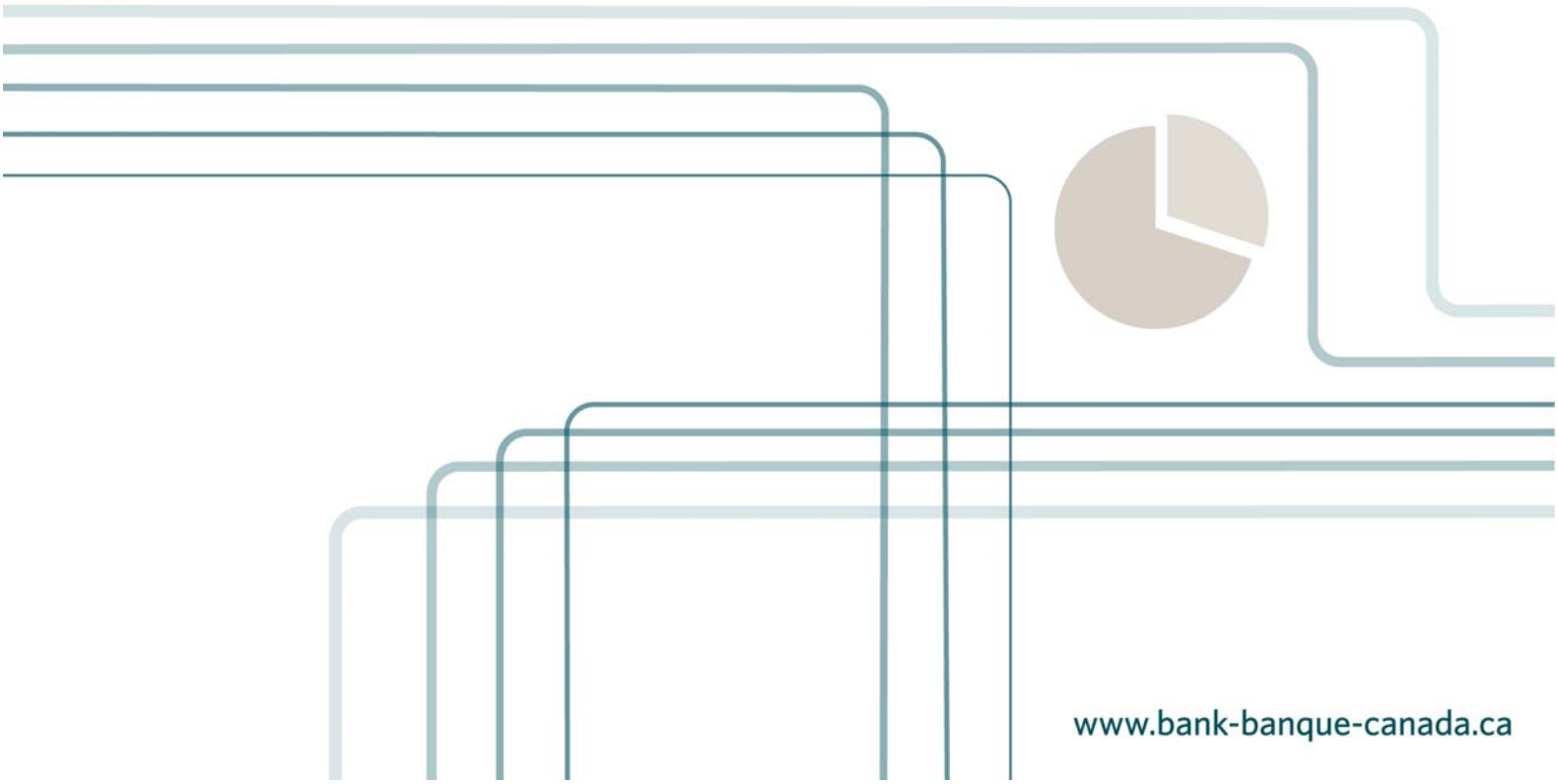
BANK OF CANADA
BANQUE DU CANADA

Technical Report No. 107 / Rapport technique nº 107

The Bank of Canada 2015 Retailer Survey on the Cost of Payment Methods: Nonresponse

by Stan Hatko



www.bank-banque-canada.ca

March 2017

The Bank of Canada 2015 Retailer Survey on the Cost of Payment Methods: Nonresponse

Stan Hatko

Currency Department
Bank of Canada
Ottawa, Ontario, Canada K1A 0G9
shatko@bankofcanada.ca

ISSN 1919-689X

Acknowledgements

I would like to thank Jean-François Beaumont and Valéry D. Jiongo for their independent reviews of this technical report. I would further like to thank Heng Chen, Maren Hansen, David Haziza, Kim P. Huynh, Anneke Kosse, Gradon Nicholls, Rallye Shen, Kyle Vincent and Angelika Welte for providing assistance and helpful comments and suggestions for this work.

Abstract

Nonresponse is a considerable challenge in the Retailer Survey on the Cost of Payment Methods conducted by the Bank of Canada in 2015. There are two types of nonresponse in this survey: unit nonresponse, in which a business does not reply to the entire survey, and item nonresponse, in which a business does not respond to particular questions within the survey. Both types may create a bias when computing statistics such as means and weighted totals for different variables. This technical report analyzes solutions to fix the problem of nonresponse in the survey data. Unit nonresponse is addressed through response probability adjustment, in which response probabilities are modelled using logistic regression (a clustering approach for the unit response probabilities is also considered) and are used in the construction of a set of survey weights. Item nonresponse is addressed through imputation, in which the gradient boosting machine (GBM) and extreme gradient boosting (XGBoost) algorithms are used to predict missing values for variables of interest.

Bank topic: Central bank research
JEL codes: C81; C83

Résumé

Dans l'enquête que la Banque du Canada a menée en 2015 sur les coûts des différents modes de paiement pour les détaillants, la non-réponse est un défi majeur. Elle se présente sous deux formes dans l'enquête en question : la non-réponse totale (une entreprise ne participe pas du tout à l'enquête) et la non-réponse partielle (une entreprise ne répond pas à certaines questions de l'enquête). Ces deux formes peuvent causer un biais dans les calculs statistiques tels que les moyennes et les totaux pondérés de différentes variables. Dans ce rapport technique, nous analysons les solutions qui permettraient de régler le problème des non-réponses dans les données d'enquête. Pour résoudre celui de la non-réponse totale, nous proposons d'ajuster les probabilités de réponse : ces dernières sont modélisées par régression logistique (le recours à une approche par grappes est également envisagé) et servent à la construction d'une série de pondérations d'enquête. Quant au problème de la non-réponse partielle, nous proposons de le résoudre par imputation. À cet effet, nous utilisons des algorithmes d'apprentissage automatique tels que « gradient boosting machine » et « extreme gradient boosting » pour prédire les valeurs manquantes des variables d'intérêt.

Sujet : Recherches menées par des banques centrales
Codes JEL : C81, C83

1 Introduction and Overview

The objective of the Bank of Canada's 2015 Study on the Cost of Payment Methods is to determine the social costs of various methods of payment. This study consists of multiple components, one of which is the Retailer Survey on the Cost of Payment Methods (RCPM survey), the focus of this report. The objective of the survey is to obtain high-quality estimates of certain variables based on a survey of retailers. Examples of variables of interest include cash at hand and acceptance of various methods of payment.

A considerable challenge when conducting surveys is nonresponse. There are two types of nonresponse: unit nonresponse, in which no response is given to the entire survey, and item nonresponse, in which no response is given to a particular survey question. Both types are present in the retailer survey, causing significant problems. There are two main problems survey nonresponse causes: the first is an increase in the variance of the response averages caused by a smaller sample size, and the second is nonresponse bias that occurs if some contacted entities are more likely to respond than others (Bethlehem et al. (2011)).

This technical report analyzes nonresponse in the RCPM survey and examines solutions for the analysis of the survey data. The report starts by describing the general approach used to handle both types of nonresponse. Unit nonresponse is handled by introducing response probabilities (predicted using logistic regression) into a set of survey weights, which adjust for the fact that some businesses are more likely to respond than others. This is followed by a discussion of the imputation algorithms used and their application to handle item nonresponse, for both the method-of-payment (MOP) acceptance variables and cash at hand.

The Bank of Canada previously conducted a similar retailer survey on the cost of payment methods in 2006 (Arango and Taylor (2008a,b)). In this previous survey, the sample size was smaller than in the current survey (approximately 500 businesses responded to the 2006 survey) and no attempt was made to correct for unit nonresponse or for item nonresponse. The current methodology is thus an improvement over the methodology in the previous survey.

This report considers only the part of the survey dealing with small and medium-sized single-location businesses, with 1–4 employees (identified as stratum A) or 5–49 employees (stratum B). A separate analysis is performed for large single-location businesses (50 or more employees, stratum C) and those that have headquarters and branch locations (Jiongo (2017)).

The report focuses primarily on the methodology used and, as a result, only a limited set of variables is considered here (cash at hand, cash acceptance, cheque acceptance, credit card acceptance, debit card acceptance, and prepaid card acceptance).

The survey frame is based on the Dun & Bradstreet (D&B) data set (a limited number of additional businesses known via personal contacts have also been included). The frame itself

was constructed internally at the Bank of Canada and is based on D&B data downloaded from the Hoover's portal between 8 August 2014 and 6 January 2015. The current version of the D&B data available from Hoover's may therefore differ from this version as the Hoover's data set is updated over time. The paper by Welte (2017) discusses the construction of the survey frame.

2 Handling Nonresponse in Weighting

In the RCPM survey, a set of survey weights is constructed, one for each business. In this section, the general approaches for handling nonresponse in the context of weighting for our survey will be discussed. When computing totals (for a variable such as cash at hand), each observation is multiplied by its weight (equation (2)), so observations with a larger weight are given more importance. The weights are necessary for a variety of reasons, including adjusting for businesses having differing inclusion probabilities in the survey sample. The weights can also help with unit nonresponse, as we discuss below.

The creation of the weights for this survey is discussed in Chen and Shen (2017). The initial weights are created using the procedure described below (in equation (1)), involving the inclusion probabilities for each business in the sample from the sampling frame (Welte (2017) discusses the calculation of the inclusion probabilities) and the response probabilities (the probability of a particular business responding to the survey). Next, the weights are calibrated based on auxiliary variables with targets from the D&B data set and the Statistics Canada Business Registry.

Unit nonresponse is addressed using unit response probabilities to adjust the weights. Item nonresponse is handled via imputation, in which missing entries are replaced by imputed values. The initial weights (prior to calibration) are

$$W_i = \frac{1}{I_p(i) \cdot R_p(i)}, \tag{1}$$

where W_i is the weight, $I_p(i)$ is the inclusion probability, and $R_p(i)$ is the unit response probability for entry i.

The estimate of the sample total Y is then computed as

$$Y = \sum_{i \in S} W_i \tilde{X}_i, \tag{2}$$

where S is the set of all survey responses (including both those who answered a particular question and those who did not answer that question) and $\tilde{X}_i$ is, for those entries that are present,

the true value of X_i, and otherwise an imputed value.

We now give an example of how this approach handles the problem of nonresponse in survey weighting. Suppose one sector of the economy is less likely to accept debit cards than other sectors, and fewer businesses in that sector respond to the survey (unit nonresponse) or to that particular question (item nonresponse) than businesses in other sectors. If nonresponse is not taken into account, the survey results will be biased towards a higher debit card acceptance rate than the true rate. To handle unit nonresponse, the lower survey response rate for that sector is taken into account by giving entries in that sector (who are less likely to accept debit cards) a larger weight. Imputation helps with the problem of item nonresponse as follows: assuming that a good predictive algorithm is used, entries with missing values in that sector will likely be assigned values similar to present values in the sector, which counteracts the effect of fewer businesses in that sector (with a lower debit card acceptance rate) responding to the question.

It is theoretically possible to handle the problem of unit nonresponse via imputation or of item nonresponse via nonresponse probabilities. In this survey, however, unit nonresponse is only modelled directly since the survey response rates are very low (less than 3 per cent for stratum A and B businesses in the sample frame and D&B). Imputation for the completely missing entries would not be a good approach, because imputed values will overwhelmingly dominate the results. Similarly, it is possible to model item nonresponse probabilities instead of imputation to handle the problem of item nonresponse here and this was previously considered. However, this is a more problematic approach since predictor variables that are correlated with response indicators, but not with the variables themselves, do not reduce the nonresponse bias and may increase the nonresponse variance. This issue is further discussed in Little and Vartivarian (2005) and Beaumont (2005).

3 Unit Nonresponse

Unit nonresponse is a considerable challenge in the RCPM survey. Only 2.3 per cent of stratum A and B businesses in the sample contacted via mailed packages and telephone calls responded to the survey. Note this figure includes businesses filtered out by address checks and screening phone calls in some cases. If such businesses are excluded the response rate rises to 2.9 per cent. To generate the response probabilities for the weighting, two approaches are considered. In the first, the probabilities predicted by logistic regression are used directly, and in the second, the logistic-regression-predicted probabilities are clustered, and the empirical response rate is taken within each cluster.

Only businesses in the D&B single-location data set (that are in stratum A or B and that are also in the survey frame) are considered here. Businesses in the survey frame filtered out by

address checks and screening phone calls are included in the nonresponse model, as opposed to the inclusion-probability model.[1] Businesses in stratum C, jumpers to or from stratum C, and businesses contacted by personal visits are not included here. Overall, there are $34{,}292$ businesses in the sample being analyzed here, of which 799 did respond and $33{,}493$ did not respond. Table 1 shows the sample sizes and response rates for different components of the survey.

