Journées de méthodologie statistique de l'Insee 2022

Treatment of unit nonresponse in surveys through machine learning methods : an empirical comparison - Working version

Khaled Larbi (*), David Haziza (**), Mehdi Dagdoug (***)

(*) Insee / Ensae
(**) University of Ottawa
(***) Université de Bourgogne France-Comté

khaled.larbi@insee.fr/khaled.larbi@ensae.fr
dhaziza@uottawa.ca
mohamed_mehdi.dagdoug@univ-fcomte.fr

Mots-clés. Non réponse totale, machine learning, calage, simulation, échantillonnage.

Domaines. Non réponse, Machine Learning / apprentissage automatique, classification.

Résumé

Cet article propose une étude empirique sur l'effet de différentes méthodes d'apprentissage automatique (machine learning) dans un contexte de repondération de la non-réponse totale pour des données d'enquête. Soit $\mathscr{U}$ une population de taille N et $\mathscr{S}$ un échantillon probabiliste de taille n tiré dans $\mathscr{U}$ selon un plan de sondage aléatoire donné. Dans un contexte de non-réponse totale, seul un sous-ensemble $\mathscr{S}_r$ de l'échantillon $\mathscr{S}$ répond à l'enquête. Le but est d'estimer le total d'une variable d'intérêt y donné par $t_y = \sum_{i\in\mathscr{U}} y_i$. Soit $(p_k)_{k\in\mathscr{S}}$ les probabilités de réponses. Si les probabilités de réponse étaient connues, un estimateur sans biais du total serait l'estimateur par double dilatation, $\hat{t}_{y,exp} = \sum_{i\in\mathscr{S}_r} \frac{y_i}{p_i\pi_i}$ où π_i désigne les probabilités d'inclusion d'ordre un de l'invididu i associées au plan de sondage. Cependant, les probabilités de réponse $(p_k)_{k\in\mathscr{S}_r}$ n'étant pas connues en pratique, elles sont estimées au moyen d'un modèle de non-réponse. À partir de ces estimations des probabilités de réponse, deux estimateurs du total sont considérés : l'estimateur PSA, $\hat{t}_{y,PSA} = \sum_{i\in\mathscr{S}_r} \frac{y_i}{\hat{p}_i\pi_i}$, et l'estimateur de Hàjek $\hat{t}_{y,Hajek} = \frac{N}{\hat{N}}\hat{t}_{y,PSA}$ où $\hat{N}$ est l'estimateur PSA du nombre d'individus dans la population.

Nous présenterons les résultats d'une étude empirique dont le but est de comparer la performance de ces deux estimateurs en termes de biais et d'efficacité. Dans un contexte de plan de sondage stratifié à probabilités inégales, nous avons utilisé plusieurs méthodes afin d'estimer les probabilités de réponse.

Trois autres méthodes dérivées des estimations obtenues par apprentissage automatique sont proposées :

— après avoir fourni un jeu de probabilités estimées (par exemple, par régression logistique), il est possible de calculer d'autres estimations des probabilités de réponses en utilisant la méthode des scores (ou groupe homogène de réponse) [HB07]. Les probabilités estimées servent à créer des groupes d'individus homogènes par rapport à la probabilité estimée (en triant l'échantillon selon la probabilité estimée et en découpant en K groupes). Pour chaque groupe homogène, la probabilité de réponse estimée correspond au taux de réponse observé dans ce groupe. Cette approche permet d'être plus robuste à des problèmes de mauvaise spécification du modèle de non-réponse.
— il est également possible de combiner plusieurs estimations des probabilités de réponse provenant de différentes méthodes (CART, SVM et BART par exemple) afin de créer un jeu de probabilités estimées. Nous considérons deux manières de combiner les estimations dans le but d'accroître la robustesse des estimateurs :
  — estimation robuste basée sur le calage de plusieurs jeux de probabilités estimées.
  — estimation robuste basée sur une méthode ensembliste appliquées aux probabilités estimées.

Les performances de ces estimateurs seront estimées à l'aide du biais relatif Monte-Carlo et du risque quadratique moyen Monte-Carlo.

Abstract en Anglais

In the context of a survey with unit non-response, reweighting provides estimators corrected for induced bias. Reweighting requires estimating response probabilities based on auxiliary informations. Machine learning methods can be used to model these probabilities in a flexible way. An empirical study was performed where the performance of two estimators of the total are compared according to the machine learning methods used to model the non-response probabilities. We studied, with which machine learning algorithm, score method [HB07] provided more robust result. We also worked on methods that aggregate several sets of response probabilities such as calibration.

1 Preliminaries

Surveys are used to get information on variables of interest in a finite population denoted by $\mathscr{U} = \{1, \ldots, N\}$. This information is computed by considering results available on a subset of units of the population $\mathscr{S}$ called *sample*. For example, a national statistical institute can use surveys for the population census : the institute wishes to obtain the total number of people in France without questioning the whole population. To achieve this, the institute's interviewers only consider a sample of households.

Samples are drawn using a probability measure on the set of all subsets of $\mathscr{U}$ endowed by the $\sigma-$field generated by it. This measure is called the survey design and it will be denoted by P in the following. For a given survey design, one can compute first order probability of a unit k, denoted by π_k, which is the probability that k belongs to the sample.

The statistician will be interested in measurable functions of the variable of interest y such as totals $t_y := \sum_{k \in \mathscr{U}} y_k$ on the whole population. Nevertheless, y is known only for the units of the sample $\mathscr{S}$ and then estimators of t_y need to be based on these values and the survey design.

Assuming that all units have a non-zero first order probability (no coverage problem) and answer

when they are drawn, a unbiased estimator of the total is the so-called Narain-Horvitz-Thompson estimator [Nar51][HT52] (or $\pi-$estimator)

$$\hat{t}_{y\pi} := \sum_{k\in\mathcal{S}} \frac{y_k}{\pi_k} = \sum_{k\in\mathcal{U}} \frac{y_k}{\pi_k} I_k \tag{1}$$

where I_k stands for the dummy variable of the belonging of k in $\mathcal{S}$.

However, in presence of unit non-response, some individuals from the sample $\mathcal{S}$ may decide not to answer : only units from a subsample $\mathcal{S}_r \subset \mathcal{S}$ answer. This phenomena can lead to bias in estimations. Unit non-response can be considered as a second survey design given the sample $\mathcal{S}$. In our work, we made some assumption on this second survey design :

— the second survey design is independent from the sample drawn at the first step.
— the second survey design is based on a Poisson sampling. First order probabilities are called response probabilities and will be denoted $(p_k)_{k\in\mathcal{S}}$. Moreover, these probabilities can depend on auxiliary informations x. We introduce R_k the dummy variable for unit k to belong in $\mathcal{S}_r$. This assumption implies that (R_k) are independent : each unit decides to answer independently.
— We assume that there are no imcompressible non-response $\forall k \in \mathcal{S},\ p_k > 0$.

Rubin [RUB76] proposed a typology of non response mechanisms. Roughly, if response probabilities can be model using variables x that not depend on the variable of interest y then the process is called missing at random (MAR). In that case, unbiased estimators can be computed using a non-response model. Otherwise, if these probabilities depend on both, then the mechanism is called not missing at random (NMAR). This mechanism can lead to severe biais.

In our setting, even if the non response mechanism is MCAR, Narain-Horvitz-Thompson leads to biased estimators. Nevertheless, under the assumption we provided above, double expansion estimator defined in (2) is unbiased for the total.

$$\hat{t}_{y,exp} := \sum_{k\in\mathcal{S}_r} \frac{y_k}{\pi_k p_k} \tag{2}$$

However, (p_k) are not known and can only be estimated using auxiliary informations. Instead we introduce propensity adjusted score estimator (PSA estimator) and Hayek estimator of t_y based on estimated response probabilities :

$$\hat{t}_{y,PSA} := \sum_{k\in\mathcal{S}_r} \frac{y_k}{\pi_k \hat{p}_k} \tag{3}$$

$$\hat{t}_{y,H} := \frac{N}{\hat{N}} \sum_{k\in\mathcal{S}_r} \frac{y_k}{\pi_k \hat{p}_k} \tag{4}$$

where $\hat{p}_k$ is the estimation of p_k based on the auxiliary informations x_k. These estimations can be done using machine learning algorithms and we propose in this work to compare the performance of each estimator for several machine learning algorithms.

2 Machine learning algorithms used

As shown in the last section, one needs to estimate response probabilities in order to compute PSA or Hajek estimator. These probabilities will be estimated using auxiliary informations on the units sampled. In this section, several algorithms will be introduced. After that, we will provide some reminders on the score method that produces more robust estimations. In the end of this part, we will introduce aggregation methods that used response probabilities estimated by different methods and then provide a new estimation of it.

2.1 Machine learning methods

In order to get an estimation of response probabilities, one can use machine learning methods. We will use the conventional notation in this section : y_k will denote the variable we want to predict and x_k the covariates used in the model.

2.1.1 Logistic regression

Logistic regression is a parametric model that explains a binary variable as a function of explanatory variables (discrete or continuous). Logistic regression can be seen as an empirical risk minimization problem with the loss ψ : $\psi(f(x), y) = \log(1 + e^{-yf(x)})$ where f is a linear function of x. For more information, one can refer to chapter 4.4 of [HTF01].

