Seminar 2

Justas Mundeikis

2019-04-04

Turinys

1 Apie seminarą

Seminaras nėra skirtas vien tam, jog studentai pasikartotų paskaitų metu įgytas žinias, bet jog ir būtų skatinami savarankiškai gilintis į studijuojamą dalyką. Todėl labai tikėtina, jog sprendžiant seminaro užduotis, studentai turės ne tik dar kartą peržiūrėti paskaitų skaidres, bet ir panaršyti literatūroje / internete, beieškant tinkamų sprendimų. Kaip dėstytojas rekomenduočiau taip spręsti seminaro užduotis:

- Pabandyti išspręsti visas užduotis vieniems
- Sulyginti savo užduotis su pateiktais rezultatais
- Sudaryti mokslo grupę (2-6 studentai)
- Sulyginti savo rezultatus. Išsiaiškinti taprusavyje, kodėl kas kaip sprendė. Labai svarbu suprasti, jog R'e visada bus galima rasti `n+1` galimą sprendimą tam pačiam uždaviniui atlikti. Rezultatas bus tas pats, bet kelias pasirinktas kitas. Svarbiausia bandyti turint sprendimą ieškoti būdų, kaip "optimizuoti" kodą, jog R galėtų veikti kuo sparčiau.

Seminaro užduočių sudėtingumas yra santykinai propocionalus testų / egzamino sudėtingumui, todėl labai svarbu, jog sprendžiant pavieniui ar grupėse, studentai suprastų ***kodėl*** ir ne tik ***kaip***.

1.1 Seminaro aptarimas:

- 2019-03-21 I srautas
- 2019-03-28 II srautas

1.2 Užduotys

Seminarą sudaro 2 dalys: pasikartojimo užduotis ir 3 funkcijų užduotys. * Pasikartojimo užduotis sudėtingumu skalėje nuo 0-10 siektia apie 5 ir atitinka (ne)anonsuotų testų sudėtingumą * Funkcijų parašymas sudėtingumu skalėje nuo 0-10 siekti tarp 8 ir 12 atitinka sudėtingumo lygiui, kurio tikėčiausi rašto darbe

2 Pasikartojimo dalis

2.1 Duomenys:

Parsisiųskite suzipintą duomenų failą: 2_seminaro_užduotis_1.zip. Unzippinkite failą ir importuokite duomenis į R. (Tiesa, tau galima apdaryti ir tiesiogiai iš R!)

2.2 Užduotys:

1. Kokie yra importuoto dataframe stulpelių pavadinimai?
2. R'e atspausdinkite pirmas 4 dataframe eilutes.
3. R'e atspausdinkite paskutines 4 dataframe eilutes.
4. Kiek elementų / stebėjimų / observacijų turi dataframe?
5. Kokia yra Ozono kiekio vertė 47 eilutėje?
6. Kokios yra vėjo greičio reikšmės nuo 50 iki 59 stebėjimo?
7. Kiek `NA` reikšmių yra Ozono kintamojo stulpelyje?
8. Koks Ozono kintamojo vidurkis visame dataframe?
9. Koks Solar.R kintamojo vidurkis, jeigu pasirenkami duomenys, kai Ozono reikmšės yra didesnis nei 31 ir temperatūra didesnė nei 90?
10. Kokia vidutinė birželio mėnesio temperatūra?
11. Kokia didžiausia pasiekta gegužės mėnesio temperatūra?

3 Funkcijų programavimas

Šioje dalyje Jums reikės parašyti tris skirtingas funkcijas, naudojantis šiuo zip failu: 2_seminaro_užduotis_2.zip, kuriame yra oro kokybės matavimo duomenys iš JAV. Zip faile yra 332 skirtingų matavimo stočių failai, savo atskiruose .csv failuose. Kiekvieno failo pavadinimas pvz., "212.csv" reiškia, jog tai 212-os matavimo stoties duomenys. Kiekvename .csv faile yra 4 stulpeliai: data, sulfatų ir nitratų kiekis bei stoties ID.

Parsisiųskite duomenis tik R pagalba. Išzipinkite duomenis.

```
#sukuriamas temporalinis failas
temp <- tempfile()
# objektui url priskirimas ur;
url <-"https://github.com/justasmundeikis/duomenu_analizes_ivadas/raw/master/seminars/2_seminaro_u%C5%BEduotis_2.zip"
# nudownloadiname į tempoainė failą url
download.file(url, temp)
# išpakuojame konteinerį
unzip(temp)

#išpakuojamas visas zip turinys, o pats zip ištrinamas
unlink(temp, recursive = TRUE, force = TRUE)
```

3.1 Pirma funkcija

Parašykite funkciją `uzterstumo_vidurkis`, kuri apskaičiuotų vieno iš pasirinktų kintamųjų vidurkį pasirinktose stotyse.b Taigi funkcija `uzterstumo_vidurkis` turi priimti 3 argumentus: direktoriją, kintamojo pavadinimą bei matavimo stoties ID. Priklausomai nuo pateikto stočių ID vektoriaus, funkcija turi nuskaityti pasirinktų stočių .csv failus iš nurodytos direktorijos ir pateikti pasirinkto kintamojo bendrą (t.y. visų pasirinktų stočių) vidurkį. Labai svarbu, jog pateikiant vidurkį, būtų panaikintos NA reikmšmės, kitaip vidurkis negalės būti apskaičiuotas.

