The R package NonProbEst for estimation in non-probability surveys

M. Rueda, R. Ferri-García, L. Castro

Abstract Different inference procedures are proposed in the literature to correct selection bias that might be introduced with non-random sampling mechanisms. The R package **NonProbEst** enables the estimation of parameters using some of these techniques to correct selection bias in non-probability surveys. The mean and the total of the target variable are estimated using Propensity Score Adjustment, calibration, statistical matching, model-based, model-assisted and model-calibratated techniques. Confidence intervals can also obtained for each method. Machine learning algorithms can be used for estimating the propensities or for predicting the unknown values of the target variable for the non-sampled units. Variance of a given estimator is performed by two different Leave-One-Out jackknife procedures. The functionality of the package is illustrated with example data sets.

Introduction

Since sampling theory was formalized in the beginning of the 20th century, surveys have been the main tool to obtain information from society and nature. Traditional surveys used telephone or face-to-face interviews for questionnaire administration, as well as mailing lists. However, the increase of costs, linked to the decrease in response rates, and the development of information and communication technologies have favored the use of new survey modes such as online or smartphone questionnaires. These modes make the sampling process cheaper and faster, but tend to amplify bias from several sources. More precisely, online surveys are often performed through a non-probability sampling, using self-selection procedures without a defined sampling frame where the inclusion probabilities are known or with deficient sampling frames with coverage issues, leading to higher levels of selection bias (Elliott and Valliant, 2017).

Some techniques can be used to correct selection bias in online non-probability surveys. A good overview of the various methods is given in Elliott and Valliant (2017). There are three important approaches: the pseudo-design based inference (or pseudo-randomisation (Buelens et al., 2018)), statistical matching and predictive inference.

In the pseudo-design based inference, the idea is to construct weights to correct for selection bias. The first method is estimating response probabilities and using them in Horvitz-Thompson or Hajek type estimators to account for unequal selection probabilities. The most used method to estimate response probabilities is Propensity Score Adjustment (see e.g. Lee and Valliant (2009)). This method uses a probability reference sample in addition to a non-probability convenience sample to construct a response propensity model. Sample matching is another approach also applied to tackle selection bias. A predictive model, with the target variable as the dependent variable, is built using data from the non-probability sample. This model is subsequently applied to a probability sample (where the target variable is not measured) to predict values of its individuals for an estimation of the population values. Similarly, predictive methods are based on superpopulation models. In this approach, a predictive model is fitted for the analysis variable from the sample and used to project the sample to the full population. This approach (that can be used with probability and non-probability samples) allows researchers to use the auxiliary information about covariates in different methods for predicting the unknown values. Most of these methods require special software for their implementation. The package **NonProbEst** implements some of these techniques.

The paper is structured as follows. First, we introduce the notation used throughout the paper and we discuss the different ways to do inference for non-probability surveys. In section 2.3 we briefly comment on the usefulness of Machine Learning (ML) Techniques in this context. Then, we describe the R package NonProbEst. In section 2.5 we briefly describe the use of the functions, including suitable examples, for each method.

Statistical methodology

Let U denote a finite population with N units, $U = \{1, \ldots, k, \ldots, N\}$. Let s_V be a volunteer non-probability sample of size n_V, self-selected from an online population U_V which is a subset of the total target population U. Let y be the variable of interest in the survey estimation. Without any auxiliary information, the population total of y, Y, is usually estimated with the following Horvitz-Thompson

type estimator:

$$\hat{Y}_{HT} = \sum_{k \in s_V} w_{vk} y_k \tag{1}$$

being w_{vk} a weight of the unit k set by the researcher to adjust the lack of response, lack of coverage, voluntariness, ... (e.g. by means of post-stratification). A simple choice is $w_{vk} = N/n_V$, that is, consider the sample of volunteers as if it was obtained with a simple random sampling design of the population U.

This estimator has a bias induced by various mechanisms regarding their application. The most important are the selection bias (due to the difference between sampled and nonsampled individuals on the probability to participate in a survey) and the coverage bias (the online population U_v is not the same of the target population U).

The key to successful weighting to remove the bias in non-probability surveys lies in the use of powerful auxiliary information. Auxiliary information can be available in different forms. We distinguish three different cases, called InfoTP, InfoES and InfoEP, depending on the information at hand.

- InfoTP: Only the population totals of the auxiliary variables are known (often called control totals). Possible sources of information are a census of the target population, an administrative register, ... One of the simplest and most frequently used control totals occurs when the information consists of known counts for a set of population groups.
- InfoES: The auxiliary variable values are available for every element in a probability sample. This reference survey is conducted on the same target population than the non-probability survey, with the main difference that the former has a better coverage and higher response rates than the latter, thus it is adequate to represent the behavior that the target population should have when a probability survey is performed on it.
- InfoEP: The auxiliary variable values are available for every element in the whole population. An example of this is when statistical agencies use auxiliary variables specified in different existing registers, for all the elements in the population.

We will now explain the main methods used to treat these biases depending on the type of information that is available.