2.1.2 Logistic regression with lasso penalization

Logistic regression with lasso penalization is the same problem than logistic regression except that a l^1- penalization on f is introduced. Since f is a linear function, the penalization holds on the coefficients of f. Lasso regression provides sparse fitted coefficients and more robust results in a high dimensional setup. For more information, one can refer to chapter 18.4 of [HTF01].

Hyperparameters : Amount of penalization λ.

2.1.3 Classification and Regression Trees (CART)

CART is a tree method that consists in recursively partitioning the space of explanatory variables and then proposing a simple predictor (majority vote for classification, mean for regression) on each part of the space.

Splits are obtained in a binary and recursive way. They are computed by optimizing homogeneity criteria. A drawback of CART methods is the instability of the fitted trees.

2.1.4 Random forests

[Bre04] proposed to overcome the instability of CART trees by using aggregation methods called bagging. The idea is to train the CART trees, not only on a single sample but on B samples drawn with replacement in the training sample (this is a bootstrap sample).

Training samples will be drawn independently and all individuals will have the same probability of being drawn. The random forest algorithm adds a second source of randomness between the different trees by allowing to choose the optimal separation only among a limited number of variables drawn uniformly without replacement. In other words, at each training step of each tree, *mtry* variables are drawn uniformly with replacement and the optimal separations are chosen from these sets

2.1.5 k-nearest neighbors (k-nn)

The k-nearest neighbors algorithm consists in using the observations closest to a given point to make a prediction. Once the space of variables x is endowed with a metric, it is possible to determine the k nearest neighbors of a point by computing all the distances between the new point and the points from the training sample. Once the neighbors identified, the prediction will be given by averaging for regression or majority voting for classification the y values observed on the nearest neighbors. For more information, one can refer to chapter 13.3 of [HTF01].

2.1.6 Bayesian additive regression tree (BART)

BART is an additive tree model based on the Bayesian framework proposed by [CGM10]. For simplicity, BART will be introduced in the regression problem.

This method assumes that :

- the variable of interest can be expressed as a sum of trees and Gaussian noise : $y_k = \sum_{j=1}^{B} g_j(x_k, M_j, T_j) + \varepsilon_k$ where $\varepsilon_k \sim \mathcal{N}(0, \sigma^2)$ and B is the number of trees computed.
- there is a prior on trees distribution (defined as a structure T and predicted values per end nodes M) used and on the variance of the Gaussian noise $(T_1, M_1, \ldots, T_B, M_B, \sigma)$.
- the prior on trees distribution and the one on the noise variance are independent.
- the priors (T_i, M_i) are mutually independent.

The prior on the tree structure is described by imposing a probability distribution on the depth of the trees, the variables and the thresholds used for the splitting. The variance is assumed to be distributed according to an inverse Gamma distribution whose hyperparameters are fixed by the user.

The final predictions are obtained by drawing samples from the the a posteriori distribution using MCMC algorithms and then sum it. For more information and a generalization for classification problem, one can refer to [CGM10].

Hyperparameters : Hyperparameters from the prior, burn-in time for the MCMC method, the number of trees.

2.1.7 Extrem Gradient Boosting (XGBoost)

Extrem Gradient Boosting algorithm is an algorithm proposed by Chen [CG16] for classification, regression and ranking based on Gradient Boosting methods. The idea is to aggregate several poorly performing models (weak learners) to build a better model. These weak learners are generally decision trees with a small depth. Unlike bagging (used in the random forest algorithm), weak learners are aggregated in a sequential way : at each step, a new tree is added to improve the predictions where the old predictor has bad results.

XGBoost differs from other boosting methods by using a second-order Taylor approximation of the loss function to improve the computation cost.

Penalization terms for the complexity of the trees (number and values taken by the leaves) are used to reduce overfitting. Other methods allow to limit this over-fitting. For more information, one can refer to [CG16].

Hyperparameters : number of trees, amount of penalization and learning rate.

2.1.8 Support vector machine (SVM)

SVM is a parametric model based on a geometric intuition that explains a binary variable as a function of explanatory variables. Linear SVM can be seen as an empirical risk minimization problem with the loss $\psi_{\text{hinge}} : \psi_{\text{hinge}}(f(x), y) = \max(1 - yf(x))$ where f is a linear function of x. A l^2-norm regularization can be added to prevent overfitting.

In order to get a more flexible model, SVM can be used with a kernel trick. Let k a positive

define kernel and $\mathscr{H}$ the reproducing Hilbert kernel space associated to k. The k-kernel SVM is the solution of the empirical risk minimization $\psi_{\text{hinge}} : \psi_{\text{hinge}}(f(\boldsymbol{x}), \boldsymbol{y}) = \max(1 - \boldsymbol{y}, f(\boldsymbol{x}))$ but f belongs to $\mathscr{H}$. We can add a regularization using the norm associated to $\mathscr{H}$. Thanks to the representer theorem, this problem can be solve considering f belongs to a finite-dimensional subspace of $\mathscr{H}$. For more information about the geometric interpretation, one can refer to chapter 12 of [HTF01].

Hyperparameters : kernel, amount of penalization.

2.1.9 Cubist

Cubist is an algorithm proposed in [Qui92] [Qui93]. It can be seen as a variant of CART algorithm. Indeed, CART algorithm predicts in each terminal node using average of the variable $\boldsymbol{y}$ (for regression problem) of the training sample belonging to this node. This approach corresponds to performing a regression in each terminal node with a constant as the only explanatory variable. According to the author, this can lead to underfiting. He thus proposes to learn in each terminal node a linear model whose explanatory variables are the variables that intervened to construct the node. Cubist provides an aggregation method close to boosting to get better results.

Hyperparameters : number of trees used in the boosting-like method.

2.1.10 MOB

MOB is an algorithm proposed in [ZHH08]. Intuitively, the author starts from the observation that it is unlikely that a parametric model is valid over the whole range of the variable $\boldsymbol{y}$: he thus proposes to learn the model on different parts of the covariates space.

For this algorithm, the covariates are decomposed into two subsets :
- stratification variables : these variables will split the space of explanatory variables.
- truly explanatory variables : these variables will be used to create models in each set of the partition created by the stratification variables.

The idea is to sequentially partition the space of explanatory variables in a recursive and binary way based on the stratification variables A part of the space will be split if the model becomes more stable if it is trained on each of the sub-parts separately. The instability is calculated using a statistic whose asymptotic distribution is known. For more information about MOB, one can refer to [ZHH08].

2.2 Score method

It is possible that the method used to estimate the response probabilities is not appropriate and leads to wrong estimates. Beaumont and Haziza [HB07] have proposed a method to provide new response probabilities using estimation from another method (for instance, by logistic regression). This method ensures more robustness to model misspecification.

The score method consists in :
- using a set of estimated response probabilities $(\hat{p}_k)_{k \in \mathcal{S}}$.
- creating classes using these response probabilities by :
  - splitting into K groups. Each group contains an equal number of probabilities.
  - or using k-means algorithm.
- estimating the response probability by the proportion of respondents per class.

The inefficiency of the PSA and Hájek estimators may be due to highly spread weights : the score method permits to reduce the discrepancy in the probabilities but may lead to bias.

2.3 Aggregation methods

There are no statistical models that are uniformly better than all other models in estimating response probabilities. In practice, the statistician may use different methods to obtain different sets of estimated probabilities. For example, logistic regression, random forests and BART can be used : each individual will thus have three different estimations of response probabilities. For each individual, we note $\mathbf{p}_i = (p_i^1, \ldots, p_i^m)$ with m stands for the number of models used.

Rather than using each of these estimations separately, it would be more interesting to combine them in order to get a new estimation. Ensemble methods provide efficiency gain and these methods are less sensitive to misspecification.

We propose here three methods to combine several probability estimates : one based on calibration, another on a linear regression where estimated coefficients are slightly modified and the last one combines these two approaches.

2.3.1 Calibration

Consider a sample $\mathcal{S}$ drawn according to a survey design P on a population $\mathcal{U}$ whose weighting is given by $(d_k)_{k\in\mathcal{S}}$. Suppose that for each individual in the sample, we have m estimations of the response probabilities based on different methods $\mathbf{p}_i = (p_i^1, \ldots, p_i^m)$. It is possible to consider $\mathbf{p}$ as an auxiliary variable and thus use margin calibration to obtain non response corrected weights. A constraint on the estimated size is also added.

The calibration problem can be expressed as follows :

$$\min_{\{w_k\}} \sum_{k\in S} \frac{G(w_k, d_k)}{q_k} \quad \text{s.c} \quad \sum_{k\in S_r} w_k \mathbf{p}_k = \sum_{k\in S} d_k \mathbf{p}_k \text{ et } \sum_{k\in S_r} w_k = \underbrace{\sum_{k\in S} d_k = N}_{\text{because sampling design of fixed size in our setting.}} \tag{5}$$

The resulting estimation from this method would be $\hat{t}_y^{\text{calibration}} = \sum_{k\in\mathcal{S}_r} w_k y_k$.

