Zadania z planowania eksperymentu

biblioteka: survey

Agnieszka Wrzos

25-03-2020

Zadanie 1

Wczytaj bibliotekę `survey`, następnie komendą `data(api)` wywołaj listę zbiorów dotyczących szkół w Kaliforni. Będzie to kilka zbiorów, a Ciebie będą interesować głównie dwa: `apisrs` oraz `apipop`. Pierwszy jest próbką 200 elementową populacji `apipop`. Na podsatwie próby oszacuj metodą przedziałową nieznany parametr wartości oczekiwanej zmiennej `enroll` na poziomie ufności 0.99. Sprawdź, czy nieznany parametr (dla Ciebie znany, bo może być wyznaczony na podstawie populacji) należy do przedziału ufności. Narysuj wykres rozkładu zmiennej `enroll` w próbie oraz oceń podstawowe statystki opisowe.

Zadanie 2

Z wczytanego wcześniej zbioru `apipop` wylosuj 100 próbek o liczebności 36 elementów każda. Następnie napisz funkcję, która będzie liczyła przedział ufności dla frakcji. Zastosuj tą funkcję do zmiennej `sch.wide` mówiącej o tym czy szkoła wypełniła założone przez państwo wymagania dotyczące rozwoju uczniów (kod "Yes"). Sprawdź ile spośród wyznaczonych w ten sposób przedziałów ufności zwierało prawdziwą proporcję szkół spełniających wymagania w populacji.

Rozwiązanie

Zadanie 1

```
library(tidyverse)
library(mosaic)
library(survey)
data(api)
```

Przedziały ufności

```
n <- nrow(apisrs)
np <- nrow(apipop)
srs_d <- svydesign(ids = ~1,fpc = ~fpc,data = apisrs )
```

Zmienna fpc mówi, że mamy 6194 szkół w Kaliforni. Zmienna `srs_d` zawiera podstawowe informacje o ankiecie.

```
summary(srs_d)
```

```
## Independent Sampling design
## svydesign(ids = ~1, fpc = ~fpc, data = apisrs)
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.03229 0.03229 0.03229 0.03229 0.03229 0.03229
## Population size (PSUs): 6194
## Data variables:
##  [1] "cds"      "stype"    "name"     "sname"    "snum"     "dname"
##  [7] "dnum"     "cname"    "cnum"     "flag"     "pcttest"  "api00"
## [13] "api99"    "target"   "growth"   "sch.wide" "comp.imp" "both"
## [19] "awards"   "meals"    "ell"      "yr.rnd"   "mobility" "acs.k3"
## [25] "acs.46"   "acs.core" "pct.resp" "not.hsg"  "hsg"      "some.col"
## [31] "col.grad" "grad.sch" "avg.ed"   "full"     "emer"     "enroll"
## [37] "api.stu"  "pw"       "fpc"
```

```
srednia <- svymean(~enroll, srs_d)
liczebnosc <- svytotal(~enroll, srs_d)
srednia
```

```
##          mean     SE
## enroll 584.61 27.368
```

```
liczebnosc
```

```
##          total     SE
## enroll 3621074 169520
```

`srednia` i `liczebnosc` są to wyestymowane średnia i liczebność całej populacji. SE jest błędem standardowym.

```
any(is.na(apipop))
```

```
## [1] TRUE
```

```
sredniapop <- mean(apipop$enroll, na.rm=T)
sredniapop
```

```
## [1] 619.0469
```

```
q <- qnorm(0.99)
sd <- sd(apisrs$enroll, na.rm = T)
sr <- as.vector(srednia)
lp <- sr-q*sqrt(1-(n/np))*(sd/sqrt(n))
pp <- sr+q*sqrt(1-(n/np))*(sd/sqrt(n))
lp
```

```
## [1] 520.9417
```

```
pp
```

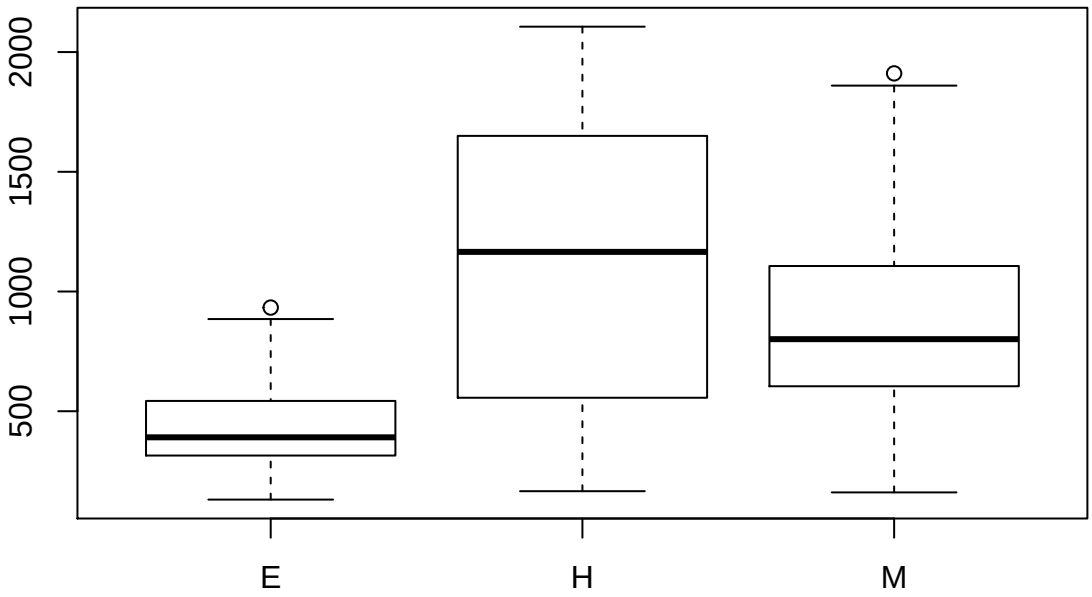
```
## [1] 648.2783
```

Przedział ufności: (521, 648).