InfoTP

Calibration

Let $\mathbf{x}_k$ be the value taken on unit k by a vector of auxiliary variables which population total is assumed to be known $\mathbf{X} = \sum_{k=1}^{N} \mathbf{x}_k$. The calibration estimation of Y consists in the computation of a new vector of weights w_k for $k \in s$ which modifies as little as possible the original sample weights, w_{vk}, which have the desirable property of producing unbiased estimations, respecting at the same time the calibration equations

$$\sum_{k \in s_V} w_k \mathbf{x}_k = \mathbf{X}. \tag{2}$$

Given a pseudo-distance $G(w_k, w_{vk})$, the calibration process consists in finding the solution to the minimization problem

$$\min_{w_k} \{ \sum_{k \in s_V} G(w_k, w_{vk}) \} \tag{3}$$

while respecting the calibration equation (2). Several distances were defined in Deville and Särndal (1992), being the linear distance one of the most commonly used. The resulting estimator of Y under the chi-square distance is the general regression estimator

$$Y_{reg} = \sum_{s_V} w_k y_k = \sum_{s_V} d_k y_k + (\mathbf{X} - \sum_{s_V} w_{vk} \mathbf{x}_k)' \hat{B}_{s_V} \tag{4}$$

where $\hat{B}_s$ is

$$\hat{B}_{s_V} = T_s^{-1} \sum_{s_V} w_{vk} \mathbf{x}_k y_k \tag{5}$$

being $T_s = \sum_{s_V} w_{vk} \mathbf{x}_k \mathbf{x}_k'$.

It is proved in Bethlehem (2010) that bias can be reduced through calibration only when the non-response due to volunteering has a Missing At Random scheme, while it cannot be equally done in Not Missing at Random situations (which are the most frequent).

InfoSP

Propensity Score Adjustment

The Propensity Score Adjustment method was originally developed by Rosenbaum and Rubin (1983) which sought to reduce the confounding bias between treatment and control groups in experimental designs. This approach would be considered in sampling research as well in combination with a reference sample (Rubin, 1986), but it was not proposed for online surveys until the early 2000's (Taylor et al., 2001).

It is expected that a sample collected by online recruitment would not follow the principles of a probability sampling, especially in those cases that the survey is filled by volunteer respondents. In such a situation, every individual is associated to a probability of participating in the survey which depends on her or his characteristics.

The propensity for an individual to take part on the non-probability survey is obtained by training a predictive model (often a logistic regression) on the dichotomous variable, I_{s_V}, which measures whether a respondent from the combination of both samples took part in the volunteer survey or in the reference survey. Covariates used in the model, $\mathbf{x}$, are measured in both samples (in contrast to the target variable which is only measured in the non-probability sample), thus the formula to compute the propensity of taking part in the volunteer survey with a logistic model, π, can be displayed as

$$\pi(\mathbf{x}) = \frac{1}{e^{-(\gamma^T \mathbf{x})} + 1} \tag{6}$$

for some vector γ, as a function of the model covariates.

We denote by s_R the reference sample and w_{Rk} the original design weight of the k individual in the reference sample

Several options for using the propensity scores in estimation are listed below:

- We can use the inverse of the estimated response propensity as a weight for constructing the estimator (Valliant, 2019):

$$\hat{Y}_{PSA1} = \sum_{k \in s_V} w_{Vk} y_k / \hat{\pi}(\mathbf{x}_k) = \sum_{k \in s_V} y_k w_k^{PSA1} \tag{7}$$

  where $\hat{\pi}(\mathbf{x}_k)$ is the estimated response propensity for the individual k of the volunteer sample as predicted using covariates $\mathbf{x}$.
- Alternatively, the approach proposed in Schonlau and Couper (2017) can be used to obtain weights for a Horvitz-Thompson type estimator using propensity scores. Weights are defined as

$$w_k^{PSA2} = \frac{1 - \hat{\pi}(\mathbf{x}_k)}{\hat{\pi}(\mathbf{x}_k)} \tag{8}$$

  and resulting estimator for the population total is given by

$$\hat{Y}_{PSA2} = \sum_{k \in s_V} y_k w_k^{PSA2} \tag{9}$$

- Valliant and Dever (2011) use the propensity scores to post-stratify the sample. The process is: sort the combined sample by $\hat{\pi}(\mathbf{x}_k)$; split the combined sample into g classes ($g = 5$ as the conventional choice following Cochran (1968)), each of which has about the same number of cases in the combined sample; and compute an average propensity, $\bar{\pi}_g$ within subclass g. Use $\bar{\pi}_g$ as the weight adjustment for every person in the subclass. Resulting estimator is:

$$\hat{Y}_{PSA3} = \sum_g \sum_{k \in s_{V_g}} w_{Vk} y_k / \bar{\pi}_g = \sum_g \sum_{k \in s_{V_g}} y_k w_k^{PSA3} \tag{10}$$

- Following the approach described in Lee and Valliant (2009) propensity scores are divided in g classes, where all units may have the same propensity score or at least be in a very narrow range and an adjustment factor is calculated as:

$$f_g = \frac{\sum_{k \in s_{R_g}} w_{Rk} / \sum_{k \in s_R} w_{Rk}}{\sum_{k \in s_{V_g}} w_{Vk} / \sum_{k \in s_V} w_{Vk}} \tag{11}$$

where s_{R_g} is the set of individuals in the reference sample that are in the gth class of propensity scores and s_{V_g} is the set of individuals in the volunteer sample that are in the gth class of propensity scores. Finally, the adjusted weights w^{PSA4} are the product of the original weights and the adjustment factor; following the same notation, the adjusted weight for individual k in s_{V_g} (i. e. the individual k of the gth propensity class in the volunteer sample) is computed as

$$w_k^{PSA4} = w_{Vk} f_g \tag{12}$$

and the estimator is given by

$$\hat{Y}_{PSA4} = \sum_g \sum_{k \in s_{V_g}} y_k w_k^{PSA4} \tag{13}$$

Research findings have shown that PSA successfully removes bias in some situations, but at the cost of increasing the variance (Lee and Valliant, 2009). Valliant and Dever (2011) showed that the estimation of a variable using PSA must be complemented with further weighting adjustment in order to make estimates less biased. The use of PSA with further calibration is studied in Lee and Valliant (2009) and Ferri-García and Rueda (2018), concluding that calibration adjustments are helpful if they are applied using the right covariates.

