Tjedan 2: Uvod u R i tidyverse

Vaš novi najdraži alat za rad s podacima

2025-03-01

Table of contents

```
library(tidyverse)
```

i Ishodi učenja

Nakon ovog predavanja moći ćete

1. Objasniti zašto je R bolji izbor od Excela i SPSS-a za statističku analizu u komunikologiji i prepoznati situacije u kojima je svaki alat prikladan.
2. Koristiti R kao kalkulator i razumjeti osnovne aritmetičke i logičke operacije.
3. Kreirati objekte u R-u, razumjeti pravila imenovanja i razlikovati tipove podataka (numerički, tekstualni, logički).
4. Konstruirati i indeksirati vektore te primijeniti vektorizirane operacije.

5. Razumjeti razliku između tibble i data.frame te koristiti tibble za organizaciju podataka.
6. Koristiti pipe operator (`|>`) za čitljivo ulančavanje operacija.
7. Učitati CSV datoteku pomoću `read_csv()`, pregledati strukturu podataka s `glimpse()` i `head()`, te identificirati tipove stupaca.
8. Instalirati i učitati R pakete te razumjeti ulogu tidyverse ekosustava.

1 Zašto R, a ne nešto drugo?

Ovo je pitanje koje studenti postavljaju na prvom satu, i to je potpuno legitimno pitanje. Zašto biste učili programski jezik kad postoje alati s grafičkim sučeljem koji rade iste stvari? Zašto ne ostati u Excelu, koji već znate koristiti, ili naučiti SPSS koji ima menije i gumbe za svaku analizu?

Odgovor ima nekoliko slojeva i vrijedi ih proći redom jer razumijevanje prednosti R-a mijenja način na koji pristupate cijelom kolegiju.

Ponovljivost. Kad radite analizu u Excelu, vaš rad je nevidljiv jer klikate po ćelijama, povlačite formule, sortirate stupce, i na kraju imate rezultat, ali nemate zapis toga kako ste do njega došli. Ako vas kolega pita kako ste izračunali nešto, morate mu pokazati i objasniti svaki korak. Ako trebate ponoviti istu analizu s novim podacima, morate sve raditi ispočetka. U R-u, vaša analiza je skripta, dakle tekstualna datoteka koja sadrži svaki korak od učitavanja podataka do konačnog rezultata. Tu skriptu možete pokrenuti ponovo jednim klikom, poslati je kolegi, objaviti uz akademski rad, ili je modificirati za novi dataset. Ovo nije trivijalna prednost. U vremenu kad se sve više govori o krizi repliciranja u društvenim znanostima, ponovljivost analize nije luksuz nego nužnost.

Fleksibilnost. SPSS ima meni za t-test. Ima meni za ANOVA-u. Ima meni za regresiju. Ali što kad trebate nešto što nije u meniju? Što kad trebate kombinirati podatke iz tri različite ankete, izračunati prilagođenu mjeru angažmana koja uključuje i klikove i vrijeme na stranici i komentare, filtrirati samo ispitanike koji zadovoljavaju tri uvjeta istovremeno, i onda vizualizirati rezultat na način koji SPSS ne podržava? U R-u, granica je samo vaša mašta (i znanje, ali znanje raste). SPSS je poput restorana s fiksnim menijem. R je kuhinja u kojoj možete pripremiti bilo što.

Vizualizacija. Ovaj argument sam po sebi opravdava učenje R-a. Paket ggplot2, koji ćemo intenzivno koristiti od petog tjedna nadalje, proizvodi grafike profesionalne kvalitete koje možete izravno staviti u akademski rad, poslovni izvještaj ili medijsku prezentaciju. Excel grafovi izgledaju kao Excel grafovi. ggplot2 grafovi izgledaju kao da ih je dizajnirao grafički dizajner.

Besplatnost i zajednica. R je besplatan i open-source. Ne trebate licencu, ne trebate institucionalnu pretplatu, ne trebate brinuti hoćete li imati pristup nakon diplome. SPSS licenca košta stotine eura godišnje. Osim toga, R ima ogromnu zajednicu korisnika i tisuće paketa za svaku zamisivu analizu. Ako imate pitanje, gotovo sigurno je netko prije vas imao isto pitanje i odgovor postoji na internetu.

Zapošljivost. Ovo je pragmatičan argument ali važan. Analitičke vještine u R-u (ili Pythonu, koji je sličan po filozofiji) sve su traženije na tržištu rada, ne samo u akademiji nego i u medijskoj industriji, marketingu, oglaševanju i PR-u. Znanje Excela se podrazumijeva. Znanje R-a vas izdvaja.

Navarro u svojoj knjizi otvoreno priznaje da je učenje R-a teže od učenja softvera s grafičkim sučeljem. Početna krivulja učenja je strmija. Ali isto tako kaže da se ta investicija višestruko isplati jer jednom kad savladate osnove, možete raditi stvari koje su u drugim alatima nemoguće ili zahtijevaju enorman ručni rad. Mi ćemo slijediti njezin pristup i ići polako, objašnjavajući svaki korak bez pretpostavke o prethodnom iskustvu s programiranjem.

> Praktični savjet
>
> Ako vas uhvati frustracija dok učite R (a uhvatit će vas, to je normalno), prisjetite se da ste u istoj situaciji kao kad ste prvi put otvorili Photoshop ili Premiere. Prvi sat je bio zbunjujući. Nakon tjedan dana ste znali osnove. Nakon mjesec dana više niste mogli zamisliti rad bez tog alata. S R-om je isto, samo što je nagrada na kraju veća jer R možete koristiti za analizu bilo čega.

---

2 R i Positron: vaš radni prostor

Prije nego što počnemo pisati kod, trebamo dva programa.

R je sam programski jezik i sustav za statističko računanje. Kad instalirate R, dobivate motor koji izvršava naredbe, ali sučelje je minimalistično, gotovo spartan. Možete koristiti R izravno u terminalu (naredbenom retku), ali to je poput pisanja romana u Notepadu. Tehnički moguće, ali nitko razuman to ne radi.

Positron je integrirano razvojno okruženje (IDE) koje olakšava rad s R-om. Daje vam uređivač teksta s bojanjem sintakse (ključne riječi R-a prikazuje u boji da kod bude čitljiviji), prozor za pregled rezultata, prozor za grafike, prozor za pomoć, i mnoštvo drugih alata koji čine rad ugodnijim. Positron je relativno novi IDE razvijen od strane Posit tima (isti ljudi koji su stvorili RStudio), i koristi Visual Studio Code arhitekturu, što znači da je moderan, brz i proširiv.

> ! Važna napomena
>
> Ako ste već koristili RStudio, Positron će vam biti intuitivan jer dijeli istu filozofiju, samo s modernijim sučeljem. Ako niste koristili niti jedan IDE, ne brinite. Mi ćemo koristiti samo osnovne funkcionalnosti poput pisanja koda u skripti (gornji lijevi panel), izvršavanja koda u konzoli (donji panel) i pregleda rezultata. Sve ostalo ćete naučiti usput, kad bude potrebno.

2.1 Kako izgleda Positron

Kad otvorite Positron i kreirate novu R datoteku, vidjet ćete sučelje podijeljeno u nekoliko panela. Gornji lijevi panel je **editor** u kojem pišete kod. Donji panel je **konzola** u kojoj se kod izvršava i prikazuju rezultati. Desna strana ima panele za pregled varijabli, grafika i pomoći.

Radni tok u Positronu je jednostavan. Pišete kod u editoru. Označite redak (ili više redova) koji želite izvršiti. Pritisnete Ctrl+Enter (ili Cmd+Enter na Macu). Kod se pošalje u konzolu, izvrši se i rezultat se prikaže. Ovo ćemo raditi stotine puta tijekom kolegija, pa će vam vrlo brzo postati automatsko.

---

3 Prve naredbe: R kao kalkulator

Najjednostavniji način da počnete koristiti R jest da ga tretirate kao kalkulator. I to ne kao obični kalkulator, nego kao vrlo moćan kalkulator koji može raditi s cijelim skupovima brojeva odjednom. Pogledajmo neke osnovne operacije.

```
# Zbrajanje
10 + 25
```

```
[1] 35
```

```
# Oduzimanje
100 - 37
```

```
[1] 63
```

```
# Množenje
12 * 8
```

```
[1] 96
```

```
# Dijeljenje
144 / 12
```

```
[1] 12
```

```r
# Potenciranje
2^10
```

```
[1] 1024
```

Ništa revolucionarno za sada. Ali primijetite par stvari. Prvo, linije koje počinju s **#** su **komentari**. R ih potpuno ignorira. Komentari služe vama (i vašim kolegama) da objasnite što kod radi. Pisanje komentara je navika koju biste trebali razviti od prvog dana jer vam štedi mnogo vremena kad se mjesec dana kasnije vraćate na vlastiti kod i pokušavate shvatiti što ste radili.

Drugo, R poštuje standardni redoslijed matematičkih operacija (množenje i dijeljenje prije zbrajanja i oduzimanja, potenciranje prije svega), ali za svaki slučaj koristite zagrade kad želite biti sigurni.

```r
# Bez zagrada: množi se prvo
5 + 3 * 2
```

```
[1] 11
```

```r
# Sa zagradama: zbraja se prvo
(5 + 3) * 2
```

```
[1] 16
```

R ima i nekoliko korisnih matematičkih funkcija koje ćemo trebati tijekom kolegija.

```r
# Kvadratni korijen
sqrt(144)
```

```
[1] 12
```

```r
# Apsolutna vrijednost
abs(-42)
```

```
[1] 42
```

```r
# Zaokruživanje
round(3.14159, 2)
```

```
[1] 3.14
```

```r
# Logaritam (prirodni)
log(100)
```

```
[1] 4.60517
```

```r
# Logaritam baze 10
log10(100)
```

```
[1] 2
```

Funkcija `sqrt()` računa kvadratni korijen, `abs()` vraća apsolutnu vrijednost, `round()` zaokružuje na zadani broj decimala. Funkcija `log()` bez drugog argumenta računa prirodni logaritam (baza e), a `log10()` računa logaritam baze 10. Logaritme ćemo koristiti kasnije kad budemo radili s podacima koji imaju ekstremno asimetričnu distribuciju, poput broja pratitelja na društvenim mrežama.

