Schematy losowania

Maciej Beręsewicz

Teoria

Poniższy dokument zawiera informację o tym jak losować jednostki według schematów nieprostych z pakietem *sampling*. Struktura dokumentu jest następująca. W pierwszej części poznamy schematy losowania jednostek w przypadku gdy prawdopodobieństwa wylosowania poszczególnych jednostek są takie same. W drugiej części poznamy schematy losowania jednostek gdy prawdopodbieństwo inkluzji jednostek jest różne. Ostatnia część poświęcona będzie pakietowi *survey* oraz temu w jaki sposób deklarujemy poszczególne schematy losowania, które poznamy we wcześniejszych podpunktach.

Jak przeprowadzić losowanie?

Pakiet `sampling` zawiera następujące funkcje:

- `srswor` – losowanie proste bez zwracania
- `srswr` – losowanie proste ze zwracaniem
- `poisson` – losowanie poissona
- `systematic` losowanie systematyczne

Losowania bardziej skomplikowane * `strata` – losowanie warstwowe * `cluster` – losowanie zespołowe * `mstage` – losowanie wielostopniowe

Przed poznaniem funkcji musimy zainstalować pakiety oraz wczytać dane. Poniższy kod umożliwia instalowanie oraz wczytywanie pakietów.

```
#### instalowanie pakietów ----------------------------------
install.packages(c('survey','sampling','ggplot2','dplyr','laeken'),
                 dependencies=T)
```

```
library(survey)
library(sampling)
library(ggplot2)
library(dplyr)
library(laeken) ## zbiór danych eusilc
```

Na zajęciach wykorzystamy syntetyczny zbiór utworzony na podstawie badania EU-SILC. Zbiór ten znajduje się w pakiecie `laeken` i uruchamiamy go w następujący sposób.

```
data(eusilc)
```

Spójrzmy na podsumowanie zbioru

```
str(eusilc)
```

```
## 'data.frame':    14827 obs. of  28 variables:
##  $ db030   : int  1 1 1 2 2 2 2 3 4 4 ...
##  $ hsize   : int  3 3 3 4 4 4 4 1 5 5 ...
##  $ db040   : Factor w/ 9 levels "Burgenland","Carinthia",..: 6 6 6 6 6 6 6 8 8 8 ...
##  $ rb030   : int  101 102 103 201 202 203 204 301 401 402 ...
##  $ age     : int  34 39 2 38 43 11 9 26 47 28 ...
##  $ rb090   : Factor w/ 2 levels "male","female": 2 1 1 2 1 1 1 2 1 1 ...
##  $ pl030   : Factor w/ 7 levels "1","2","3","4",..: 2 1 NA 7 1 NA NA 7 3 1 ...
##  $ pb220a  : Factor w/ 3 levels "AT","EU","Other": 1 3 NA 1 1 NA NA 1 3 1 ...
```

```
##  $ py010n  : num  9756 12472 NA 12487 42821 ...
##  $ py050n  : num  0 0 NA 0 0 ...
##  $ py090n  : num  0 0 NA 0 0 ...
##  $ py100n  : num  0 0 NA 0 0 NA NA 0 0 0 ...
##  $ py110n  : num  0 0 NA 0 0 NA NA 0 0 0 ...
##  $ py120n  : num  0 0 NA 0 0 NA NA 0 0 0 ...
##  $ py130n  : num  0 0 NA 0 0 NA NA 0 0 0 ...
##  $ py140n  : num  0 0 NA 0 0 NA NA 0 0 0 ...
##  $ hy040n  : num  4274 4274 4274 0 0 ...
##  $ hy050n  : num  2428 2428 2428 1550 1550 ...
##  $ hy070n  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hy080n  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hy090n  : num  33.39 33.39 33.39 2.13 2.13 ...
##  $ hy110n  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hy130n  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ hy145n  : num  0 0 0 0 0 ...
##  $ eqSS    : num  1.8 1.8 1.8 2.1 2.1 2.1 2.1 1 2.8 2.8 ...
##  $ eqIncome: num  16091 16091 16091 27076 27076 ...
##  $ db090   : num  505 505 505 493 493 ...
##  $ rb050   : num  505 505 505 493 493 ...
```