Variance estimation in PSA is not a simple issue. Valliant (2019) proposes an estimator of the variance for an estimator of a mean, $\hat{\bar{y}}$, based on linearization, but this estimator does not take into account the randomness of weight estimation, therefore it will tend to underestimate the variance.

Jackknife's variance estimator (Quenouille (1956)) can be seen as an acceptable alternative in nonprobability samples after applying PSA. Let $\hat{\bar{y}} = \frac{1}{N}\sum_{k \in s_V} w_k^{PSA} y_k$ be the estimator of the mean of y, his Leave-One-Out Jackknife estimator of the variance is given by:

$$\hat{V}(\hat{\bar{y}}) = \frac{n-1}{n} \sum_{j=1}^{n} (\bar{y}_{(j)} - \bar{y})^2 \tag{14}$$

where $\bar{y}_{(j)}$ is the value of the estimator $\hat{\bar{y}}$ after dropping unit j from s_V and where $\bar{y}$ is the mean of values $\bar{y}_{(j)}$.

Given that PSA weights are estimated from the available data, the exclusion of one unit can have an impact on the values of w_i and affect the variability of the estimator. This variability can be taken into account if propensities are recalculated for each of the n Leave-One-Out partitions. Thus a Jackknife estimator with recalculating weights is defined as:

$$\hat{V}_{rw}(\hat{\bar{y}}) = \frac{n-1}{n} \sum_{j=1}^{n} (\bar{y}_{rw(j)} - \bar{y}_{rw})^2 \tag{15}$$

where $\bar{y}_{rw(j)} = \frac{1}{N} \sum_{k \in s_V - \{j\}} w_k^{PSA}(j) y_k$, with $w_k^{PSA}(j)$ the PSA weight obtained from the sample $s_V - \{j\}$ and $\bar{y}_{rw}$ is the mean of values $\bar{y}_{rw(j)}$.

Statistical matching

The statistical matching method was introduced by Rivers (2007). The idea is to model the relationship between y_k and $\mathbf{x}_k$ using the volunteer sample s_V in order to predict y_k for the reference sample. That is, the matching estimator is given by:

$$\hat{Y}_{SM} = \sum_{s_R} \hat{y}_k w_{Rk}$$

being $\hat{y}_k$ the predict value of y_k.

The key is how to predict the values y_k. Usually $\hat{y}_k = \mathbf{x}_k' \hat{\beta}$ being $\hat{\beta} = \sum_{k \in s_V} y_k \mathbf{x}_k / \sum_{k \in s_V} \mathbf{x}_k' \mathbf{x}_k$ but other methods can be considered as donor imputation (Rivers, 2007) or fractional donor imputation (Kim and Fuller, 2004).

A major drawback of matching is that the precision of the non-probability sample reduces to the standard error of the reference sample (Buelens et al., 2018). These authors also justify that matching is based on strong ignorability assumptions and can lead biased estimators if the assumptions are not met.

InfoUP

The prediction approach is based on superpopulation models, which assume that the population under study $\mathbf{y} = (y_1, ..., y_N)'$ is a realization of super-population random variables $\mathbf{Y} = (Y_1, ..., Y_N)'$ having a superpopulation model ξ. To incorporate auxiliary information $\mathbf{x}_k$ available for all $k \in U$ on assume a superpopulation for y built on some mean function of $\mathbf{x}$:

$$Y_k = m(\mathbf{x}_k) + e_k, \quad k = 1, ..., N. \tag{16}$$

The random vector $e = (e_1, ..., e_N)'$ is assumed to have zero mean and a positive definite covariance matrix which is diagonal (Y_k are mutually independent).

Using a set of covariates, $\mathbf{x}$, measured in s_V and $\bar{s}_V = U - s_V$ it is possible to estimate the values of y in $\bar{s}_V$ with regression modeling such that the estimated value of y for an individual k can be calculated through the following expression:

$$\hat{y}_k = E_m(y_k|\mathbf{x}_k) \tag{17}$$

m alludes to the specific model which provides the expectation of y_k, and $\mathbf{x}_k$ are the values of the k-th individual in the covariates $\mathbf{x}$.

We can use the auxiliary information in several ways to define several estimators:

- the model-based estimator:

$$\hat{Y}_m = \sum_{k \in s_V} y_k + \sum_{k \in \bar{s}_V} \hat{y}_k \tag{18}$$

- the model-assisted estimator:

$$\hat{Y}_{ma} = \sum_{k \in U} \hat{y}_k + \sum_{k \in s_V} (y_k - \hat{y}_k) w_{Vk} \tag{19}$$

- the model-calibrated estimator:

$$\hat{Y}_{mcal} = \sum_{k \in s_V} y_k w_k^{CAL} \tag{20}$$

  where w_k^{CAL} are such that they minimize $\sum_{k \in s} G\left(w_k^{CAL}, w_{Vk}\right)$, where $G(\cdot, \cdot)$ is a particular distance function, subject to

$$\sum_{k \in s_V} w_k^{CAL} \hat{y}_k = \sum_{k \in U} \hat{y}_k.$$

Usually the linear regression model is used, $E_m(y_k|\mathbf{x}_k) = \mathbf{x}_k'\beta$ and the above estimators can be rewritten as a type of regression estimators.