The probability of a business responding to the survey is modelled using logistic regression, using the following variables:

- the region the business was in (British Columbia, Prairies, Ontario, Quebec or Atlantic)
- the two-digit primary North American Industry Classification System (NAICS) code of the business
- presence of a phone number
- presence of a fax number
- presence of a web address
- the D&B prescreen score (low risk, high risk or missing value)
- the age of the business (zero if missing in D&B)

Table 2 shows the coefficients generated by logistic regression for all of the predictor variables. We see that the response rate is higher when a phone number is present, when a fax number is present, when a web address is present, for older businesses, for businesses with a post-office (PO) box, and for businesses with a D&B prescreen score of "low risk" (as opposed to "high risk" or missing). Significant variation in the response rate is also observed by region and by NAICS. A histogram of the predicted response probabilities is shown in Figure 1. Probit regression has been tested previously and produces results almost identical to logistic regression for this data set.

When modelling nonresponse, it is important that the predictor variables are also good predictors of the variables of interest, and not simply of the response indicators. This issue is discussed in Little and Vartivarian (2005) and Beaumont (2005). Fortunately, all of the variables listed above meet this requirement for at least some of the variables of interest. Logistic regression is performed on each variable (by itself) against the variable of interest (considering only those entries for which both variables are present). The significance (p-value) of the

[1] This follows the recommendation made by J.-F. Beaumont, Statistics Canada, and is discussed in Welte (2017).

variable as a predictor is then found. The p-values obtained from logistic regression are shown in Table 3. We see that all of the variables are significant at the 5 per cent level for at least one of the variables of interest, and many have very small p-values.

The quality of the logistic regression is assessed using receiver operating characteristic (ROC) curves and the area under these curves (the ROC AUC or AUROC). A k-fold cross-validation approach is used (with $k = 10$), in which for each fold, the training set consists of entries outside the current fold and the testing set is the current fold, and the predicted probabilities are then combined for all the folds. Figure 2 shows an ROC curve of the results obtained, and Figure 3 shows an ROC curve with the entire data set.[2] We see that the results obtained in both cases are very similar and show fairly good predictive performance.

A nonparametric clustering approach is now taken, in which k-means clustering is applied to the response probabilities with various numbers of clusters. First, the logistic-regression-predicted response probabilities found above are taken, and k-means clustering is used to group them into k clusters. Within each cluster, the empirical response rate is found (by dividing the number of responders within the cluster by the size of the cluster). This empirical response rate is taken to be the response probability for all entries within that cluster. Figure 4 shows the ROC curve with 10 clusters, which is very similar to the ROC curve with logistic regression. The response probabilities obtained with 5 and 10 clusters are shown in Figure 5 and in Tables 4 and 5, in which the response rate within each cluster is compared with the mean predicted probability within the corresponding clusters.

In general, the logistic-regression-predicted probabilities agree quite well with the clustered response rates, with the exception of a very small cluster (the rightmost cluster with $k = 10$; these entries also have a similar but much smaller effect on the rightmost $k = 5$ cluster) whose predicted response probabilities are much larger than their empirical response rates. This is not a significant problem for us since the high predicted response probabilities do not have much influence on the estimates. The response probabilities in this small cluster are larger, which means that the weights of these few entries are less than other observations, so they do not have the large influence problem that outliers with very small predicted response probabilities would have. Since there are only six respondents in the rightmost cluster when $k = 10$ (from Table 5), it is very unlikely that those businesses having their weights reduced would have a significant effect on the final results.

These unit nonresponse probabilities are used in the generation of one of the sets of survey weights. Chen and Shen (2017) discuss the creation of the set of weights for this survey.

[2] The ROC curves and AUROC have been computed using the R package PRROC (Keilwagen et al. (2014)).

4 Variables of Interest

There are many variables of interest in this survey. Among the businesses that responded, many did not answer some of these questions (the rates varying depending on the question). For the purposes of this technical report, six key variables are considered: acceptance of cash, acceptance of cheques, acceptance of credit cards, acceptance of debit cards, acceptance of prepaid cards, and cash at hand. Within the data set, these have the names `acc_cash`, `acc_cheque`, `acc_credit`, `acc_debit`, `acc_prepaid`, and `cash_at_hand`, respectively. The first five are binary MOP acceptance variables, while the last is a numeric variable. This section describes the variables. Table 6 shows the missing rates for these six questions. Additional variables of interest will be considered in the future using the methodology developed in this technical report.

In total, there are 841 businesses in this data set, with weights given for each business, including personal visits and stratum jumpers (for which the weights are generated using mean imputation within the cell by NAICS, number of employees, and region) (Chen and Shen (2017)). These include personal visits and stratum jumpers that are not considered in the unit nonresponse model. Of these businesses, 814 are not stratum jumpers to stratum C (such stratum jumpers are given zero weight).

4.1 MOP acceptance variables

The MOP acceptance variables indicate whether or not a business accepts a particular method of payment, and are all yes/no questions in the survey, for which a blank response is treated as missing. There are additional MOP acceptance variables not considered in this technical report, namely, accept mobile payment, accept Bitcoin, accept contactless credit cards, accept contactless debit cards, accept chip & PIN credit cards, accept chip & PIN debit cards, accept magnetic strip credit cards with PIN, accept magnetic strip debit cards with PIN, accept magnetic strip credit cards with signature, accept magnetic strip debit cards with signature, accept magnetic strip credit cards without signature, accept magnetic strip debit cards without signature, and accept other methods of payment. All of these variables (with the exception of "accept other," which is a text field) are yes/no questions in the same format as before. The missing rates for these variables vary but are generally higher than those of the MOP acceptance variables considered in this report.

4.2 Cash-at-hand variable

The cash-at-hand variable is a numeric variable indicating the value in bank notes and coins that the business has at the start of a typical business day. This variable is usually present (92 per cent of the time), but is sometimes missing. The cash-at-hand variable has a heavy-tailed distribution (with some entries being much larger than others), as shown in Figure 6; this is quite problematic when attempting to perform regressions to impute missing values. An alternative with a much less skewed distribution (as seen in Figure 7) that produces better results with imputation is to apply the transformation $x \mapsto \log(x+1)$ to the cash at hand, where we add one before taking the logarithm to avoid negative infinity with entries that have zero cash at hand.

5 Imputation of Missing Values

To fix the problem of item nonresponse and the associated bias, imputation of missing values is employed. This section discusses the general imputation algorithm used in the report. Specific methods used for the MOP acceptance variables and for the cash-at-hand variable are discussed in the subsequent sections. These techniques were partially inspired by Loh (2015) and He (2006).

Here we use two variants of gradient boosting for imputation, the first being GBM (gradient boosting machine; the R implementation used is also known as a generalized boosted regression model) and the second being XGBoost (extreme gradient boosting, a newer algorithm based on GBM but with some modifications). Both GBM and XGBoost are well-known classifiers that have a very good classification/regression accuracy for many data sets, are very flexible (able to handle many redundant features and messy data well), and can handle missing predictors directly without any need to impute them. Both methods are described below.

For the imputation of cash at hand, the GBM algorithm is used, while for the MOP acceptance variables, the XGBoost algorithm is used for imputation. The reason for these choices is that in earlier testing, XGBoost was found to improve the predictive quality, while GBM remained superior for cash-at-hand imputation. Both algorithms can be used for either type of variable (categorical or numeric) and for future variables, both algorithms should be tested.

To improve the stability of the estimates and assist with providing error estimates, a bootstrap imputation approach is used, in which repeated bootstrap samples of the data set are taken and are used for the imputation, while entries outside the corresponding bootstrap sample are held out and used for estimating the quality of the imputation. This resembles the "out-of-bag error" method that can be used to estimate the error of a random forest (Liaw and Wiener (2002)).

5.1 Gradient boosting machines

Gradient boosting machines provide a method for solving classification and regression problems. GBM is an ensemble technique, in which the predictions of many models or learners are combined into a single stronger prediction. One well-known ensemble technique is the random forest, in which many classification or regression trees are grown independently (with bootstrap samples of the data and random subsets of features considered at each split) and the trees vote or are averaged for the result (Liaw and Wiener (2002)). In boosting methods, instead of growing each tree or building each model independently, an attempt is made to improve or "boost" the model at each step from the previous model (so the new tree depends on the previous trees and tries to improve the existing model). In gradient boosting machines, a loss function is specified, and at each step, a new learner is fit to the negative gradient of the loss function over the existing combined model. The combined model is then updated, including the new learner. In the R package `gbm`, each individual learner is a regression tree. Missing values can be handled directly by the `gbm` package since the tree-growing algorithm can handle missing values.[3,4]

Suppose we have a data set with points $x_1, x_2, \ldots, x_n$ and labels $y_1, y_2, \ldots, y_n$. We would like to estimate an unknown function f that maps input points to labels. In gradient boosting, we construct a predictor of the form

$$\hat{f}(x) = \sum_{i=0}^{k} \hat{f}_i(x), \tag{3}$$

with $\hat{f}_0$ being an initial estimate (for instance, a constant function) and $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_k$ being base learners. In gradient-boosted trees (which are implemented in the R `gbm` package), each of the base learners $\hat{f}_i$ $(1 \leq i \leq n)$ is a regression tree.

The algorithm for gradient-boosted trees (implemented in the R package `gbm`) works as follows (the description is based on Hastie et al. (2009) and Ridgeway (2015)):

(i) We start with a training data set containing n labelled points, with points $x_1, x_2, \ldots, x_n$ and labels $y_1, y_2, \ldots, y_n$. We fix a number of iterations M for our algorithm and a shrinkage parameter λ (discussed below).

[3] When performing a variable split, missing values for a variable are assigned into their own node at each split, which is called "MissingNode" in the code. I would like to thank the users at `http://stackoverflow.com/questions/14718648/r-gbm-handling-of-missing-values` for bringing this to light.

[4] Further information about the GBM algorithm and the R package `gbm` is available in Friedman (2002); Ridgeway (2015, 2012); Elith and Leathwick (2016); Natekin and Knoll (2013); and Hastie et al. (2009).

(ii) We start with an initial constant prediction that minimizes the total loss function,

$$\hat{f}_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} \ell(y_i, \gamma). \tag{4}$$

(iii) For each iteration m from 1 to M, we update the current estimate to obtain a new estimate as follows (if subsampling [discussed below] is enabled, only a subset of the observations will be considered at each step):

(a) We compute the "pseudo-residuals" r_{im} (with $1 \leq i \leq n$), which are the components of the negative gradient of the loss function with respect to the existing estimate at each point, that is, for $1 \leq i \leq n$, we compute

$$r_{im} = -\frac{\partial}{\partial \hat{f}_{m-1}(x_i)} \ell(y_i, \hat{f}_{m-1}(x_i)) \tag{5}$$

$$= -\frac{\partial}{\partial \hat{y}_i} \ell(y_i, \hat{y}_i) \text{ where } \hat{y}_i = f_{m-1}(x_i). \tag{6}$$

(b) We then fit a regression tree to the pseudo-residuals r_{im}, with the regions R_{ji} being the leaves of the tree (with J_m being the number of leaves and $1 \leq j \leq J_m$).

(c) For each $j \in \{1, 2, \ldots, J_m\}$, we determine the optimal update γ_{jm} for points in the region R_{ji}, by computing

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} \ell\left(y_i, f_{m-1}(x_i) + \gamma\right). \tag{7}$$

(d) For each point in the region R_{ji}, we adjust the current estimate by γ_{jm} (by adding γ_{jm} to points in that region),

$$f_m(x) = f_{m-1}(x) + \lambda \sum_{j=1}^{J_m} \gamma_{jm} \mathbb{1}_{\{x \in R_{jm}\}}. \tag{8}$$

(iv) We return the final estimate $\hat{f} = \hat{f}_M$.

One important parameter for GBM is the choice of loss function. In the R package `gbm`, this is controlled by the `distribution` parameter. With "`distribution = gaussian`," the mean squared error (MSE) is used as the loss function (no normality assumptions are made in this case, only that the MSE is a useful loss function). With "`distribution = bernoulli`," a logistic loss for binary outcomes is used, and several other distributions can be specified as well.

The number of trees also has to be specified, controlled by the `n.trees` parameter in the `gbm` package. Adding more trees to the model will generally improve the model up to a point; however, if too many are added, the model will start to overfit. It is possible to train the model using many trees, and to then generate predictions with a smaller number of trees (this is done by specifying `n.trees` to the predict function, which can take values up to the number of trees used to train the model). In addition, adding trees to an existing model can be done (using the `gbm.more` function). This flexibility allows different choices for the number of trees to be quickly compared on a validation set, for which we can compare the results and select the number of trees that appears optimal.

The effect of the shrinkage parameter λ in equation (8) is to control how much the model is adjusted by each individual tree: smaller values make the algorithm more conservative, reducing the effect of adding each individual tree at each step. It is very common for higher prediction accuracies to be achieved by using a smaller shrinkage parameter and a larger number of trees, which makes the algorithm take many small steps instead of a few large ones (however, this increases the computational cost owing to the need to compute more trees).

Subsampling can be incorporated into the GBM algorithm, in which for each decision tree only a random subset of the observations are considered instead of all the observations (a different random subsample is chosen at each step). This helps to prevent overfitting. In the `gbm` package, subsampling is controlled by the `bag.fraction` parameter.