---

4 Objekti: pohranjivanje vrijednosti

Kalkulator je koristan, ali prava snaga programiranja dolazi od mogućnosti da rezultate pohranite i koristite kasnije. U R-u, vrijednosti pohranjujete u **objekte** (neki ih zovu varijablama, ali da izbjegnemo zabunu sa statističkim varijablama, koristit ćemo termin objekti).

Objekt se kreira operatorom pridruživanja `<-` (strelica lijevo). Čita se kao "dodijeli vrijednost desne strane objektu na lijevoj strani".

```r
# Pohranjujemo broj ispitanika
n_ispitanika <- 500

# Pohranjujemo prosječno dnevno korištenje (u minutama)
prosjek_minuta <- 87.3

# Sada možemo koristiti te objekte
n_ispitanika
```

```
[1] 500
```

```r
prosjek_minuta
```

```
[1] 87.3
```

Kad kreirate objekt, R ga tiho pohrani u memoriju bez ikakvog ispisa. Da biste vidjeli što objekt sadrži, jednostavno upišite njegovo ime. Objekte možete koristiti u izračunima baš kao i brojeve.

```
# Ukupno vrijeme svih ispitanika (u minutama)
ukupno_minuta <- n_ispitanika * prosjek_minuta
ukupno_minuta
```

```
[1] 43650
```

```
# Pretvaranje u sate
ukupno_sati <- ukupno_minuta / 60
ukupno_sati
```

```
[1] 727.5
```

```
# Prosječno korištenje u satima
prosjek_sati <- prosjek_minuta / 60
round(prosjek_sati, 1)
```

```
[1] 1.5
```

Svih 500 ispitanika u našem imaginarnom uzorku zajedno provede gotovo 44 tisuće minuta dnevno na društvenim mrežama. To je oko 727 sati, ili nešto više od 30 punih dana. Svaki dan. Statistika, čak i ovako jednostavna, pomaže vizualizirati razmjere fenomena koji istražujemo.

4.1 Pravila imenovanja objekata

R ima nekoliko pravila i mnogo dobrih praksi za imenovanje objekata. Pravila su stroga i traže da ime mora počinjati slovom (ne brojem), ne smije sadržavati razmake ni specijalne znakove osim točke i podvlake, i R razlikuje velika i mala slova (`prosjek` i `Prosjek` su dva različita objekta).

Dobre prakse su mekše ali važne za čitljivost. Na ovom kolegiju koristimo konvenciju **snake_case**, u kojoj su riječi odvojene podvlakom i sve je malim slovima (npr. `prosjek_minuta`, `n_ispitanika`, `dnevno_koristenje`). Ova konvencija je standard u tidyverse zajednici i čini kod čitljivijim od alternativa poput `prosjekMinuta` (camelCase) ili `prosjek.minuta` (dot notation).

```r
# Dobra imena (snake_case, opisna)
broj_portala <- 15
prosjecni_ctr <- 0.034
ime_studije <- "medijske navike 2025"

# Funkcionalna ali loša imena (nejasna ili nekonzistentna)
x <- 15
bp <- 15
BrojPortala <- 15
```

Sva četiri donja primjera rade, ali samo `broj_portala` odmah komunicira što objekt sadrži. Kad budete imali skriptu s 50 objekata, razlika između opisnih i kriptičnih imena postaje enormna. Navarro u knjizi koristi lijep savjet: imenujte objekte tako da ih možete razumjeti kad se vratite na kod nakon dva tjedna bez spavanja.

> Praktični savjet
>
> R ima neke rezervirane riječi koje ne smijete koristiti kao imena objekata jer imaju posebno značenje u jeziku. Na primjer, `TRUE`, `FALSE`, `NULL`, `NA`, `if`, `else`, `for`, `function`. Ako pokušate, dobit ćete grešku. Također, izbjegavajte imena koja se poklapaju s postojećim R funkcijama, poput `mean`, `sum`, `data` ili `c`. Tehnički možete kreirati objekt nazvan `mean`, ali to će pregaziti funkciju `mean()` i uzrokovati zbunjujuće greške. Zato je dobra praksa koristiti opisna imena poput `prosjek_dobi` umjesto generičnog `mean`.

4.2 Prepisivanje objekata

Važno je razumjeti da kad dodijelite novu vrijednost postojećem objektu, stara vrijednost nestaje bez upozorenja.

```r
temperatura <- 22
temperatura
```

```
[1] 22
```

```r
# Nova dodjela prepisuje staru vrijednost
temperatura <- 35
temperatura
```

```
[1] 35
```

R vas neće pitati jeste li sigurni. Neće vam reći da ste upravo izgubili prethodnu vrijednost. Jednostavno će to napraviti. Ovo je razlog zašto je dobra praksa koristiti opisna imena i ne reciklirati isti objekt za različite svrhe. Ako vam trebaju temperatura zraka i temperatura vode, napravite `temp_zraka` i `temp_vode`, nemojte koristiti isti objekt `temp` za oboje.

5 Vektori: rad s više vrijednosti odjednom

Do sada smo pohranjivali pojedinačne brojeve, ali u statistici gotovo nikad ne radimo s jednim brojem. Radimo sa skupovima podataka. Najjednostavnija struktura za pohranjivanje više vrijednosti u R-u je **vektor**.

Vektor je uređeni niz vrijednosti istog tipa. Kreiramo ga funkcijom `c()` (od "combine" ili "concatenate").

```
# Dnevno korištenje TikToka za 8 ispitanika (u minutama)
dnevno_tiktok <- c(95, 22, 112, 45, 78, 8, 135, 55)
dnevno_tiktok
```

```
[1]  95  22 112  45  78   8 135  55
```

```
# Dobne skupine istih ispitanika
dobne_skupine <- c("18-24", "45+", "18-24", "25-34", "18-24", "55+", "18-24", "35-44")
dobne_skupine
```

```
[1] "18-24" "45+"   "18-24" "25-34" "18-24" "55+"   "18-24" "35-44"
```

Primijetite da se tekstualne vrijednosti stavljaju u navodnike, a numeričke ne. Ovo je razlika između tipova podataka o kojoj ćemo govoriti detaljnije za trenutak.

Snaga vektora je u tome što R automatski primjenjuje operacije na sve elemente odjednom. Ovo se zove **vektorizacija** i jedna je od najvažnijih karakteristika R-a.

```
# Pretvorba u sate (dijeli SVAKI element sa 60)
dnevno_sati <- dnevno_tiktok / 60
round(dnevno_sati, 1)
```

```
[1] 1.6 0.4 1.9 0.8 1.3 0.1 2.2 0.9
```

```
# Tjedno korištenje (množi SVAKI element sa 7)
tjedno_tiktok <- dnevno_tiktok * 7
tjedno_tiktok
```

```
[1] 665 154 784 315 546  56 945 385
```

```r
# Koliko minuta iznad prosjeka?
prosjek <- mean(dnevno_tiktok)
iznad_prosjeka <- dnevno_tiktok - prosjek
round(iznad_prosjeka, 1)
```

```
[1]  26.2 -46.8  43.2 -23.8   9.2 -60.8  66.2 -13.8
```

Kad napišete `dnevno_tiktok / 60`, R ne dijeli vektor kao cjelinu sa 60 (to ne bi imalo smisla), nego dijeli svaki element vektora sa 60. Rezultat je novi vektor iste duljine. Isto vrijedi za sve aritmetičke operacije. Ovo je enormno praktično jer nam omogućuje da jednom naredbom transformiramo stotine ili tisuće vrijednosti.

Na vektore možemo primijeniti i funkcije koje sažimaju podatke u jednu vrijednost.

```r
# Broj elemenata
length(dnevno_tiktok)
```

```
[1] 8
```

```r
# Prosjek
mean(dnevno_tiktok)
```

```
[1] 68.75
```

```r
# Medijan
median(dnevno_tiktok)
```

```
[1] 66.5
```

```r
# Standardna devijacija
sd(dnevno_tiktok)
```

```
[1] 44.18064
```

```r
# Minimum i maksimum
min(dnevno_tiktok)
```

```
[1] 8
```

```r
max(dnevno_tiktok)
```

```
[1] 135
```

```r
# Zbroj svih vrijednosti
sum(dnevno_tiktok)
```

```
[1] 550
```

Ove funkcije uzimaju čitav vektor i vraćaju jednu vrijednost. `mean()` računa aritmetičku sredinu, `median()` srednju vrijednost, `sd()` standardnu devijaciju, `min()` i `max()` najmanju i najveću vrijednost, `sum()` zbroj svih elemenata. Detaljno ćemo objasniti svaku od ovih mjera u tjednu o deskriptivnoj statistici. Za sada je dovoljno znati da postoje i da rade na vektorima.

5.1 Indeksiranje vektora

Ponekad trebamo pristupiti samo jednom ili nekoliko elemenata vektora. To radimo uglatim zagradama `[]`.

```r
# Treći element
dnevno_tiktok[3]
```

```
[1] 112
```

```r
# Elementi od drugog do petog
dnevno_tiktok[2:5]
```

```
[1]  22 112  45  78
```

```r
# Elementi koji zadovoljavaju uvjet
dnevno_tiktok[dnevno_tiktok > 100]
```

```
[1] 112 135
```

```r
# Koliko ispitanika koristi TikTok više od 100 minuta dnevno?
sum(dnevno_tiktok > 100)
```

```
[1] 2
```

Posebno je korisna mogućnost filtriranja po uvjetu. Izraz `dnevno_tiktok > 100` proizvodi logički vektor (niz TRUE i FALSE vrijednosti), a kad ga stavimo u uglate zagrade, R vraća samo one elemente za koje je uvjet TRUE. Funkcija `sum()` primijenjena na logički vektor broji koliko je TRUE vrijednosti, jer R tretira TRUE kao 1 i FALSE kao 0.

> **!** Važna napomena
>
> R indeksira od 1, ne od 0. Prvi element vektora je `vektor[1]`, ne `vektor[0]`. Ako ste učili Python ili JavaScript, ovo je važna razlika. Većina početničkih grešaka s indeksiranjem u R-u dolazi od zaboravljanja da R počinje brojati od 1.

---

6 Tipovi podataka

Svaka vrijednost u R-u ima tip koji određuje što možete s njom raditi. Četiri osnovna tipa koja ćete koristiti su numerički, tekstualni, logički i faktorski.