Prediction estimators need complete information about the auxiliary variables (InfoEP) and can fail if the model is not true, but might potentially be fruitful to correct for selection bias in informative sampling (Buelens et al., 2018).

Use of machine learning algorithms in non-probability samples

The emerging data sources like Big Data can be used in combination to traditional survey samples for construct more valid inferences. Machine Learning (ML) methods can be used for the matter, given their known advantages in high dimensional environments. There are several types of learning algorithms but for this package we focus on classification and regression. Classification aims to identify the category to which a new observation belongs while regression is used for prediction in real-valuated variables. Both are trained with known observations to make predictions based on some covariates.

There is a vast spectrum of classification and regression algorithms to take into account, starting from the basic linear and logistic regressions and its extensions, like Ridge regression (Hoerl and Kennard, 1970). Other examples are decision trees which uses tree-like graphs , like the C4.5 (Quinlan, 1993). More modern approaches even build ensembles of decision trees with outstanding results, like XGBoost (Chen and Guestrin, 2016). During the last few years, deep learning models have been dramatically improving the state-of-the-art (LeCun et al., 2015). However, many other techniques are still being widely used and developed, like some bayesian methods (Park and Casella, 2008). Having so many different options, choosing the right learning algorithm for each problem is key for obtaining optimal results.

Regarding survey research, the use of ML algorithms has been studied in the last few years for deriving model-assisted estimators (Montanari and Ranalli (2007); Baffetta et al. (2009); Breidt et al. (2017)). In the prediction approach ML algorithms uses the sample to train a model capturing the behaviour of a target variable which is to be estimated, and applies it to the nonsampled individuals to obtain population-level estimates. Applications of machine learning algorithms in PSA for nonresponse propensity have been studied for classification and regression trees (Phipps et al., 2012) and Random Forests (Buskirk and Kolenikov, 2015); their efficacy on reducing nonresponse bias in comparison to logistic regression depends on the available covariates and the complexity of the relationships. (Chen et al., 2019) use LASSO for calibrating non-probability surveys. (Buelens et al., 2018) review existing inference methods to correct for selection bias and recommend adding ML methods to deal with non-probability samples.

NonProbEst allows the use of a wide variety of classification and regression algorithms for model-based, model-assisted and model-calibrated estimators, matching and PSA (which only works with classification). It offers so many alternatives by relying on **caret** (Kuhn, 2018), a well known machine learning package.

The R package NonProbEst

The package **NonProbEst** implements in R a set of techniques for estimation in non-probability surveys, using various approaches which correspond to several frameworks. Functions in the package allow to obtain calibration weights via `calib_weights`, propensity scores via `propensities` and matching predictions for a reference sample via `matching`. Propensity scores can be transformed into weights by all of the approaches mentioned in previous sections via functions `lee_weights`, `sc_weights`, `valliant_weights`, `vd_weights`. These weights can be used for estimation of total, mean and proportion of a given target variable measured in a sample using functions `total_estimation`, `mean_estimation`, `prop_estimation`. Alternatively, total and mean can also be calculated using a model-based, a model-assisted or a model-calibrated approach with the functions `model_based`, `model_assisted` and `model_calibrated` respectively. The variance of the estimators can be calculated using the Leave-One-Out Jackknife method, this is, recalculating the set of weights after substracting one unit or not, by means of the functions `generic_jackknife_variance` and `jackknife_variance`, and without recalculating the weights via `fast_jackknife_variance`. Frequentist confidence intervals of the estimates can be directly computed with the `confidence_interval` function.

Calibration weights are obtained using the `calib` function of the **sampling** package (Tillé and Matei, 2016) for g-weights computation. `calib_weights` offers a wrapper for calculation of final weights straight from the dataset. Functions that require prediction techniques, such as `propensities`, `matching`, `model_based`, `model_assisted`, `model_calibrated` and `jackknife_variance`, use the `train` function from the **caret** package (Kuhn, 2018). This function allows the user to use any of the algorithms in the large list of functions which are covered by `train`, with the possibility of optimizing hyperparameters for a better performance of the predictors. For propensity estimation, only classification algorithms should be used as the target variable is binary (participation in the probability sample vs participation in the non-probability sample). Case weights are used to balance both classes (for models that accept them). For matching, model-based and model-assisted estimations, algorithms should account for the type of variable of the target feature.

Note that weighting formulas for PSA from Lee (2006) and Valliant and Dever (2011) require applying a stratification procedure. In both `lee_weights` and `vd_weights` the same procedure is applied: the vector of propensities is sorted increasingly, and the individuals are equally divided in g strata of the same length according to their position in the sorted vector. g is defined by the user, and the procedure results in a vector with the strata number (from 1 to g) to which a given individual corresponds. This stratification avoids errors that could arise from the lack of unique values.