Other parameters of interest include `weights`, which specifies case weights for each observation; `minobsinnode`, which specifies the minimum number of observations in each node of the tree; and `interaction.depth`, which specifies the maximum depth of the trees that are grown by GBM. There are additional parameters, documented in Ridgeway (2015).

5.2 XGBoost algorithm

The extreme boosting machine is a newer method based on gradient tree boosting, implemented in the R package `xgboost`. The `xgboost` package also implements gradient boosting with some other base learners; however, in this technical report we use only the version with regression trees as base learners. In XGBoost, regularization is performed when growing each tree to prevent overfitting. The design of XGBoost also emphasizes efficient computation of the trees so that the classifier can be trained and applied quickly.

In XGBoost, we have a predictor of the form in equation (3) (in Chen and Guestrin (2016), the initial estimate is taken to be zero, $\hat{f}_0 = 0$; the initial estimate should not be too important since there will be many trees added that adjust this initial estimate). When growing each tree, we include a regularization term $\Omega(\hat{f}_i)$ in the objective to be minimized to limit tree complexity

and prevent overfitting,

$$\Omega(f_i) = \gamma T + \frac{1}{2}\lambda \|\boldsymbol{w}\|^2, \tag{9}$$

where T is the number of leaves in the tree and $\boldsymbol{w}$ is the vector of leaf weights (the values assigned to each leaf or terminal node of the tree).

Suppose that $\hat{y}_i^{(t)} = f_t(x_i)$ is the prediction at step t. From step $t-1$, we add a new base learner f_t at step t, where we would like to minimize the objective

$$\mathcal{L}_{(t)} = \sum_{i=1}^{n} \ell\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t). \tag{10}$$

We define g_i to be the i^{th} component of the gradient and h_i to be the i^{th} component of the second-order gradient (equivalently, the diagonal of the Hessian matrix) of the loss function with respect to the current estimate,

$$g_i = \frac{\partial}{\partial \hat{y}_i^{(t-1)}} \ell\left(y_i, \hat{y}_i^{(t-1)}\right), \tag{11}$$

$$h_i = \frac{\partial^2}{\partial \hat{y}_i^{(t-1)^2}} \ell\left(y_i, \hat{y}_i^{(t-1)}\right). \tag{12}$$

We take a second-order approximation to equation (10) that can be optimized quickly,

$$\mathcal{L}_{(t)} \approx \sum_{i=1}^{n} \left(\ell\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right) + \Omega(f_t). \tag{13}$$

Since we are interested in minimizing the above expression, we can drop constant terms to obtain

$$\tilde{\mathcal{L}}_{(t)} = \sum_{i=1}^{n} \left(g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right) + \Omega(f_t). \tag{14}$$

Chen and Guestrin (2016) discuss how the XGBoost algorithm optimizes the above objective, using a greedy algorithm to iteratively add branches to the tree and then finding optimal weights for the leaves of the tree. The algorithm for constructing trees can handle missing values (discussed in the paper in the context of sparse input data) by assigning a default direction for each branch, where the optimal default direction is learned from the data.

We notice that the only dependency the expression in equation (14) has on the loss function is in the g_i and h_i. This means we can apply any loss function for which we know the gradient and second-order gradient for XGBoost, without requiring any modifications to the algorithm. This allows us to easily use a wide variety of loss functions with XGBoost, including custom loss functions with weights for each instance. In the `xgboost` package, this can be done

using the `objective` parameter, which can be either a user-defined R function returning the gradient and second-order gradient or a string that specifies some options built into `xgboost` (such as `reg:linear` for linear regression or `binary:logistic` for logistic regression for binary classes, returning probabilities). Case weights for XGBoost can be specified to the `xgb.DMatrix` function by specifying a `weight` parameter.

When invoking XGBoost in the `xgboost` R package, the `nrounds` parameter specifies the number of iterations (equivalent to `n.trees` for GBM), the `eta` parameter specifies the learning rate (equivalent to `shrinkage` for GBM), the `max_depth` parameter specifies the maximum depth of the trees (equivalent to `interaction.depth` for GBM), `gamma` specifies the value of γ in equation (9), and the `subsample` and `colsample_bytree` parameters control row and column subsampling when building each tree (so a different random subsample of the rows and of the columns can be used when building each tree). There are additional parameters, which are documented in the R package (Chen et al. (2016)).

5.3 Bootstrap imputation

A bootstrap imputation approach is used with the GBM and XGBoost algorithms, in which repeated bootstrap subsamples of the non-missing entries of the data set are used to predict the missing entries. The advantage of the bootstrap approach is that it helps stabilize the result, reveals the variability of the imputation predictions, and allows us to create error estimates by comparing the predictions for the non-missing entries not in the current bootstrap sample (that is, the predicted values for out-of-sample entries for which the true values are known) against the true values in each round.

For each variable to impute, the data set is split into two sets: one in which the variable of interest is present, and the other in which it is missing. We then take $B = 1{,}000$ bootstrap samples of the non-missing entries (the constant B can be adjusted downward or upward if desired) for the bootstrap imputation. For each round i of the bootstrap imputation, the i^{th} bootstrap sample is used as a training set, and predictions for the entries, both missing and outside the current bootstrap sample, are generated using GBM or XGBoost with the predictor variables listed above. Equivalently, if we let B_i be the i^{th} bootstrap sample, a model (for us, GBM or XGBoost) is trained using the entries in B_i, and predictions are generated for all entries in B_i^{c} (the complement of B_i, which is the set of all entries not in B_i). The bootstrap imputation results are saved, along with their means and standard deviations.

Combining the above, the general algorithm for imputation is as follows:

- For the variable of interest, the bootstrap samples are created as described above.

- For each bootstrap sample, a GBM or XGBoost model is trained, using parameters fixed for the particular variable of interest.
- For each bootstrap sample, predictions are generated only for the entries outside the bootstrap sample (both out-of-sample entries for which the variable of interest is known and entries for which the variable of interest is unknown), using the model found in the previous step for that bootstrap sample. No predictions are generated for entries in the current bootstrap sample.
- The means and standard deviations of all of the generated predictions for each entry (with the number of trees fixed) are computed (we notice that every prediction is generated without the current entry being in the bootstrap sample). A separate averaged prediction is generated for each specified number of trees.

Depending on the variable of interest (in this technical report, the cash-at-hand variable in particular), preprocessing may be done before the imputation algorithm is run and a transformation may be applied after the imputation. This is discussed below for the variables of interest where this is performed.

6 Imputation of MOP Acceptance Variables

In this section, we discuss imputation of the MOP acceptance variables (accept cash, accept cheque, accept credit, accept debit, and accept prepaid). All of these variables are binary (either "yes" or "no"), with missing rates and observed acceptance rates given in Table 6.

6.1 Imputation method

For each MOP acceptance variable, the bootstrap imputation discussed above is applied with $B = 1{,}000$ bootstrap replications. For each bootstrap sample, an XGBoost model is trained, with the learning rate parameter `eta` set to 10^{-2} and a total of $1{,}000$ trees computed (using the `nrounds` parameter). The survey weights (based on the unit response probabilities with 10 clusters) are used in the XGBoost. All other parameters have been left at their defaults in the XGBoost package. Predictions are generated for each of $10, 20, 30, \ldots, 990, 1000$ trees (the determination of the number of trees to use for prediction of the final result is discussed in the next section).

For the imputation, probability scores in $[0, 1]$ are imputed for the missing entries, using the XGBoost algorithm. The present entries are converted to 0 for the "no" entries and 1 for the "yes" entries. The means and totals (both unweighted and weighted) can then easily be computed

from this (where the imputed entries take on fractional values, reflecting the uncertainty in the prediction). It is theoretically possible to convert the imputed values to the quantities 0 or 1 (this can be done by taking all values below a threshold to be zero and otherwise to be 1), but for our purposes (the calculation of weighted totals), this is unnecessary and increases variance.

The XGBoost algorithm requires its input (including predictor variables) to be numeric. Categorical values are therefore coded as a set of binary indicator variables for XGBoost ("one-hot encoding") (Chen et al. (2016)).

6.2 Imputation results

Several methods are used to determine the quality of the imputed values. Unless otherwise specified, each quality measure in this section is based on the mean of the out-of-sample predictions (that is, the mean of the predictions each time the entry was outside the bootstrap sample). We first look at the cross-entropy loss, a weighted version of which is used to determine the number of trees to use in prediction. For binary variables (with $y \in 0, 1$ being a true value and $x \in (0, 1)$ being a predictor) the cross-entropy loss function is defined as[5]

$$\ell(x, y) = -y \log(x) - (1 - y) \log(1 - x). \tag{15}$$

For a sample of n predicted values $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and true labels $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$, we can take the unweighted mean cross-entropy as a loss function,

$$\ell(\boldsymbol{x}, \boldsymbol{y}) = -\frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i). \tag{16}$$

Given a set of survey weights $w_1, w_2, \ldots, w_n$, we can define a weighted mean cross-entropy as follows

$$\ell(\boldsymbol{x}, \boldsymbol{y}) = -\frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \ell(x_i, y_i). \tag{17}$$

Figures 8 and 9 show the unweighted and weighted cross-entropy results, respectively. We see that both functions are stable (no sudden fluctuations or jumps—instead, they smoothly decrease to a minimum and start to increase again) and have the same general shape, for all of the MOP acceptance variables. The exact minimum observed differs between the weighted and unweighted versions. However, both types of minima are in the same region in all cases, and the minimum for one graph is still a good point (i.e., the loss function is small) for the other graph. This means that choosing the minimum based on the weighted or the unweighted version should

[5] This can be extended to the case where both x and y are zero or both are 1 by defining the cross-entropy to be zero in these two cases.

have little effect on the final results. Since the weighted totals are the quantities of ultimate interest, the weighted cross-entropy is used to determine the number of trees to use. Table 7 shows the number of trees chosen for the imputation of each MOP acceptance variable.

At the number of trees chosen, additional quality measures are computed. The ROC curves are plotted in Figures 10, 11, 12, 13 and 14, with the AUROC shown for each variable in Table 7 (computed using the R package PRROC, Keilwagen et al. (2014)). We see that the predictive accuracy appears good in each case. Within the bootstrap imputation, we compute the standard deviation as well as the mean for each entry (that is, the standard deviation of the predictions for each entry, every time the entry is outside the bootstrap sample). This is an indicator of the variability of the predictions in the bootstrap imputation, and is displayed as well in Table 7.

In addition, bootstrapped estimates of the bias associated with the imputation are found. The unweighted bias is computed in the usual manner

$$\text{bias}_{\text{unweighted}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i), \tag{18}$$

while the weighted bias can be computed as

$$\text{bias}_{\text{weighted}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\sum_{i=1}^{n} w_i}\sum_{i=1}^{n} w_i(x_i - y_i). \tag{19}$$

A bootstrap procedure is used with $B = 25{,}000$ bootstrap samples of the mean imputation results, in which, for each bootstrap sample, the weighted and unweighted bias is computed. Histograms of the results obtained are shown in Figures 15, 16, 17, 18 and 19 and the mean unweighted and weighted bias observed is shown in Table 7.

7 Imputation of Cash at Hand

In this section, we discuss imputation of the cash-at-hand variable. This is a numeric variable with entries that have quite a bit of variability. A logarithmic transformation $x \mapsto \log(x+1)$ is computed for each present entry x prior to imputation. This is done because $\log(x+1)$ has a much less skewed distribution (as seen in Figures 6 and 7) and can be imputed more accurately. The bootstrap imputation approach discussed earlier is used, and the GBM regression is used to generate the predictions. The quality of the logarithm-transformed imputed values is analyzed. The problem of back-transforming the imputed values in the logarithmic scale to the original scale without creating bias is then discussed.

A comparison of the values predicted by GBM with the logarithmic transformation and

GBM with the original values is performed. The imputation predictions generated by applying GBM directly to the data are compared with the back-transformed predictions (without the use of any correction factor, as discussed below) generated with the logarithm-transformed values for GBM (so both sets of predictions are directly comparable). The MSE is found to be similar in both cases (being slightly higher for the logarithm-transformed version), but the mean absolute error (MAE) and a "bins" score (the fraction of points such that the true value is within a certain distance from the imputed value) are found to be significantly better for the logarithm-transformed version. This suggests that the predictions and the MSE are affected by outliers, and the logarithm-transformed version is a "safe" choice to make.

7.1 Imputation of logarithm-transformed variable

First, the logarithmic transformation $x \mapsto \log(x+1)$ is applied to every entry with cash at hand present. The imputation algorithm attempts to predict the transformed cash at hand value for entries with cash at hand missing.

The bootstrap imputation approach discussed previously is used for cash at hand, with 100 rounds of bootstrap samples taken. For generating predicted values, GBM regression is performed, using the MSE as the loss function (this is done by specifying the distribution parameter in GBM to be `gaussian`; note that this specifies the MSE loss function and does not make further normality assumptions). Most of the other parameters for GBM are kept at their defaults, with the exception of the weights being specified (the survey weights based on the response probabilities with 10 clusters are used here), the `interaction.depth` (the maximum depth of trees to grow) being set to 6, the `shrinkage` (a factor controlling how much the model is updated by each new tree) set to 10^{-2}, and `n.trees` (the number of trees to grow) set to $2{,}000$ (with predictions generated using $10, 20, 30, \ldots, 1990, 2000$ trees). For each number of trees and each bootstrap sample, predictions are generated for all out-of-sample entries (both entries with unknown cash at hand and entries with known cash at hand not in the current bootstrap sample). For each number of trees, the mean and standard deviation of the predictions for each entry from all the bootstrap samples are computed.