2.3.2 COMPRESS

As in subsection 2.3.1, we consider a sample $\mathcal{S}$ drawn according to a design P on a population $\mathcal{U}$. The idea is to find a simple way to summarize all the information from the vector $\mathbf{p}_i = (p_i^1, \ldots, p_i^m)$. The method we propose is to perform the linear regression (without the constant) of the response indicator variable R_i on $\mathbf{p}_i$:

$$R_i = \beta^1 p_i^1 + \ldots + \beta^m p_i^m + \varepsilon_i \tag{6}$$

.

Let $\hat{\boldsymbol{\beta}} = (\hat{\beta}^1, \ldots, \hat{\beta}^m)$ be the ordinary least squares estimation of the regression given in 6. We define $\tilde{\boldsymbol{\beta}} := (\tilde{\beta}^1, \ldots, \tilde{\beta}^m) = \frac{1}{<\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}>}((\hat{\beta}^1)^2, \ldots, (\hat{\beta}^m)^2)$. where $<.,.>$ denotes the canonical inner product of $\mathbb{R}^m$.

Therefore, $\tilde{\boldsymbol{\beta}} \in [0;1]^m$ and $\sum_{i=1}^{m} \tilde{\boldsymbol{\beta}}^i = 1$. These two properties ensure that for any vector of estimated probabilities $\mathbf{p}_i$ then $< \tilde{\boldsymbol{\beta}}, \mathbf{p}_i > \in [0;1]$ (as a convex combination of estimated probabilities).

The aggregate value of the probability vector p_i^{compress} is thus given by $p_i^{\text{compress}} =< \tilde{\boldsymbol{\beta}}, \mathbf{p}_i >$ and the resulting estimation from this method would be $\hat{t}_y^{\text{compress}} = \sum_{k \in \mathcal{S}_r} \frac{y_k}{p_k^{\text{compress}} \pi_k}$.

2.3.3 COMPRESS and calibration

We can combine the two methods presented above to get a third way to mix several set of response probabilities. We will consider the same setup with $\mathbf{p}_i$ denotes the fitted probabilities for unit i using m different algorithms. As in subsection 2.3.2, we can compute a compressed version p_i^{compress} for each unit i. There is no reason for this new set p^{compress} to be already calibrated. At this point, we can use calibration. Unlike in subsection 2.3.1, we will use other calibration constraints :

$$\min_{\{w_k\}} \sum_{k \in S} \frac{G(w_k, d_k)}{q_k} \quad \text{s.c} \quad \sum_{k \in S_r} w_k \log(p_k^{\text{compress}}) = \sum_{k \in S} d_k \log(p_k^{\text{compress}}) \text{ and } \sum_{k \in S_r} w_k = \sum_{k \in S} d_k = N \tag{7}$$

3 Simulation study

This section aims to present how the simulations are done and how the estimators will be compared using these simulations.

3.1 Simulation protocol

In order to estimate the bias and the mean squared error (MSE) of the PSA and Hájek estimators, 1000 iterations will be performed. The estimation will be provided using the classical Monte-Carlo method.

To simulate a population $\mathcal{U}$ consisting of N individuals and a sample $\mathcal{S} \subset \mathcal{U}$ of n individuals, we need to :

1. simulate a stratification variable $X^{(s)}$ for each unit k of $\mathcal{U}$ and create stratum based on the empirical quantiles from the realization of the stratification variable $(x_1^{(s)}, \ldots, x_N^{(s)})$. In our work, we chose $X^{(s)} \sim \gamma(3,2)$.
2. Simulate, for each unit k in the population $\mathcal{U}$, auxiliary variables :
   — continuous variables :
   $$(X^{(c_1)}, X^{(c_2)}, X^{(c_3)}) \sim \mathcal{N}(0,1) \otimes \gamma(3,2) \otimes \gamma(3,2) \tag{8}$$
   — discrete variables :
   $$(X^{(d_1)}, X^{(d_2)}, X^{(d_3)}) \sim \mathcal{M}(\mathbf{p}) \otimes \mathcal{B}(0.5) \otimes \mathcal{U}_{\{1;\ldots;5\}} \tag{9}$$
   where $\mathbf{p} = (p_1, p_2, p_3, p_4, p_5) = (0.5, 0.05, 0.05, 0.1, 0.3)$
3. compute the Neyman's allocation for each stratum using the continuous variable $X^{(c_1)}$ and then draw a sample using a simple random sampling without replacement as survey design based on the allocation.

4. simulate a survey variable Y using a model $Y = f(X^{(c_1)}, \ldots, X^{(c_3)}, X^{(d_1)}, \ldots, X^{(d_3)}, X^{(s)}) + \varepsilon$, où $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$. The variance of the error term is used to ajust the correlation between y and x.
5. simulate response probabilities using all auxiliary variables by the following processes (see figure 1) :
   (a) $p_i^{(1)} = \text{logit}^{-1}(\beta_0 + \beta_1^{(s)} X_1^{(s)} + \beta_2^{(s)} X_2^{(s)} + \beta_3^{(s)} X_3^{(s)} + \beta_1^{(c)} X_1^{(c)} + \beta_2 X_2^{(c)} + \beta_3 X_3^{(c)} + \sum_{k=2}^{5} \beta_{1,k}^{(d)} 1_{\{X_1^{(c)}=k\}} + \beta_2^{(d)} X_2^{(d)} + \sum_{k=2}^{5} \beta_{3,k}^{(d)} 1_{\{X_3^{(d)}=k\}})$.
   where $p_i = \mathbb{E}_{PQ}(R_i \mid S, X)$
   (b) $p_i^{(2)} = 0.1 + 0.9 p_i^{(1)}$.
   (c) $p_i^{(3)} = 0.1 + 0.9 \,\text{logit}^{-1}\left(-1 + \text{sgn}(X_1^c)(X_1^c)^2 + 3 \times 1_{\left\{X_1^{(d)}<4\right\} \cap \left\{X_2^{(d)}=1\right\}}\right)$.
   (d) $p_i^{(4)} = 0.55 + 0.45 \tanh(0.05 y_i - 0.5)$.
   (e) $p_i^{(5)} = 0.1 + 0.9 \,\text{logit}^{-1}(0.2 y_i - 1.2)$.

   In the following, we will denote these processes by non-response mechanisms.
6. the sample of respondents is generated using these probabilities (for each individual, a draw is made according to a Bernouilli distribution whose parameter is the probability of non-response). Thus, we obtain as many sets of response indicators $R_i^{(k)}$ as there are non response mechanisms.
7. for each non response mechanism and for each algorithm, the response probabilities are estimated and then the PSA and Hájek estimators are computed.
   Hence we obtain as many simulations as there are couples of non-response mechanism and machine learning algorithm (for the PSA and Hájek estimator).

Remarks :

— since the allocation is performed according to the variable $X^{(c_2)}$ independent of y, this variable does not impact the construction of y. As a result, very low empirical Pearson correlation coefficients between y and $\frac{1}{\pi}$ are observed. This case corresponds to an non-informative survey design : the survey design does not contain any information on the variable y.

— auxiliary variables are drawn independently. This configuration can seem unrealistic. We propose a scenario in which the variables from the step (2) have the same marginal law but different joined distribution using Gaussian copula.

— $(p_i^{(4)})$ and $(p_i^{(5)})$ are NMAR mechanisms : we can use these mechanisms to test the limit of each estimator.

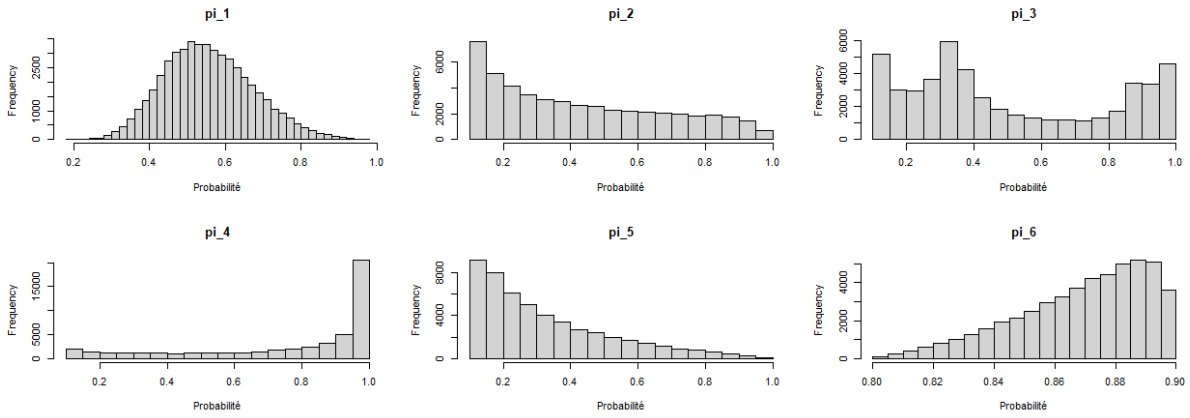

FIGURE 1 – Distribution of response probabilities on the population $\mathscr{U}$

3.2 Monte-Carlo estimation of bias and relative mean squared error

For the purpose of quantifying the performance of each method, we will use two statistics : the relative Monte-Carlo bias and the relative Monte-Carlo mean squared error.