Three datasets are available in the package: `sampleP`, `sampleNP` and `population`. These fictitious datasets were created as described in Ferri-García and Rueda (2018); `sampleP` represents a probability sample of size $n_r = 500$ extracted by simple random sampling from a frame covering the entire population, while `sampleNP` represents a non-probability sample of size $n_v = 1000$ extracted by simple random sampling from a frame covering only the subpopulation of individuals who have access to Internet. The dataset of the complete population of size $N = 50000$ is available in `population`. Variables available in each dataset differ, with `sampleNP` having the largest amount of variables. In the aforementioned dataset, three variables (`vote_gen`, `vote_pens`, `vote_pir`) measuring whether an individual would vote to a given party ("gen", "pens" or "pir") in an election or not. Probabilities of voting to party "gen", "pens" or "pir" are higher if the individual is a woman, and elder person and has access to the Internet, respectively. These variables are only measured in `sampleNP`, meaning that adjustment methods have to be applied in order to produce reliable estimates of voting intentions. For the matter, the rest of the available variables in the dataset, which are also included in `sampleP`

(except for the language) and `population`, can be used. `education_primaria`, `education_secundaria`, `education_terciaria` are three disjunct variables measuring the education level of the individual (Primary, Secondary or Tertiary Education), while `age` and `sex` measures the numeric age and the gender (0 female, 1 male). Finally, `language` measures whether the individual's native language is the official language or not. The absence of certain variables in the datasets accounts for real situations where not all the information is available at individual level.

It must be mentioned that the use of `jackknife_variance` for calculating the variance of the estimators via Leave-One-Out Jackknife will be computationally slower than the `fast_jackknife_variance` alternative. Recalculating the weights in each iteration means that the weighting procedure has to be repeated as many times as individuals are in the non-probability sample. If Propensity Score Adjustment is used for weighting, the models have to be rebuilt in each iteration, resulting in larger computation times which will depend on the computational costs of the algorithms used for propensity estimation. Note that `generic_jackknife_variance` will behave similarly if the estimator passed as argument involves predictive modelling algorithms or other costly procedures. To show the difference of procedures, we calculated the Leave-One-Out Jackknife estimated variance of the estimator of the mean for the variable `vote_pir` in a non-probability sample of size $n_v = 100$ extracted by simple random sampling on the `sampleNP` dataset, using a probability sample of size $n_r = 100$ extracted by simple random sampling on the `sampleP` dataset as the reference sample data. Considering a population of $N = 50000$, variance estimates of the estimator weighted by PSA using different algorithms were computed, measuring the computation elapsed time. All the calculations were performed in a Intel(R) Core(TM) i7-3770 CPU up to 3.40GHz. Results can be consulted in Table 1

Weight recalculation	PSA algorithm	R function	Elapsed time (seconds)
No	Logistic regression	glm	0.004999876
Yes	Logistic regression	glm	75.56034
Yes	CART	rpart	102.3409
Yes	Random Forest	rf	203.7737
Yes	GBM	gbm	453.731
Yes	Neural Network	nnet	719.733

Table 1: Total elapsed time of Leave-One-Out Jackknife variance estimation under recalculation of weights in each iteration for a set of predictive models, with sample sizes of 100 for both the probability and the non-probability sample

In this example, the variance estimation with recalculations takes more than 15000 times the seconds that it takes without recalculations if logistic regression is the method used for propensity estimation, and almost 144000 times if feed-forward neural networks are used. Time differences might be different depending on the data, the estimator and the algorithm, but they will be largely appreciable in all cases.

In order to ilustrate how the resources in the package can be used for estimation in non-probability surveys, some examples of each adjustment covered by the package are developed in the following section.

Inference in non-probability samples with NonProbEst

InfoTP: Calibration

Suppose that a non-probability sample of 1000 individuals recruited via online surveying is available for estimating the vote intention in a given election. For the matter, `sampleNP` will be used as the non-probability sample data.

```
> library(NonProbEst)
> head(sampleNP)
  vote_gen vote_pens vote_pir education_primaria education_secundaria education_terciaria age sex language
1        0         1        0                  1                    0                   0  66   1        1
2        0         0        1                  0                    0                   1  30   1        1
3        1         0        0                  0                    1                   0  62   0        1
4        0         0        1                  1                    0                   0  33   0        1
5        0         0        1                  0                    1                   0  30   0        1
6        0         0        0                  1                    0                   0  69   1        1
```

Some auxiliary information is available in the sample; more precisely, individual data on education, age, gender and language (as described in the previous Section) can be used for mitigating the effects of coverage error. Population totals are available for all of these auxiliar variables, as they have been measured for the entire population. They can be retrieved from the `population` dataset:

```
> head(population)
  education_primaria education_secundaria education_terciaria age sex language
1                  0                    1                   0  39   1        1
2                  0                    0                   1  55   0        1
3                  1                    0                   0  35   0        1
4                  1                    0                   0  58   1        1
5                  1                    0                   0  36   1        1
6                  0                    1                   0  61   1        1
> totals <- colSums(population)
> totals
  education_primaria education_secundaria  education_terciaria                 age                 sex            language
               25287                10546                14167             2539340               24430               45429
```

If the variables of which population totals are available are not disjunct, Raking calibration can be applied in order to estimate cell counts and account for the lack of information. This can be done with the `calib_weights` function; in this case, the `Xs` argument were the dataset `sampleNP` selecting the auxiliar variables only. Other arguments involve the totals previously obtained and the initial weights, which allows the user to specify whether sampling design weights were used or not. In the latter case, unitary weights should be provided as a vector of ones of length equal to the number of individuals in the non-probability sample. Population size and method to be used by the `calib` function from **sampling** have to be specified.

```
> covariates <- colnames(sampleNP)[4:9]
> initial_weights <- rep(1, nrow(sampleNP))
> w <- calib_weights(sampleNP[, covariates], totals, initial_weights,
        N = 50000, method = "raking")
```