Średnia mieści się w przedziale ufności.

boxplot rozkładu enroll ze względu na typ szkoły

```
svyboxplot(enroll~stype,srs_d)
```



Średnie uczniów w każdym typie szkoły się różnią.

Statystyki opisowe dla `enroll` w próbie

```
min <- min(apisrs$enroll, na.rm=T)
max <- max(apisrs$enroll, na.rm=T)
```

```
srednia
```

```
##           mean     SE
## enroll 584.61 27.368
```

```
sd
```

```
## [1] 393.4514
```

```
min
```

```
## [1] 131
```

```
max
```

```
## [1] 2106
```

```
n
```

```
## [1] 200
```

Zadanie 2

100 próbek o liczebności 36 elementów każda

```
apipop$sch.wide <- as.numeric(apipop$sch.wide)-1

set.seed(2020)
probki <- 1:100 %>%
  map(~sample_n(apipop, size=36))
```

funkcja licząca przedział ufności dla frakcji

```
przedzialyufnosci <- function(x, alpha=0.05) {
q <- qnorm(1-alpha)
sd1 <- sd(x, na.rm=T)
sred <- mean(x, na.rm=T)
l <- sred-q*sqrt(1-36/6157)*(sd1/sqrt(36))
p <- sred+q*sqrt(1-36/6157)*(sd1/sqrt(36))
return( c(l, p, sred))
}
```

czy szkoła wypełniła założone przez państwo wymagania dotyczące rozwoju uczniów

```
wynik <- probki %>%
  map(~przedzialyufnosci(.x$sch.wide, alpha=0.01))
wynik
```

```
## [[1]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[2]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[3]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[4]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[5]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[6]]
## [1] 0.5138389 0.8750500 0.6944444
##
## [[7]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[8]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[9]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[10]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[11]]
## [1] 0.5466113 0.8978331 0.7222222
##
## [[12]]
## [1] 0.7656721 1.0121056 0.8888889
```

```
##
## [[13]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[14]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[15]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[16]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[17]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[18]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[19]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[20]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[21]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[22]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[23]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[24]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[25]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[26]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[27]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[28]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[29]]
## [1] 0.9077906 1.0366538 0.9722222
##
## [[30]]
## [1] 0.4818415 0.8514918 0.6666667
```

```
##
## [[31]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[32]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[33]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[34]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[35]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[36]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[37]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[38]]
## [1] 0.5138389 0.8750500 0.6944444
##
## [[39]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[40]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[41]]
## [1] 0.9077906 1.0366538 0.9722222
##
## [[42]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[43]]
## [1] 0.4505672 0.8272106 0.6388889
##
## [[44]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[45]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[46]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[47]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[48]]
## [1] 0.5802273 0.9197727 0.7500000
```

```
##
## [[49]]
## [1] 0.5138389 0.8750500 0.6944444
##
## [[50]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[51]]
## [1] 0.5466113 0.8978331 0.7222222
##
## [[52]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[53]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[54]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[55]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[56]]
## [1] 0.5466113 0.8978331 0.7222222
##
## [[57]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[58]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[59]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[60]]
## [1] 0.4818415 0.8514918 0.6666667
##
## [[61]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[62]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[63]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[64]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[65]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[66]]
## [1] 0.6503838 0.9607273 0.8055556
```

```
##
## [[67]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[68]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[69]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[70]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[71]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[72]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[73]]
## [1] 0.5138389 0.8750500 0.6944444
##
## [[74]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[75]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[76]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[77]]
## [1] 0.7255202 0.9967020 0.8611111
##
## [[78]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[79]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[80]]
## [1] 0.5138389 0.8750500 0.6944444
##
## [[81]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[82]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[83]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[84]]
## [1] 0.6147773 0.9407782 0.7777778
```

```
##
## [[85]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[86]]
## [1] 0.8546356 1.0342533 0.9444444
##
## [[87]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[88]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[89]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[90]]
## [1] 0.6503838 0.9607273 0.8055556
##
## [[91]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[92]]
## [1] 0.9077906 1.0366538 0.9722222
##
## [[93]]
## [1] 0.8546356 1.0342533 0.9444444
##
## [[94]]
## [1] 0.6872162 0.9794504 0.8333333
##
## [[95]]
## [1] 0.5802273 0.9197727 0.7500000
##
## [[96]]
## [1] 0.6147773 0.9407782 0.7777778
##
## [[97]]
## [1] 0.7656721 1.0121056 0.8888889
##
## [[98]]
## [1] 0.4505672 0.8272106 0.6388889
##
## [[99]]
## [1] 0.8083033 1.0250300 0.9166667
##
## [[100]]
## [1] 0.6503838 0.9607273 0.8055556
```

```
sredniapop_sw <- mean(apipop$sch.wide)
test <- wynik %>%
  map_dbl(~{ifelse(.x[1]<sredniapop_sw & .x[2]>sredniapop_sw,1,0)}) %>%
  sum()
test
```

```
## [1] 95
```

W 95 przypakach przedział ufności zawiera średnią dla populacji.