

Predavanje 4 – data.frame, operacije nad vrsticami in stolpci

Odstranjevanje stolpca po imenu

V R-ju lahko odstranimo stolpec, tudi če poznamo le njegovo ime. Poglejmo si ponovno primer data.frame-a od zadnjič.

```
visina <- c(179, 185, 183, 172, 174, 185, 193, 169, 173, 168)
teza <- c(75, 89, 70, 80, 58, 86, 73, 63, 72, 70)
spol <- c("f", "m", "m", "m", "f", "m", "f", "f", "m", "f")
imena <- c('Micka', 'Marko', 'Gregor', 'Tomaz', 'Ana', 'Peter',
           'Mojca', 'Katja', 'Anze', 'Alja')
df <- data.frame(spol, visina, teza, imena)
print(df)
```

```
##      spol visina teza  imena
## 1      f    179   75  Micka
## 2      m    185   89  Marko
## 3      m    183   70 Gregor
## 4      m    172   80  Tomaz
## 5      f    174   58   Ana
## 6      m    185   86  Peter
## 7      f    193   73  Mojca
## 8      f    169   63  Katja
## 9      m    173   72   Anze
## 10     f    168   70   Alja
```

Če želimo odstraniti stolpec z imenom **teza**, najprej poiščemo indeks tega stolpca. Pri tem si pomagamo s funkcijo `which()`, ki nam vrne indeks iskanega elementa v zaporedju.

```
ind <- which(names(df) == 'teza')
```

Vidimo, da je stolpec **teza** tretji v data.frame-u. V spremenljivko **ind** smo shranili njegov indeks. Sedaj ga lahko odstranimo:

```
df[, -ind]
```

```
##      spol visina  imena
## 1      f    179  Micka
## 2      m    185  Marko
## 3      m    183 Gregor
## 4      m    172  Tomaz
## 5      f    174   Ana
## 6      m    185  Peter
## 7      f    193  Mojca
```

```
## 8      f      169  Katja
## 9      m      173   Anze
## 10     f      168   Alja
```

Ko želimo odstraniti več stolpcev katerih imena poznamo, si ponavadi pomagamo s funkcijo `setdiff()`, ki nam vrne razliko v elementih dveh množic. Naša prva množica bodo vsa imena stolpcev v `data.frame`-u.

```
mn1 <- names(df)
print(mn1)
```

```
## [1] "spol" "visina" "teza" "imena"
```

Druga množica pa bodo imena, ki jih ne želimo prikazati.

```
mn2 <- c('spol', 'imena')
print(mn2)
```

```
## [1] "spol" "imena"
```

Sedaj pogledamo, katera imena so v množici **mn1** in jih v **mn2** ni. V bistvu nekako iz množice **mn1** odstranimo imena, ki so **mn2**.

```
mnd <- setdiff(mn1, mn2)
print(mnd)
```

```
## [1] "visina" "teza"
```

Prikažemo `data.frame` brez stolpcev **spol** in **imena**.

```
print(df[,mnd])
```

```
##      visina teza
## 1      179   75
## 2      185   89
## 3      183   70
## 4      172   80
## 5      174   58
## 6      185   86
## 7      193   73
## 8      169   63
## 9      173   72
## 10     168   70
```

Operacije nad vrsticami ali stolpci `data.frame`-a

Numerične vrstice ali stolpe lahko med seboj seštevamo, odštevamo, množimo ali delimo.

Lahko izračunamo BMI udeležencev:

```
df$teza / (df$visina / 100)^2
```

```
## [1] 23.40751 26.00438 20.90239 27.04164 19.15709 25.12783 19.59784 22.05805  
## [9] 24.05693 24.80159
```

Ne moremo pa izvajati aritmetičnih operacij med numeričnimi in znakovnimi stolpci ali vrsticami.

```
df$visina + df$spol
```

```
## Error in df$visina + df$spol: non-numeric argument to binary operator
```

Ne moremo npr. sešteti celotnih vrstic med seboj, ker se znakovni tipi ne seštevajo med seboj:

```
df[1, ] + df[2, ]
```

```
## Error in FUN(left, right): non-numeric argument to binary operator
```

Izvajamo lahko matematične operacije nad posameznimi stolpci.

Stolpcu **visina** prištejemo 5 cm.

```
df$visina + 5
```

```
## [1] 184 190 188 177 179 190 198 174 178 173
```

Stolpec lahko logaritmiramo.

```
log(df$teza)
```

```
## [1] 4.317488 4.488636 4.248495 4.382027 4.060443 4.454347 4.290459 4.143135  
## [9] 4.276666 4.248495
```

Obdelava podatkov o delcih PM10 v Kranju

Poglejmo si podatke o vrednostih raznih snovi v delcih PM10. Podatki so priloženi v repozitoriju.

```
dat <- read.table('./data_raw/delci2.csv',  
                  header = TRUE,  
                  dec = ".",  
                  sep = ",",  
                  quote = "\"")
```