The quality of the imputed values is assessed using both unweighted and weighted versions of the MSE for the logarithm-transformed variables. Suppose we have a sample of n entries with observed cash at hand, with w_i being the weights, x_i being the predicted values in the logarithm scale, and y_i being the observed values for cash at hand with the $x \mapsto \log(x+1)$ transformation applied (here $1 \leq i \leq n$). The unweighted MSE is defined as

$$\text{MSE}_{\text{unweighted}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2. \tag{20}$$

The weighted MSE is defined as

$$\text{MSE}_{\text{weighted}} = \frac{\sum_{i=1}^{n} w_i (x_i - y_i)^2}{\sum_{i=1}^{n} w_i}. \tag{21}$$

For each number of trees, both the weighted and unweighted MSE are computed (based on the predictions averaged over the bootstrap entries). Plots of these are shown in Figures 20 and 21. We see that both graphs are stable and have the same shape, with the minimum being attained in approximately the same place for both graphs. Overall, the smallest unweighted MSE is 3.12 and is attained with 580 trees, while the smallest weighted MSE is 3.42 and is attained with 550 trees (here the average standard deviation of the bootstrapped predictions is 0.43). This can be compared with the mean imputation (where the missing values are replaced by the mean of the logarithm-transformed cash at hand), which has an unweighted MSE of 4.55 and a weighted MSE of 4.90, both of which are much worse than the result obtained with GBM. The average standard deviation of the bootstrapped predictions is 0.44 and 0.43 with the 580 and 550 trees, respectively.

Since the ultimate quantity of interest is the weighted total cash at hand, the point with the optimal weighted MSE is chosen (with 550 trees). The unweighted MSE serves as a robustness check and confirms that this still produces good results in another quality measure that does not involve the weights. The graph is stable and the MSE is relatively small in a wide region around this point (slowly increasing as we move away from the point), without sudden unexpected jumps or fluctuations.

7.2 Back-transformation of logarithm-transformed variable

Intuition suggests back-transforming the variables with the $x \mapsto \log(x+1)$ transformation by applying the inverse function $y \mapsto \exp(y) - 1$. The problem with this is that a bias may be created and the resulting estimates using the mean of the imputed values may be biased. This is a well-known issue, noticed as early as 1941 in Finney (1941), in a variety of contexts where estimates are based on logarithm-transformed variables. Here is an explanation of the problem in our case. Suppose X_i is the true value we are trying to predict (for each entry i in the sample), Z_i is a vector of predictor variables, and ϕ_i is a Borel-measurable function of Z_i. We can let ϕ_i be an estimate of $\log(X_i + 1)$ based on Z_i. We then let ϵ_i be a random variable that corresponds to the error of our estimate, so we have

$$\log(X_i + 1) = \phi_i + \epsilon_i. \tag{22}$$

Rearranging the above equation and taking the conditional expectation on both sides, we

find

$$\mathbb{E}\left[X_i|Z_i\right] = \mathbb{E}\left[\exp\left(\phi_i + \epsilon_i\right) - 1|Z_i\right]. \tag{23}$$

Since ϕ_i is a Borel-measurable function of Z_i, this becomes

$$\mathbb{E}\left[X_i|Z_i\right] = \mathbb{E}\left[\exp\left(\epsilon_i\right)|Z_i\right]\exp\left(\phi_i\right) - 1. \tag{24}$$

If the errors ϵ_i are identically distributed, the $\mathbb{E}\left[\exp\left(\epsilon_i\right)|Z_i\right]$ term can be replaced by a constant C,

$$\mathbb{E}\left[X_i|Z_i\right] = C\exp\left(\phi_i\right) - 1. \tag{25}$$

We see that if $C = 1$, then the expression simplifies to $\exp(\phi_i) - 1$ and our original estimate is unbiased. However, this is not necessarily the case, and it is possible for it to be not equal to one. In this case, our original estimate will be biased. To avoid this bias, we can multiply the exponential term by a correction factor C, to obtain an expression of the form $C\exp(\phi_i) - 1$. A variety of methods for finding such a correction factor C are discussed in the literature (in various contexts such as logarithmic regression). Some papers discussing possible choices of correction factor are: Finney (1941); Baskerville (1972); Snowdon (1991); and Zeng and Tang (2011). Some of the proposed correction factors are sensitive to normality assumptions, which we would like to avoid here.

We find a method-of-moments estimator for C as follows, which we denote as C_m. We first take the expectation on both sides of equation (25) and apply the law of total expectation to obtain

$$\mathbb{E}\left[X_i\right] = C_m\mathbb{E}\left[\exp\left(\phi_i\right)\right] - 1. \tag{26}$$

We then rearrange the equation for C_m,

$$C_m = \frac{\mathbb{E}\left[X_i\right] + 1}{\mathbb{E}\left[\exp\left(\phi_i\right)\right]}. \tag{27}$$

We now need to replace the terms in equation (27) with empirical estimates based on the observed sample values x_i and our predicted estimates $\phi_i(z_i)$. Let S_r be the sample of observed entries and let $n = |S_r|$ be the number of observed entries. Substituting in the empirical values, we obtain

$$C_m = \frac{\frac{1}{n}\sum_{i\in S_r}\left(x_i\right) + 1}{\frac{1}{n}\sum_{i\in S_r}\exp\left(\phi_i(z_i)\right)}. \tag{28}$$

Simplifying, we find that

$$C_m = \frac{\sum_{i\in S_r}\left(x_i + 1\right)}{\sum_{i\in S_r}\exp\left(\phi_i(z_i)\right)}. \tag{29}$$

We can modify the above estimator in equation (29) to incorporate the survey weights w_i, to obtain an alternative correction factor

$$C_w = \frac{\sum_{i \in S_r} w_i \left(x_i + 1\right)}{\sum_{i \in S_r} w_i \exp\left(\phi_i(z_i)\right)}. \tag{30}$$

7.3 Results of back-transformation

We now compute and evaluate the back-transformation results, with the null correction constant $C_{\text{null}} = 1$ (as a control), the unweighted correction constant C_m (from equation (29)), and the weighted correction constant C_w (from equation (30)). A bootstrap method is used to evaluate each of these correction constants to find which one is the most stable and produces results with the smallest bias.

In the bootstrap approach, we take $B = 25{,}000$ bootstrap samples of the entries with known cash at hand (true value present in data set). For each bootstrap sample, we then calculate each of the correction factors based only on those entries inside the bootstrap sample (using the formulas above). The bias and various error measures are then computed for those entries that are outside the corresponding bootstrap sample by applying the back-transformation $x \mapsto C \exp(x+1) - 1$ (with C being a correction factor computed based on the bootstrap factor; here C_{null}, C_m and C_w are computed). This approach avoids the problem of the correction factors being fit to known values (with the same points being used for both determining the values and assessing their quality) and thereby biasing the error estimates.

We first plot histograms of the values of the correction constants (from the bootstrap samples) in Figure 22. The histograms in Figures 23, 24 and 25 contain the bias results with no correction factor C_{null}, the unweighted correction factor C_m and the weighted correction factor C_w. Table 8 contains a summary of the results obtained with these three correction factors.

From this, we see that the weighted correction factor has significantly less variance than the unweighted version. It appears that the weights have a stabilizing effect on the correction factor. The weighted correction factor also produces the smallest weighted bias (approximately zero for our purposes) and has an unweighted bias that is much smaller than with no correction factor (although larger than with the unweighted correction factor, which produces an unweighted bias of approximately zero for our purposes).

We notice that when the correction factor is applied, the MSE of the variables with the $x \mapsto \log(x+1)$ transformation applied becomes much higher, in some cases higher than the mean imputation. This suggests that if we want to obtain accurate predictions for individual entries, we should keep the biased estimate with $C_{\text{null}} = 1$. However, we are interested in correcting for nonresponse bias in the weighted total, so what is important for us is that the bias

of the prediction for all of the missing entries is small, and not that each individual prediction has a small error. In our case, the imputation will generate logarithmic-transformed predictions that are as accurate as possible for each missing entry, and the correction factor in the back-transformation will help fix the overall bias of the predictions, so the weighted total will be as close to the true value as possible.

Overall, the weighted correction factor is significantly more stable than the unweighted correction factor, produces the smallest weighted bias (which is important since the final result of interest is a weighted total), and produces a much smaller unweighted bias than with no correction factor. For these reasons, it is recommended that the weighted correction factor C_w be used for the final imputed values.

8 Conclusion

Nonresponse is one of the major challenges in the Retailer Survey on the Cost of Payment Methods, which must be accounted for to avoid bias in the final results. Unit nonresponse is handled by a logistic-regression-based response probability model followed by weight adjustment, and item nonresponse is handled by imputation using the GBM and XGBoost algorithms.

These solutions are effective at addressing the problem of nonresponse in our survey. The logistic regression model has good predictive accuracy for the task at hand. Two sets of response probabilities are generated, one with a direct approach and another using a clustering approach with logistic regression. Both sets of response probabilities produce similar results and both should be considered for the creation of the set of weights. Chen and Shen (2017) discuss the creation of the weights, for which these response probabilities are used. The unit nonresponse methodology is able to handle very low response rates, provided that the results are checked carefully. For the imputation, the GBM and XGBoost algorithms are successfully able to handle the messy data with a large number of predictors and produce imputed values whose quality ranges from good to excellent for our data set.

References

Arango, C. and Taylor, V. (2008a). Merchant acceptance, costs, and perceptions of retail payments: A Canadian survey. Bank of Canada Discussion Paper No. 2008-12.

Arango, C. and Taylor, V. (2008b). Merchants' costs of accepting means of payment: Is cash the least costly? *Bank of Canada Review* (Winter): 15-23.

Baskerville, G. L. (1972). Use of logarithmic regression in the estimation of plant biomass. *Canadian Journal of Forest Research* 2(1) 49–53.

Beaumont, J.-F. (2005). On the use of data collection process information for the treatment of unit nonresponse through weight adjustment. *Survey Methodology* 31(2) 227–231.

Bethlehem, J., Cobben, F., and Schouten, B. (2011). *Handbook of Nonresponse in Household Surveys*. Wiley.

Chen, H. and Shen, R. (2017). The Bank of Canada 2015 retailer survey on the cost of payment methods: Calibration for single-location retailers. Bank of Canada Technical Report No. 109.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. arXiv preprint 1603.02754.

Chen, T., He, T., and Benesty, M. (2016). *XGBoost: Extreme Gradient Boosting*. R package version 0.4-3.

Elith, J. and Leathwick, J. (2016). *Boosted Regression Trees for Ecological Modeling*. Provided as a vignette for R package `dismo`.

Finney, D. J. (1941). On the distribution of a variate whose logarithm is normally distributed. *Supplement to the Journal of the Royal Statistical Society* 7(2) 155–161.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis* 38(4) 367–378.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, 2nd edition.

He, Y. (2006). *Missing Data Imputation for Tree-Based Models*. PhD thesis.

Jiongo, V. D. (2017). The Bank of Canada 2015 retailer survey on the cost of payment methods: Estimation of the total private cost for large businesses. Bank of Canada Technical Report No. 110.

Keilwagen, J., Grosse, I., and Grau, J. (2014). Area under precision-recall curves for weighted and unweighted data. *PLOS ONE* 9(3) 3. R package version 1.1.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News* 2(3) 18–22.

Little, R. J. and Vartivarian, S. (2005). Does weighting for nonresponse increase the variance of survey means? *Survey Methodology* 31(2) 161–168.

Loh, W.-Y. (2015). Regression tree approach to analysis of survey data with missing observations. Presented at CRM-CANSSI Workshop on Statistical Inference for Complex Surveys with Missing Observations.

Natekin, A. and Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics* 7 21.

Ridgeway, G. (2012). *Generalized Boosted Models: A Guide to the gbm Package*. Available from `http://ftp.auckland.ac.nz/software/CRAN/doc/vignettes/gbm/gbm.pdf`.

Ridgeway, G. (2015). *GBM: Generalized Boosted Regression Models*. R package version 2.1.1.

Snowdon, P. (1991). A ratio estimator for bias correction in logarithmic regressions. *Canadian Journal of Forest Research;* 5 720.

Welte, A. (2017). The Bank of Canada 2015 retailer survey on the cost of payment methods: Sampling. Bank of Canada Technical Report No. 108.

Zeng, W. S. and Tang, S. Z. (2011). Bias correction in logarithmic regression and comparison with weighted regression for nonlinear models. Available from Nature Precedings.