6.1 Numerički tip (numeric / double)

Svaki broj u R-u je po defaultu tipa `double`, što znači da se pohranjuje kao decimalni broj čak i kad izgleda kao cijeli broj. Postoji i podtip `integer` (cijeli broj) koji se kreira dodavanjem slova L: `42L`. U praksi, razlika rijetko bitna i R se uglavnom sam snalazi.

```
x <- 42
class(x)
```

```
[1] "numeric"
```

```
y <- 42L
class(y)
```

```
[1] "integer"
```

```
# Oboje radi jednako u većini situacija
x == y
```

```
[1] TRUE
```

6.2 Tekstualni tip (character)

Tekst u R-u se označava navodnicima, bilo jednostrukim ili dvostrukim. U ovom kolegiju koristimo dvostruke navodnike jer je to konvencija u tidyverse zajednici.

```r
platforma <- "TikTok"
class(platforma)
```

```
[1] "character"
```

```r
poruka <- "Ispitanik koristi platformu 95 minuta dnevno"
poruka
```

```
[1] "Ispitanik koristi platformu 95 minuta dnevno"
```

S tekstualnim vrijednostima ne možete raditi aritmetiku. Ako pokušate zbrojiti dva teksta pomoću `+`, dobit ćete grešku. Za spajanje tekstova koristi se funkcija `paste()` ili `paste0()`.

```r
ime <- "Portal"
broj <- "Index.hr"

# paste() spaja s razmakom (po defaultu)
paste(ime, broj)
```

```
[1] "Portal Index.hr"
```

```r
# paste0() spaja bez razmaka
paste0("n = ", n_ispitanika)
```

```
[1] "n = 500"
```

6.3 Logički tip (logical)

Logičke vrijednosti su samo dvije. Kako se pojavljuju: `TRUE` i `FALSE` nastaju kad R evaluira uvjete.

```r
# Usporedbe vraćaju logičke vrijednosti
10 > 5
```

```
[1] TRUE
```

```r
10 < 5
```

```
[1] FALSE
```

```r
10 == 10   # jednako (dva znaka jednakosti!)
```

```
[1] TRUE
```

```r
10 != 5    # nije jednako
```

```
[1] TRUE
```

```r
# Logički vektor
minuta <- c(95, 22, 112, 45, 78)
visoko_koristenje <- minuta > 60
visoko_koristenje
```

```
[1]  TRUE FALSE  TRUE FALSE  TRUE
```

Obratite pažnju na razliku između = i ==. Jedan znak jednakosti (=) je operator pridruživanja (isto kao <-). Dva znaka jednakosti (==) je operator usporedbe koji provjerava jesu li dvije vrijednosti jednake i vraća TRUE ili FALSE. Zamjena jednog s drugim je jedna od najčešćih pogrešaka u R-u.

6.4 Faktorski tip (factor)

Faktori su poseban tip za kategorijalene podatke. Iznad smo vidjeli da razine mjerenja određuju što smijemo raditi s varijablom. Faktori su način na koji R implementira kategorijalne varijable, posebno nominalne i ordinalne.

```r
# Kreiranje faktora
platforme <- factor(c("Instagram", "TikTok", "YouTube", "TikTok", "Instagram", "YouTube",
platforme
```

```
[1] Instagram TikTok    YouTube   TikTok    Instagram YouTube   TikTok
Levels: Instagram TikTok YouTube
```

```r
# Razine faktora (unique kategorije)
levels(platforme)
```

```
[1] "Instagram" "TikTok"    "YouTube"
```

```r
# Uređeni faktor (ordinalni)
obrazovanje <- factor(
  c("srednja", "prvostupnik", "magistar", "srednja", "magistar"),
  levels = c("srednja", "prvostupnik", "magistar", "doktor"),
  ordered = TRUE
)
obrazovanje
```

```
[1] srednja     prvostupnik magistar    srednja     magistar
Levels: srednja < prvostupnik < magistar < doktor
```

Faktori postaju bitni kad počnemo raditi vizualizacije i statističke testove. Na primjer, ako želite da se kategorije na grafikonu pojave u specifičnom redoslijedu (ne abecednom), morate koristiti faktore s definiranim razinama. Za sada je dovoljno znati da postoje, a detaljnije ćemo ih koristiti od tjedna vizualizacije.

6.5 Provjera i pretvorba tipova

R ima funkcije za provjeru tipa (`class()`, `is.numeric()`, `is.character()`) i za pretvorbu (`as.numeric()`, `as.character()`).

```r
# Provjera tipa
tekst_broj <- "42"
class(tekst_broj)
```

```
[1] "character"
```

```r
# Ovo je tekst, ne broj! Ne možemo računati s njim.
# tekst_broj + 10 bi dalo grešku

# Pretvorba u broj
pravi_broj <- as.numeric(tekst_broj)
class(pravi_broj)
```

```
[1] "numeric"
```

```r
pravi_broj + 10
```

```
[1] 52
```

Razumijevanje tipova podataka postaje ključno kad učitavate podatke iz CSV datoteka. Ponekad R pogrešno protumači stupac (na primjer, stupac s brojevima koji sadrži jedno slovo "N/A" umjesto prazne ćelije bit će učitan kao tekst umjesto broja). Znanje o tipovima i pretvorbama pomaže vam dijagnosticirati i popraviti takve probleme.

---

7 Tibble: moderna tablica podataka

Vektor može sadržavati samo vrijednosti istog tipa. Ali u stvarnim podacima imamo i brojeve (dob, minuta korištenja) i tekst (ime platforme, spol) i logičke vrijednosti, sve za istog ispitanika. Za organizaciju takvih podataka koristimo **tablicu** ili, u R žargonu, **data frame**.

U tidyverse ekosustavu koristimo poboljšanu verziju data framea koja se zove **tibble** (iz paketa tibble koji je dio tidyverse). Tibble je tablica u kojoj svaki stupac može biti drugog tipa, svaki redak predstavlja jedno opažanje i svaki stupac predstavlja jednu varijablu. Ovo odgovara onome što Wickham naziva **tidy data** (uredni podaci), i to je filozofija oko koje je cijeli tidyverse izgrađen.

Kreirajmo mali tibble s podacima o našim zamišljenim ispitanicima.

```
anketa <- tibble(
  id = 1:8,
  dob = c(19, 52, 21, 35, 23, 61, 20, 42),
  spol = c("ženski", "muški", "ženski", "muški", "ženski", "muški", "muški", "ženski"),
  platforma = c("TikTok", "Facebook", "Instagram", "LinkedIn", "TikTok", "Facebook", "TikT
  dnevno_min = c(95, 22, 112, 45, 78, 8, 135, 55)
)

anketa
```

```
# A tibble: 8 x 5
     id   dob spol   platforma dnevno_min
  <int> <dbl> <chr>  <chr>          <dbl>
1     1    19 ženski TikTok            95
2     2    52 muški  Facebook          22
3     3    21 ženski Instagram        112
4     4    35 muški  LinkedIn          45
5     5    23 ženski TikTok            78
6     6    61 muški  Facebook           8
7     7    20 muški  TikTok           135
8     8    42 ženski Instagram         55
```

Nekoliko stvari koje vrijedi primijetiti. Kad ispišete tibble, R automatski prikazuje tip svakog stupca ispod imena (`<int>` za cijele brojeve, `<dbl>` za decimalne, `<chr>` za tekst). Ovo je enormno korisno jer na prvi pogled vidite kakve su varijable u vašim podacima. Tibble također automatski prikazuje samo prvih 10 redova, što sprječava da vam konzola bude poplavljena tisućama redova kad radite s velikim datasetima.

Usporedite ovo s klasičnim data frameom.

```
# Klasični data.frame
df <- data.frame(
  id = 1:3,
  ime = c("Ana", "Marko", "Petra"),
  dob = c(22, 35, 28)
)

# tibble
tb <- tibble(
  id = 1:3,
  ime = c("Ana", "Marko", "Petra"),
  dob = c(22, 35, 28)
)

# Razlika u ispisu
class(df)
```

```
[1] "data.frame"
```

```
class(tb)
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

Razlika postaje očitija s većim podacima, ali ključna poanta je da je tibble modernija, čistija verzija data framea i da je standard u tidyverse ekosustavu. Na ovom kolegiju ćemo uvijek koristiti tibble.

7.1 Pristupanje stupcima

Pojedinom stupcu tibble pristupamo operatorom `$` ili pomoću funkcije `pull()`.

```
# Dolar operator
anketa$dnevno_min
```

```
[1]  95  22 112  45  78   8 135  55
```

```r
# Izračun prosjeka jednog stupca
mean(anketa$dnevno_min)
```

```
[1] 68.75
```

```r
# Provjera koliko ispitanika koristi TikTok
sum(anketa$platforma == "TikTok")
```

```
[1] 3
```

Operator `$` je brz i praktičan za pristup jednom stupcu. U tidyverse pristupu ćemo češće koristiti `select()` i `pull()`, ali `$` je savršeno ispravan i često najbrži način da dohvatite jedan stupac.

---

8 Pipe operator: čitljivo ulančavanje

Sada dolazimo do jednog od najvažnijih koncepata u tidyverse pristupu: **pipe operatora** `|>`. Pipe je jednostavan ali transformativan koncept koji čini R kod dramatično čitljivijim.

Zamislite da želite napraviti sljedeće: trebate uzeti naš tibble `anketa`, filtrirati samo ispitanike mlađe od 30 godina i izračunati prosjek njihovog dnevnog korištenja. Bez pipea, ovo možete napisati na dva načina, i oba su neelegantna.

```r
# Pristup 1: ugniježđene funkcije (čitanje iznutra prema van)
mean(filter(anketa, dob < 30)$dnevno_min)
```

```
[1] 105
```

```r
# Pristup 2: međuobjekti (stvara nepotrebne objekte)
mladi <- filter(anketa, dob < 30)
prosjek_mladi <- mean(mladi$dnevno_min)
prosjek_mladi
```

```
[1] 105
```

Prvi pristup je kompaktan ali nečitljiv. Trebate čitati iznutra prema van, pa najprije vidite `mean()` pa se trebate probiti do `filter()` unutra da shvatite na što se mean primjenjuje. Drugi pristup je čitljiviji ali stvara objekt `mladi` koji nam zapravo ne treba i zatrpava radni prostor.

Pipe operator rješava oba problema. Čita se kao "uzmi ovo i onda napravi ono".