Assume that B simulations have been done. For each couple machine learning algorithm / non response (for instance, (BART, $p_i^{(4)}$) mechanism, there are B estimations from the PSA estimator and from the Hájek estimator.

Let us denote by $\hat{t}_{y,k}$ the total of y on $\mathscr{U}$ estimated (by using the PSA or Hájek estimator) at the k^{th} simulation and $t_{y,k}$ the real total of y on $\mathscr{U}$ (this quantity is known because we simulate y before drawing the sample).

The relative Monte-Carlo bias is defined by :

$$\mathbb{B}_{MC}(\hat{t}_y) = \frac{100}{B}\sum_{k=1}^{B}\frac{\hat{t}_{y,k} - t_{y,k}}{t_{y,k}} \tag{10}$$

The mean squared error is defined by :

$$\text{MSE}_{MC}(\hat{t}_y) = \frac{1}{B}\sum_{k=1}^{B}\left(\hat{t}_{y,k} - t_{y,k}\right)^2 \tag{11}$$

However the mean squared error may be hard to interpret. Instead, we will introduce the relative Monte-Carlo efficiency of a estimator $\hat{t}_y$ (PSA or Hájek) as

$$\text{Eff}_{MC}(\hat{t}_y) = 100\frac{\text{MSE}_{MC}(\hat{t}_y)}{\text{MSE}_{MC}(\hat{t}_{y,\pi})} \tag{12}$$

where $\text{MSE}_{MC}(\hat{t}_{y,\pi})$ is the mean squared error of the Narain-Horvitz-Thompson estimator. $\hat{t}_{y,\pi} = \sum_{k\in S}\frac{y_k}{\pi_k}$. This statistic allows us to know if an estimator is more efficient than the Narain-Horvitz-Thompsion estimation obtained if the true response probabilities were known.

3.3 Alternative simulation scenario

So far, some assumptions was made in order to make the simulations : auxiliary variables are generated independently (as shown in equation 8 and 9), the design survey is non-informative and the survey function is a linear function of the auxiliary variables.

We decided to introduce several alternative scenarios, in which one or more assumptions change, in order to compare each estimator in more realistic setup. These scenarios are obtained using :

- a more important weight on the variable used for the allocation (which is here $X^{(c_1)}$ in the function that links the survey variable Y and the auxiliary variables X. For instance, if the function between Y and X is linear then we will increase (in absolute value) of the coefficient related to $X^{(c_1)}$. We control how the plan is informative by computing the empirical correlation between the survey variable $\{y_i\}_{i\in\mathscr{U}}$ and the weight of the survey sampling without non-response. $\{\frac{1}{\pi_i}\}_{i\in\mathscr{U}}$
- another function for Y. For instance, we can use functions that involves interactions between auxiliary variables.
- gaussian copulas that keep the same marginal distribution but introduce correlation between variables. We used Gaussian copulas because we can easily express the correlation between variables in term of the parameters of these copulas. We control how variables are correlated using the empirical correlation matrix.

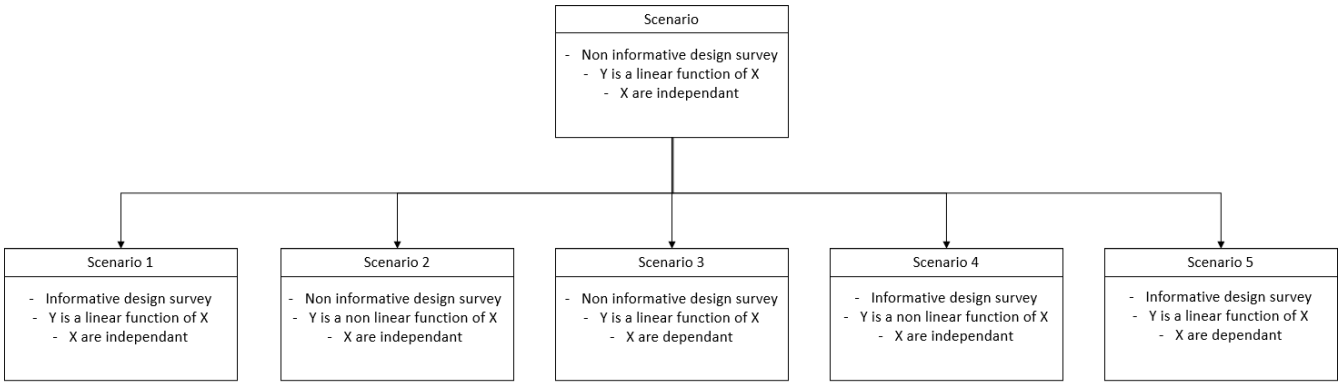

FIGURE 2 – Alternative simulation scenario

3.4 Results

3.4.1 Comparison of machine learning methods

In this section, the efficiency of the PSA estimator and the Hájek estimator will be studied. For each algorithm of machine learning, an estimate of the efficiency of the estimators is available for 42 configurations (6 non-response mechanisms and 7 general scenarios).

These tables shows us different results :

- algorithms such as cb3, svm2 and mob that provide the worst result in the worst case (Max column) also provide worst results in other cases. It seems that using Cubist with extrapolation and bias estimations provides very poor results.
- results obtained using svm2 are extremely bad : perhaps the polynomial kernel is too restrictive.

Algorithm	Min	Q1	Med	Q3	Max	Mean
knn	16	25	13	6	15	16
knn_reg	10	15	15	9	18	19
bart	3	2	12	3	12	5
bart_reg	9	3	24	26	32	32
cart1	17	14	9	16	6	7
cart2	19	12	6	17	7	8
cart3	18	13	5	19	9	9
cart4	11	16	11	14	3	3
CART_reg	15	20	8	13	8	4
cb1	23	22	23	22	21	21
cb2	20	11	25	28	26	27
cb3	32	32	31	32	32	32
cb4	24	17	17	20	23	20
cb5	25	19	18	25	24	24
logit	2	5	28	29	32	28
logit_lasso	7	27	26	21	22	26
mob	1	30	30	30	32	32
rf1	30	29	29	24	13	23
rf2	26	23	20	18	10	13
rf3	22	18	22	15	4	14
rf4	21	9	16	11	5	12
rf5	27	21	21	27	27	25
rf_reg	29	28	27	23	25	22
svm1	4	24	14	1	19	17
svm2	31	31	32	32	32	32
xgb1	14	8	1	10	20	18
xgb2	13	6	10	8	14	11
xgb3	8	10	4	7	16	10
xgb4	12	7	3	12	17	15
calibration	28	26	19	4	1	2
COMPRESS	5	1	7	5	11	6
COMPRESS + calibration	6	4	2	2	2	1

Algorithm	Min	Q1	Med	Q3	Max	Mean
knn	29	26	18	25	18	23
knn_reg	27	24	21	26	21	25
bart	15	5	4	9	15	13
bart_reg	21	12	29	30	32	32
cart1	22	23	14	21	6	14
cart2	24	21	15	22	8	15
cart3	23	22	16	23	11	16
cart4	5	20	11	8	3	3
CART_reg	16	25	8	18	9	8
cb1	25	19	12	5	25	21
cb2	3	10	13	4	19	7
cb3	31	32	31	31	30	30
cb4	18	17	9	6	22	19
cb5	17	18	10	7	24	20
logit	2	13	27	24	29	27
logit_lasso	28	28	28	28	27	28
mob	1	31	32	32	32	32
rf1	8	15	26	17	4	9
rf2	11	6	20	13	7	6
rf3	14	3	22	15	10	10
rf4	12	1	17	16	12	11
rf5	10	7	23	3	13	4
rf_reg	9	11	25	10	5	5
svm1	26	27	24	27	23	26
svm2	32	30	30	29	28	29
xgb1	20	16	1	20	26	24
xgb2	7	8	6	14	17	18
xgb3	6	4	5	12	16	17
xgb4	19	14	3	19	20	22
calibration	30	29	19	2	1	2
COMPRESS	13	2	2	11	14	12
COMPRESS + calibration	4	9	7	1	2	1

TABLE 1 – Rank efficiency of PSA estimator (left) and Hájek estimator (right).
The case **(cb3, Min) = 32** means that cb3 has the 32^{th} better minimum efficiency. The minimum efficiency for an algorithm is the minimum MSE we get among all the simulations done with this algorithm. We define in the same way the first quartile efficiency (Q1), the median efficiency (Med), the third quartile efficiency (Q3), the max efficiency (Max) and the mean efficiency (Mean).

- unlike Bart for classification, Bart for regression provide poor results in the worst cases.
- using PSA estimators :
  - algorithms based on XGboost provide best results if we consider the median case for each algorithm.
  - aggregation methods (calibration, COMPRESS and COMPRESS + calibration) presented in section 2.3 perform well in the worst case. We can notice that Bart gives good results in almost every configuration.
- using Hájek estimators :
  - calibration based estimator perform better in the worst cases.
  - calibration has bad results on the more friendly case.