Once we obtain the weights, estimates for the mean (proportion if the variable is binary) or the total of any variable present in the non-probability sample can be obtained using `mean_estimation` or `total_estimation` respectively. For example, the estimated proportion of votes for each party can be obtained with the following code:

```
> mean_estimation(sampleNP, w, "vote_gen", N = 50000)
  vote_gen
0.09824163
> mean_estimation(sampleNP, w, "vote_pens", N = 50000)
vote_pens
0.3726149
> mean_estimation(sampleNP, w, "vote_pir", N = 50000)
 vote_pir
0.3905399
```

If these estimates are compared to those which would be obtained if no adjustment was used, the effect of calibration is notorious. As the presence of "gen" voters in the sample is MCAR, estimates do not differ, but in the case of "pens" voters whose presence is MAR, the calibration approach gives a larger estimate which can be explained by the fact that the overrepresentation of younger people in the sample has been corrected up to a point. To a much lesser extent, this correction is also noticeable in the estimation of vote to "pir" (presence of their voters in the sample is NMAR).

```
> sum(sampleNP$vote_gen)/nrow(sampleNP)
[1] 0.096
> sum(sampleNP$vote_pens)/nrow(sampleNP)
[1] 0.346
> sum(sampleNP$vote_pir)/nrow(sampleNP)
[1] 0.404
> sum(sampleNP$vote_gen)/nrow(sampleNP) -
+        mean_estimation(sampleNP, w, "vote_gen", N = 50000)
   vote_gen
-0.00224163
> sum(sampleNP$vote_pens)/nrow(sampleNP) -
+        mean_estimation(sampleNP, w, "vote_pens", N = 50000)
  vote_pens
-0.02661494
> sum(sampleNP$vote_pir)/nrow(sampleNP) -
+        mean_estimation(sampleNP, w, "vote_pir", N = 50000)
  vote_pir
0.01346014
```

The variance of the estimates can be assessed through Leave-One-Out Jackknife, both with or without reweighting in each iteration. In the former case, a function must be created by the user for

such a task. In the following lines, a function example is developed for estimating the variance on the estimation of the proportion of votes for the "pir" party:

```
### Leave-One-Out Jackknife variance estimation with reweighting
> estimator <- function(s){
     initial_weights <- rep(1, nrow(s))
     w <- calib_weights(s[,covariates], totals, initial_weights, N = 50000,
                method = "raking")
     return(mean_estimation(s, w, "vote_pir", N = 50000))
     }
> v_r <- generic_jackknife_variance(sampleNP, estimator, N = 50000)
> v_r
[1] 0.0003352199
### Leave-One-Out Jackknife variance estimation without reweighting
> v_nr <- fast_jackknife_variance(sampleNP, w, estimated_vars = "vote_pir", N = 50000)
> v_nr
    vote_pir
0.0003189449
```

These estimates of the variance can be used for the construction of confidence intervals for the estimation of the proportion via `confidence_interval` function. This function requires the point estimator and the standard deviation as arguments, with the option to fix the confidence level. If not specified by the user, the confidence interval is calculated at 95% confidence level.

```
> ic_r <- confidence_interval( mean_estimation(sampleNP, w, "vote_pir", N = 50000),
        sqrt(v_r)
        )
> ic_r
lower.vote_pir upper.vote_pir
     0.3546549      0.4264249
> ic_nr <- confidence_interval( mean_estimation(sampleNP, w, "vote_pir", N = 50000),
        sqrt(v_nr)
        )
> ic_nr
lower.vote_pir upper.vote_pir
     0.3555368      0.4255429
```

InfoSP: Propensity Score Adjustment

Suppose that, in addition to the non-probability sample, a probability sample of the same target population is available as auxiliary information. The target variable is not measured, but some other variables which are also available in the non-probability sample have been measured on it. For the matter, `sampleP` will be used as data from the probability sample.

```
> head(sampleP)
  education_primaria education_secundaria education_terciaria age sex
1                  1                    0                   0  35   1
2                  0                    0                   1  64   0
3                  1                    0                   0  55   1
4                  0                    1                   0  61   1
5                  0                    0                   1  35   0
6                  1                    0                   0  51   1
```

In order to reduce the selection bias, Propensity Score Adjustment can be used in this case for reweighting. This procedure is implemented in the `propensities` function; it requires both samples, the list of covariates to be used to build the models for propensity estimation, and three arguments regarding technical aspects of the adjustment: the prediction algorithm (must match any of the list of `caret` supported algorithms), a boolean indicating whether smoothing of propensities is applied or not, and a vector of strings specifying the preprocessing procedures to be passed to `train` (by default, preprocessing is not applied). Further arguments to be passed to `train` can be specified.