```
anketa |>
  filter(dob < 30) |>
  pull(dnevno_min) |>
  mean()
```

```
[1] 105
```

Čitate ovaj kod odozgo prema dolje, s lijeva na desno, baš kao tekst. Uzmi `anketa`, **zatim** filtriraj retke gdje je dob manja od 30, **zatim** izvuci stupac `dnevno_min`, **zatim** izračunaj prosjek. Svaki `|>` znači "uzmi rezultat prethodnog koraka i proslijedi ga kao prvi argument sljedećoj funkciji".

Evo još jednog primjera koji pokazuje zašto je pipe tako koristan.

```
anketa |>
  filter(dob < 40) |>
  select(id, platforma, dnevno_min) |>
  arrange(desc(dnevno_min))
```

```
# A tibble: 5 x 3
     id platforma dnevno_min
  <int> <chr>          <dbl>
1     7 TikTok           135
2     3 Instagram        112
3     1 TikTok            95
4     5 TikTok            78
5     4 LinkedIn          45
```

Uzmi anketu, zadrži samo ispitanike mlađe od 40, odaberi tri stupca i sortiraj po dnevnom korištenju od najvećeg prema najmanjem. Svaki korak je jasan i čitljiv. Bez pipea, trebate napisati: `arrange(select(filter(anketa, dob < 40), id, platforma, dnevno_min), desc(dnevno_min))`. Isti rezultat, ali mozak se muči dok ga parsira.

> **Praktični savjet**
>
> Tipkovnička kratica za pipe operator `|>` u Positronu je Ctrl+Shift+M (ili Cmd+Shift+M na Macu). Budući da ćete pipe koristiti u gotovo svakom redu koda od sada pa nadalje, ova kratica će vam uštedjeti mnogo tipkanja. Provjerite u postavkama Positrona da

je kratica podešena na native pipe `|>`, a ne na magrittr pipe `%>%`. Oba rade gotovo identično, ali `|>` je noviji i preporučen.

> Pipe operator je poput veznika "i onda" u rečenici. Bez njega, R kod se čita kao telegram. S njim, čita se kao priča.

---

9 Paketi: proširivanje R-a

R sam po sebi dolazi s osnovnim funkcijama (base R), ali prava snaga leži u **paketima** koje je zajednica korisnika razvila za specifične zadatke. Paket je kolekcija funkcija, podataka i dokumentacije koju netko drugi napisao i koju vi možete koristiti u svom radu.

Na ovom kolegiju dominira jedan paket, zapravo kolekcija paketa, koji se zove **tidyverse**.

9.1 Što je tidyverse?

Tidyverse nije jedan paket nego skup od osam paketa koji dijele zajedničku filozofiju dizajna i besprijekorno surađuju. Kad učitate tidyverse naredbom `library(tidyverse)`, zapravo učitavate sljedeće pakete.

ggplot2 za vizualizaciju podataka, **dplyr** za manipulaciju podacima (filter, select, mutate, summarise, group_by), **tidyr** za preoblikovanje podataka (pivot_longer, pivot_wider), **readr** za učitavanje podataka (read_csv), **tibble** za moderne tablice podataka, **stringr** za rad s tekstom, **forcats** za rad s faktorima, te **purrr** za funkcionalno programiranje.

Od ovih osam, na ovom kolegiju ćemo najintenzivnije koristiti dplyr, ggplot2, tidyr i readr. S ostalima ćemo se susresti po potrebi.

9.2 Instalacija i učitavanje paketa

Paketi se instaliraju jednom, a učitavaju svaki put kad pokrenete R sesiju. Razmislite na analógiju gdje je instalacija poput kupnje knjige (radite to jednom), dok je učitavanje poput otvaranja knjige (radite svaki put kad ju trebate).

```
# Instalacija (samo jednom, u konzoli)
install.packages("tidyverse")

# Učitavanje (na početku svake skripte)
library(tidyverse)
```

Naredbu `install.packages()` pokrećete u konzoli, ne u skripti, jer ne želite da se paket reinstalira svaki put kad pokrenete skriptu. Naredbu `library()` stavljate na početak svake skripte jer R mora znati koje pakete koristite.

> **!** Važna napomena
>
> Kad prvi put instalirate tidyverse, proces može trajati nekoliko minuta jer se instalira mnogo paketa i njihovih ovisnosti. To je normalno. Kad instalacija završi, ne morate ju ponavljati osim ako ne želite ažurirati na noviju verziju. Ako dobijete grešku tijekom instalacije, najčešći uzrok je nedostatak sistemskih biblioteka na Linuxu ili zastarjela verzija R-a. U tom slučaju, ažurirajte R na najnoviju verziju i pokušajte ponovo.

---

10 Učitavanje podataka: read_csv()

Teorija je lijepa, ali prava zabava počinje kad počnemo raditi sa stvarnim (ili barem realistično simuliranim) podacima. Najčešći format za podatke je CSV (comma-separated values), običan tekstualni fajl u kojem su vrijednosti odvojene zarezima. CSV možete otvoriti u bilo čemu, od Excela do Notepada, i gotovo svaki softver ga može izvesti.

Za učitavanje CSV datoteka koristimo funkciju `read_csv()` iz paketa readr (dio tidyverse). Učitajmo dataset o korištenju društvenih mreža koji ćemo koristiti na ovom predavanju.

```
social <- read_csv("../resources/datasets/social_media_survey.csv")
```

Funkcija `read_csv()` čita datoteku i automatski pogađa tipove stupaca. Vraća tibble koji smo pohranili u objekt nazvan `social`. Primijetite da smo koristili `read_csv()` (s podvlakom), a ne `read.csv()` (s točkom). Ovo nije kozmetička razlika. `read_csv()` je brža, automatski stvara tibble (ne data.frame), bolje pogađa tipove stupaca i daje informativnije poruke.

10.1 Prvi pogled na podatke

Kad učitate novi dataset, prva stvar koju uvijek radite jest pogledati što se unutra nalazi. Nekoliko funkcija je korisno za to.

```
# Struktura podataka sa stupcima, tipovima i prvim vrijednostima
glimpse(social)
```

```
Rows: 500
Columns: 12
$ respondent_id       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,~
$ age                 <dbl> 35, 43, 40, 21, 36, 23, 45, 52, 59, 33, 32, 20, 49~
$ age_group           <chr> "35-44", "35-44", "35-44", "18-24", "35-44", "18-
2~
$ gender              <chr> "male", "female", "female", "female", "male", "fem~
$ education           <chr> "srednja_skola", "prvostupnik", "srednja_skola", "~
$ primary_platform    <chr> "Instagram", "Facebook", "Twitter", "TikTok", "You~
$ daily_minutes       <dbl> 92, 70, 79, 158, 14, 79, 38, 23, 41, 173, 153, 100~
$ num_platforms       <dbl> 4, 3, 3, 5, 3, 1, 2, 1, 2, 2, 2, 4, 2, 2, 2, 4, 3,~
$ trust_social_news   <dbl> 2, 6, 3, 4, 5, 9, 6, 2, 3, 2, 3, 5, 4, 5, 2, 4, 2,~
$ primary_news_source <chr> "portal", "drustvene_mreze", "portal", "portal", "~
$ weekly_posts        <dbl> 2, 9, 6, 14, 6, 0, 0, 0, 0, 3, 3, 19, 0, 2, 0, 13,~
$ privacy_concern     <dbl> 7, 6, 8, 7, 5, 5, 5, 5, 1, 5, 7, 7, 8, 8, 9, 5, 7,~
```

Funkcija `glimpse()` je jedna od najkorisnijih u tidyverse. Na jednom ekranu vidite broj redova i stupaca, ime svakog stupca, tip svakog stupca i prvih nekoliko vrijednosti. To je dovoljno da stvorite mentalnu sliku dataseta.

```
# Prvih 10 redova
head(social, 10)
```

```
# A tibble: 10 x 12
   respondent_id   age age_group gender education primary_platform daily_minutes
           <dbl> <dbl> <chr>     <chr>  <chr>     <chr>                     <dbl>
 1             1    35 35-44     male   srednja_~ Instagram                    92
 2             2    43 35-44     female prvostup~ Facebook                     70
 3             3    40 35-44     female srednja_~ Twitter                      79
 4             4    21 18-24     female srednja_~ TikTok                      158
 5             5    36 35-44     male   magistar  YouTube                      14
 6             6    23 18-24     female magistar  Instagram                    79
 7             7    45 45-54     female srednja_~ Facebook                     38
 8             8    52 45-54     female srednja_~ Twitter                      23
 9             9    59 55+       male   srednja_~ Instagram                    41
10            10    33 25-34     male   srednja_~ Twitter                     173
# i 5 more variables: num_platforms <dbl>, trust_social_news <dbl>,
#   primary_news_source <chr>, weekly_posts <dbl>, privacy_concern <dbl>
```

Funkcija `head()` prikazuje prvih N redova (po defaultu 6, ali možete zadati drugi broj). Korisna je kad želite vidjeti kako stvarni redovi izgledaju.

```
# Broj redova i stupaca
nrow(social)
```

```
[1] 500
```

```
ncol(social)
```

```
[1] 12
```

```
# Imena stupaca
names(social)
```

```
 [1] "respondent_id"       "age"                 "age_group"
 [4] "gender"              "education"           "primary_platform"
 [7] "daily_minutes"       "num_platforms"       "trust_social_news"
[10] "primary_news_source" "weekly_posts"        "privacy_concern"
```

Naš dataset sadrži 500 ispitanika i 12 varijabli. Varijable uključuju demografske podatke (dob, spol, obrazovanje), podatke o korištenju društvenih mreža (primarna platforma, dnevne minute, broj platformi, tjedno objavljenih postova) i stavove (povjerenje u vijesti na društvenim mrežama, briga za privatnost). Također imamo varijablu o primarnom izvoru vijesti.

10.2 Provjera tipova stupaca

Ponekad `read_csv()` ne protumači stupac onako kako bismo željeli. Dobra praksa je provjeriti tipove i napraviti eventualne korekcije.