3.4.2 Comparison between raw estimators and score methods

The score method allows to be more robust against misspecification problems. It can allow efficiency gains. But is this the case for all learning methods used for modeling response probabilities ? In this section, we are interested in the gain of the 10-class score method after using a learning method to estimate the response probabilities. The gain for keeping the raw version of

Algorithm	Min	Q1	Med	Q3	Max	Mean
knn	172	282	392	921	11 513	1 621
knn_reg	144	261	413	1 020	12 398	1 745
bart	129	199	384	852	10 595	1 314
bart_reg	143	208	571	2 479	Inf	Inf
cart1	172	259	351	1 448	9 373	1 370
cart2	175	256	348	1 464	9 472	1 376
cart3	175	259	345	1 506	9 627	1 393
cart4	145	262	369	1 382	8 881	1 231
CART_reg	162	269	350	1 367	9 522	1 293
cb1	194	270	524	1 814	14 125	2 002
cb2	181	241	598	3 239	23 578	3 385
cb3	304	53 745	890 538	Inf	Inf	Inf
cb4	197	263	456	1 592	16 376	1 948
cb5	199	267	466	2 406	17 395	2 249
logit	123	215	962	5 786	Inf	84 503
logit_lasso	141	331	636	1 739	15 895	2 520
mob	121	833	10 846	106 423	Inf	Inf
rf1	228	345	1 147	2 152	10 973	2 208
rf2	199	278	487	1 470	9 717	1 482
rf3	192	264	522	1 419	9 215	1 488
rf4	188	235	417	1 133	9 341	1 413
rf5	200	269	508	2 847	25 181	2 408
rf_reg	225	343	821	1 989	19 596	2 203
svm1	129	280	407	780	12 482	1 639
svm2	297	32 212	Inf	Inf	Inf	Inf
xgb1	155	225	324	1 124	12 551	1 677
xgb2	148	215	368	1 016	11 479	1 405
xgb3	143	239	344	928	11 581	1 394
xgb4	148	221	330	1 139	12 111	1 589
calibration	222	318	472	875	7 475	1 031
COMPRESS	137	199	348	906	10 382	1 317
COMPRESS_CAL	139	208	328	798	7 772	908

Algorithm	Min	Q1	Med	Q3	Max	Mean
knn	198	253	449	2 219	10 875	1 826
knn_reg	187	251	485	2 352	11 932	1 998
bart	159	202	306	1 417	10 201	1 457
bart_reg	173	217	1 401	5 545	Inf	Inf
cart1	173	248	421	1 807	9 369	1 485
cart2	174	240	422	1 807	9 472	1 487
cart3	174	243	430	1 844	9 627	1 510
cart4	145	229	362	1 413	8 879	1 255
CART_reg	163	252	344	1 733	9 515	1 382
cb1	182	228	363	1 365	12 281	1 680
cb2	138	211	419	1 291	10 922	1 367
cb3	223	1 605	3 246	8 241	60 590	7 404
cb4	165	224	345	1 389	12 223	1 675
cb5	163	224	346	1 398	12 255	1 680
logit	123	218	671	2 202	27 493	2 770
logit_lasso	193	305	679	2 759	15 670	2 833
mob	122	976	8 259	374 131	Inf	Inf
rf1	150	219	572	1 598	9 149	1 382
rf2	156	202	477	1 512	9 397	1 348
rf3	159	198	489	1 529	9 607	1 401
rf4	156	195	437	1 555	9 721	1 406
rf5	153	202	493	1 275	9 890	1 327
rf_reg	151	212	547	1 458	9 159	1 345
svm1	187	279	516	2 691	12 231	2 069
svm2	237	452	1 491	3 723	23 959	3 966
xgb1	171	220	295	1 751	12 305	1 864
xgb2	148	206	315	1 520	10 817	1 567
xgb3	147	201	307	1 508	10 815	1 560
xgb4	170	219	296	1 741	11 783	1 778
calibration	222	318	472	875	7 475	1 031
COMPRESS	158	196	296	1 470	10 144	1 443
COMPRESS_CAL	139	208	328	798	7 772	908

TABLE 2 – Relative Monte-Carlo efficiency for PSA estimator (left) and Hájek estimator (right).
The case **(xgb4,Med) = 397** means that the median of all the relative Monte-Carlo efficiencies (as defined in equation 12) using xgb4 as algorithm.

an estimator $\hat{t}_y$ (PSA or Hájek for instance) is quantified by $G = \frac{\text{Eff}_{MC}\hat{t}_{y,\text{score}}}{\text{Eff}_{MC}\hat{t}_y}$. Figure 3 allows us to observe whether the gain is greater than 1 (so the 10-class score method deteriorates - in blue on the graph) for each pair of method and non response mechanism.

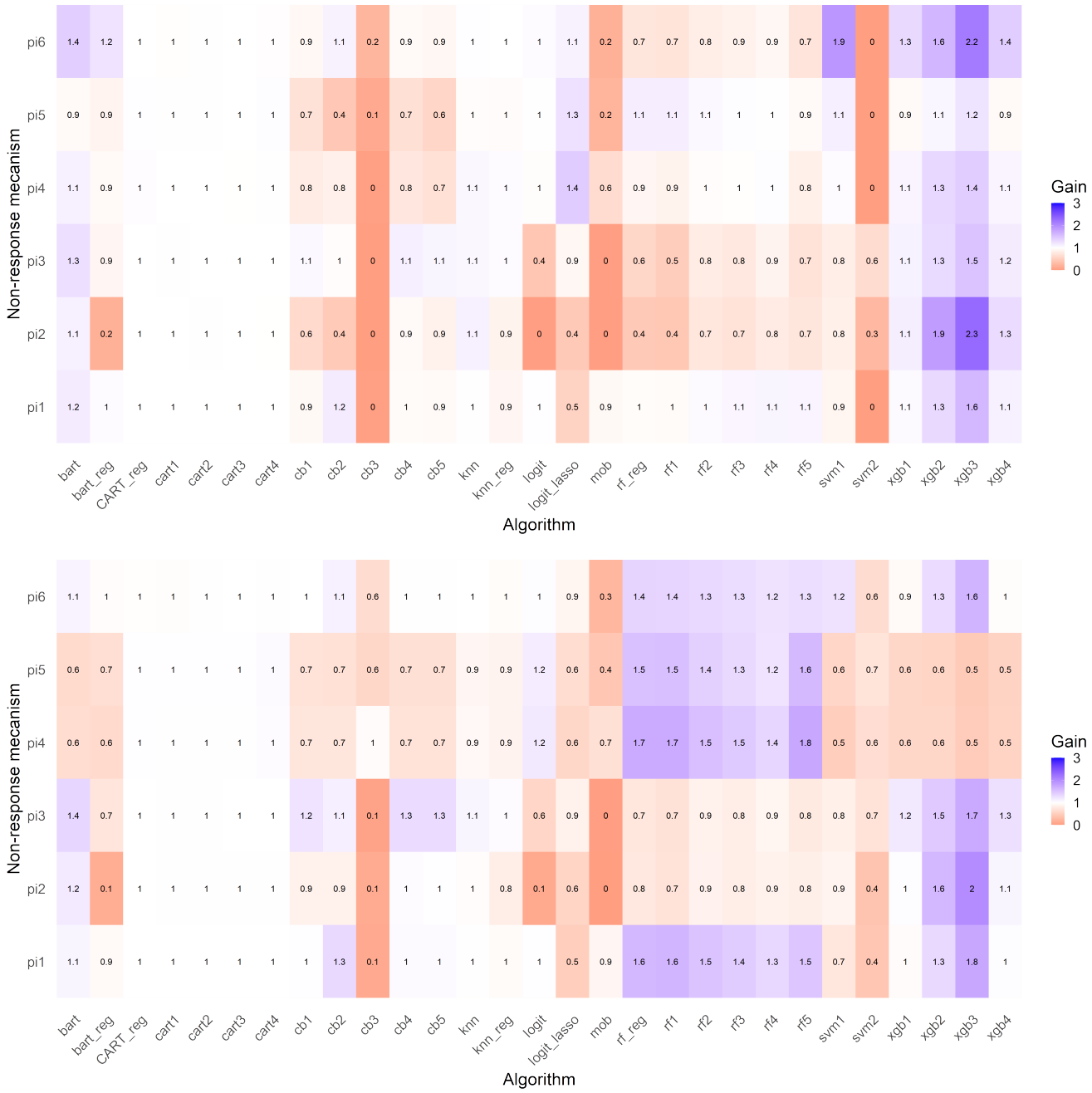

FIGURE 3 – Gain (in term of efficiency) between raw estimators and score method estimators.

Several results emerge from this graph :

— the results with and without the score method provide the same results when the response probabilities are estimated by the CART method. It is possible that there is no efficiency gain because the probabilities estimated by the CART methods belong to a finite support with a small cardinal and therefore the score method does not allow to decrease significantly the size of the support of the estimated probabilities.

— similar results are found for the k-nearest neighbors. Indeed, the cardinal of the support of the estimated probabilities is at most k + 1.

— the method of scores damage the efficiency of Hájek estimators for random forests mostly.

— the method of scores decreases the efficiency of PSA estimators when the probabilities are estimated with the XGBoost algorithm.

— overall, the score method increases the efficiency of the estimators when the estimates of the response probabilities are very bad such as the SVM method with linear kernel or the logistic regression when the true response probabilities are between 0.1 and 1.