Appendix A Figures

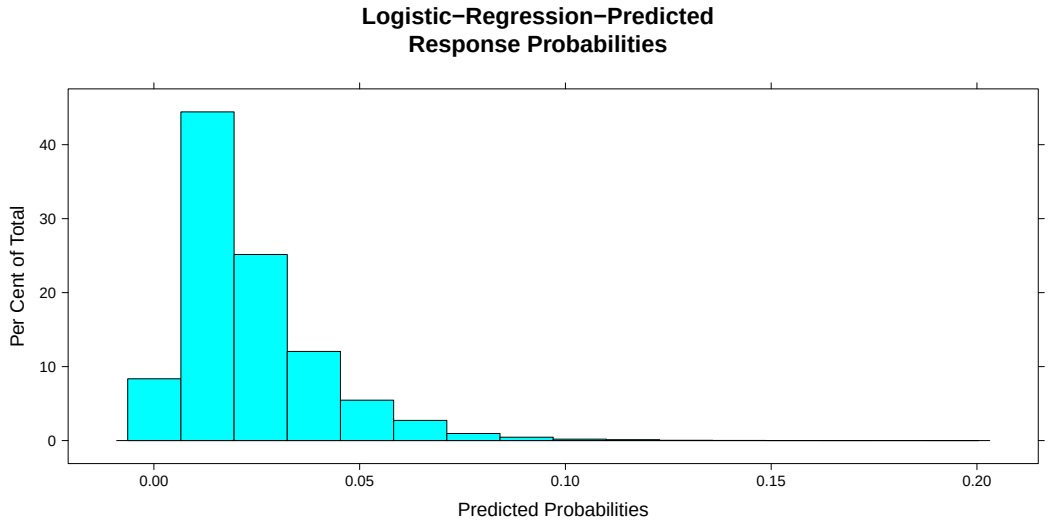


Figure 1: Histogram of the logistic-regression-predicted response probabilities.

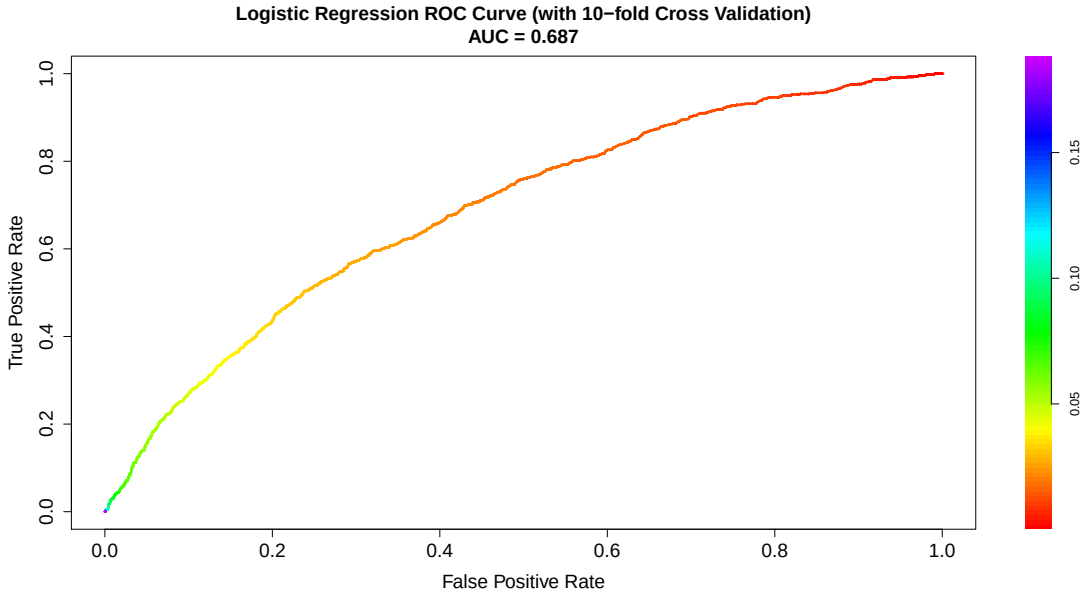


Figure 2: Plot of receiver operating characteristic (ROC) curve of the logistic-regression-predicted response probabilities, using a k-fold cross-validation approach (with $k = 10$ here). The colour indicates the minimum threshold at that point on the ROC curve to classify as responded.

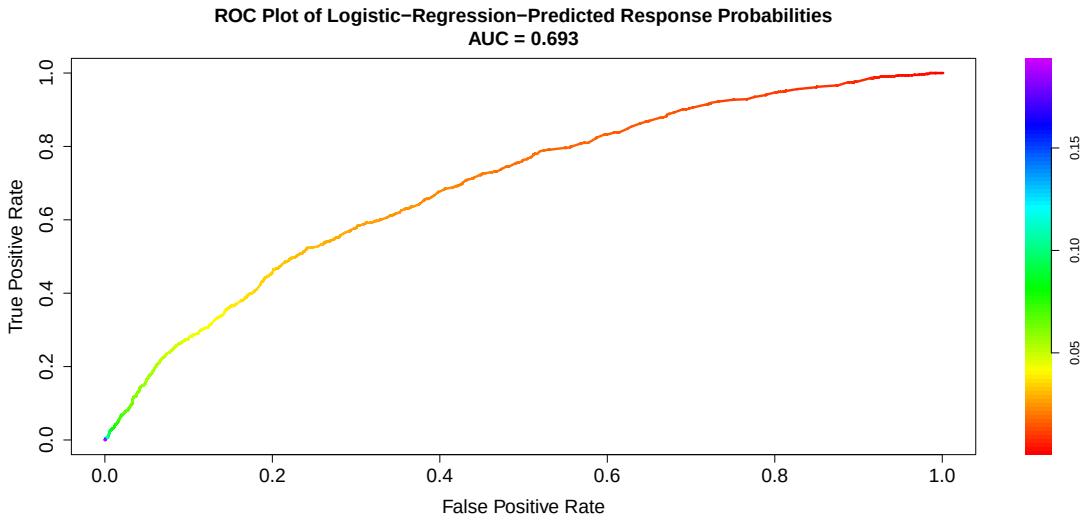


Figure 3: Plot of receiver operating characteristic (ROC) curve of the logistic-regression-response probability predictions (using the entire data set). The colour indicates the minimum threshold at that point on the ROC curve to classify as responded.

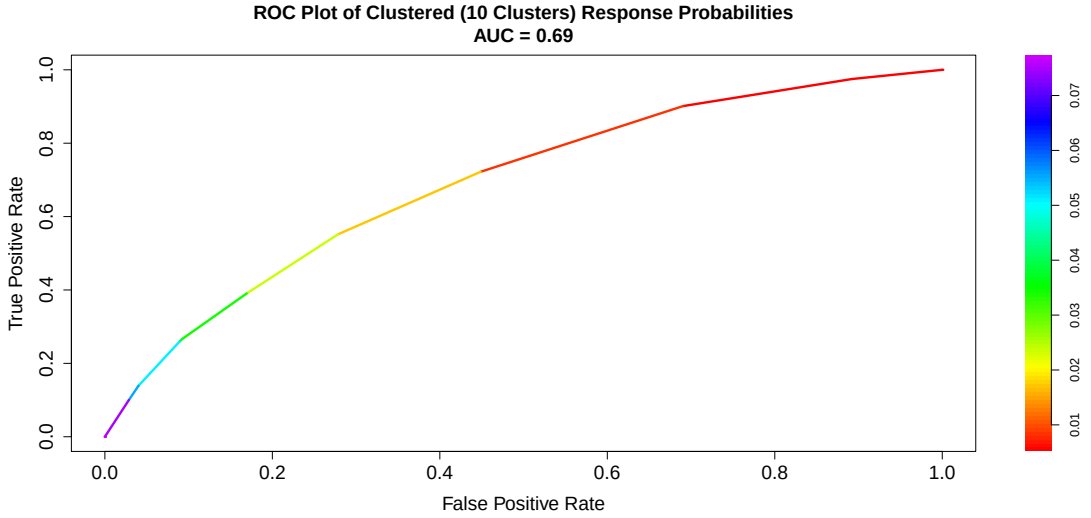


Figure 4: Plot of receiver operating characteristic (ROC) curve of the clustered response rates with 10 clusters. The colour indicates the minimum threshold at that point on the ROC curve to classify as responded.

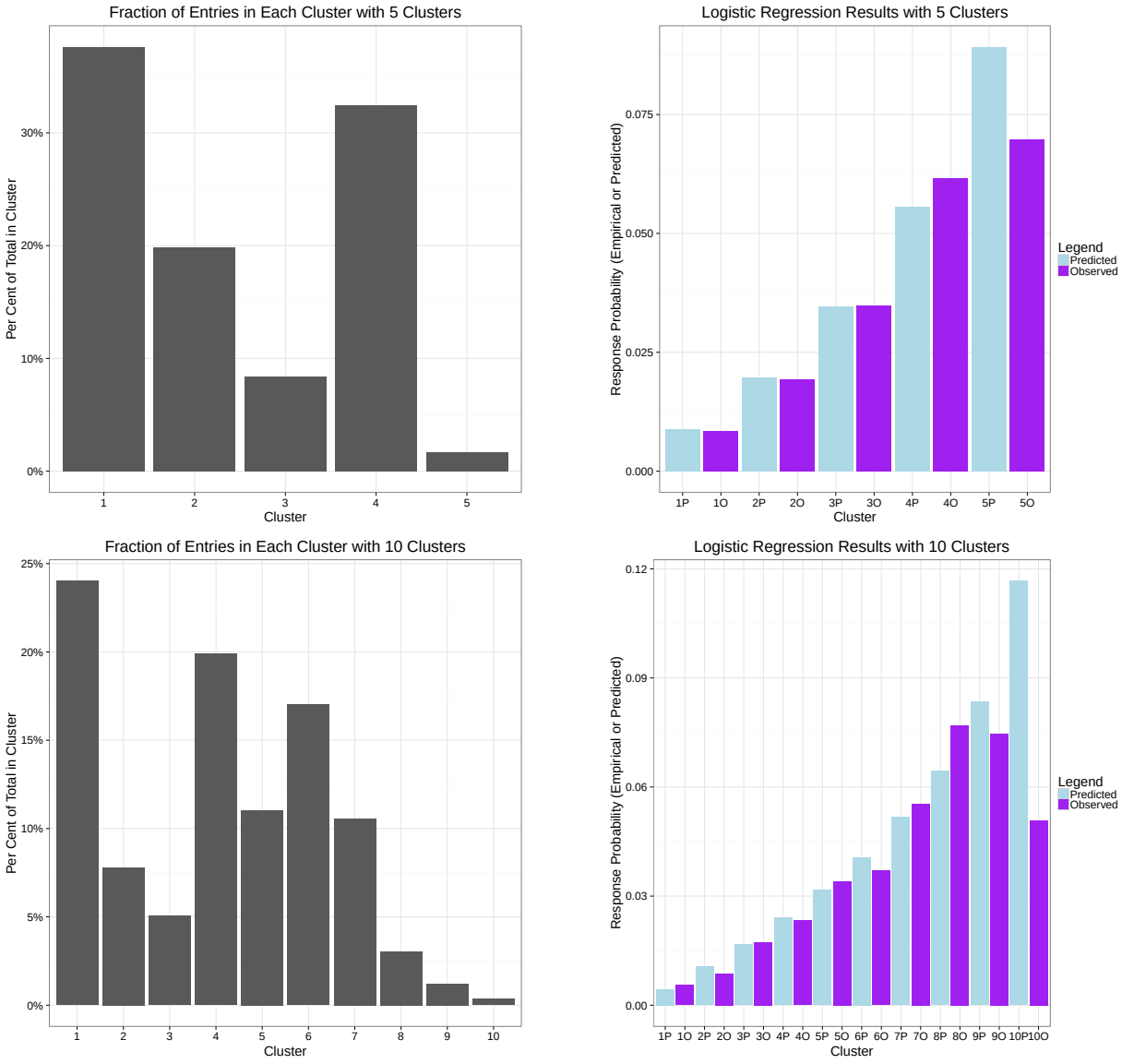


Figure 5: Results of clustering logistic regression predictions (for the second set of variables) using k-means clustering, with 5 and 10 clusters. A histogram of the cluster sizes is on the left and a bar plot of the mean predicted response probability and the true response rate in each cluster is on the right.

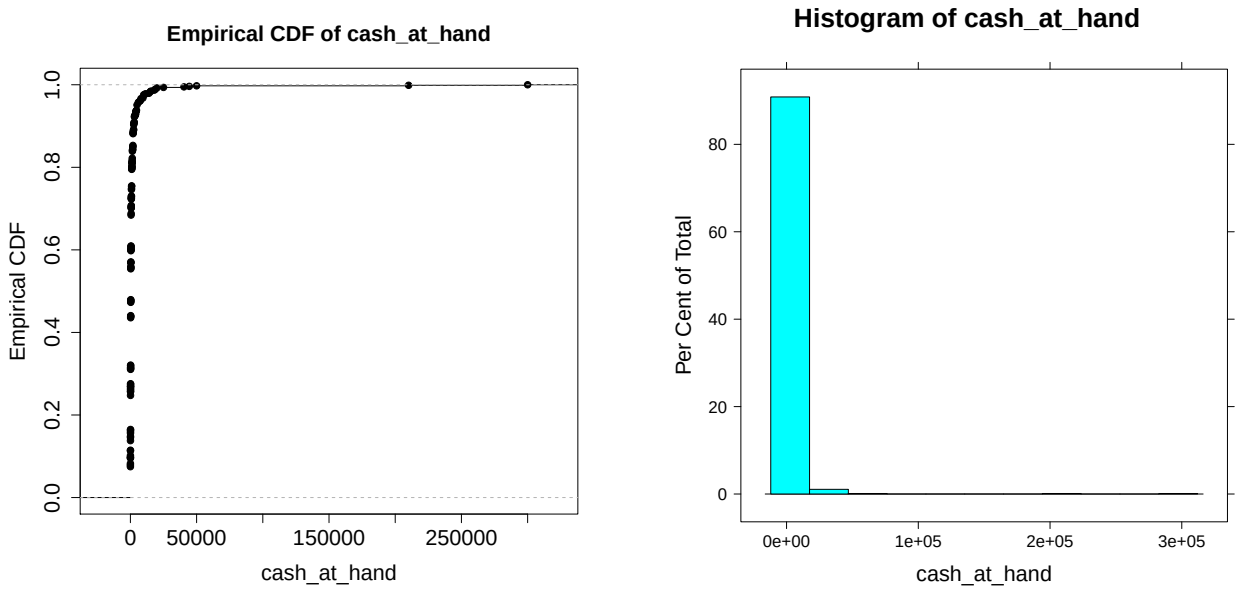


Figure 6: Empirical CDF and histogram of cash-at-hand variable. We see that this has a highly skewed distribution with some entries being many times larger than the vast majority of the entries.