```
# Pregled prvih redova odabranih stupaca
social |>
  select(respondent_id, age, gender, primary_platform, daily_minutes) |>
  head()
```

```
# A tibble: 6 x 5
  respondent_id   age gender primary_platform daily_minutes
          <dbl> <dbl> <chr>  <chr>                    <dbl>
1             1    35 male   Instagram                   92
2             2    43 female Facebook                    70
3             3    40 female Twitter                     79
4             4    21 female TikTok                     158
5             5    36 male   YouTube                     14
6             6    23 female Instagram                   79
```

Vidimo da je `respondent_id` učitan kao broj (`<dbl>`), `age` kao broj, `gender` kao tekst (`<chr>`), `primary_platform` kao tekst i `daily_minutes` kao broj. Ovo je razumno za naše podatke. U kasnijim tjednima naučit ćemo kako pretvoriti tekstualne stupce u faktore kad nam to bude trebalo za analizu ili vizualizaciju.

---

11 Istraživanje podataka: prvi uvidi

Sad kad imamo podatke učitane, napravimo nekoliko osnovnih istraživanja da stvorimo osjećaj za ono s čime radimo. Ovo je korak koji biste uvijek trebali napraviti prije ikakve ozbiljne analize.

```
# Osnovna deskriptivna statistika za numeričke varijable
summary(social$daily_minutes)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.00   51.00   86.50   93.58  136.00  272.00
```

```
summary(social$age)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  18.00   23.00   32.00   34.05   42.00   71.00
```

Funkcija `summary()` daje brzi pregled distribucije, uključujući minimum, prvi kvartil, medijan, prosjek, treći kvartil i maksimum. Detaljno ćemo objasniti sve ove mjere u tjednu o deskriptivnoj statistici. Za sada je dovoljno vidjeti da prosječni ispitanik provodi otprilike 90 minuta dnevno na društvenim mrežama i da su dobi raspoređene od 18 do 71 godinu.

Pogledajmo distribuciju kategoričkih varijabli.

```
# Koliko ispitanika po platformi?
social |>
  count(primary_platform, sort = TRUE)
```

```
# A tibble: 8 x 2
  primary_platform     n
  <chr>            <int>
1 Instagram          103
2 Facebook            93
3 TikTok              90
4 YouTube             86
5 LinkedIn            49
6 Twitter             41
7 Snapchat            20
8 Reddit              18
```

Funkcija `count()` je iz paketa dplyr i radi nešto vrlo jednostavno ali korisno. Evo što radi — prebrojava koliko redova pripada svakoj kategoriji. Argument `sort = TRUE` sortira rezultat po frekvenciji od najveće prema najmanjoj. Vidimo da su Instagram, Facebook i TikTok najzastupljenije platforme u našem uzorku.

```r
# Odakle ispitanici dobivaju vijesti?
social |>
  count(primary_news_source, sort = TRUE)
```

```
# A tibble: 5 x 2
  primary_news_source     n
  <chr>               <int>
1 drustvene_mreze       199
2 portal                132
3 TV                    102
4 print                  39
5 radio                  28
```

Zanimljivo je da najveći broj ispitanika navodi društvene mreže kao primarni izvor vijesti, što je konzistentno s trendom koji vidimo u istraživanjima diljem svijeta, osobito kod mlađih dobnih skupina.

Kombinirajmo pipe operator s nečim složenijim. Pogledajmo prosječno dnevno korištenje po dobnim skupinama.

```r
social |>
  group_by(age_group) |>
  summarise(
    n = n(),
    prosjek_min = round(mean(daily_minutes), 1),
    medijan_min = median(daily_minutes)
  )
```

```
# A tibble: 5 x 4
  age_group     n prosjek_min medijan_min
  <chr>     <int>       <dbl>       <dbl>
1 18-24       167       146.        148
2 25-34       130        95.9        94.5
3 35-44       107        61.2        62
4 45-54        58        40.1        38
5 55+          38        26.8        27
```

Ovo je prvi primjer obrasca `group_by() |> summarise()` koji će postati vaš najvažniji alat u tjednima koji dolaze. Logika je jednostavna. `group_by()` dijeli podatke u grupe po varijabli `age_group`, a `summarise()` izračunava statistike za svaku grupu zasebno. Vidimo jasnu razliku u korištenju društvenih mreža između dobnih skupina, s mladima koji provode daleko više vremena na platformama.

Praktični savjet

Funkcija `n()` unutar `summarise()` vraća broj opažanja u svakoj grupi. Uvijek je dobra praksa uključiti `n = n()` u svaki `summarise()` poziv jer vam govori koliko podataka stoji iza svake izračunate statistike. Prosjek izračunat na 5 opažanja je puno manje pouzdan od prosjeka izračunatog na 500 opažanja, i bez `n()` to ne biste znali.

---

12 Logički operatori: kombiniranje uvjeta

U prvom dijelu predavanja vidjeli smo jednostavne usporedbe poput `dob < 30` ili `platforma == "TikTok"`. Ali u stvarnoj analizi rijetko vas zanima samo jedan uvjet. Tipičnije je da tražite ispitanike koji su mlađi od 25 **i** koriste TikTok, ili ispitanike koji koriste Instagram **ili** Facebook, ili ispitanike koji **ne** pripadaju najstarijoj dobnoj skupini. Za kombiniranje uvjeta koristimo logičke operatore.

I operator (`&`) vraća TRUE samo kad su oba uvjeta ispunjena.

```
# Ispitanici mlađi od 25 koji koriste TikTok
social |>
  filter(age < 25 & primary_platform == "TikTok") |>
  head(8)
```

```
# A tibble: 8 x 12
  respondent_id   age age_group gender education  primary_platform daily_minutes
          <dbl> <dbl> <chr>     <chr>  <chr>      <chr>                     <dbl>
1             4    21 18-24     female srednja_s~ TikTok                      158
2            31    21 18-24     female magistar   TikTok                      218
3            33    24 18-24     female magistar   TikTok                       77
4            44    22 18-24     female srednja_s~ TikTok                      112
5            45    24 18-24     female magistar   TikTok                      115
6            70    23 18-24     male   prvostupn~ TikTok                       67
7            77    24 18-24     female srednja_s~ TikTok                      182
8            83    20 18-24     male   magistar   TikTok                      131
# i 5 more variables: num_platforms <dbl>, trust_social_news <dbl>,
#   primary_news_source <chr>, weekly_posts <dbl>, privacy_concern <dbl>
```

ILI operator (`|`) vraća TRUE kad je barem jedan uvjet ispunjen.

```
# Ispitanici koji koriste Instagram ili TikTok
social |>
  filter(primary_platform == "Instagram" | primary_platform == "TikTok") |>
  count(primary_platform)
```

```
# A tibble: 2 x 2
  primary_platform     n
  <chr>            <int>
1 Instagram          103
2 TikTok              90
```

NE operator (`!`) preokretne logičku vrijednost. Evo kako to radi: TRUE postaje FALSE i obrnuto.

```
# Ispitanici koji NE koriste Facebook
social |>
  filter(!primary_platform == "Facebook") |>
  count(primary_platform, sort = TRUE)
```

```
# A tibble: 7 x 2
  primary_platform     n
  <chr>            <int>
1 Instagram          103
2 TikTok              90
3 YouTube             86
4 LinkedIn            49
5 Twitter             41
6 Snapchat            20
7 Reddit              18
```

Kad trebate provjeriti pripada li vrijednost jednoj od više kategorija, umjesto dugačkog niza ILI uvjeta koristite operator `%in%`.

```
# Ispitanici koji koriste jednu od tri platforme
vizualne_platforme <- c("Instagram", "TikTok", "Snapchat")

social |>
  filter(primary_platform %in% vizualne_platforme) |>
  count(primary_platform, sort = TRUE)
```

```
# A tibble: 3 x 2
  primary_platform     n
  <chr>            <int>
1 Instagram          103
2 TikTok              90
3 Snapchat            20
```

Operator `%in%` je ekvivalentan pisanju `primary_platform == "Instagram" | primary_platform == "TikTok" | primary_platform == "Snapchat"`, ali je dramatično čitljiviji i manje podložan greškama. Kad imate pet ili više kategorija, `%in%` je jedini razuman izbor.

Logičke operatore možete kombinirati i u kontekstu vektora izvan tibbleova.

```
dobi <- c(19, 52, 21, 35, 23, 61, 20, 42)

# Koliko ispitanika je između 20 i 30 godina?
sum(dobi >= 20 & dobi <= 30)
```

```
[1] 3
```

```
# Koliko ih je mlađe od 20 ILI starije od 50?
sum(dobi < 20 | dobi > 50)
```

```
[1] 3
```

> Praktični savjet
>
> Česta greška je pisati `platforma == "Instagram" | "TikTok"` umjesto `platforma == "Instagram" | platforma == "TikTok"`. Prva verzija ne radi jer R interpretira `"TikTok"` kao samostalnu logičku vrijednost (neprazan tekst je uvijek TRUE), pa uvjet uvijek vraća TRUE. Koristite `%in%` da izbjegnete ovakve zamke.

---

13 Nedostajuće vrijednosti: NA

U stvarnom svijetu podaci gotovo nikad nisu potpuni. Ispitanik preskoči pitanje u anketi, senzor ne zabilježi podatak, sistem zapiše grešku. R koristi posebnu oznaku **`NA`** (not available) za nedostajuće vrijednosti i ove vrijednosti zahtijevaju posebnu pažnju od prvog dana jer se ponašaju drugačije od svega ostalog.

Temeljno pravilo je jednostavno i nemilosrdno. Svaka operacija koja uključuje NA vraća NA.

```
# Vektor s nedostajućom vrijednošću
ocjene <- c(4, 5, NA, 3, 4)

# Prosjek vektora s NA
mean(ocjene)
```

```
[1] NA
```

```
# Zbroj vektora s NA
sum(ocjene)
```

```
[1] NA
```

Rezultat je NA, ne broj. R ne pretpostavlja da možete ignorirati nedostajuću vrijednost jer ne znate što bi ta vrijednost bila. Možda je nedostajuća ocjena bila 1, možda 5, a možda nešto između. Prosjek s tom vrijednošću i bez nje bio bi različit. R vas prisiljava da svjesno odlučite što ćete učiniti.

Najčešće rješenje je argument `na.rm = TRUE` koji govori R-u da ignorira NA vrijednosti.

```
# Prosjek bez NA vrijednosti
mean(ocjene, na.rm = TRUE)
```

```
[1] 4
```

```
# Zbroj bez NA vrijednosti
sum(ocjene, na.rm = TRUE)
```

```
[1] 16
```

```
# Medijan, SD, min, max - svi imaju na.rm argument
median(ocjene, na.rm = TRUE)
```

```
[1] 4
```

```
sd(ocjene, na.rm = TRUE)
```

```
[1] 0.8164966
```

13.1 Provjera i prepoznavanje NA

Za otkrivanje NA vrijednosti koristimo funkciju `is.na()`, nikad usporedbu s `==`.

