

Predavanja 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$steza / df$visina^2
```

```
## [1] 0.002340751 0.002600438 0.002090239 0.002704164 0.001915709 0.002512783
## [7] 0.001959784 0.002205805 0.002405693 0.002480159
```

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$steza)
```

```
## [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.csv('./data_raw/delci2.csv')
```

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')
```

```
## Warning in file(file, ifelse(append, "a", "w")): cannot open file './data_clean/
## delci_popravljeni.csv': No such file or directory
```

```
## Error in file(file, ifelse(append, "a", "w")): cannot open the connection
```

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
## 47.388889  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      N03
## 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      N03 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
```

Operacije nad množicami

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

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.

Ča 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 **nm1** in jih ni v **nm3**. To pa ni enako elementom, ki so v **nm3** in jih ni v **nm1**.

Domača naloga

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.
- V izbranih podatkih izračunajte povprečne vrednosti za vse snovi.
- Dodajte stolpec, ki ponazarja seštevek Na in Cl.
- Za vsako vrstico izračunajte seštevek vseh meritev.