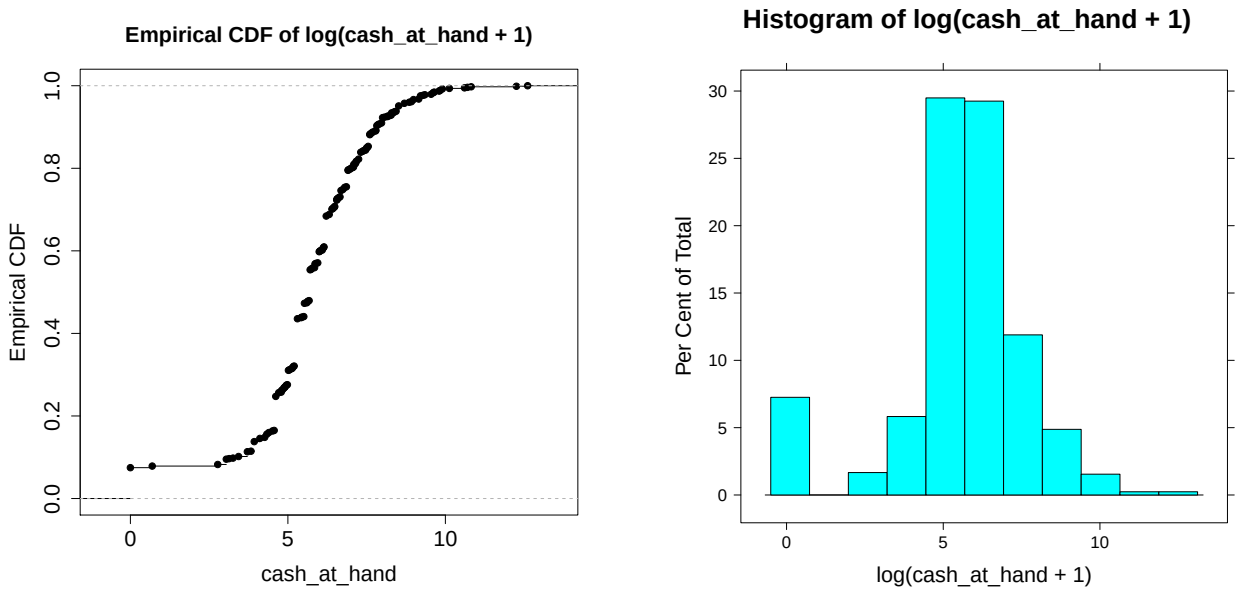


Figure 7: Empirical CDF and histogram of the logarithm of the cash-at-hand variable plus one. We see that this has a much less skewed distribution than the original cash-at-hand variable.

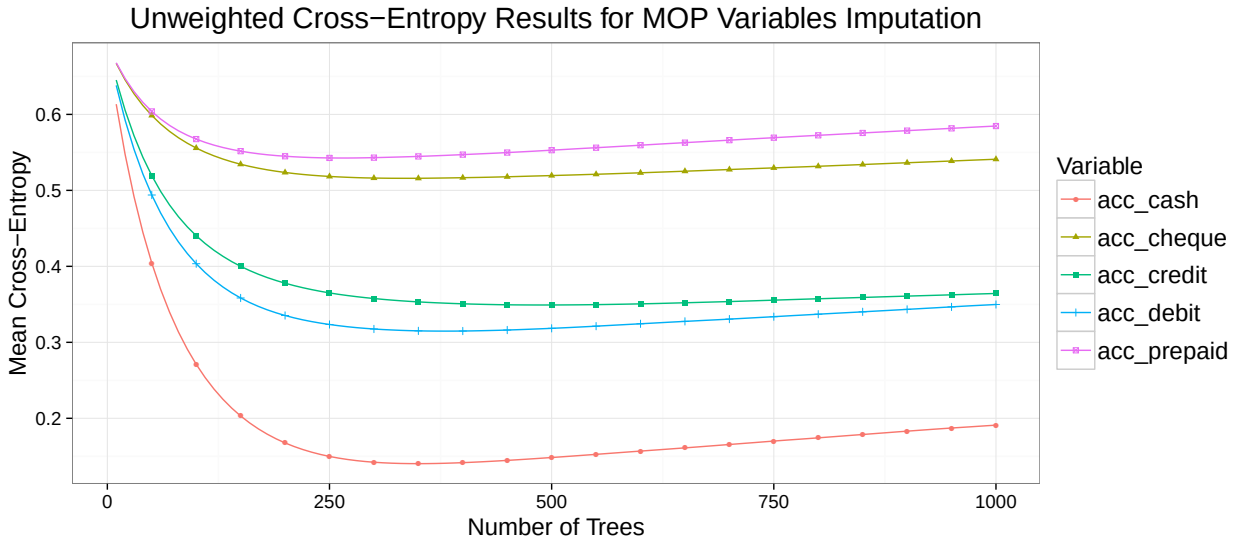


Figure 8: The unweighted mean cross-entropy for MOP variables imputation, as a function of the number of trees, for each of the MOP acceptance variables.

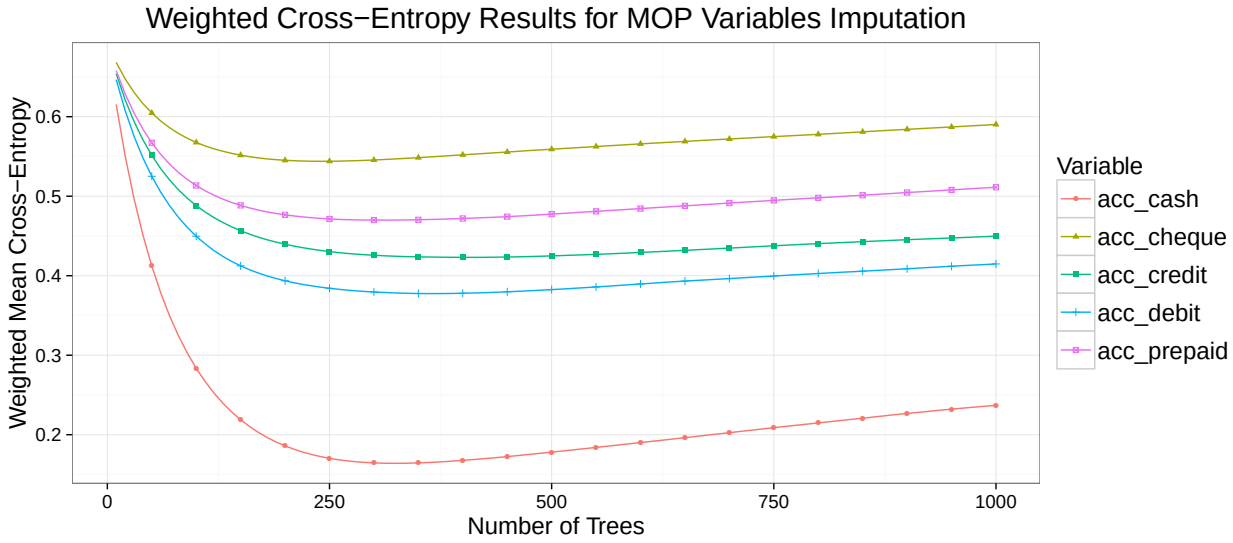


Figure 9: The weighted cross-entropy for MOP variables imputation, as a function of the number of trees, for each of the MOP acceptance variables.

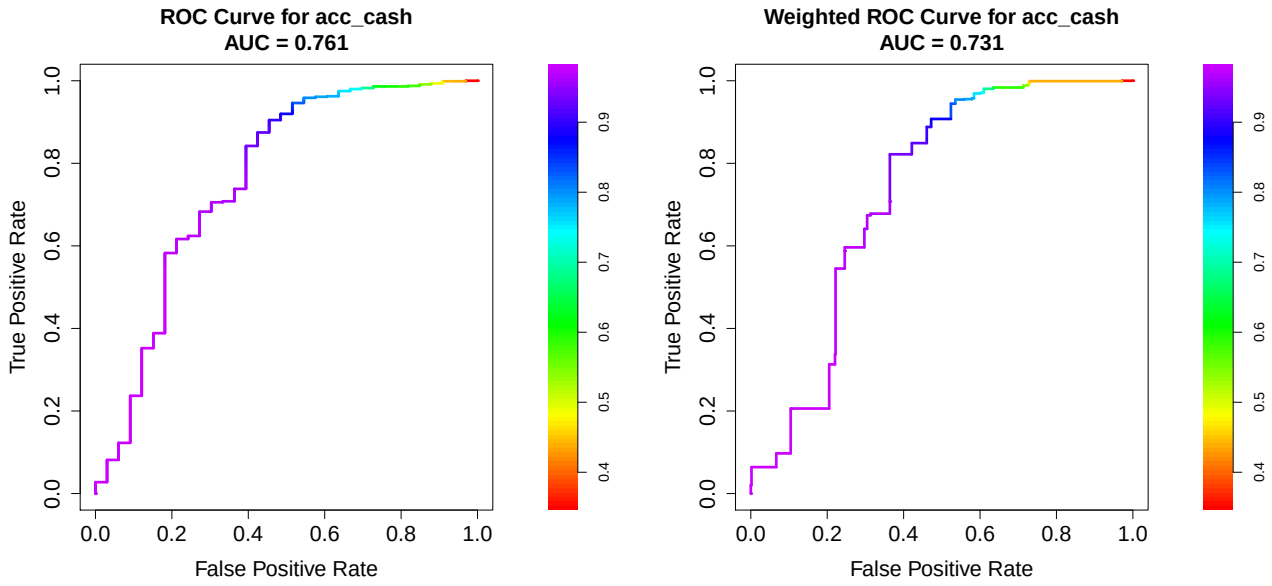


Figure 10: Unweighted and weighted ROC curves for imputation of the "accept cash" variable. The colour indicates the minimum threshold at that point on the ROC curve to classify as "yes" for accept cash.

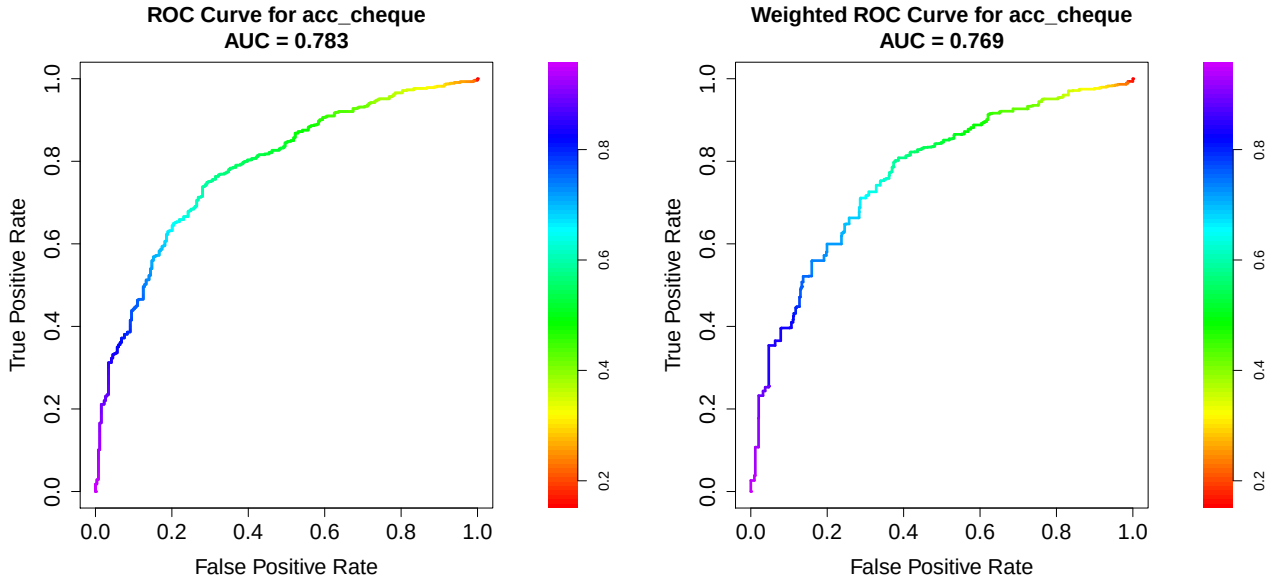


Figure 11: Unweighted and weighted ROC curves for imputation of the "accept cheque" variable. The colour indicates the minimum threshold at that point on the ROC curve to classify as "yes" for accept cheque.