```
# ISPRAVNO: is.na()
is.na(ocjene)
```

```
[1] FALSE FALSE  TRUE FALSE FALSE
```

```
# Koliko je NA vrijednosti?
sum(is.na(ocjene))
```

```
[1] 1
```

```
# NEISPRAVNO: usporedba s == ne radi za NA
ocjene == NA
```

```
[1] NA NA NA NA NA
```

Usporedba `ocjene == NA` vraća niz NA vrijednosti, ne TRUE/FALSE. To je zato što je NA nepoznata vrijednost, a usporedba nečega nepoznatog s nepoznatim daje nepoznat rezultat. Ovo je logično kad se zamisli — ako ne znate koliko je Ana visoka i ne znate koliko je Marko visok, ne možete reći jesu li jednako visoki. Odgovor je "ne znam", dakle NA.

13.2 NA u tibbleovima

Kad učitavate podatke, prazne ćelije i tekstualne oznake poput "N/A" ili "missing" automatski se pretvaraju u NA (ili bi se trebale, ovisno o formatu). U tidyverse okruženju, provjera NA u cijelom datasetu izgleda ovako.

```
# Provjera NA za svaki stupac
social |>
  summarise(across(everything(), ~sum(is.na(.x))))
```

```
# A tibble: 1 x 12
  respondent_id   age age_group gender education primary_platform daily_minutes
          <int> <int>     <int>  <int>     <int>            <int>         <int>
1             0     0         0      0         0                0             0
# i 5 more variables: num_platforms <int>, trust_social_news <int>,
#   primary_news_source <int>, weekly_posts <int>, privacy_concern <int>
```

Naš simulirani dataset nema nedostajućih vrijednosti, ali u stvarnim podacima ih gotovo uvijek ima. Navikavanje na provjeru NA od prvog kontakta s podacima je navika koja vam štedi sate frustracije. Detaljno ćemo obraditi strategije za rad s nedostajućim vrijednostima u tjednu o deskriptivnoj statistici, uključujući razliku između podataka koji nedostaju nasumično i onih koji nedostaju sustavno.

> **!** Važna napomena
>
> Nikada nemojte pretpostaviti da vaši podaci nemaju NA. Čak i kad su podaci "čisti", funkcije poput `read_csv()` ponekad stvore NA na neočekivanim mjestima (prazna ćelija, razmak umjesto broja, tekst u numeričkom stupcu). Pravilo je jednostavno — uvijek provjerite, nikad ne pretpostavljajte.

---

14 Korisne funkcije za vektore

Prije nego prijeđemo na pisanje skripti, vrijedi proći još nekoliko funkcija koje ćete često koristiti. Sve rade na vektorima i pojavljuju se u gotovo svakoj analizi.

14.1 Generiranje nizova

```
# Niz cijelih brojeva od 1 do 10
1:10
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
# Niz s zadanim korakom
seq(from = 0, to = 100, by = 10)
```

```
 [1]   0  10  20  30  40  50  60  70  80  90 100
```

```
# Niz zadane duljine
seq(from = 0, to = 1, length.out = 5)
```

```
[1] 0.00 0.25 0.50 0.75 1.00
```

```
# Ponavljanje
rep("kontrolna", times = 5)
```

```
[1] "kontrolna" "kontrolna" "kontrolna" "kontrolna" "kontrolna"
```

```
rep(c("A", "B"), times = 3)
```

```
[1] "A" "B" "A" "B" "A" "B"
```

```r
rep(c("A", "B"), each = 3)
```

```
[1] "A" "A" "A" "B" "B" "B"
```

Funkcija `seq()` stvara pravilne nizove. Koristit ćemo je kad budemo trebali osi za grafike ili sekvence za simulacije. Funkcija `rep()` ponavlja vrijednosti i korisna je kad kreirate testne podatke ili oznake za eksperimentalne grupe. Obratite pažnju na razliku između `times` (ponavlja cijeli vektor) i `each` (ponavlja svaki element).

14.2 Sortiranje i redoslijed

```r
minute <- c(95, 22, 112, 45, 78, 8, 135, 55)

# Sortirano uzlazno
sort(minute)
```

```
[1]   8  22  45  55  78  95 112 135
```

```r
# Sortirano silazno
sort(minute, decreasing = TRUE)
```

```
[1] 135 112  95  78  55  45  22   8
```

```r
# Rang (pozicija u sortiranom nizu)
rank(minute)
```

```
[1] 6 2 7 3 5 1 8 4
```

```r
# Indeksi koji bi sortirali vektor
order(minute)
```

```
[1] 6 2 4 8 5 1 3 7
```

Funkcija `sort()` vraća sortirane vrijednosti. Funkcija `rank()` vraća rang svakog elementa (najmanji dobiva rang 1). Funkcija `order()` vraća indekse koji bi sortirali vektor, što je korisno za sortiranje jednog vektora prema redoslijedu drugog. U tidyverse pristupu češće koristimo `arrange()` za sortiranje tibbleova, ali `sort()` i `rank()` ostaju korisni za rad s pojedinačnim vektorima.

14.3 Jedinstvene vrijednosti i tablice frekvencija

```
platforme <- c("TikTok", "Instagram", "TikTok", "YouTube", "Instagram", "TikTok", "Faceboo

# Jedinstvene vrijednosti
unique(platforme)
```

```
[1] "TikTok"    "Instagram" "YouTube"   "Facebook"
```

```
# Broj jedinstvenih vrijednosti
length(unique(platforme))
```

```
[1] 4
```

```
# Tablica frekvencija (base R)
table(platforme)
```

```
platforme
 Facebook Instagram    TikTok   YouTube
        1         2         3         1
```

Funkcija `unique()` vraća sve različite vrijednosti u vektoru. `table()` prebrojava koliko se puta svaka vrijednost pojavljuje. U tidyverse pristupu, `count()` radi isto ali elegantnije i vraća tibble umjesto tablice. Ipak, `unique()` i `length(unique())` su toliko korisni da ih vrijedi znati neovisno o tidyverse.

14.4 Zaokruživanje i formatiranje

```
x <- 3.14159265

# Zaokruživanje na N decimala
round(x, 2)
```

```
[1] 3.14
```

```
round(x, 4)
```

```
[1] 3.1416
```

```r
# Zaokruživanje prema gore i dolje
ceiling(2.3)
```

```
[1] 3
```

```r
floor(2.9)
```

```
[1] 2
```

```r
# Značajne znamenke
signif(x, 3)
```

```
[1] 3.14
```

Funkcija `round()` se pojavljuje konstantno jer je rezultate statističkih izračuna gotovo uvijek potrebno zaokružiti prije prikazivanja. Konvencija u akademskim radovima je obično 2 decimale za korelacije i p-vrijednosti, 1 decimala za prosjeke i standardne devijacije. Na ovom kolegiju ćemo se držati tih konvencija.

---

15 Pisanje čistih R skripti

Do sada smo pisali kod redak po redak, ali u praksi ćete pisati **skripte**, datoteke koje sadrže sav kod za jednu analizu od početka do kraja. Čista skripta je nešto što možete dati kolegi, i kolega može pokrenuti vaš kod i dobiti identične rezultate. To je suština ponovljivosti o kojoj smo govorili na početku.

Dobra R skripta ima jasnu strukturu.

```r
# ============================================
# Analiza korištenja društvenih mreža
# Kolegij: Statistika za komunikologe
# Datum: 2025-03-01
# Autor: Ime Prezime
# ============================================

# 1. Učitavanje paketa ----
library(tidyverse)

# 2. Učitavanje podataka ----
```

```
social <- read_csv("resources/datasets/social_media_survey.csv")

# 3. Pregled podataka ----
glimpse(social)

# 4. Deskriptivna statistika po dobnim skupinama ----
social |>
  group_by(age_group) |>
  summarise(
    n = n(),
    prosjek_min = round(mean(daily_minutes), 1),
    sd_min = round(sd(daily_minutes), 1),
    .groups = "drop"
  )

# 5. Najpopularnije platforme ----
social |>
  count(primary_platform, sort = TRUE)
```

Primijetite nekoliko stvari o ovoj skripti. Na vrhu je zaglavlje u komentarima koje objašnjava što skripta radi, za koji kolegij je, tko ju je napisao i kad. Sekcije su označene komentarima s četiri crtice na kraju (`# Naslov ----`), što Positron prepoznaje i prikazuje kao navigacijske točke u bočnom panelu. Svaka sekcija ima jasan opis. Kod teče logički — najprije paketi, pa podaci, pa pregled, pa analiza.

15.1 Radni direktorij i putanje do datoteka

Jedna od najčešćih frustracija za početnike je problem s putanjama do datoteka. Kad napišete `read_csv("social_media_survey.csv")`, R traži tu datoteku u **radnom direktoriju** (working directory). Ako datoteka nije tamo, dobit ćete grešku.

```
# Koji je trenutni radni direktorij?
getwd()
```

```
[1] "C:/Users/lsikic/Dropbox/HKS/Kolegiji/Osnove statistike/GHub/lectures"
```

U Positronu, radni direktorij se obično automatski postavlja na mapu u kojoj je otvorena R datoteka ili projekt. To znači da ako je vaša skripta u mapi `projekt/analize/` i dataset u mapi `projekt/podaci/`, putanja u skripti bi bila `"../podaci/social_media_survey.csv"` (dvije točke znače "idi jednu mapu gore").

Praktični savjet

Najbolja praksa je koristiti **R projekte** (ili Quarto projekte) jer automatski postavljaju radni direktorij na korijensku mapu projekta. Kad otvorite projekt u Positronu, svi putovi su relativni prema toj mapi, i nikad ne morate razmišljati o apsolutnim putanjama poput `C:/Users/Ana/Documents/faks/statistika/podaci/...`. Apsolutne putanje su problem jer ne rade na tuđem računalu (kolega nema mapu `Ana` na svom disku). Relativne putanje rade svugdje jer polaze od mape projekta.

---

16 Spremanje podataka: write_csv()

Jednako važno kao učitavanje podataka jest njihovo spremanje. Nakon što očistite podatke ili izračunate nove varijable, želite pohraniti rezultat da ne morate ponavljati iste korake svaki put. Funkcija `write_csv()` sprema tibble u CSV datoteku.