Pogledamo kakšen je prebrani data.frame:

```
head(dat)
```

```
##      Datum PM10    Ca    Cl    K    Mg    Na    NH4    NO3 kraj
## 1 1/17/2014  22 0.186 0.297 0.577 0.0374 0.1450 0.639 1.98 Kranj
## 2 1/18/2014  32 0.132 0.528 0.735 0.0235 0.1090 0.877 2.71 Kranj
## 3 1/19/2014  30 0.145 0.381 0.577 0.0363 0.1590 1.080 2.72 Kranj
## 4 1/20/2014  16 0.127 0.170 0.383 0.0428 0.0608 0.628 2.01 Kranj
## 5 1/21/2014  24 0.202 0.160 0.418 0.0365 0.0346 1.220 3.62 Kranj
## 6 1/22/2014  32 0.610 0.231 0.615 0.0734 0.0468 1.140 3.83 Kranj
```

```
summary(dat)
```

```
##      Datum      PM10      Ca      Cl
## Length:336      Min.   : 2.80      Min.   :0.0239      Min.   :0.01140
## Class :character 1st Qu.: 12.00      1st Qu.:0.1495      1st Qu.:0.04085
## Mode  :character Median : 18.00      Median :0.2500      Median :0.06245
##                      Mean  : 22.44      Mean  :0.3506      Mean  :0.15003
##                      3rd Qu.: 28.25      3rd Qu.:0.4420      3rd Qu.:0.19750
##                      Max.   :100.00      Max.   :1.6200      Max.   :1.47000
##      K      Mg      Na      NH4
## Min.   :0.0161      Min.   :0.00139      Min.   :0.00396      Min.   :0.0250
## 1st Qu.:0.0882      1st Qu.:0.03485      1st Qu.:0.03578      1st Qu.:0.4323
## Median :0.1770      Median :0.05635      Median :0.05840      Median :0.7610
## Mean   :0.3257      Mean   :0.06930      Mean   :0.08895      Mean   :1.2008
## 3rd Qu.:0.4828      3rd Qu.:0.09613      3rd Qu.:0.10450      3rd Qu.:1.4600
## Max.   :4.1300      Max.   :0.30400      Max.   :0.77600      Max.   :6.6500
##      NO3      kraj
## Min.   : 0.0487      Length:336
## 1st Qu.: 0.3347      Class :character
## Median : 0.8095      Mode  :character
## Mean   : 2.0964
## 3rd Qu.: 2.7250
## Max.   :19.6000
```

Recimo, da želimo izbrati vrednosti PM10 v Celju.

Če nas zanimajo imena krajev v data.frame-u in ne želimo izpisati večkrat istih imen, uporabimo funkcijo `unique()`. Če imamo vektor:

```
vek <- c(8, 8, 8, 9, 9, 9, 9, 7, 7, 7, 2)
unique(vek)
```

```
## [1] 8 9 7 2
```

```
unique(dat$kraj)
```

```
## [1] "Kranj"      "Ljubljana" "Celje"
```

Poleg tega, da izberemo podatke za Celje, izberemo le tiste, kjer je vrednost PM10 večja od 30 in jih shranimo kot drugi data.frame (`datPM10`)

```
datPM10 <- dat[dat$kraj == 'Celje' & dat$PM10 > 30, ]
```

Recimo, da merilnik izmeri 22% nižjo vrednost, ko so vrednosti nad 30. To napako želimo popraviti. Izračunamo vrednost napake:

```
datPM10$PM10err <- datPM10$PM10 * 0.22
```

Preverimo data.frame:

```
head(datPM10)
```

```
##      Datum PM10    Ca    Cl    K    Mg    Na NH4   NO3 kraj PM10err
## 235  9/8/2014  31 1.260 0.0114 0.326 0.225 0.0750 1.02 0.575 Celje    6.82
## 245  9/18/2014 33 1.170 0.1150 0.298 0.175 0.0508 1.84 0.652 Celje    7.26
## 263 10/6/2014  37 0.857 0.0621 0.381 0.188 0.0906 3.89 0.809 Celje    8.14
## 264 10/7/2014  41 1.190 0.0739 0.357 0.256 0.0756 3.00 1.120 Celje    9.02
## 285 10/28/2014 36 0.895 0.1710 0.435 0.155 0.0864 1.67 3.510 Celje    7.92
## 286 10/29/2014 42 0.977 0.2190 0.548 0.173 0.0521 1.97 4.660 Celje    9.24
```

```
summary(datPM10)
```

```
##      Datum      PM10      Ca      Cl
## Length:36      Min.   : 31.00      Min.   :0.0870      Min.   :0.0114
## Class :character 1st Qu.: 38.50      1st Qu.:0.2062      1st Qu.:0.2505
## Mode  :character Median : 44.00      Median :0.4130      Median :0.4275
##                      Mean  : 47.39      Mean  :0.5206      Mean  :0.4902
##                      3rd Qu.: 50.25      3rd Qu.:0.8488      3rd Qu.:0.6232
##                      Max.   :100.00      Max.   :1.6200      Max.   :1.4700
##      K      Mg      Na      NH4
## Min.   :0.2980      Min.   :0.02380      Min.   :0.03690      Min.   :0.389
## 1st Qu.:0.6232      1st Qu.:0.05445      1st Qu.:0.06482      1st Qu.:1.617
## Median :0.7900      Median :0.09460      Median :0.09180      Median :2.030
## Mean   :0.9083      Mean   :0.10893      Mean   :0.16787      Mean   :2.555
## 3rd Qu.:0.8760      3rd Qu.:0.14375      3rd Qu.:0.24750      3rd Qu.:3.163
## Max.   :4.1300      Max.   :0.30400      Max.   :0.71400      Max.   :6.620
##      NO3      kraj      PM10err
## Min.   : 0.575      Length:36      Min.   : 6.82
## 1st Qu.: 3.470      Class :character 1st Qu.: 8.47
## Median : 5.690      Mode  :character Median : 9.68
## Mean   : 5.645                      Mean  :10.43
## 3rd Qu.: 6.955                      3rd Qu.:11.05
## Max.   :13.100                      Max.   :22.00
```