Zbiór zawiera następujące zmienne:

- db030 - integer; the household ID.
- hsize - integer; the number of persons in the household.
- db040 - factor; the federal state in which the household is located (levels Burgenland, Carinthia, Lower Austria, Salzburg, Styria, Tyrol, Upper Austria, Vienna and Vorarlberg).
- rb030 - integer; the personal ID.
- age - integer; the person's age.
- rb090 - factor; the person's gender (levels male and female).
- pl030 - factor; the person's economic status (levels 1 = working full time, 2 = working part time, 3 = unemployed, 4 = pupil, student, further training or unpaid work experience or in compulsory military or community service, 5 = in retirement or early retirement or has given up business, 6 = permanently disabled or/and unfit to work or other inactive person, 7 = fulfilling domestic tasks and care responsibilities).
- pb220a - factor; the person's citizenship (levels AT, EU and Other).
- py010n - numeric; employee cash or near cash income (net).
- py050n - numeric; cash benefits or losses from self-employment (net).
- py090n - numeric; unemployment benefits (net).
- py100n - numeric; old-age benefits (net).
- py110n - numeric; survivor's benefits (net).
- py120n - numeric; sickness benefits (net).
- py130n - numeric; disability benefits (net).
- py140n - numeric; education-related allowances (net).
- hy040n - numeric; income from rental of a property or land (net).
- hy050n - numeric; family/children related allowances (net).
- hy070n - numeric; housing allowances (net).
- hy080n - numeric; regular inter-household cash transfer received (net).
- hy090n - numeric; interest, dividends, profit from capital investments in unincorporated business (net).
- hy110n - numeric; income received by people aged under 16 (net).
- hy130n - numeric; regular inter-household cash transfer paid (net).
- hy145n - numeric; repayments/receipts for tax adjustment (net).
- eqSS - numeric; the equivalized household size according to the modified OECD scale.
- eqIncome - numeric; a slightly simplified version of the equivalized household income.
- db090 - numeric; the household sample weights.

- rb050 - numeric; the personal sample weights.

Losowanie z wykorzystaniem funkcji sample oraz pakietu sampling

W tej części zajmiemy się różnymi schematami losowania, które są dostępne w pakiecie sampling, jak również z domyślną funkcją *sample*.

Losowanie proste bez zwracania

Pierwszym i najprostrzym schematem losowania jest losowanie proste bez zwracania. W takim przypadku losujemy określony podzbiór jednostek bez uwzględniania zmiennych pomocniczych (np. płci, wieku). W `R` możemy wykorzystać w tym celu funkcję `sample` lub `srswor`.

Poniżej przykład wykorzystania funkcji dla zbioru studenci. Załóżmy, że naszym celem jest oszacowanie średniej wielkości gospodarstwa domowego.

```
### ID osób w gospodarstwach
id_hh <- eusilc$rb030

### Frakcja, którą losujemy
prop_wylos <- 0.1

### Wielkość próby
n_pop <- length(id_hh) ## wielkość populacji
n_sample <- round(n_pop*prop_wylos) ## wielkość próby

### identyfikatory wylosowanych studetów
set.seed(123) ## ustawienie ziarna losowania tak aby każda osoba dostała taki sam wynik
id_hh_wylos <- sample(x = id_hh,
                      size = n_sample)

### wylosowani studenci
eusilc_samp <- subset(eusilc, rb030 %in% id_hh_wylos)

dim(eusilc_samp)
```

```
## [1] 1483   28
```

Sprawdzmy teraz jakie wyniki otrzymujemy w przypadku losowania prostego i porównamy do populacji.

```
summary(eusilc_samp$hsize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.207   4.000   9.000
```

```
summary(eusilc$hsize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.234   4.000   9.000
```

Teraz wykorzystamy funkcję sampling::srswor

```
## ID osób w gospodarstwach
id_hh <- eusilc$rb030

### Frakcja, którą losujemy
```

```
prop_wylos <- 0.1

### Wielkość próby
n_pop <- length(id_hh) ## wielkość populacji
n_sample <- round(n_pop*prop_wylos) ## wielkość próby

### losujemy
set.seed(123)
row_wylos <- srswor(n_sample,n_pop)

### sprawdzmy jak wygląda wynik
head(row_wylos)
```

