Survey.jl - An Efficient Framework for Analysing Complex Surveys

Ayush Patnaik[1]

[1]XKDR Forum

ABSTRACT

In the domain of survey data analysis, a persistent challenge involves accurately estimating variances while accounting for complex survey designs. The Survey.jl package implements

Keywords

Julia, Survey, Statistics, Sampling

1. Introduction

2. Survey design

The data, along with the sampling design, can be used to make a `SurveyDesign` object.
The parameters are:

(1) `data::DataFrame`, data in the form of a `DataFrame`
(2) `clusters::Symbol`, name of the column containing clusters.
(3) `strata::Symbol`, name of the column containing the strata.
(4) `weights::Symbol`, name of the column containing the weights.
(5) `popsize::Symbol`, name of the column containing the population size.

2.1 Example: Clustered and stratified

```
julia> nhanes = load_data("nhanes")

julia> SurveyDesign(nhanes; clusters=:SDMVPSU,
                    strata=:SDMVSTRA, weights=:WTMEC2YR)
SurveyDesign:
data: 8591 x 11 DataFrame
strata: SDMVSTRA
    [83, 84, 86  ...  81]
cluster: SDMVPSU
    [1, 1, 2  ...  2]
popsize: [244586.316, 43527.8366, 36124.9061
...  19331.022]
sampsize: [3, 3, 3  ...  3]
weights: [81528.772, 14509.2789, 12041.6354  ...
6443.674]
allprobs: [0.0, 0.0001, 0.0001  ...  0.0002]
```

There is only 1 constructor for all kinds of surveys. Every survey is assumed to be a complex survey. If there is no stratification, we assume that everything is part of 1 strata.

3. Estimation

Univariate : mean, median, total, quantile, etc. For example, the average height of adult men.

Multivaraite : ratio, regression, etc. For example, the relationship between height and weight.

3.1 Univariate

```
julia> mean(:api99, survey_design)
1x1 DataFrame
    Row | mean
        | Float64
-----|--------
    1 | 624.685
julia> quantile(:api99, survey_design, 0.7)
1x1 DataFrame
    Row | 0.7th percentile
        | Float64
-----|-----------------
    1 |           708.0
```

3.2 Multivariate

```
julia> glm(@formula(y ~ x), my_design, Normal(),

IdentityLink())

julia> ratio(:y, :x, my_design)
```

4. Replicate weights

The standard error of an estimator measures the average amount of variability or uncertainty in the estimated value. Standard errors are often provided alongside point estimates in various statistical packages, and these are suitable for simple random samples.
Estimate design based standard errors by simulation.

—Construction:
  —Replicate samples generated through resampling techniques (e.g., bootstrap, jackknife, BRR).
  —Each replicate sample represents a plausible variation of the original sample.
  —Standard error can be thought of as the variation if the sampling was done repeated.

—Usage:

(1) Generate replicate weights using bootstrap, jackknife, BRR, etc.
(2) Using each replicate weight, calculate the estimate.
(3) Calculate the standard error using the new set of estimates.

4.1 Bootstrapping

For bootstrap replicate $r (r = 1, \ldots, R)$, an SRS of $n_h - 1$ PSUs is selected with replacement from the n_h sample PSUs in stratum h. $m_{hj}(r)$ represents the number of times PSU j of stratum h is selected in replicate r.
The adjusted weight $w'_i(r)$ for observation i in replicate r is calculated as:

$$w'_i(r) = w_i(r) \times \frac{n_h}{n_h - 1} \times m_{hj}(r) \tag{1}$$

Here, $w_i(r)$ denotes the initial weight for observation i within replicate r, n_h is the total number of observations in stratum h, and $m_{hj}(r)$ is a multiplier term specific to observation i in PSU j of stratum h for replicate r.

```
    julia> srs = SurveyDesign(apisrs; weights=:pw);

    julia> bsrs = bootweights(srs; replicates =
1000)
    ReplicateDesign{BootstrapReplicates}:
    data: 200x1045 DataFrame
    strata: none
    cluster: none
    popsize: [6194.0, 6194.0, 6194.0  ...  6194.0]
    sampsize: [200, 200, 200  ...  200]
    weights: [30.97, 30.97, 30.97  ...  30.97]
    allprobs: [0.0323, 0.0323, 0.0323  ...
0.0323]
    type: bootstrap
    replicates: 1000
```

$\hat{\theta}^*_r$ is the estimator of θ, calculated the same way as $\hat{\theta}$ but using weights $w_i(r)$ instead of the original weights w_i.

$$\hat{V}_B(\hat{\theta}) = \frac{1}{R-1} \sum_{r=1}^{R} (\hat{\theta}^*_r - \hat{\theta})^2. \tag{2}$$

4.2 Jackknife

$$w_{i(hj)} = \begin{cases} w_i & i \notin h \\ 0 & i \in j_h \\ \dfrac{n_h}{n_h - 1} w_i & i \in h \text{ and } i \notin j_h \end{cases}$$