sep=";",
                 fileEncoding = "ISO-8859-13",
                 stringsAsFactors = FALSE)
head(df)
tail(df)
str(df)
```

Galvos skausmas Excel formatai

- iš Sodros
- parsipūčiam .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 = "curl")
```

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_01.xlsx",
                  sheet = "Duomenys")
```

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 *info

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

XML formatas

- XML - Extensible markup language
- Dažnai naudojama internetinių duomenų formavimui
- Išgaunamas dažniausiai *web scraping* būdu
- Susideda iš
- Markup - labels, nustatančiais struktūrą (*Tags*, *attributes*)
- Content - turiniu (tarp TAGS)
- https://en.wikipedia.org/wiki/XML

XML formatas

- http://finmin.lrv.lt/lt/kontaktai –> Atviri duomenys
- XML nuoroda
- atsidarome su Sublime

```
DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
URL <-
"http://finmin.lrv.lt/lt/kontaktai/exportPublicData?export_data_type=xml&do
download.file(url = URL,
              destfile = paste("./data/finmin_kontaktai_",
                               DownloadDate,
                               ".xml",
                               sep = ""),
              method = "curl")
```

XML formatas

- XML paketas

```
install.packages("XML")
library(XML)

doc <-xmlTreeParse(URL, isURL = TRUE)

root_node <- xmlRoot(doc)
xmlName(root_node)
names(root_node)
root_node[[1]]
root_node[[1]][[1]]
xmlSApply(root_node,xmlValue)
xmlSApply(root_node,xmlChildren)
```

JSON

- JSON
- dažnai naudojmas API(*Application programming interface*) formatas
- `jsonlite` paketas

LSD

- LSD sukelia daug galvos skausmų, tačiau LSD turi API *rsdmx* paketas sutvarko LSD failus

```
library(rsdmx)

#metadata
url_meta <- "https://osp-rs.stat.gov.lt/rest_xml/dataflow/"
meta <- readSDMX(url_meta)
meta <- as.data.frame(meta)

DownloadDate <- format(Sys.time(), format="%Y_%m_%d")
write.csv(meta, paste("./data/meta_",
                      DownloadDate,
                      ".csv",
                      sep = ))
```

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)
S3R838_M3010608 <- as.data.frame(S3R838_M3010608 ,
                                 labels = TRUE)
```

RSDMX

- padeda imporuoti ir duomenis iš OECD
- žr. Github

Eurostat

- `eurostat` paketas
- google “eurostat cheatsheet”

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

Inspection

```
head(df)
tail(df)
str(df)
summary(df)
quantile(df$Apdraustųjų.skaičius)
quantile(df$Apdraustųjų.skaičius, probs = seq(0,1,0.1))
table(df$Mėnesio.pajamos)
table(df$Metai, df$Mėnuo)
sum(is.na(df$Apdraustųjų.skaičius))
table(df$Mėnesio.pajamos=="virš 5001 \u0080")
```

Subsetting

```
df[c(1:5),]
df[c(1:5),c("Amžius", "Apdraustųjų.skaičius")]
df[(df$Metai==2019 &
             df$Mėnuo=="Sausis"&
             df$Amžius=="Visos amžiaus grupės"&
             df$Lytis=="Visi apdraustieji"), ]


df[(df$Metai>=2010 &
             df$Mėnuo=="Sausis"&
             (df$Mėnesio.pajamos=="iki 400 \u0080 (MMA)" |
                      df$Mėnesio.pajamos=="iki 555 \u0080 (MMA)") &
             df$Amžius=="Visos amžiaus grupės"&
             df$Lytis=="Visi apdraustieji"), ]
```

Subsetting

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

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

df[which(df$Apdraustųjų.skaičius>300),]
```

Sorting

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

```
a <- c(1,NA,20,NA,40)
sort(a)
sort(a, decreasing = TRUE)
sort(a, decreasing = TRUE, na.last = TRUE)
df[order(df$Metai, df$Mėnuo),]
```

Naujų stulpelių sukūrimas

- ifelse(condition, value_true, value_false)

```
df2 <- df[(df$Metai==2018 &
                  df$Mėnesio.pajamos=="401-450 \u0080"&
                  df$Amžius=="Visos amžiaus grupės"&
                  df$Lytis=="Visi apdraustieji"), ]

mean(df2$Apdraustųjų.skaičius)

df2$log <-
ifelse(df2$Apdraustųjų.skaičius>=mean(df2$Apdraustųjų.skaičius), 1,0)
# alternatyva
df2 <-
cbind(df2,
new=ifelse(df2$Apdraustųjų.skaičius>=mean(df2$Apdraustųjų.skaičius),1,0))
```

Cross tabs

```
df2 <- df[(df$Metai==2018 &
                    df$Mėnesio.pajamos=="401-450 \u0080"&
                    df$Amžius=="Visos amžiaus grupės"&
                    df$Lytis!="Visi apdraustieji"), ]

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]
# kur problema?
df2$men <- men[as.character(df2$Mėnuo)]
plot(df2$men, df2$Apdraustųjų.skaičius,
     ylim=c(0,105),
     type="l",
     col="blue")
plot(df2$men, df2$Apdraustųjų.skaičius-mean(df2$Apdraustųjų.skaičius),
     type="l",
     col="blue")
```

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(df$Apdraustųjų.skaičius, breaks=4)
table(cut(df$Apdraustųjų.skaičius, breaks=quantile(df$Apdraustųjų.skaičius)

library(Hmisc)
table(cut2(df$Apdraustųjų.skaičius, g=4))
```

Cut

- cut2() iš Hmisc

```
library(Hmisc)
df2$fact <-cut2(df2$Apdraustųjų.skaičius, g=4)
df2$fact_num <-as.numeric(cut2(df2$Apdraustųjų.skaičius, 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

```
library(reshape2)
#View(mtcars)
head(mtcars,4)
##                mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4     21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710    22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive 21.4  6  258 110 3.08 3.215 19.44  1  0    3    1
```

reshape2 - melt

- `measure.vars` = sukuria factor

```
mtcars$car_name <- rownames(mtcars)

mtcars_melt <- melt(mtcars, id=c("car_name", "gear", "cyl"), measure.vars =

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
b <- data.frame(id=sample(1:10), name=sample(letters[1:10],10), val2=sample
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 %>%

dplyr

```
library(dplyr)
dim(nama_10_gdp)
str(nama_10_gdp)
names(nama_10_gdp)
```

select

```
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")
```