Funkcijos prototipas:

```
pollutantmean <- function(direktorija, kintamasis, id=1:332){
        ## `direktorija` yra charackter string nurodanti, kur randasi csv failai

        ## `kintamasis` yra charackter string nurodantis pavadinimą,
        ## kurios medžiagos vidrukis turi būti skaičiuojamas:
        ## arba "sulfate" arba "nitrate"

        ## `id` integer vektorius nurodantis kurių stočių duomenis naudoti skaičiavimuose

        ## apskaičiuojamas pasirinktų stočių bei pasirinkto kintamojo vidurkis.
        ## vidurkis neapvalinimas, bet skaičiavimui pašalinimi NA
}
```

Sprendimas

```
pollutantmean <-  function(direktorja, kintamasis, id=1:332){

        #sukuriamas tuščias dataframe
        df <- data.frame()

        #sukuriamas failų pavadinimų su adresu (full.names=T) vektorius
        failas <- list.files(direktorja, full.names = TRUE)

        #pradedamas for loop kiekvimam i iš id (i perima konkrečius id įveriučius)
        #check: for(i in 3:5) print(i)
        for (i in id) {
                #df prikabinaimas jau turimas df ir duomenys iš csv
                df <- rbind(df,read.csv(failas[i],header = TRUE))
        }

        #apskaičiuojamas vidurkis
        mean(df[,kintamasis], na.rm = TRUE)

}
```

Pasitikrinimui, ar funkcija veikia teisingai:

```
pollutantmean("specdata", "sulfate", 1:10)
```

```
## [1] 4.064128
```

```
pollutantmean("specdata", "nitrate", 70:72)
```

```
## [1] 1.706047
```

```
pollutantmean("specdata", "nitrate", 23)
```

```
## [1] 1.280833
```

3.2 Antra funckija

Parašykite funkciją, kuri nuskaitytų norimą direktoriją ir grąžintų dataframe, kurios pirmame stulpelyje būtų failo pavadinimas, o antrame pilnų observacijų skaičius.

Funkcijos prototipas:

```
complete <- function(direktorija, id=1:332){
        ## `direktorija` yra charackter string nurodanti, kur randasi csv failai

        ## `id` integer vektorius nurodantis kurių stočių duomenis naudoti

        ## grąžinama dataframe turi turėti toki forma
        ## id nobs
        ## 1 117
        ## 2 1041
```

```
}
```

```
complete <- function(direktorija, id=1:332){
        #sukurimas dataframe su 2 stulpeliais ir tuščiom vertem
        df <- data.frame(id=NA, nobs=NA)

        #sukuriamas failų pavadinimų su adresu (full.names=T) vektorius
        failas <- list.files(direktorija, full.names = TRUE)

        #sukuriamas pagalbinis skaitiklis j, kur j=1,
        #tam kad išnaudoti įrašant df 1 eilutėje
        j <- 1
        for (i in id) {
                # df j-toje eilutėje 1 stulp[elyje įrašoma i reikšmė]
                df[j,1] <-i
                # df j-toje eilutėje 2 stulp[elyje įrašoma nuskaityti i failo complete cases suma
                # complete cases gržina loginį vektoriu (1,0,1) tad galima sumuot TRUE
                df[j,2] <- sum(complete.cases(read.csv(failas[i],header = TRUE)))

                #tam kad prasisuktų for loope į sekančią eilutę
                j <- j+1
        }

        #pabaigus atspausdina visa df
        print(df)
}
```

Pasitikrinimui, ar funkcija veikia teisingai:

```
complete("specdata", 1)
```

```
##   id nobs
## 1  1  117
```

```
complete("specdata", c(2, 4, 8, 10, 12))
```

```
##   id nobs
## 1  2 1041
## 2  4  474
## 3  8  192
## 4 10  148
## 5 12   96
```

```
complete("specdata", 30:25)
```

```
##   id nobs
## 1 30  932
## 2 29  711
## 3 28  475
## 4 27  338
## 5 26  586
## 6 25  463
```

```
complete("specdata", 3)
```

```
##   id nobs
## 1  3  243
```