```
# Kreiranje sažetka
sazetak_po_dobi <- social |>
  group_by(age_group) |>
  summarise(
    n = n(),
    prosjek_min = round(mean(daily_minutes), 1),
    sd_min = round(sd(daily_minutes), 1),
    prosjek_platformi = round(mean(num_platforms), 1),
    .groups = "drop"
  )

# Spremanje u CSV
write_csv(sazetak_po_dobi, "rezultati/sazetak_po_dobi.csv")
```

Funkcija `write_csv()` prima dva argumenta: tibble koji želite spremiti i putanju s imenom datoteke. Mapa `rezultati/` mora postojati prije nego pozovete funkciju, inače ćete dobiti grešku. Možete je kreirati ručno u datotečnom pregledniku ili iz R-a naredbom `dir.create("rezultati")`.

---

17 Traženje pomoći

Čak i iskusni R korisnici redovito trebaju pomoć. R ima ugrađeni sustav dokumentacije koji je izuzetno detaljan, i postoji nekoliko načina da mu pristupite.

```r
# Pomoć za specifičnu funkciju
?mean
help(mean)

# Pretraživanje pomoći po ključnoj riječi
??correlation

# Primjeri korištenja funkcije
example(mean)
```

Upitnik ispred imena funkcije (`?mean`) otvara stranicu pomoći za tu funkciju. Stranica sadrži opis, listu argumenata, detalje o ponašanju, povratnu vrijednost i primjere. Na početku stranice pomoći djeluju zastrašujuće jer su pisane tehničkim jezikom, ali brzo ćete naučiti preskočiti na sekciju **Examples** na dnu, koja gotovo uvijek postoji i pokazuje kako se funkcija koristi u praksi.

17.1 Kad pomoć ne pomaže: internet

Realno, za većinu problema ćete koristiti internet. Tri resursa su daleko najkorisnija.

Stack Overflow je forum za programerska pitanja. Gotovo svako pitanje o R-u koje možete zamisliti već je postavljeno i odgovoreno na Stack Overflowu. Ključ je znati kako formulirati pitanje za pretragu. Umjesto "moj kod ne radi", tražite "r dplyr filter multiple conditions" ili "r ggplot change axis labels".

Posit Community (community.rstudio.com) je forum specifičan za R i tidyverse. Atmosfera je prijateljska i odgovori su obično vrlo detaljni.

R dokumentacija i vinjete. Mnogi paketi dolaze s vinjetama (vignettes), dugačkim dokumentima koji objašnjavaju filozofiju paketa i pokazuju tipične radne tokove. Vinjete za dplyr (`vignette("dplyr")`) i ggplot2 su izvrsni resursi.

> Praktični savjet
>
> Kad tražite pomoć na internetu, uvijek uključite "tidyverse" ili ime paketa u pretragu. Bez toga, odgovori će često biti u base R sintaksi koja je drugačija od onoga što koristimo na kolegiju. Na primjer, tražite "tidyverse filter rows by condition" umjesto "R filter rows".

---

18 Česte greške i kako ih popraviti

Greške su normalan i neizbježan dio programiranja. Čak i nakon godina iskustva, R korisnici redovito dobivaju poruke o greškama. Razlika između početnika i iskusnog korisnika nije u tome koliko grešaka prave, nego u tome koliko brzo ih prepoznaju i poprave. Pogledajmo najčešće greške s kojima ćete se susresti.

18.1 Greška: objekt nije pronađen

```
# Error: object 'prosjek' not found
prosjek_minuta <- mean(social$daily_minutes)
prosjek   # krivi naziv, nedostaje "_minuta"
```

Ova greška znači da R ne može pronaći objekt s tim imenom. Najčešći uzroci su pogrešno ime (tipfeler), zaboravljeno pokretanje koda koji kreira objekt, ili pokretanje koda izvan redoslijeda (pokušavate koristiti objekt prije nego ste ga kreirali). Što trebate učiniti: provjerite ime i provjerite jeste li pokrenuli sve prethodne retke koda.

18.2 Greška: neočekivani simbol

```
# Error: unexpected symbol in "social |> filter(age < 30 daily_minutes > 60)"
social |> filter(age < 30 daily_minutes > 60)   # nedostaje &
```

R je naišao na nešto što ne očekuje. Najčešće nedostaje operator (`&`, `,`, `|>`), nedostaje zatvarajuća zagrada, ili ste zaboravili zarez između argumenata. Rješenje: pažljivo pregledajte redak i usporedite s ispravnom sintaksom.

18.3 Greška: datoteka nije pronađena

```
# Error: 'podaci.csv' does not exist in current working directory
read_csv("podaci.csv")
```

R ne može pronaći datoteku na zadanoj putanji. Provjerite je li naziv datoteke ispravan (uključujući velika i mala slova), je li datoteka u radnom direktoriju, i je li putanja ispravna. Koristite `getwd()` da vidite gdje R traži datoteke.

18.4 Upozorenje naspram greške

Važno je razlikovati **greške** (errors) od **upozorenja** (warnings). Greška zaustavlja izvršavanje koda jer R ne može nastaviti. Upozorenje ne zaustavlja izvršavanje ali vas obavještava da se nešto neobično dogodilo. Na primjer, kad pretvarate tekst u broj, a tekst sadrži slova.

```
# Ovo daje upozorenje ali ne grešku
as.numeric(c("10", "20", "trideset"))
```

```
Warning: NAs introduced by coercion
```

```
[1] 10 20 NA
```

R je uspio pretvoriti "10" i "20" u brojeve, ali "trideset" nije mogao pretvoriti i stavio je NA. Upozorenje vam govori da je nešto pošlo po krivu, ali kod se izvršio do kraja. Uvijek čitajte upozorenja jer vam govore o potencijalnim problemima s podacima.

> **!** Važna napomena
>
> Kad dobijete grešku, pročitajte poruku o grešci. Zvuči očito, ali većina početnika reagira panikom umjesto čitanjem. R poruke o greškama su obično informativne i govore vam što je pošlo po krivu. Tekst "object 'x' not found" vam doslovno govori da objekt x ne postoji. Tekst "unexpected symbol" govori da je nešto krivo sa sintaksom. Čitanje poruke je prvi i najvažniji korak u rješavanju problema.

---

19 Sve zajedno: kompletna mini analiza

Zaokružimo ovo predavanje tako da povežemo sve što smo naučili u jednu koherentnu analizu. Istražit ćemo koji izvor vijesti dominira u različitim dobnim skupinama i kako je korištenje društvenih mreža povezano s povjerenjem u vijesti na tim platformama.

```
# Primarni izvor vijesti po dobnim skupinama
social |>
  group_by(age_group, primary_news_source) |>
  summarise(n = n(), .groups = "drop") |>
  group_by(age_group) |>
  mutate(postotak = round(n / sum(n) * 100, 1)) |>
  arrange(age_group, desc(postotak))
```

```
# A tibble: 25 x 4
# Groups:   age_group [5]
   age_group primary_news_source     n postotak
   <chr>     <chr>                <int>    <dbl>
 1 18-24     drustvene_mreze         80     47.9
 2 18-24     portal                  50     29.9
 3 18-24     TV                      25     15
 4 18-24     print                    7      4.2
 5 18-24     radio                    5      3
 6 25-34     drustvene_mreze         55     42.3
 7 25-34     portal                  37     28.5
 8 25-34     TV                      18     13.8
 9 25-34     print                   10      7.7
10 25-34     radio                   10      7.7
# i 15 more rows
```

Ovaj kod radi nešto složenije nego što smo do sada vidjeli. Najprije grupira podatke po dobnoj skupini i izvoru vijesti te broji ispitanike. Zatim, unutar svake dobne skupine, računa postotak. Funkciju `mutate()` koristimo za kreiranje novog stupca, a detalje ćemo objasniti sljedeći tjedan. Za sada je dovoljno vidjeti obrazac i rezultat.

Vidimo jasnu razliku između generacija. Mladi (18 do 24) dominantno koriste društvene mreže kao primarni izvor vijesti, dok stariji ispitanici (55+) preferiraju televiziju. Ovo je konzistentno s istraživanjima diljem svijeta i ilustrira zašto je raščlamba po dobnim skupinama toliko važna, tema koju smo obradili i u prvom tjednu kad smo govorili o Simpsonovom paradoksu.

Pogledajmo sada vezu između dnevnog korištenja i povjerenja u vijesti na društvenim mrežama.

```
social |>
  group_by(age_group) |>
  summarise(
    n = n(),
    prosjek_min = round(mean(daily_minutes), 1),
    prosjek_trust = round(mean(trust_social_news), 1),
    korelacija = round(cor(daily_minutes, trust_social_news), 2),
    .groups = "drop"
  )
```

```
# A tibble: 5 x 5
  age_group     n prosjek_min prosjek_trust korelacija
  <chr>     <int>       <dbl>         <dbl>      <dbl>
1 18-24       167       146.            5.4      -0.06
2 25-34       130        95.9           4.5       0.12
3 35-44       107        61.2           3.7      -0.04
```

```
4 45-54        58        40.1           3         0.04
5 55+          38        26.8           2.5       0.33
```

Ovo je tablica koja komunicira mnogo informacija. Za svaku dobnu skupinu vidimo koliko ispitanika imamo, koliko u prosjeku koriste društvene mreže, koliko im vjeruju kao izvoru vijesti i kakva je korelacija između korištenja i povjerenja unutar svake skupine. Ovakve tablice čine okosnicu svakog deskriptivnog izvještaja u komunikologiji.

Na kraju, pogledajmo koliko platformi u prosjeku koriste različite dobne skupine i kako se to razlikuje po spolu.