Izračunamo še vrednosti PM10 s popravkom:

```
datPM10$PM10corr <- datPM10$PM10 + datPM10$PM10err
```

Zanima nas še vrednost soli (NaCl) v delcih, zato seštejemo vrednosti Na in Cl in to vrednost shranimo:

```
datPM10$NaCl <- datPM10$Na + datPM10$Cl
```

Posodobljeni data.frame shranimo.

```
write.csv(datPM10, './data_clean/delci_popravljeni.csv')
```

Funkcija `apply()`

Funkcija `apply()` nam omogoča, da apliciramo neko funkcijo na vse vrstice ali stolpce. Recimo, da želimo izračunati maksimalne vrednosti vseh stolpcev našega `data.frame`-a. Zato lahko uporabimo funkcijo `max()`: Npr. izračunamo maksimalno vrednost kalcija:

```
max(datPM10$Ca)
```

```
## [1] 1.62
```

Glede na to, da je stolpcev različnih snovi 9, je uporaba funkcije `max()` na vsakem stolpcu posebej dolgotrajna. Tukaj nam pomaga funkcija `apply()`, da funkcijo `max()` apliciramo na vseh stolpcih hkrati.

Funkcijo `apply()` uporabimo samo na numeričnih elementih, zato odstranimo nenumerične stolpce. Odstraniti moramo stolpca **datum** in **kraj**:

```
datPM10n <- datPM10[, setdiff(names(datPM10), c('Datum', 'kraj'))]
```

Na primer želimo izračunati največje vrednosti vsakega stolpca posebej.

```
apply(datPM10n, 2, max)
```

```
##      PM10      Ca      Cl      K      Mg      Na      NH4      NO3
## 100.000   1.620   1.470   4.130   0.304   0.714   6.620  13.100
## PM10err PM10corr   NaCl
##   22.000  122.000   1.796
```

Funkciji `apply` podamo 3 argumente:

- 1) Numerični `data.frame`, za katerega želimo izračun.
- 2) Število 1 ali 2. Če želimo izračun za vsak stolpec, uporabimo 2. Če želimo izračun za vsako vrstico, uporabimo 1.
- 3) Ime funkcije, katere vrednost želimo izračunati (v narekovajih).

Izračunamo najmanjše vrednosti stolpcev.

```
apply(datPM10n, 2, min)
```

```
##      PM10      Ca      Cl      K      Mg      Na      NH4      NO3
##  31.0000   0.0870   0.0114   0.2980   0.0238   0.0369   0.3890   0.5750
## PM10err PM10corr   NaCl
##   6.8200  37.8200   0.0864
```

Izračunamo povprečne vrednosti stolpcev:

```
apply(datPM10n, 2, mean)
```

```
##      PM10      Ca      Cl      K      Mg      Na      NH4
## 47.3888889 0.5205833 0.4902056 0.9083333 0.1089306 0.1678694 2.5549722
##      NO3      PM10err      PM10corr      NaCl
## 5.6454444 10.4255556 57.8144444 0.6580750
```

Seštejemo vrednosti v stolpcih:

```
apply(datPM10n, 2, sum)
```

```
##      PM10      Ca      Cl      K      Mg      Na      NH4      NO3
## 1706.0000 18.7410 17.6474 32.7000 3.9215 6.0433 91.9790 203.2360
##      PM10err      PM10corr      NaCl
## 375.3200 2081.3200 23.6907
```