In this example, the propensity of participating will be estimated using k-Nearest Neighbors with further smoothing and a parameter grid of all the odd numbers between 3 and 11 for optimization of k. The covariates will be all the variables measured in `sampleP`. The result will be a list with two vectors: the estimated propensities for individuals in the non-probability (convenience) and the probability (reference) sample respectively.

```
> covariates <- colnames(sampleP)
> pi <- propensities(sampleNP, sampleP, covariates,
        algorithm = "knn", smooth = T, tuneGrid = data.frame(k = seq(3, 11, by = 2)))
> summary(pi$convenience)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.3079  0.6249  0.6873  0.6834  0.7584  0.9995
> summary(pi$reference)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.3079  0.5384  0.6388  0.6236  0.6998  0.9469
```

The propensities must be subsequently transformed into weights for their application in survey estimation. Transformations available in **NonProbEst** include approaches developed by Lee (2006) and Lee and Valliant (2009) in the `lee_weights` function, Valliant and Dever (2011) in the `vd_weights` function, Schonlau and Couper (2017) in the `sc_weights` function and Valliant (2019) in the `valliant_weights` function. `lee_weights` and `vd_weights` require propensities of both samples and a number of strata (5 by default), while `sc_weights` and `valliant_weights` only require propensities of the non-probability sample.

For example, if we want to apply propensities via weights developed in Valliant and Dever (2011) for the estimation of voting intention to party "pir", we can do it with the following code:

```
> wi <- vd_weights(convenience_propensities = pi$convenience,
                   reference_propensities = pi$reference)
> summary(wi)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.233   1.376   1.493   1.505   1.632   2.011
> mean_estimation(sample = sampleNP, weights = wi,
                  estimated_vars = "vote_pir")
 vote_pir
0.4006072
#Estimation of the 95% confidence interval
> estim <- mean_estimation(sample = sampleNP, weights = wi,
                  estimated_vars = "vote_pir")
> std_dev <- fast_jackknife_variance(sample = sampleNP, weights = wi,
                          estimated_vars = "vote_pir", N = 50000)
> confidence_interval(estimation = estim, std_dev = std_dev, confidence = 0.95)
lower.vote_pir upper.vote_pir
     0.4001341      0.4010803
```

Note that for those weights that are calculated by means of propensity stratification, propensities of the individuals in the convenience and reference sample are needed. If they are calculated by inverting propensities, only those for the individuals in the convenience sample are needed. For example, if we calculate weights via the formula developed in Schonlau and Couper (2017), the code is:

```
> wi <- sc_weights(propensities = pi$convenience)
> summary(wi)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
0.0004998 0.3185741 0.4549419 0.5044062 0.6003197 2.2479720
```

Apart from direct estimation, resulting weights can be used as inputs in the `initial_weights` argument of the `calib_weights` function for the estimation with PSA and calibration, or with the package **survey** (Lumley, 2018) for more complex analysis.

InfoUP: superpopulation estimators

In this case, in addition to the non-probability sample, the population itself is avaliable for some covariates. However, the target variable is only measured in the non-probability sample. For the matter, `sampleNP` will be used as the non-probability sample data and `population` will be used as the population data.

The model-based estimator can be used to estimate the population total (or mean) for the target variable. In this example, the expected number of votes for "pens" will be estimated with regularized logistic regression as learning algorithm. This procedure is implemented in the `model_based` function. It requires the sample, the population, the covariates names and the target variable as arguments. In our example, the specific algorithm and a normalization preprocessing are passed to change default behaviour. Since no optimization strategy is specified in this case, a default bootstrap will be applied.

```
> covariates <- c("education_primaria", "education_secundaria",
        "education_terciaria", "age", "sex", "language")
> mySample = sampleNP
> mySample$vote_pens = factor(mySample$vote_pens, c(0, 1), c('F', 'T'))
> model_based(mySample, population, covariates, "vote_pens",
        positive_label = 'T', algorithm = "glmnet", proc = c("center", "scale"))
[1] 18282.51
```

If the proportion of votes has to be estimated, rather than the total, it would be as simple as adding the `estimate_mean` argument as follows:

```
> model_based(mySample, population, covariates, "vote_pens", positive_label = 'T',
        algorithm = "glmnet", proc = c("center", "scale"), estimate_mean = TRUE)
[1] 0.366757
```

Alternatively, model-calibrated estimator can be used to achieve higher efficiency in some situations. In that case, design weights have to be specified in the argument "weights", in addition to the rest of arguments previously described. If no sampling design was followed in data collection, which is the case that we suppose in our example, we can specify unitary weights by turning the parameter to 1, as it is done in the following code:

```
> model_calibrated(sample_data = mySample, weights = 1, full_data = population,
+                  covariates = covariates, estimated_var = "vote_pens", positive_label = 'T',
+                  algorithm = "glmnet", proc = c("center","scale"),
+                  estimate_mean = TRUE)
[1] 0.365945
```

Conclusion and future developments

In this paper we show how the **NonProbEst** package can simplify the application of different weighting methods to correct selection bias in non-probability surveys. This package is, to the best of our knowledge, the first package that supports the user beyond estimation in PSA, PSA+calibration, statistical matching or model-calibration. Another important feature is that a wide range of ML techniques can be used to optimize the information provided by the auxiliary variables.

Additional features will be integrated in future versions of the package. Some simplified wrappers will be developed for some methods so non-expert users can also easily apply them, more parameters will be avaliable for estimation and further support for weighted models will be added. Also, other techniques for variance estimation can be considered. Many of these features can already be applied combining **NonProbEst** with the **survey** package, as noted before.

Regarding Machine Learning, methods for variable selection will be studied as well as the use of more advanced deep learning libraries outside of **caret**'s scope. Variable selection would help explaining the bias and choosing the best covariates for its correction. Better deep learning libraries would allow the use of state-of-the-art algorithms.