3.4.3 Comparaison between PSA and Hájek estimator

We decided to compute for each simulation the estimates based on PSA and Hayek estimators. Are there methods for which one of the estimators is better than the other ?

In order to answer this question empirically, simulations were performed. The following figure 4 allows us to compute the ratio between the mean square error of Hayek's estimator and that of the PSA estimator. When this indicator is smaller than 1 (red on the figure) then the Hayek estimator is better than the PSA estimator. Symmetrically, if this indicator is greater than 1 (blue on the figure) then the PSA estimator is better.

First, we observe that the performances are the same for both estimators as soon as we use a method based on calibration (calibration or COMPRESS and calibration) : this is due to the fact that these estimators are calibrated on several auxiliary variables including the total size of the population and thus the two estimators are similar.

The PSA estimator seems to provide better results when the probabilities are estimated using the BART or MOB methods. The Hayek estimator appears to produce better estimators when the methods produce very poor estimates of the response probabilities (cb3 or svm2). With several nonresponse mechanisms, the performance of both estimators seems to be the same when the probabilities are estimated by the CART method.

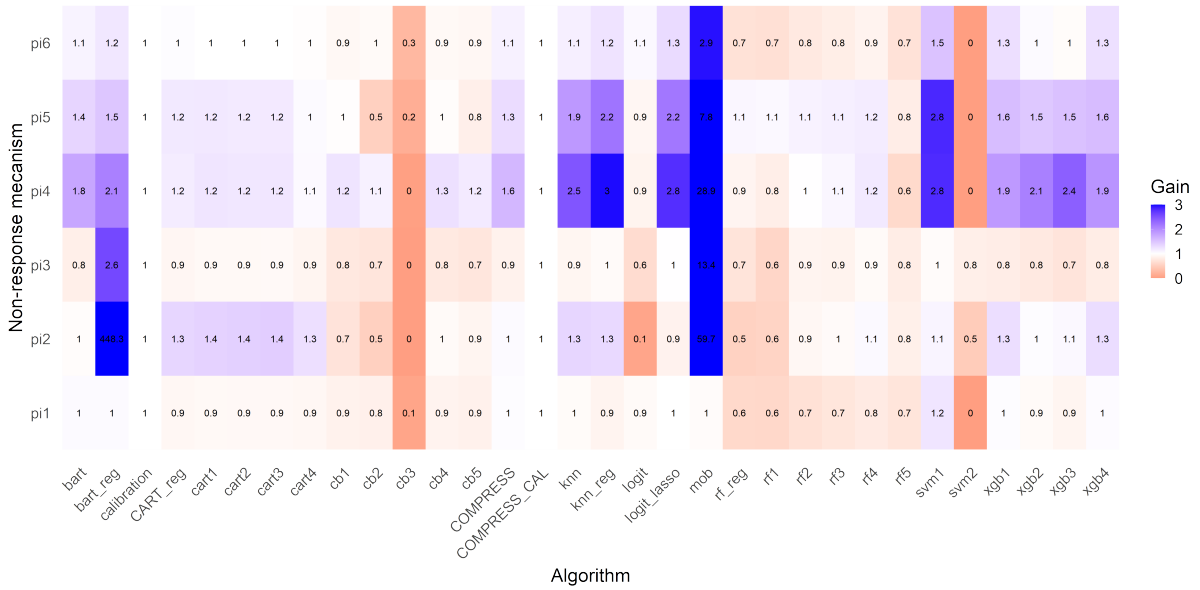

FIGURE 4 – Gain (in term of efficiency) between Hayek and PSA estimators.

3.4.4 COMPRESS and calibration methods : the more we add probability estimations the better is the resulting estimator ?

As mentioned in section 2.3, it is possible to aggregate m sets of estimated probabilities to create new ones (using the COMPRESS method, based on calibration or COMPRESS +

calibration). Empirically, we have found that these estimators provide more efficient estimators when considering the least favorable situations. However, do these estimators keep these good properties when the number of estimated probability sets m increases ?

In order to verify this hypothesis empirically, simulations were performed. We kept the same setup that was described in section 3.1. From 6 learning methods (logistic regression with the 10-class score method, rf4, bart, cb1, xgb3, svm1), the probabilities of answers will be estimated. Then each aggregation method will be used based on all the combination of one method ($m = 1$), then two ($m = 2$ - there are $\binom{6}{2}$ possibilities) and up to 6 methods. This gives 63 different estimates for each aggregation method. These estimations will be repeated 1000 times in order to have an estimate of the biases and relative efficiencies.

In the figure 5, we have displayed, for each aggregated method, the relative efficiency of the PSA estimator (We provided same results on the bias in the annex (figure 7, 6 and 8) as a function of the number of methods m used. This figure consists of three curves :

- *Min* curve describes, for a number of sets of probabilities used m, the minimum of the relative efficiencies obtained among the possible $\binom{6}{m}$ relative efficiencies.
- *Med* curve describes, for a number of sets of probabilities used m, the median of the relative efficiencies obtained among the possible $\binom{6}{m}$ relative efficiencies.
- *Max* curve describes, for a number of sets of probabilities used m, the maximum of the relative efficiencies obtained among the possible $\binom{6}{m}$ relative efficiencies.

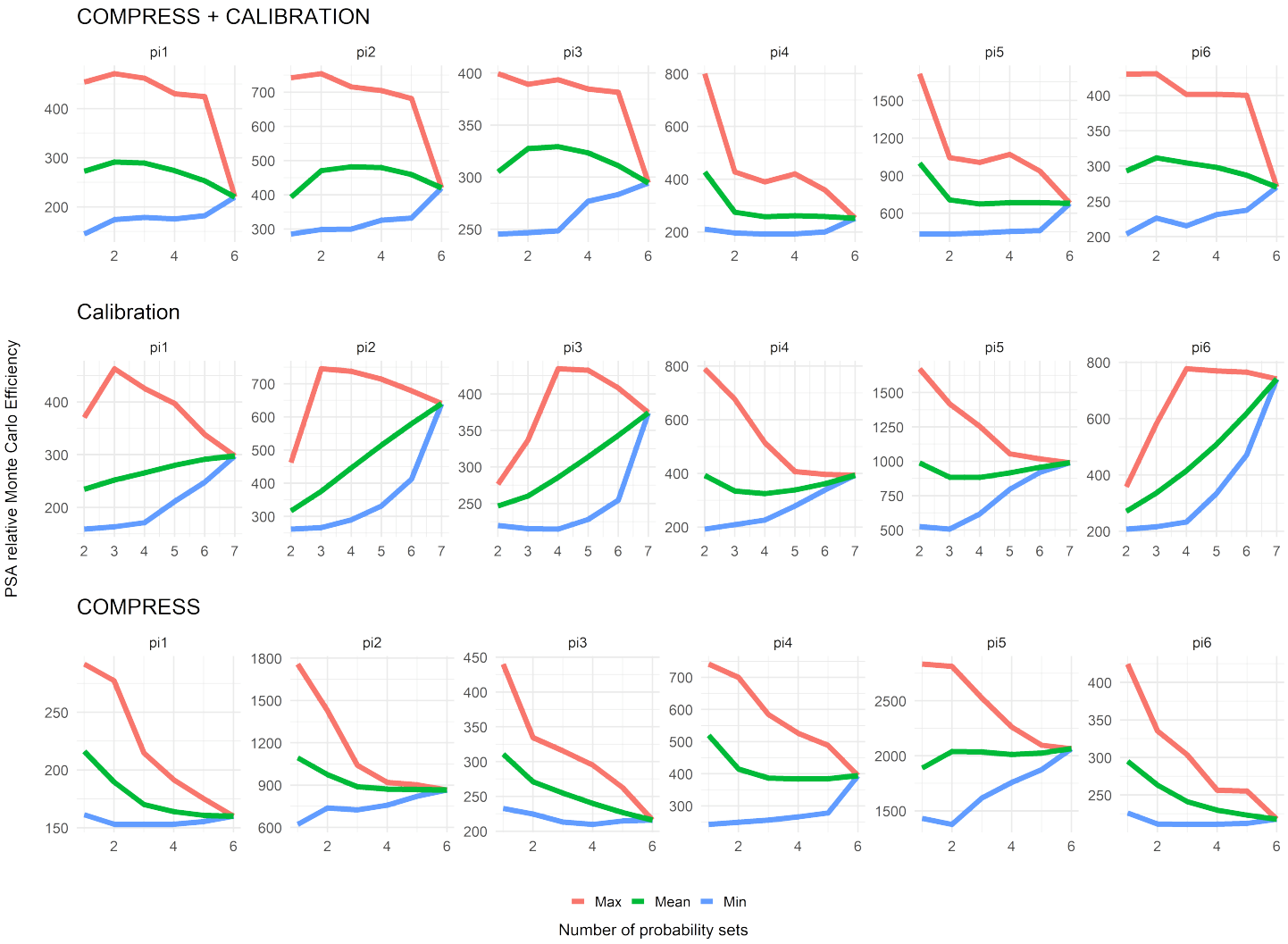

FIGURE 5 – Efficiency of aggregation methods and number of methods aggregated

When the number of probability sets increases with the COMPRESS estimator, the median of the mean square errors for a number of probability sets m decreases with m except in the case pi5 which corresponds to a NMAR process. The maximum of the mean square errors also decreases. Nevertheless, the minimum of the mean square errors seems to increase but with a smaller amplitude than the decrease of the maximum of the mean square errors : the increase of the number of sets used seems to improve more significantly the performances in the less favorable situations than it deteriorates it in the most favorable cases.