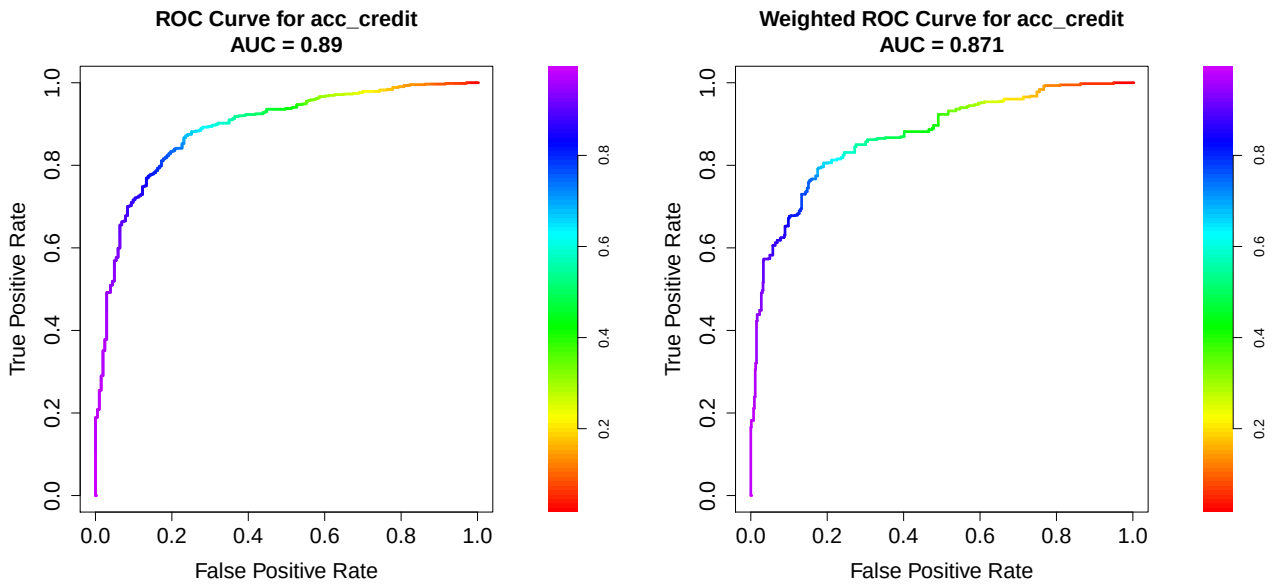


Figure 12: Unweighted and weighted ROC curves for imputation of the "accept credit" variable. The colour indicates the minimum threshold at that point on the ROC curve to classify as "yes" for accept credit.

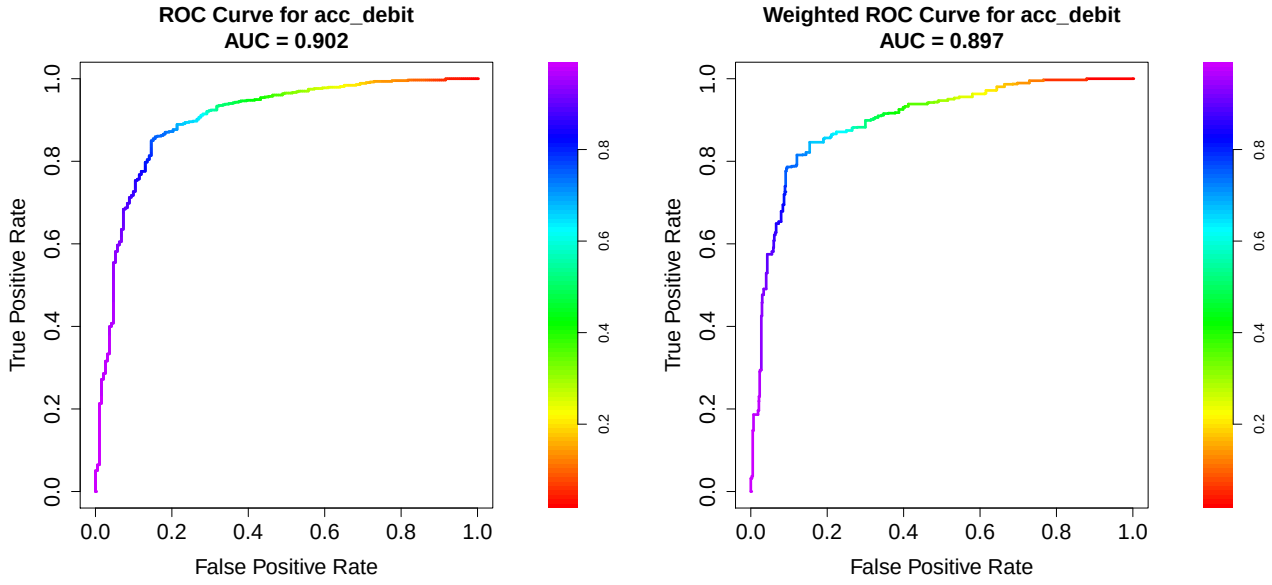


Figure 13: Unweighted and weighted ROC curves for imputation of the "accept debit" variable. The colour indicates the minimum threshold at that point on the ROC curve to classify as "yes" for accept debit.

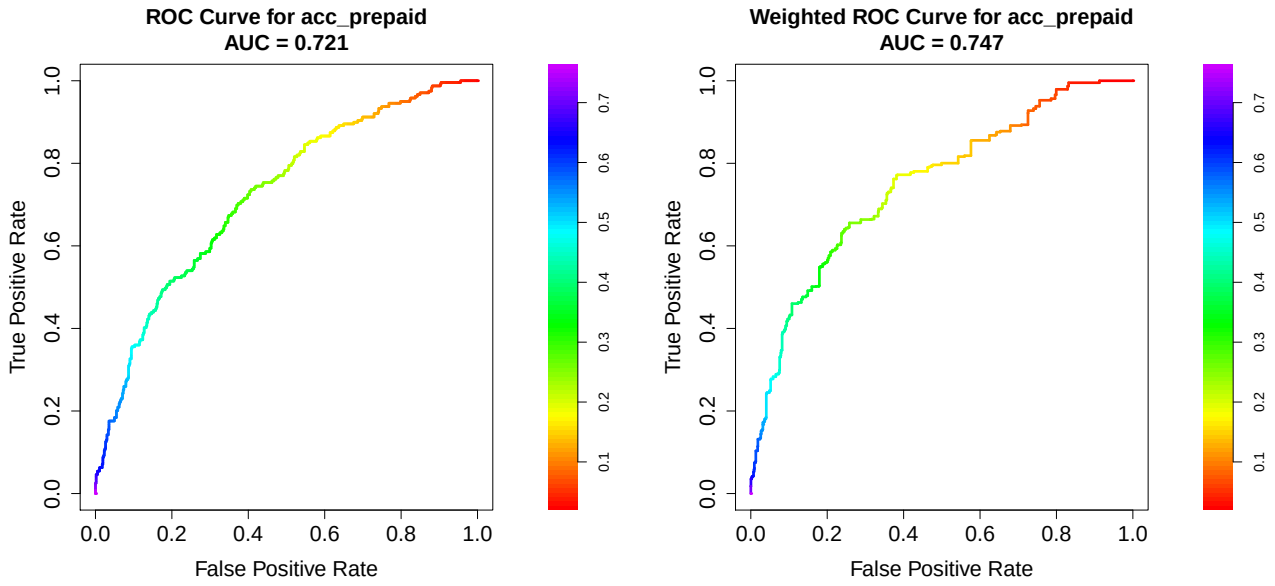


Figure 14: Unweighted and weighted ROC curves for imputation of the "accept prepaid" variable. The colour indicates the minimum threshold at that point on the ROC curve to classify as "yes" for accept prepaid.

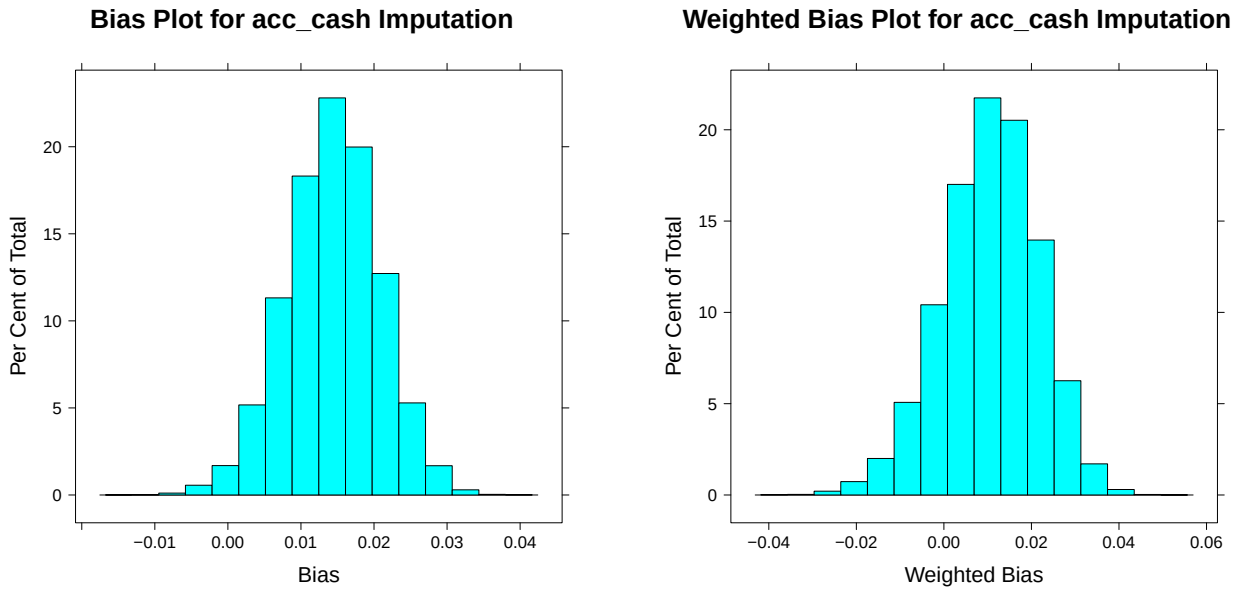


Figure 15: Histograms showing unweighted and weighted bias of imputation for the "accept cash" variable.

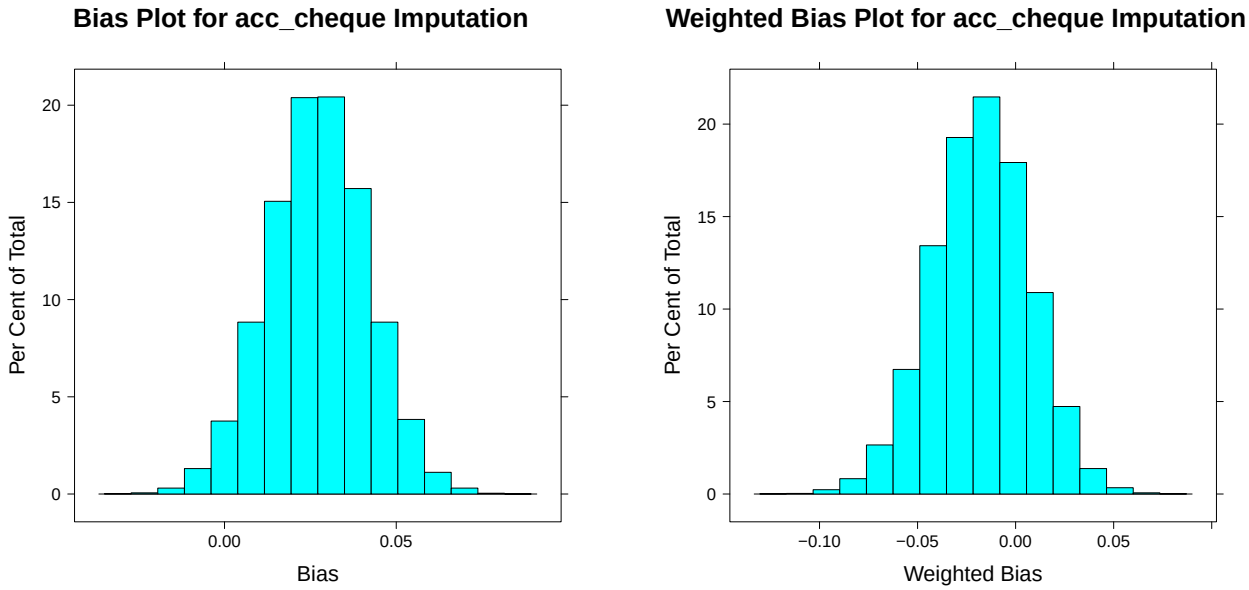


Figure 16: Histograms showing unweighted and weighted bias of imputation for the "accept cheque" variable.