3.3 Trečia funkcija

Parašykite funkciją kuri priimtu argumentus: direktoriją ir “threshold”. Threshold nusako, kiek turi būti pilnų observacijų, jog matavimo stoties rodikliai būtų įtraukti į skaičiavimus. Funkcija turi apskaičiuoti koreliaciją tarp “sulfit” ir “nitrat”

```
corr <- function(direktorija, threshold=0){
        ## `direktorija` yra charackter string nurodanti, kur randasi csv failai
```

```
        ## `threshold` numerinis vektorius, nurodantis, kiek stotyje turi būti mažiausiai pilnų observacijų

        ## grąžinamas numerinis vektorius su koreliacijomis tarp nitrato ir sulfato

}
```

```
corr <- function(direktorija, threshold=0){

        #sukuriamas failų pavadinimų su adresu (full.names=T) vektorius
        failas <- list.files(direktorija, full.names = TRUE)

        # sukurimas tuščias (=0) skaitinis vektorius nummeric(), kurio ilgumas length(failas)
        ax <- numeric(length=length(failas))

        #startuojamas for loopas kurio pagalba ax užpildomas pilnų case skaičiumi
        #taigi gauname vektoriu, kuriame ax atspindi kiek kuris .csv failas turi pilnų case (eilučių be NA)
        for(i in 1:length(failas)){
                ax[i] <- sum(complete.cases(read.csv(failas[i],header = TRUE)))
                }

        #seq_alon sukuria eilės sekos voktroių 1:tokio ilgumo kaip failas
        #iš to vektoriaus paimami tie skaičiai, kurių failai atitinka threshold
        #taigi sužinomo kuri failas[i] turėsime naudoti
        bx <- seq_along(failas)[ax>threshold]

        # sukuriamas  tuščias cx vektorius, į kurį kalsime korelaicijas
        cx <- numeric(length = length(bx))

        j <- 1
        #išnaudojame sukurt bx su skaičiais, kurie parodo kuris failas atitinka reikalavimus
        for (i in bx) {

                #nuskaitome reikalinga failą
                temp <- na.omit(read.csv(failas[i],header = TRUE))
                #apskaičiuojame kotreliacija tarp 2 ir 3 stulpelio ir reikmšė priskiriame  j-tai vektoriaus cx pozicijai
                cx[j] <- cor(temp[,2], temp[,3])
                # prastumiam j +1
                j <- j+1

        }
        #atspausindam cx vektoriu
        cx
}
```

Pasitikrinimui, ar funkcija veikia teisingai:

```
cr <- corr("specdata", 150)
head(cr)
```

```
## [1] -0.01895754 -0.14051254 -0.04389737 -0.06815956 -0.12350667 -0.07588814
```

```
summary(cr)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.21057 -0.04999  0.09463  0.12525  0.26844  0.76313
```

```
cr <- corr("specdata", 400)
head(cr)
```

```
## [1] -0.01895754 -0.04389737 -0.06815956 -0.07588814  0.76312884 -0.15782860
```

```
summary(cr)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.17623 -0.03109  0.10021  0.13969  0.26849  0.76313
```

```
cr <- corr("specdata", 5000)
summary(cr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

```
length(cr)
```

```
## [1] 0
```

```
cr <- corr("specdata")
summary(cr)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.00000 -0.05282  0.10718  0.13684  0.27831  1.00000
```

```
length(cr)
```

```
## [1] 323
```

4 Klausimai funkcijų daliai:

- Pateiktie suapvalintą iki 3 ženklų po kablelio atsakymą:

```
pollutantmean("specdata", "sulfate", 1:10)
```

- Pateiktie suapvalintą iki 3 ženklų po kablelio atsakymą:

```
pollutantmean("specdata", "nitrate", 70:72)
```

- Pateiktie suapvalintą iki 3 ženklų po kablelio atsakymą:

```
pollutantmean("specdata", "sulfate", 34)
```

- Koks gaunamas šio kodo rezultatas:

```
cc <- complete("specdata", c(6, 10, 20, 34, 100, 200, 310))
print(cc$nobs)
```

- Koks gaunamas šio kodo rezultatas:

```
set.seed(42)
cc <- complete("specdata", 332:1)
use <- sample(332, 10)
print(cc[use, "nobs"])
```

- Koks gaunamas šio kodo rezultatas:

```
cr <- corr("specdata")
cr <- sort(cr)
set.seed(868)
out <- round(cr[sample(length(cr), 5)], 4)
print(out)
```

- Koks gaunamas šio kodo rezultatas:

```
cr <- corr("specdata", 129)
cr <- sort(cr)
n <- length(cr)
set.seed(197)
out <- c(n, round(cr[sample(n, 5)], 4))
print(out)
```

- Koks gaunamas šio kodo rezultatas:

```
cr <- corr("specdata", 2000)
n <- length(cr)
cr <- corr("specdata", 1000)
cr <- sort(cr)
print(c(n, round(cr, 4)))
```