```
social |>
  group_by(age_group, gender) |>
  summarise(
    n = n(),
    prosjek_platformi = round(mean(num_platforms), 1),
    .groups = "drop"
  ) |>
  filter(gender != "non-binary") |>
  arrange(age_group, gender)
```

```
# A tibble: 10 x 4
   age_group gender     n prosjek_platformi
   <chr>     <chr>  <int>             <dbl>
 1 18-24     female    72               4
 2 18-24     male      89               4
 3 25-34     female    71               3.2
 4 25-34     male      55               3.1
 5 35-44     female    44               3
 6 35-44     male      63               2.7
 7 45-54     female    34               1.7
 8 45-54     male      23               1.7
 9 55+       female    17               1.5
10 55+       male      21               1.7
```

Mladi koriste više platformi od starijih ispitanika, što je logično. Razlika između spolova je relativno mala u usporedbi s razlikom između dobnih skupina. Ovo je opet ilustracija važnog principa. Kad gledate ukupne prosjeke, gubite informaciju o tome koja varijabla zapravo objašnjava razlike.

> Svaka analiza podataka počinje s pitanjem. Dobar analitičar ne otvara dataset i "vidi što će pronaći". Dobar analitičar ima pitanje, prevede ga u kod i interpretira rezultat u kontekstu tog pitanja.

! Ključni zaključci

1. R je programski jezik za statističko računanje koji nudi ponovljivost, fleksibilnost i profesionalnu vizualizaciju. Početna krivulja učenja je strmija od softvera s grafičkim sučeljem, ali dugoročna isplativost je znatno veća.

2. Positron je moderno razvojno okruženje (IDE) koje čini rad s R-om ugodnijim. Radni tok uključuje pisanje koda u editoru, izvršavanje u konzoli i pregled rezultata.

3. Objekti pohranjuju vrijednosti za kasniju upotrebu. Koristite opisna imena u snake_case konvenciji i ne reciklirajte objekte za različite svrhe.

4. Vektori su uređeni nizovi vrijednosti istog tipa. R automatski primjenjuje operacije na sve elemente vektora (vektorizacija), što omogućuje efikasan rad s podacima.

5. Četiri osnovna tipa podataka su numerički (`numeric`), tekstualni (`character`), logički (`logical`) i faktorski (`factor`). Razumijevanje tipova ključno je za dijagnosticiranje problema s podacima.

6. Tibble je moderna tablica podataka u kojoj svaki stupac može biti drugog tipa. Standard je u tidyverse ekosustavu.

7. Pipe operator (`|>`) čini kod čitljivijim jer omogućuje ulančavanje operacija u prirodnom redoslijedu, odozgo prema dolje. Koristite ga uvijek kad imate više od jednog koraka.

8. Funkcija `read_csv()` učitava CSV datoteke u tibble. Nakon učitavanja, uvijek pregledajte podatke s `glimpse()`, `head()` i `count()`.

9. Logički operatori (`&`, `|`, `!`, `%in%`) omogućuju kombiniranje uvjeta za precizno filtriranje i odabir podataka.

10. NA (nedostajuće vrijednosti) zahtijevaju svjesnu odluku o tretmanu. Uvijek provjerite ima li ih u podacima i koristite `na.rm = TRUE` kad je prikladno.

11. Greške su normalan dio programiranja. Čitajte poruke o greškama, provjerite imena objekata i zagrade, i koristite internet kao resurs za rješavanje problema.

12. Čista R skripta ima jasnu strukturu s zaglavljem, učitavanjem paketa, učitavanjem podataka, pregledom podataka i analizom. Koristite komentare i relativne putanje.

⚠ Priprema za sljedeći tjedan

Sljedeći tjedan nastavljamo s radom s podacima u tidyverse. Naučit ćemo temeljne funkcije za manipulaciju podacima — `filter()` za odabir redova po uvjetu, `select()` za odabir stupaca, `mutate()` za kreiranje novih varijabli, `arrange()` za sortiranje, te `pivot_longer()` i `pivot_wider()` za preoblikovanje podataka. Ove funkcije, u kombinaciji s `group_by()` i `summarise()` koje smo već upoznali, čine okosnicu svake analize podataka u R-u.
Za pripremu napravite sljedeće:

1. Provjerite da vam R i Positron rade ispravno. Otvorite Positron, stvorite novu R datoteku i pokrenite `library(tidyverse)`. Ako ne dobijete grešku, spremni ste.
2. Ponovite sve primjere iz ovog predavanja. Nemojte samo čitati kod, nego ga upišite i pokrenite. Trebate eksperimentirati primjenom promjena brojeva, dodavanja novih vektora i testiranja što se dogodi kad nešto napravite krivo.
3. Učitajte dataset `social_media_survey.csv` i pokušajte odgovoriti na sljedeća pitanja koristeći `filter()`, `count()` i `group_by() |> summarise()`:
   - Koja je najčešća platforma među ispitanicima starijim od 45 godina?
   - Koliki je prosječni broj dnevnih minuta za korisnike Instagrama u usporedbi s korisnicima Facebooka?
   - Koliko ispitanika koristi 5 ili više platformi?
4. Pročitajte poglavlja 3 i 4 iz knjige R for Data Science (besplatno online na r4ds.hadley.nz). Poglavlje 3 pokriva osnove tidyverse radnog toka, a poglavlje 4 transformaciju podataka s dplyr.

---

20 Dodatno čitanje

Obavezno

Navarro, D. (2018). *Learning Statistics with R*, Chapter 3: Getting Started with R. Besplatno dostupno na learningstatisticswithr.com. Poglavlje pokriva iste teme kao ovo predavanje, ali koristi base R pristup umjesto tidyverse. Korisno za razumijevanje osnova jezika, ali kod iz knjige ne koristimo izravno na kolegiju.

Wickham, H. & Grolemund, G. (2023). *R for Data Science* (2nd edition), Chapters 1, 3 i 4. Besplatno dostupno na r4ds.hadley.nz. Poglavlje 1 daje motivaciju za tidyverse pristup, poglavlje 3 pokriva transformaciju podataka, a poglavlje 4 organizaciju radnog toka.

Preporučeno

Ismay, C. & Kim, A. (2020). *Statistical Inference via Data Science: A ModernDive into R and the Tidyverse*. Besplatno dostupno na moderndive.com. Alternativni udžbenik koji od početka koristi tidyverse i naglašava vizualno razmišljanje.

Bryan, J. & Hester, J. *What They Forgot to Teach You About R*. Besplatno dostupno na rstats.wtf. Pokriva praktične aspekte rada s R-om: projekte, putanje, radne direktorije, organizaciju datoteka. Čitanje za one koji žele profesionalizirati svoj radni tok.

---

21 Pojmovnik

Pojam	Objašnjenje
R	Programski jezik i okruženje za statističko računanje i vizualizaciju. Besplatan i open-source.
Positron	Moderno integrirano razvojno okruženje (IDE) za rad s R-om, razvijeno od strane Posit tima.
IDE (integrirano razvojno okruženje)	Program koji kombinira editor teksta, konzolu, pregled varijabli i druge alate u jednom sučelju.
Objekt	Pohranjena vrijednost u R-u kojoj se pristupa putem imena. Kreira se operatorom `<-`.
Vektor	Uređeni niz vrijednosti istog tipa. Temeljna struktura podataka u R-u. Kreira se funkcijom `c()`.
Vektorizacija	Svojstvo R-a da automatski primjenjuje operacije na sve elemente vektora odjednom.
Tip podataka	Klasifikacija vrijednosti koja određuje moguće operacije. Osnovni tipovi: numeric, character, logical, factor.
Faktor (factor)	Poseban tip podataka za kategorijalne varijable. Sadrži unaprijed definirane razine (levels).
Tibble	Moderna verzija data framea iz tidyverse ekosustava. Prikazuje tipove stupaca i ograničava ispis na prvih 10 redova.
Data frame	Tablica u R-u u kojoj svaki stupac može biti drugog tipa. Tibble je poboljšana verzija.
Pipe operator (`|>`)	Operator koji proslijeđuje rezultat jednog izraza kao prvi argument sljedećoj funkciji. Čini kod čitljivijim.
Tidyverse	Kolekcija R paketa za rad s podacima koji dijele zajedničku filozofiju dizajna. Uključuje ggplot2, dplyr, tidyr, readr, tibble i druge.

Pojam	Objašnjenje
Paket (package)	Kolekcija R funkcija, podataka i dokumentacije koja proširuje mogućnosti R-a. Instalira se s `install.packages()`, učitava s `library()`.
`read_csv()`	Funkcija iz paketa readr za učitavanje CSV datoteka. Vraća tibble i automatski pogađa tipove stupaca.
`write_csv()`	Funkcija iz paketa readr za spremanje tibble u CSV datoteku. Korisna za izvoz obrađenih podataka.
`glimpse()`	Funkcija iz tidyverse koja prikazuje strukturu dataseta: stupce, tipove i prvih nekoliko vrijednosti.
`count()`	Funkcija iz dplyr koja prebrojava opažanja po kategorijama.
`group_by()`	Funkcija iz dplyr koja dijeli podatke u grupe po jednoj ili više varijabli. Koristi se u kombinaciji sa `summarise()`.
`summarise()`	Funkcija iz dplyr koja izračunava sažetke (prosjek, medijan, SD i sl.) za svaku grupu ili za cijeli dataset.
`filter()`	Funkcija iz dplyr koja odabire retke koji zadovoljavaju zadani uvjet.
`mutate()`	Funkcija iz dplyr koja kreira nove stupce ili mijenja postojeće.
`arrange()`	Funkcija iz dplyr koja sortira retke po vrijednostima jednog ili više stupaca.
`select()`	Funkcija iz dplyr koja odabire stupce po imenu.
NA (not available)	Oznaka za nedostajuću vrijednost u R-u. Svaka operacija s NA vraća NA osim ako se eksplicitno kaže `na.rm = TRUE`.
`is.na()`	Funkcija koja provjerava jesu li vrijednosti NA. Jedini ispravan način provjere (nikad koristiti `== NA`).
Logički operatori	Operatori za kombiniranje uvjeta: `&` (i), `|` (ili), `!` (ne), `%in%` (pripada skupu).
Skripta	Tekstualna datoteka (.R) koja sadrži R kod od početka do kraja analize. Omogućuje ponovljivost.
Radni direktorij	Mapa u kojoj R traži datoteke i sprema rezultate. Provjerava se s `getwd()`.

Pojam	Objašnjenje
snake_case	Konvencija imenovanja: riječi odvojene podvlakom, sve malim slovima (npr. `dnevno_koristenje`). Standard u tidyverse zajednici.
CSV (comma-separated values)	Tekstualni format za pohranu tabličnih podataka u kojem su vrijednosti odvojene zarezima.
Komentar	Tekst u kodu koji počinje s `#` i koji R ignorira. Služi za objašnjavanje koda.