However, different results are observed for the calibration-based estimator. The minimum and the median of the squared errors seem to increase with the number of probability sets used. This result could be due to the instability of the calibration-based estimator when the number of variables used increases ([SS97]). Moreover, contrary to the estimator based on the COMPRESS methods, the maximum of the mean square errors only decreases after a certain rank.

Finally, when the COMPRESS and calibration methods are combined, the median of the squared errors seems to increase at first before decreasing, contrary to the estimator based only on the calibration. Moreover, the maximum of the mean squared errors decreases with the number of

probability sets used m.

Bibliography

[BFOS83] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and C. J. Stone. Classification and regression trees. 1983.

[BGC18] David Haziza Brigitte Gelein and David Causeur. Pondération pour correction de la non-réponse totale et machine learning. *Acte des JMS 2018*, 1, 2018.

[BGV92] Bernhard E. Boser, Isabelle Guyon, and Vladimir Naumovich Vapnik. A training algorithm for optimal margin classifiers. In *COLT '92*, 1992.

[Bre04] Leo Breiman. Random forests. *Machine Learning*, 45 :5–32, 2004.

[CG16] Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[CGM10] Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. Bart : Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1), Mar 2010.

[DGH20] Mehdi Dagdoug, Camelia Goga, and David Haziza. Imputation procedures in surveys using nonparametric and machine learning methods : an empirical comparison, 2020.

[DS92] Jean-Claude Deville and Carl-Erik Sarndal. Calibration estimators in survey sampling. *Journal of the American Statistical Association*, 87(418) :376–382, 1992.

[HB07] David Haziza and Jean-Francois Beaumont. On the construction of imputation classes in surveys. *International Statistical Review*, 75 :25–43, 2007.

[HT52] D.G. Horvitz and DJ Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260) :663–685, 1952.

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[LD04] Jared K Lunceford and Marie Davidian. Stratification and weighting via the propensity score in estimation of causal treatment effects : a comparative study. *Statistics in medicine*, 23 19 :2937–60, 2004.

[Nar51] R. D. Narain. On sampling without replacement with varying probabilities. *Journal of the Indian Society of Agricultural Statistics*, 3 :169–175, 1951.

[Pla99] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.

[Qui92] J. Ross Quinlan. Learning with continuous classes. 1992.

[Qui93] J. Ross Quinlan. Combining instance-based and model-based learning. In *ICML*, 1993.

[RBK05] Susana Rubin-Bleuer and Ioana Schiopu Kratina. On the two-phase framework for joint model and design-based inference. *The Annals of Statistics*, 33(6) :2789–2810, 2005.

[RUB76] DONALD B. RUBIN. Inference and missing data. *Biometrika*, 63(3) :581–592, 12 1976.

[SS97] P.L.D. Nascimento Silva and C.J. Skinner. Variable selection for regression estimation in finite populations. *Survey Methodology*, 23(1) :23–32, 1997.

[ZH07] Achim Zeileis and Kurt Hornik. Generalized m-fluctuation tests for parameter instability. *Statistica Neerlandica*, 61 :488–508, 2007.

[ZHH08] Achim Zeileis, Torsten Hothorn, and Kurt Hornik. Model-based recursive partitioning. *Journal of Computational and Graphical Statistics*, 17 :492 – 514, 2008.

4 Annexes

4.1 Algorithm used

Label	Algorithm	Continuous or discrete variable	Hyperparameters
bart	BART	Discrete	
bart_reg	BART	Continue	
CART_reg	CART	Continuous	Min number of observations in each leaf : 20
cart1	CART pruned	Discrete	Min number of observations in each leaf : 10
cart2	CART pruned	Discrete	Min number of observations in each leaf : 20
cart3	CART pruned	Discrete	Min number of observations in each leaf : 30
cart4	CART non pruned	Discrete	Min number of observations in each leaf : 20
cb1	Cubist	Continuous	Not biased 10 agregated models With extrapolation
cb2	Cubist	Continuous	Biased 10 agregated models Without extrapolation
cb3	Cubist	Continuous	Biased 10 agregated models With extrapolation
knn	k-nearest neighbours	Discrete	
knn_reg	k-nearest neighbours	Discrete	
logit	Logistic regression	Discrete	
logit_lasso	Lasso logistic regression	Discreate	Lambda : obtained using 10-fold cross-validation
mob	Model-Based Recursive Partitioning	Discrete	Variable used for the stratification : $X^{(x)}$
rf_reg	Random forests	Continue	Min number of observations in each leaf : 20 Nombre d'arbres aggregés : 200
rf1	Random forests	Discrete (Probabilities estimation trees)	Min number of observations in each leaf : 10 Number of trees : 100
rf2	Random forests	Discrete (Probabilities estimation trees)	Min number of observations in each leaf : 10 Number of trees : 500
rf3	Random forests	Discrete (Probabilities estimation trees)	Min number of observations in each leaf : 30 Number of trees : 100
rf4	Random forests	Discrete (Probabilities estimation trees)	Min number of observations in each leaf : 30 Number of trees : 500
rf5	Random forests	Discrete (Probabilities estimation trees)	Min number of observations in each leaf : 30 Number of trees : 5E00
svm1	SVM with RBF kernel Platt method to get probabilities	Discrete	Gamma : 0.025 nu : 0.7
svm2	SVM with polynomial kernel	Discreate	Gamma : 0.0001 nu = 0.7 Degré = 1
xgb1	XGBoost	Continue	Number of trees : 500 Gamma : 10 Proportion for subset : 75% Learning rate : 0.05 Max deepth : 2
xgb2	XGBoost	Continue	Number of trees : 2000 Gamma : 2 Proportion for subset : 100% Learning rate : 0.5 Max deepth : 2
xgb3	XGBoost	Continue	Number of trees : 1000 Gamma : 1 Proportion for subset : 75% Learning rate : 0.01 Max deepth : 1
xgb4	XGBoost	Continue	Number of trees : 500 Gamma : 10 Proportion for subset : 75% Learning rate : 0.05 Max deepth : 3