Acknowledgments

This work is partially supported by Ministerio de Economía y Competitividad of Spain (grant MTM2015-63609-R) and by Ministerio de Ciencia, Innovación y Universidades (grant FPU17/02177).

Bibliography

F. Baffetta, L. Fattorini, S. Franceschi, and P. Corona. Design-based approach to k-nearest neighbours technique for coupling field and remotely sensed data in forest surveys. *Remote Sensing of Environment*, 113(3):463–475, 2009. [p6]

J. Bethlehem. Selection bias in web surveys. *International Statistical Review*, 78(2):161–188, 2010. [p3]

F. J. Breidt, J. D. Opsomer, et al. Model-assisted survey estimation with modern prediction techniques. *Statistical Science*, 32(2):190–205, 2017. [p6]

B. Buelens, J. Burger, and J. A. van den Brakel. Comparing inference methods for non-probability samples. *International Statistical Review*, 86(2):322–343, 2018. [p1, 4, 5, 6]

T. D. Buskirk and S. Kolenikov. Finding respondents in the forest: A comparison of logistic regression and random forest models for response propensity weighting and stratification. *Survey Methods: Insights from the Field*, page 17, 2015. [p6]

J. K. T. Chen, R. L. Valliant, and M. R. Elliott. Calibrating non-probability surveys to estimated control totals using lasso, with an application to political polling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 68(3):657–681, 2019. [p6]

T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016. [p5]

W. G. Cochran. The effectiveness of adjustment by subclassification in removing bias in observational studies. *Biometrics*, pages 295–313, 1968. [p3]

J.-C. Deville and C. E. Särndal. Calibration estimators in survey sampling. *Journal of the American statistical Association*, 87(418):376–382, 1992. [p2]

M. R. Elliott and R. Valliant. Inference for nonprobability samples. *Statistical Science*, 32(2):249–264, 2017. [p1]

R. Ferri-García and M. d. M. Rueda. Efficiency of propensity score adjustment and calibration on the estimation from non-probabilistic online surveys. *SORT-Statistics and Operations Research Transactions*, 1(2):159–182, 2018. [p4, 6]

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. [p5]

J. Kim and W. Fuller. Fractional hot deck imputation. *Biometrika*, 91(3):559–578, 2004. [p4]

M. Kuhn. *caret: Classification and Regression Training*, 2018. URL `https://CRAN.R-project.org/package=caret`. R package version 6.0-81. [p6]

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. [p5]

S. Lee. Propensity score adjustment as a weighting scheme for volunteer panel web surveys. *Journal of official statistics*, 22(2):329–349, 2006. [p6, 10]

S. Lee and R. Valliant. Estimation for volunteer panel web surveys using propensity score adjustment and calibration adjustment. *Sociological Methods & Research*, 37(3):319–343, 2009. [p1, 3, 4, 10]

T. Lumley. *survey: Analysis of Complex Survey Samples*, 2018. URL `https://CRAN.R-project.org/package=survey`. [p10]

G. E. Montanari and M. G. Ranalli. Multiple and ridge model calibration for sample surveys. In *Proceedings of the Workshop in Calibration and estimation in surveys, Ottawa*, 2007. [p6]

T. Park and G. Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103(482): 681–686, 2008. [p5]

P. Phipps, D. Toth, et al. Analyzing establishment nonresponse using an interpretable regression tree model with linked administrative data. *The Annals of Applied Statistics*, 6(2):772–794, 2012. [p6]

M. H. Quenouille. Notes on bias in estimation. *Biometrika*, 43(3/4):353–360, 1956. [p4]

J. R. Quinlan. *C4. 5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993. [p5]

D. Rivers. Sampling for web surveys. *Presented in Joint Statistical Meetings*, 2007. Salt Lake City, UT. [p4]

P. R. Rosenbaum and D. B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983. [p3]

D. B. Rubin. Statistical matching using file concatenation with adjusted weights and multiple imputations. *Journal of Business & Economic Statistics*, 4(1):87–94, 1986. [p3]

M. Schonlau and M. P. Couper. Options for conducting web surveys. *Statistical Science*, 32(2):279–292, 2017. [p3, 10]

H. Taylor, J. Bremer, C. Overmeyer, J. W. Siegel, and G. Terhanian. The record of internet-based opinion polls in predicting the results of 72 races in the november 2000 us elections. *International Journal of Market Research*, 43(2):127–135, 2001. [p3]

Y. Tillé and A. Matei. *sampling: Survey Sampling*, 2016. URL https://CRAN.R-project.org/package=sampling. R package version 2.8. [p6]

R. Valliant. Comparing alternatives for estimation from nonprobability samples. *Journal of Survey Statistics and Methodology*, 2019. [p3, 4, 10]

R. Valliant and J. A. Dever. Estimating propensity adjustments for volunteer web surveys. *Sociological Methods & Research*, 40(1):105–137, 2011. [p3, 4, 6, 10]

María del Mar Rueda
Department of Statistics and Operations Research
University of Granada
Spain
ORCiD: 0000-0002-2903-8745
mrueda@ugr.es

Ramón Ferri-García
Department of Statistics and Operations Research
University of Granada
Spain
ORCiD: 0000-0002-9655-933X
rferri@ugr.es

Luis Castro
Department of Statistics and Operations Research
University of Granada
Spain
luiscastro193@correo.ugr.es