Podobno lahko uporabimo `apply()` na vrsticah. Izračunaj seštevka PM10 in drugih snovi v vsaki vrstici:

```
datPM10$PM10total <- apply(datPM10n, 1, sum)
head(datPM10)
```

```
##      Datum PM10      Ca      Cl      K      Mg      Na NH4      NO3 kraj PM10err
## 235  9/8/2014  31 1.260 0.0114 0.326 0.225 0.0750 1.02 0.575 Celje 6.82
## 245  9/18/2014 33 1.170 0.1150 0.298 0.175 0.0508 1.84 0.652 Celje 7.26
## 263 10/6/2014  37 0.857 0.0621 0.381 0.188 0.0906 3.89 0.809 Celje 8.14
## 264 10/7/2014  41 1.190 0.0739 0.357 0.256 0.0756 3.00 1.120 Celje 9.02
## 285 10/28/2014 36 0.895 0.1710 0.435 0.155 0.0864 1.67 3.510 Celje 7.92
## 286 10/29/2014 42 0.977 0.2190 0.548 0.173 0.0521 1.97 4.660 Celje 9.24
##      PM10corr      NaCl      PM10total
## 235 37.82 0.0864 79.2188
## 245 40.26 0.1658 84.9866
## 263 45.14 0.1527 96.7104
## 264 50.02 0.1495 106.2620
## 285 43.92 0.2574 95.0198
## 286 51.24 0.2711 111.3502
```

Delo z datumi

Datum je posebna podatkovna struktura. Prej smo videli, da so datumi v `data.frame` prebrani kot besedilo. Če želimo delati z datumi moramo R-ju povedati, da nek stolpec vsebuje datume.

Če v R-ju želimo definirati datum:

```
as.Date("2020-02-01")
```

```
## [1] "2020-02-01"
```

Na tak način poskusimo spremeniti celo stolpec v datume.

```
print(as.Date(datPM10$Datum))
```

```
## [1] "0009-08-20" NA "0010-06-20" "0010-07-20" NA
## [6] NA NA NA "0011-01-20" "0011-02-20"
## [11] "0011-03-20" NA NA NA NA
## [16] NA NA NA NA NA
## [21] NA NA NA NA NA
## [26] "0001-01-20" "0001-02-20" "0001-03-20" "0001-06-20" "0001-07-20"
## [31] "0001-08-20" "0001-09-20" "0001-10-20" "0001-12-20" NA
## [36] NA
```

Datum mora biti v ISO formatu, če ni, moramo format definirati.

```
datPM10$Datum <- as.Date(datPM10$Datum, format = "%m/%d/%y")
print(head(datPM10,10))
```

```
## Datum PM10 Ca Cl K Mg Na NH4 NO3 kraj PM10err
## 235 2020-09-08 31 1.260 0.0114 0.326 0.2250 0.0750 1.02 0.575 Celje 6.82
## 245 2020-09-18 33 1.170 0.1150 0.298 0.1750 0.0508 1.84 0.652 Celje 7.26
## 263 2020-10-06 37 0.857 0.0621 0.381 0.1880 0.0906 3.89 0.809 Celje 8.14
## 264 2020-10-07 41 1.190 0.0739 0.357 0.2560 0.0756 3.00 1.120 Celje 9.02
## 285 2020-10-28 36 0.895 0.1710 0.435 0.1550 0.0864 1.67 3.510 Celje 7.92
## 286 2020-10-29 42 0.977 0.2190 0.548 0.1730 0.0521 1.97 4.660 Celje 9.24
## 287 2020-10-30 51 1.320 0.2190 0.731 0.1980 0.0693 1.84 6.000 Celje 11.22
## 288 2020-10-31 49 0.846 0.2690 0.671 0.1790 0.0894 2.43 7.330 Celje 10.78
## 289 2020-11-01 44 0.418 0.2530 0.517 0.0910 0.0622 2.89 8.130 Celje 9.68
## 290 2020-11-02 40 0.437 0.2430 0.679 0.0971 0.0369 1.46 2.210 Celje 8.80
## PM10corr NaCl PM10total
## 235 37.82 0.0864 79.2188
## 245 40.26 0.1658 84.9866
## 263 45.14 0.1527 96.7104
## 264 50.02 0.1495 106.2620
## 285 43.92 0.2574 95.0198
## 286 51.24 0.2711 111.3502
## 287 62.22 0.2883 135.1056
## 288 59.78 0.3584 131.7328
## 289 53.68 0.3152 120.0364
## 290 48.80 0.2799 103.0429
```

Če želimo bolj napredno delati z datumi imamo na voljo paket **lubridate**.

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
## date, intersect, setdiff, union
```

Iz datumov lahko izluščimo dneve:


```
day(datPM10$Datum)
```

```
## [1] 8 18 6 7 28 29 30 31 1 2 3 22 25 26 27 28 29 17 19 22 23 27 29 30 31
## [26] 1 2 3 6 7 8 9 10 12 15 16
```

...mesece:

```
month(datPM10$Datum)
```

```
## [1] 9 9 10 10 10 10 10 10 11 11 11 11 11 11 11 11 12 12 12 12 12 12 12
## [26] 1 1 1 1 1 1 1 1 1 1 1 1
```

... ali leta

```
year(datPM10$Datum)
```

```
## [1] 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
## [16] 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 2020
## [31] 2020 2020 2020 2020 2020 2020
```

Vrne nam tudi zaporedni dan v letu (Julijanski dan):

```
yday(datPM10$Datum)
```

```
## [1] 252 262 280 281 302 303 304 305 306 307 308 327 330 331 332 333 334 352 354
## [20] 357 358 362 364 365 366 1 2 3 6 7 8 9 10 12 15 16
```

S pomočjo funkcije **weekdays** lahko ugotovimo tudi, kateri dan v tednu je bil:

```
weekdays(datPM10$Datum)
```

```
## [1] "Tuesday" "Friday" "Tuesday" "Wednesday" "Wednesday" "Thursday"
## [7] "Friday" "Saturday" "Sunday" "Monday" "Tuesday" "Sunday"
## [13] "Wednesday" "Thursday" "Friday" "Saturday" "Sunday" "Thursday"
## [19] "Saturday" "Tuesday" "Wednesday" "Sunday" "Tuesday" "Wednesday"
## [25] "Thursday" "Wednesday" "Thursday" "Friday" "Monday" "Tuesday"
## [31] "Wednesday" "Thursday" "Friday" "Sunday" "Wednesday" "Thursday"
```

Primer izbire podatkov samo za en mesec:

```
datPM10[month(datPM10$Datum) == 11,]
```

```
## Datum PM10 Ca Cl K Mg Na NH4 NO3 kraj PM10err
## 289 2020-11-01 44 0.418 0.253 0.517 0.0910 0.0622 2.89 8.13 Celje 9.68
## 290 2020-11-02 40 0.437 0.243 0.679 0.0971 0.0369 1.46 2.21 Celje 8.80
## 291 2020-11-03 49 0.693 0.429 0.789 0.1400 0.0531 1.67 5.84 Celje 10.78
## 296 2020-11-22 45 0.494 0.455 0.791 0.1140 0.0585 2.49 6.40 Celje 9.90
## 299 2020-11-25 35 0.157 0.221 0.639 0.0293 0.0657 4.75 6.83 Celje 7.70
## 300 2020-11-26 47 0.414 0.277 0.828 0.0735 0.0564 6.06 10.40 Celje 10.34
```

```
## 301 2020-11-27 54 0.546 0.289 0.830 0.1060 0.0505 6.62 13.10 Celje 11.88
## 302 2020-11-28 61 0.244 0.648 1.150 0.0480 0.0909 5.98 9.10 Celje 13.42
## 303 2020-11-29 46 0.154 0.442 0.875 0.0545 0.0927 3.72 6.39 Celje 10.12
##      PM10corr  NaCl PM10total
## 289      53.68 0.3152 120.0364
## 290      48.80 0.2799 103.0429
## 291      59.78 0.4821 129.6562
## 296      54.90 0.5135 121.1160
## 299      42.70 0.2867 98.3787
## 300      57.34 0.3334 133.1223
## 301      65.88 0.3395 153.6410
## 302      74.42 0.7389 166.8398
## 303      56.12 0.5347 124.5029
```

Primer izbire podatkov od septembra do začetka decembra:

```
datPM10[month(datPM10$Datum) >= 9 & month(datPM10$Datum)<= 11, ]
```

```
##      Datum PM10    Ca    Cl    K    Mg    Na NH4    NO3 kraj PM10err
## 235 2020-09-08 31 1.260 0.0114 0.326 0.2250 0.0750 1.02 0.575 Celje 6.82
## 245 2020-09-18 33 1.170 0.1150 0.298 0.1750 0.0508 1.84 0.652 Celje 7.26
## 263 2020-10-06 37 0.857 0.0621 0.381 0.1880 0.0906 3.89 0.809 Celje 8.14
## 264 2020-10-07 41 1.190 0.0739 0.357 0.2560 0.0756 3.00 1.120 Celje 9.02
## 285 2020-10-28 36 0.895 0.1710 0.435 0.1550 0.0864 1.67 3.510 Celje 7.92
## 286 2020-10-29 42 0.977 0.2190 0.548 0.1730 0.0521 1.97 4.660 Celje 9.24
## 287 2020-10-30 51 1.320 0.2190 0.731 0.1980 0.0693 1.84 6.000 Celje 11.22
## 288 2020-10-31 49 0.846 0.2690 0.671 0.1790 0.0894 2.43 7.330 Celje 10.78
## 289 2020-11-01 44 0.418 0.2530 0.517 0.0910 0.0622 2.89 8.130 Celje 9.68
## 290 2020-11-02 40 0.437 0.2430 0.679 0.0971 0.0369 1.46 2.210 Celje 8.80
## 291 2020-11-03 49 0.693 0.4290 0.789 0.1400 0.0531 1.67 5.840 Celje 10.78
## 296 2020-11-22 45 0.494 0.4550 0.791 0.1140 0.0585 2.49 6.400 Celje 9.90
## 299 2020-11-25 35 0.157 0.2210 0.639 0.0293 0.0657 4.75 6.830 Celje 7.70
## 300 2020-11-26 47 0.414 0.2770 0.828 0.0735 0.0564 6.06 10.400 Celje 10.34
## 301 2020-11-27 54 0.546 0.2890 0.830 0.1060 0.0505 6.62 13.100 Celje 11.88
## 302 2020-11-28 61 0.244 0.6480 1.150 0.0480 0.0909 5.98 9.100 Celje 13.42
## 303 2020-11-29 46 0.154 0.4420 0.875 0.0545 0.0927 3.72 6.390 Celje 10.12
##      PM10corr  NaCl PM10total
## 235      37.82 0.0864 79.2188
## 245      40.26 0.1658 84.9866
## 263      45.14 0.1527 96.7104
## 264      50.02 0.1495 106.2620
## 285      43.92 0.2574 95.0198
## 286      51.24 0.2711 111.3502
## 287      62.22 0.2883 135.1056
## 288      59.78 0.3584 131.7328
## 289      53.68 0.3152 120.0364
## 290      48.80 0.2799 103.0429
## 291      59.78 0.4821 129.6562
## 296      54.90 0.5135 121.1160
## 299      42.70 0.2867 98.3787
## 300      57.34 0.3334 133.1223
## 301      65.88 0.3395 153.6410
## 302      74.42 0.7389 166.8398
## 303      56.12 0.5347 124.5029
```

Operacije nad množicami

Poleg funkcije `setdiff()` poznamo še nekaj funkcij za delo z množicami kot so `union()` (unija) in `intersect()` (preseki).

Vzamemo množico **mn1**, ki vsebuje vsa imena stolpcev data.frame-a **df**. Naredimo še množico **mn3**, ki je enaka:

```
mn3 <- c('spol', 'imena', 'starost', 'st_noge')
```

Če naredimo unijo množic **mn1** in **mn3**, dobimo:

```
print(union(mn1, mn3))
```

```
## [1] "spol"      "visina"    "teza"      "imena"     "starost"   "st_noge"
```

Kaj pa če vrstni red obrnemo?

```
print(union(mn3, mn1))
```

```
## [1] "spol"      "imena"     "starost"   "st_noge"   "visina"    "teza"
```

Dobimo enako.

Če naredimo presek, dobimo:

```
print(intersect(mn1, mn3))
```

```
## [1] "spol"      "imena"
```

Kaj pa če vrstni red obrnemo?

```
print(intersect(mn3, mn1))
```

```
## [1] "spol"      "imena"
```

Dobimo enako.

Če pogledamo razliko, dobimo:

```
print(setdiff(mn1, mn3))
```

```
## [1] "visina"    "teza"
```

Kaj pa če tukaj zamenjamo vrstni red?

```
print(setdiff(mn3, mn1))
```

```
## [1] "starost"   "st_noge"
```

V tem primeru pa ne dobimo enako. Funkcijo `setdiff(mn1, mn3)` si lahko predstavljamo, da vrne elemente, ki so v **mn1** in jih ni v **mn3**. To pa ni enako elementom, ki so v **mn3** in jih ni v **mn1**.

Delo s faktorji

Kategorične spremenljivke so spremenljivke, ki lahko zavzamejo samo končno mnogo vnaprej določenih vrednosti. Delimo jih na:

- **Nominalne spremenljivke.** To so spremenljivke brez ureditve: tip izdelka ali znamka avtomobila.
- **Ordinalne spremenljivke.** To so spremenljivke, ki imajo smiselno ureditev: stopnja izobrazbe ali šolski uspeh.

V R uporabljamo za delo s kategoričnimi spremenljivkami t. i. **faktorje** (ang. **factor**). Ti se od spremenljivk tipa niz razlikujejo v tem, da se spremenljivka hrani tudi informacijo o vseh možnih vrednostih. Spremenljivki tipa faktor ni mogoče dodati vrednosti, ki je ni v množici možnih vrednosti, kar služi kot varovalka pred napakami pri vnosu podatkov.

Poglejmo si uporabo faktorja na dveh preprostih primerih, kjer bomo sami ustvarili spremenljivki. Kasneje si bomo pogledali še delo s faktorji na primeru realnih podatkov, kjer bomo ponovno uporabili podatke o zaposlitvah na področju podatkovnih ved.

Kot primer nominalne spremenljivke si oglejmo vreme. Obstajajo 4 možne vrednosti za oceno oblačnosti (oblačno, pretežno_oblačno, zmerno_oblačno, jasno). Ustvarimo vektor oblačnosti za pet dni:

```
vreme <- c("oblačno", "pretežno_oblačno", "jasno",  
           "pretežno_oblačno", "pretežno_oblačno")
```

Sedaj je vektor `vreme` shranjen kot niz. Kaj so slabosti takšnega shranjevanja kategoričnih podatkov? Prvič, nimamo nobenega varovala pred tipkarskimi napakami – R je zadnji vnos prebral kot *pretečno_oblačno* in ga tako tudi shranil, čeprav ta beseda ne obstaja:

```
vreme
```

```
## [1] "oblacno"          "pretežno_oblacno" "jasno"            "pretežno_oblacno"  
## [5] "pretecno_oblacno"
```

Če imamo spremenljivko, za katero vemo, da bo zasedla eno od vnaprej določenih vrednosti, je bolje, da jo shranimo kot faktor. V R za to uporabimo funkcijo `factor()`. Naredimo faktor iz spremenljivke `vreme`:

```
vreme_fac <- factor(vreme)  
vreme_fac
```

```
## [1] oblacno          pretežno_oblacno jasno            pretežno_oblacno  
## [5] pretecno_oblacno  
## Levels: jasno oblacno pretecno_oblacno pretežno_oblacno
```

Opazimo, da je sedaj spremenljivka drugačnega tipa, saj hrani tudi informacijo o možnih vrednostih oziroma **ravnih** (ang. **levels**). Ampak v tem primeru so te ravni napačne (ne zajame vseh 4 možnih vrednosti, poleg tega pa vsebuje tudi eno napačno vrednost). Funkcija `factor()` privzeto kot ravni nastavi vse vrednosti v podani spremenljivki. Če želimo, ji lahko podamo dodaten argument `levels`, kateri ročno določimo ravni spremenljivke. V kolikor poznamo ravni vnaprej, je dobra praksa, da podamo tudi ta argument:

```
vreme_fac <- factor(vreme, levels = c("oblačno", "pretežno_oblačno", "zmerno_oblačno", "jasno"))  
vreme_fac
```

```
## [1] oblacno      pretežno_oblacno jasno      pretežno_oblacno
## [5] <NA>
## Levels: oblacno pretežno_oblacno zmerno_oblacno jasno
```

Ker element seznama *pretežno_oblačno* ni na seznamu ravni spremenljivke, se ta samodejno spremeni v manjkajočo vrednost (). V kolikor želimo, da faktor hrani tudi informacijo o smiselni razvrstitvi po velikosti, dodamo argument `ordered = TRUE`.

```
vreme_fac2 <- factor(vreme, levels = c("jasno", "zmerno_oblačno", "pretežno_oblačno", "oblačno"),
                     ordered = TRUE)
vreme_fac2
```

```
## [1] oblacno      pretežno_oblacno jasno      pretežno_oblacno
## [5] <NA>
## Levels: jasno < zmerno_oblacno < pretežno_oblacno < oblacno
```

V zgornjem primeru (podatki o PM10) bi bilo pametno shraniti kraj kot **faktor**.