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

```
### wybieranie jednostek
stu_samp <- eusilc[row_wylos, ]
```

Losowanie proste ze zwracaniem

Losowanie systematyczne

Losowanie systematyczne polega na wylosowaniu określonej liczby jednostek wykorzystując stały interwał między wybieranymi jednostkami (np. co 5). Interwał określany jest przez stosunek wielkości populacji do wielkości próby. Losowanie systematyczne może również uwzględniać fakt, że jednostki maja nierówne prawdopodobieństwa dostania się do próby. W tym przykładzie zastosujemy dwa podejścia - losowanie z równymi prawdopodobieństwami oraz losowanie z nierównymi prawdopodobieństwami.

W pierwszym przypadku wykorzystamy funkcję *seq*, która umożliwia tworzenie wektorów z pewnym krokiem. Istotny jest również element określający start losowania systematycznego, co wpływa na dobór jednostek.

```
## krok
k <- n_pop/n_sample

## losowanie początku
set.seed(123)
start <- sample(1:k,1)

## losowanie systematyczne
id_wylos <- seq(from = start,
                to = n_pop,
                by = round(k))

### wylosowanie jednostki
systematyczne <- eusilc[id_wylos,]
```

Poniżej przedstawiamy porównanie w przypadku losowania systematycznego

```
### porównanie rozkładu
summary(systematyczne$hsize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.211   4.000   9.000
```

```
summary(eusilc$hsize)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.234   4.000   9.000
```

Losowanie proporcjonalne do zmiennej pomocniczej

W tym celu wykorzystamy funkcję `sampling::UPpoisson` - jeden ze schematów losowania wykorzystywany w przypadku przyjęcia za zmienną pomocniczą zmienną ciągłą. Najczęściej takie losowanie wykorzysujemy gdy chcemy losować jednostki proporcjonalnie do ich wielkości (np. liczby zatrudnionych, przychodów).

Przyjrzyjmy się zmiennej `eqIncome` określającej dochód ekwiwalentny.

```
summary(eusilc$eqIncome)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0   13460   18080   19910   24190  152200
```

Wykorzystamy teraz funkcję `sampling::UPpoisson` oraz `sampling::inclusionprobabilities`, która służy do określenia prawdopodobieństw inkluzji poszczególnych jednostek.

```
### wielkość próby
n <- 90

### obliczamy prawdopodobieństwa
pik <- inclusionprobabilities(eusilc$eqIncome,n)
summary(pik)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000000 0.004105 0.005513 0.006070 0.007377 0.046410
```

```
### wykres jak zmienia jak wygląda prawdopodobieństwo
n_sch <- nrow(eusilc)
```

Po ustaleniu prawdopodobieństw wylosowania poszczególnych jednostek stosujemy funkcję `sampling::UPpoisson`. W wyniku otrzymujemy wektor składający się z 0-1, który określa czy dana jednostka jest wylosowana (1) lub niejest wylosowania (0).

```
### losowanie z wykorzystaniem funkcji UPpoisson (losowa wielkość jednak zbliżona do 38)
set.seed(123)
pp_wylos <- UPpoisson(pik)
sum(pp_wylos)
```

```
## [1] 87
```

```
### wybranie jednostek
stu_pp <- eusilc[pp_wylos, ]
stu_pp$pik <- pik[pp_wylos]
stu_pp$fpc <- nrow(stu_pp)
```

Wartość globalną możemy oszacować jako suma iloczynów kolumny *eqIncome* oraz odwrotności *pik*, która będzie określać wagi dla poszczególnych szkół.

```
sum(stu_pp$eqIncome * 1/stu_pp$pik)
```

```
## [1] 285320473
```

Porównamy teraz szacunki z wykorzystaniem funkcji survey::svydesign

```
### deklaracja schematu losowania
des_up <- svydesign(ids =  ~ 1,
                    probs = ~ pik,
```

```
                    fpc = ~ fpc,
                    data = stu_pp)

### szacowanie globalnego dochodu
svytotal(~eqIncome,des_up)
```

```
##              total SE
## eqIncome 285320473  0
```