TABLE 3 – Labels and hyperparameters for each algorithm

4.2 Efficiency of aggregation methods and number of methods aggregated

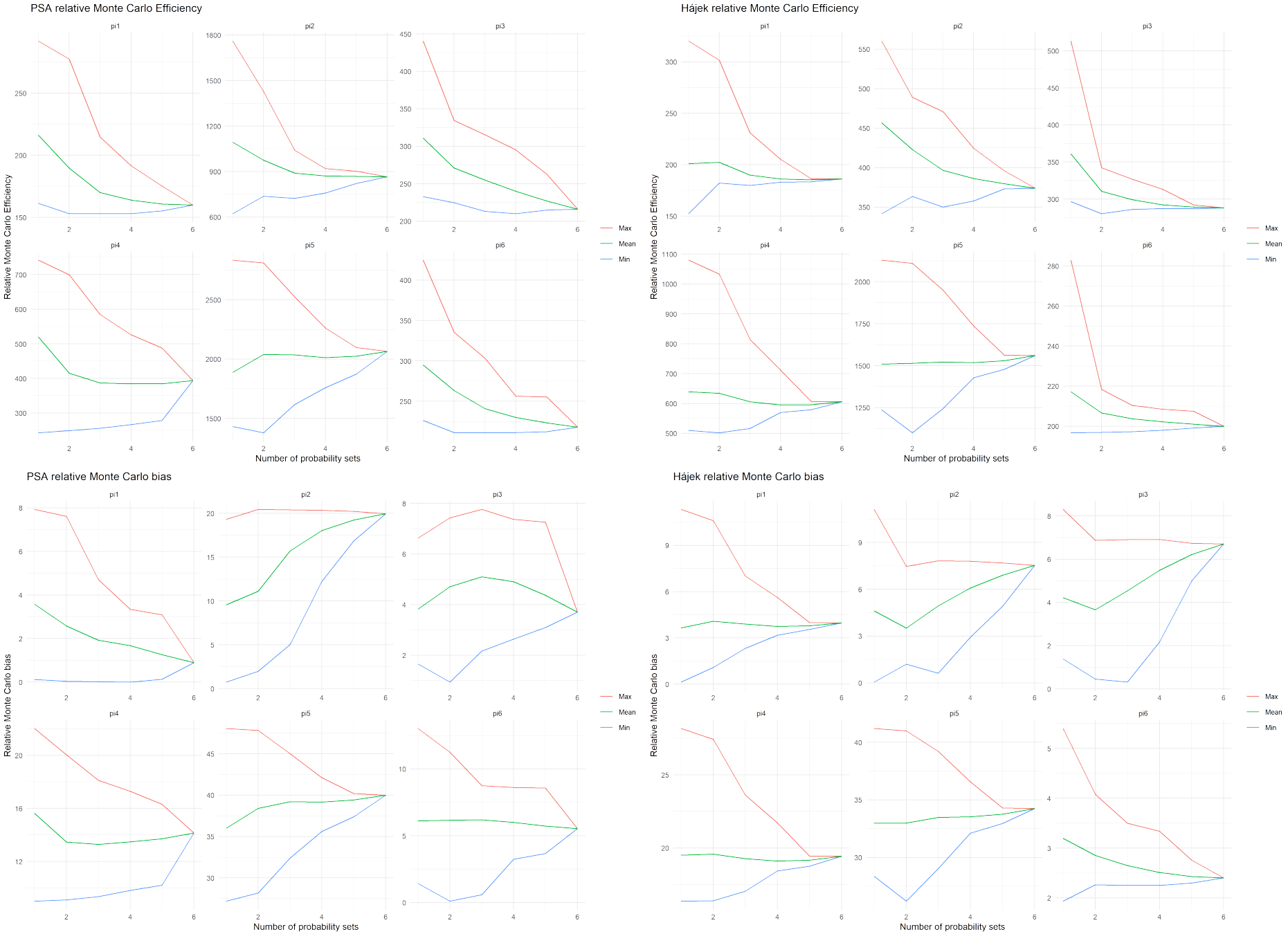

FIGURE 6 – Efficiency of COMPRESS methods

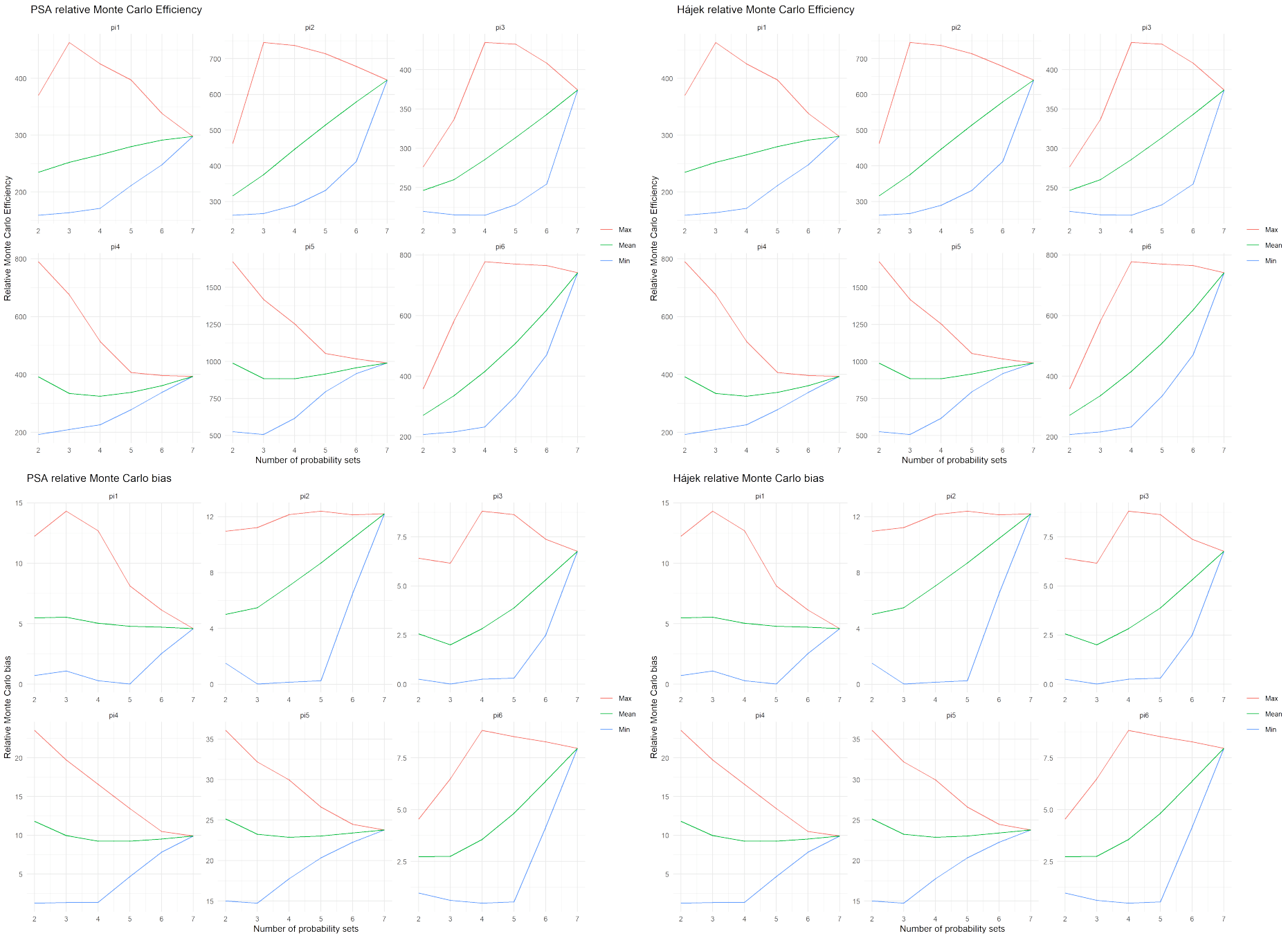

FIGURE 7 – Efficiency of calibration methods

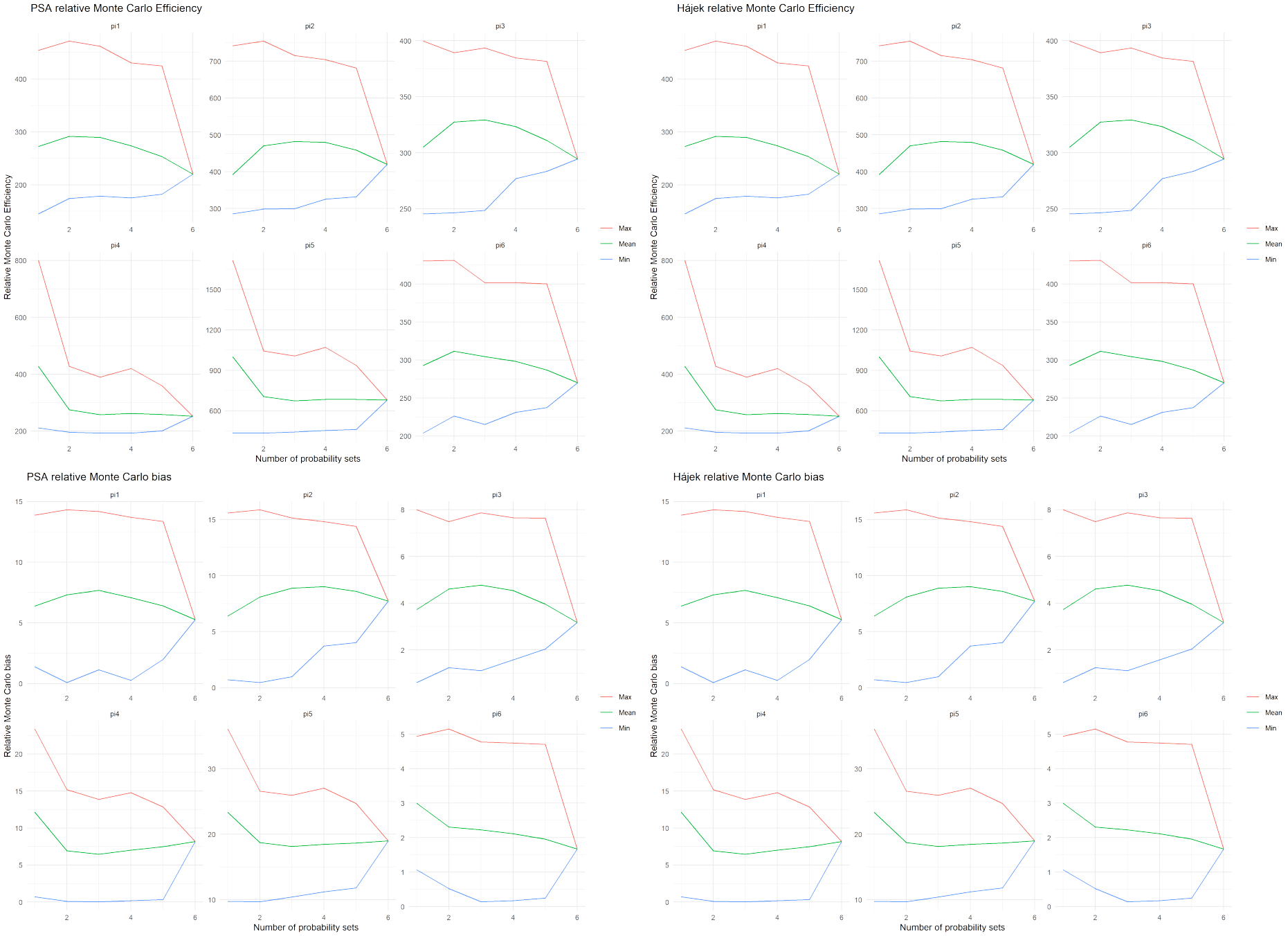

FIGURE 8 – Efficiency of COMPRESS + calibration methods