```
dat$kraj <- factor(dat$kraj)
summary(dat)
```

```
##      Datum      PM10      Ca      Cl
## Length:336    Min.   : 2.80    Min.   :0.0239    Min.   :0.01140
## Class :character 1st Qu.: 12.00    1st Qu.:0.1495    1st Qu.:0.04085
## Mode  :character Median : 18.00    Median :0.2500    Median :0.06245
##                      Mean   : 22.44    Mean   :0.3506    Mean   :0.15003
##                      3rd Qu.: 28.25    3rd Qu.:0.4420    3rd Qu.:0.19750
##                      Max.    :100.00    Max.    :1.6200    Max.    :1.47000
##      K      Mg      Na      NH4
## Min.   :0.0161    Min.   :0.00139    Min.   :0.00396    Min.   :0.0250
## 1st Qu.:0.0882    1st Qu.:0.03485    1st Qu.:0.03578    1st Qu.:0.4323
## Median :0.1770    Median :0.05635    Median :0.05840    Median :0.7610
## Mean   :0.3257    Mean   :0.06930    Mean   :0.08895    Mean   :1.2008
## 3rd Qu.:0.4828    3rd Qu.:0.09613    3rd Qu.:0.10450    3rd Qu.:1.4600
## Max.   :4.1300    Max.   :0.30400    Max.   :0.77600    Max.   :6.6500
##      NO3      kraj
## Min.   : 0.0487    Celje      :120
## 1st Qu.: 0.3347    Kranj      :120
## Median : 0.8095    Ljubljana: 96
## Mean   : 2.0964
## 3rd Qu.: 2.7250
## Max.   :19.6000
```

Domača naloga

1. Med podatki o delcih PM10 izberite podatke za Ljubljano.

- Izberite samo tiste, ki imajo vrednost kalcija (Ca) večje od 0.3 in vrednosti natrija (Na) manjše od 0.05.

##	Datum	PM10	Ca	Cl	K	Mg	Na	NH4	NO3	kraj
## 144	6/9/2014	18	0.391	0.0388	0.0846	0.0931	0.0270	0.421	0.202	Ljubljana
## 145	6/10/2014	20	0.467	0.0383	0.0980	0.1030	0.0327	0.539	0.235	Ljubljana
## 146	6/11/2014	20	0.581	0.0392	0.0760	0.1030	0.0437	0.415	0.294	Ljubljana
## 147	6/12/2014	18	0.596	0.0415	0.0699	0.0794	0.0369	0.135	0.352	Ljubljana
## 152	6/17/2014	16	0.332	0.0376	0.0939	0.0701	0.0496	0.637	0.342	Ljubljana
## 155	6/20/2014	15	0.446	0.0388	0.0573	0.0807	0.0396	0.468	0.374	Ljubljana

- V podatkih izračunajte povprečne vrednosti za vse snovi.

##	PM10	Ca	Cl	K	Mg	Na
## 12.58020833	0.27463646	0.03781562	0.08191875	0.05635625	0.05557250	
##	NH4	NO3				
## 0.50023958	0.37222500					

- Dodajte stolpec, ki ponazarja seštevke Ca in Cl.

##	Datum	PM10	Ca	Cl	K	Mg	Na	NH4	NO3	kraj
## 1	1/17/2014	22	0.186	0.297	0.577	0.0374	0.1450	0.639	1.98	Kranj
## 2	1/18/2014	32	0.132	0.528	0.735	0.0235	0.1090	0.877	2.71	Kranj
## 3	1/19/2014	30	0.145	0.381	0.577	0.0363	0.1590	1.080	2.72	Kranj
## 4	1/20/2014	16	0.127	0.170	0.383	0.0428	0.0608	0.628	2.01	Kranj
## 5	1/21/2014	24	0.202	0.160	0.418	0.0365	0.0346	1.220	3.62	Kranj
## 6	1/22/2014	32	0.610	0.231	0.615	0.0734	0.0468	1.140	3.83	Kranj

- Za vsako vrstico izračunajte seštevke vseh meritev.

##	121	122	123	124	125	126	127	128
## 8.18270	13.63050	15.03430	18.41950	21.29210	24.51910	35.53220	30.26150	
##	129	130	131	132	133	134	135	136
## 21.02130	15.15980	13.32240	8.20260	9.81020	12.80870	10.32030	12.20590	
##	137	138	139	140	141	142	143	144
## 15.47740	12.23080	12.28930	12.42570	14.48300	15.32360	16.12080	19.25750	
##	145	146	147	148	149	150	151	152
## 21.51300	21.55190	19.31070	13.98760	14.53290	12.37660	14.15850	17.56220	
##	153	154	155	156	157	158	159	160
## 22.80830	20.38800	16.50440	10.36130	13.49470	18.28330	10.09540	7.38290	
##	161	162	163	164	165	166	167	168
## 8.35730	13.09770	13.43900	13.55840	5.99980	7.71190	8.69760	10.87550	
##	169	170	171	172	173	174	175	176
## 16.31170	18.64360	14.09930	20.23940	12.30540	3.78600	6.25666	8.75420	
##	177	178	179	180	181	182	183	184
## 9.37290	7.47490	9.69390	12.75240	12.08170	16.31270	18.38840	25.08440	
##	185	186	187	188	189	190	191	192
## 23.12350	18.22120	15.14060	11.87640	15.20440	16.28600	13.10760	13.06550	
##	193	194	195	196	197	198	199	200
## 16.55430	14.20570	8.75260	3.59740	9.46380	13.35070	13.60550	13.36580	
##	201	202	203	204	205	206	207	208
## 5.78400	10.80320	12.28140	15.06980	14.17690	13.26230	21.22040	9.98490	
##	209	210	211	212	213	214	215	216
## 15.90320	7.24960	8.02940	8.49670	7.64320	10.50010	12.20360	13.59400	

2. Naložite tabelo nakupov v trgovini, ki smo jo pridobili iz spletne strani <https://www.kaggle.com/aungpyaeap/supermarket-sales>.

- Tabeli odstranite stolpce `branch`, `invoice_ID`, `cogs`, `gross_margin_percentage` in `gross_income`.
- Z uporabo funkcije `apply` izračunajte povprečja za stolpce `unit_price`, `quantity`, `total` in `rating`.

```
## unit_price    quantity      total      rating
##    55.67213     5.51000   322.96675    6.97270
```

- Iz datumov nakupov izvečite dan nakupa, nato pogledajte, kako se spreminja število nakupov glede na dan v tednu (*Namig: pogledajte funkcijo "table"*). Spomnite se, da morate najprej R-ju povedati, da je "date" stolpec datumov.

```
##
##    1    2    3    4    5    6    7
## 133 125 158 143 138 139 164
```

- (Težje) Vsem včlanjenim strankam dodajte 7.5% popust na končni ceni in nove vrednosti shranite v stolpec "price_member_disc".

```
##      city customer_type gender      product_line unit_price quantity
## 1   Yangon      Member Female  Health and beauty      74.69         7
## 2 Naypyitaw    Normal Female Electronic accessories      15.28         5
## 3   Yangon    Normal   Male   Home and lifestyle      46.33         7
## 4   Yangon    Member   Male   Health and beauty      58.22         8
## 5   Yangon    Normal   Male   Sports and travel      86.31         7
## 6 Naypyitaw    Normal   Male Electronic accessories      85.39         7
##  tax_5_percent    total      date  time      payment rating price_member_disc
## 1         26.1415  548.9715 2019-01-05 13:08      Ewallet     9.1         507.7986
## 2          3.8200   80.2200 2019-03-08 10:29        Cash     9.6          80.2200
## 3         16.2155  340.5255 2019-03-03 13:23 Credit card     7.4         340.5255
## 4         23.2880  489.0480 2019-01-27 20:33      Ewallet     8.4         452.3694
## 5         30.2085  634.3785 2019-02-08 10:37      Ewallet     5.3         634.3785
## 6         29.8865  627.6165 2019-03-25 18:30      Ewallet     4.1         627.6165
```