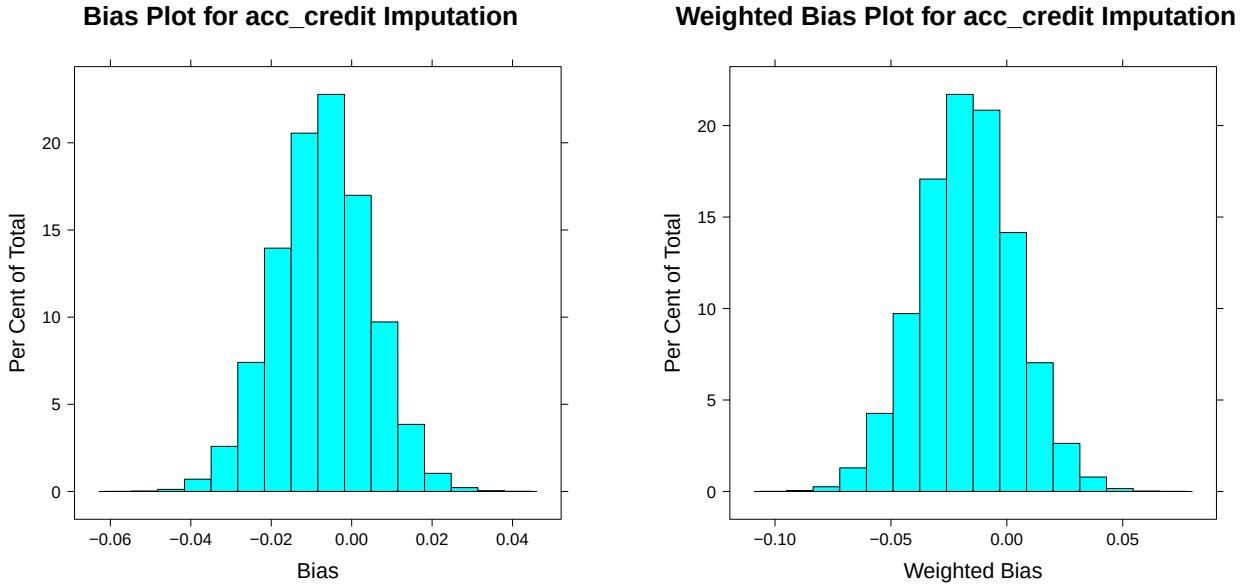


Figure 17: Histograms showing unweighted and weighted bias of imputation for the "accept credit" variable.

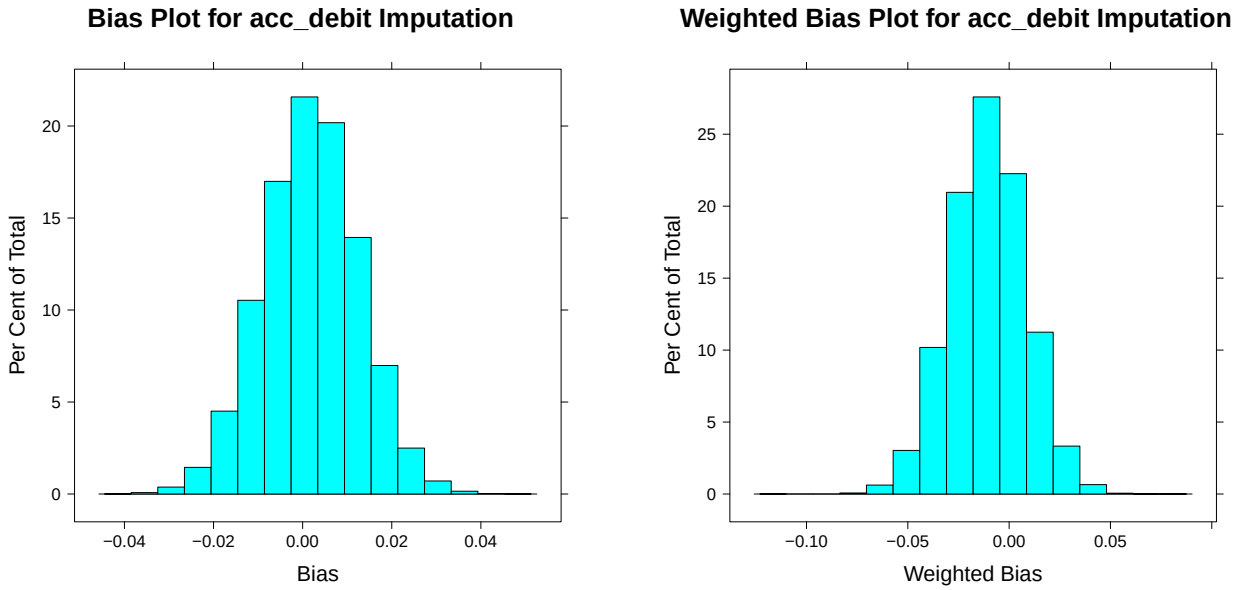


Figure 18: Histograms showing unweighted and weighted bias of imputation for the "accept debit" variable.

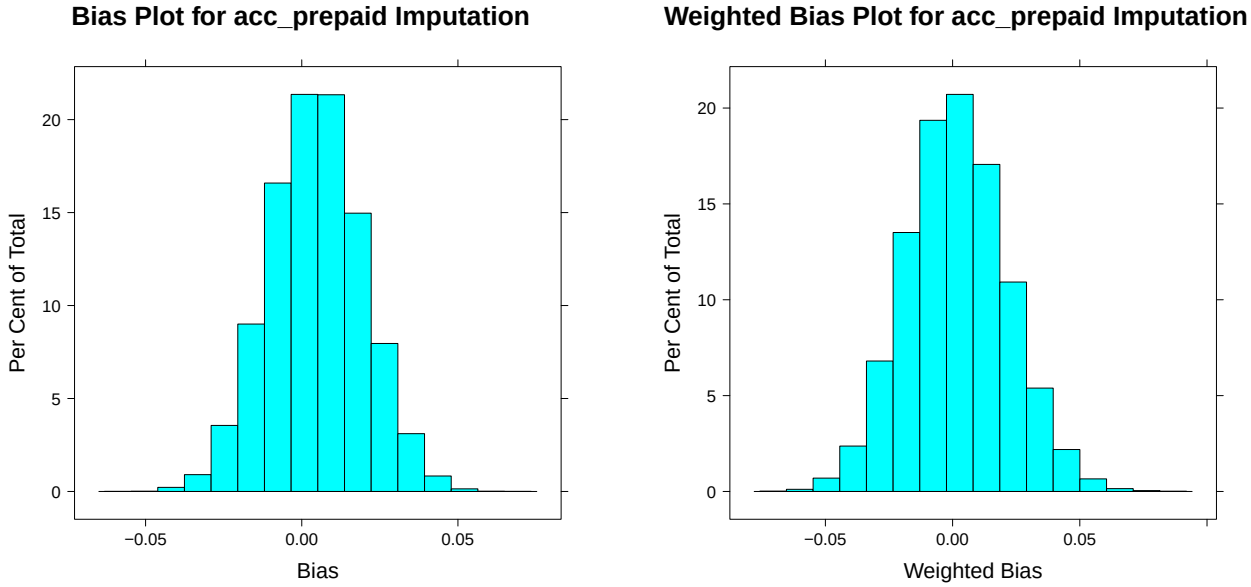


Figure 19: Histograms showing unweighted and weighted bias of imputation for the "accept prepaid" variable.

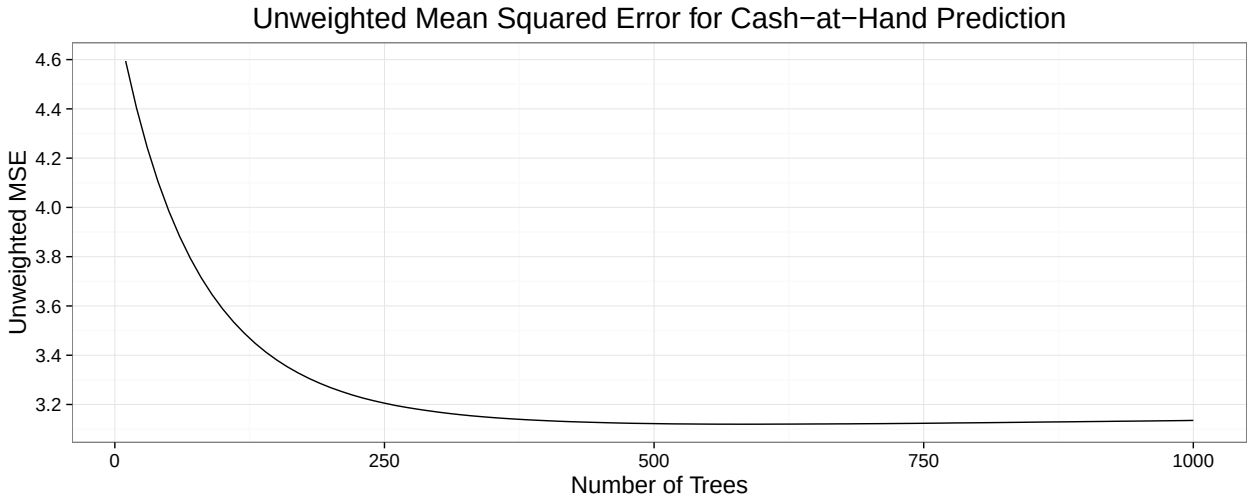


Figure 20: The unweighted MSE for cash-at-hand imputation (with logarithmic transformation), as a function of the number of trees.

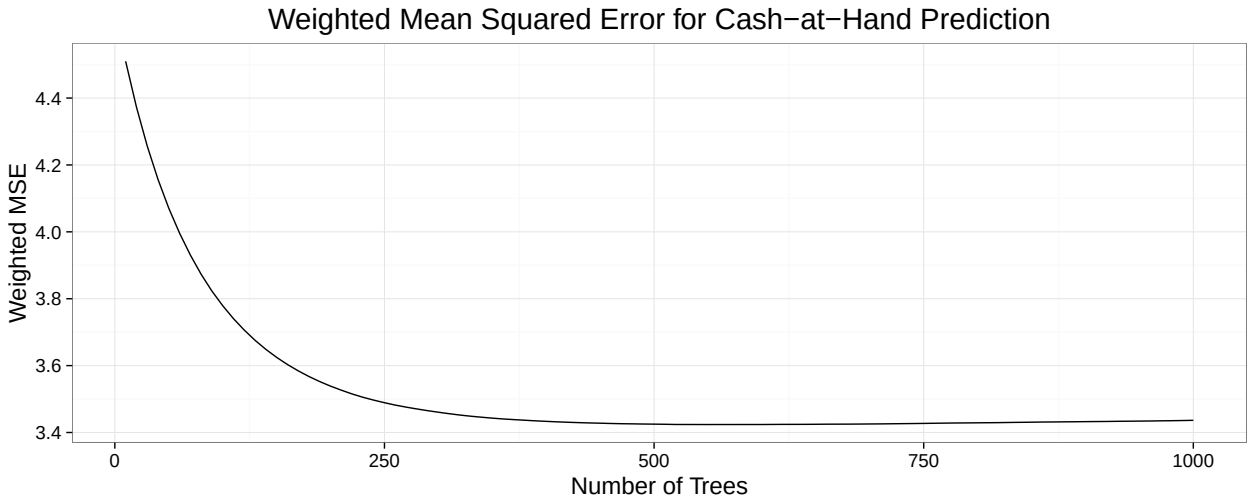


Figure 21: The weighted MSE for cash-at-hand imputation (with logarithmic transformation), as a function of the number of trees.

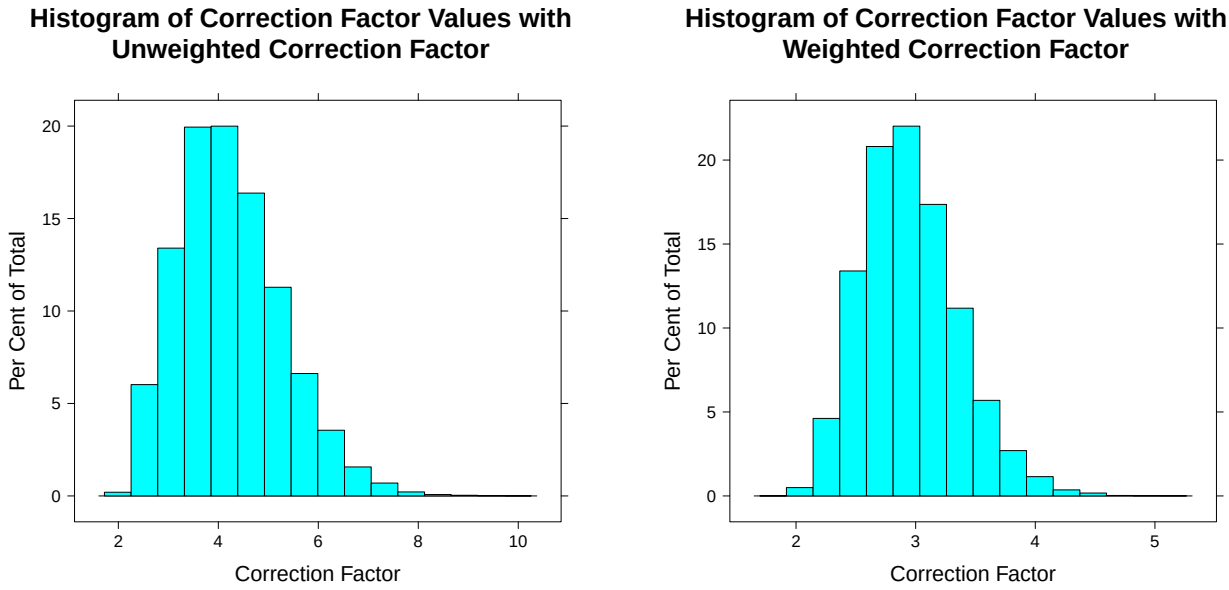


Figure 22: Histograms of correction factor values, for the bootstrap samples for cash-at-hand imputation. The histogram on the left is for the unweighted method-of-moments correction factor; the one on the right is for the weighted method-of-moments correction factor.

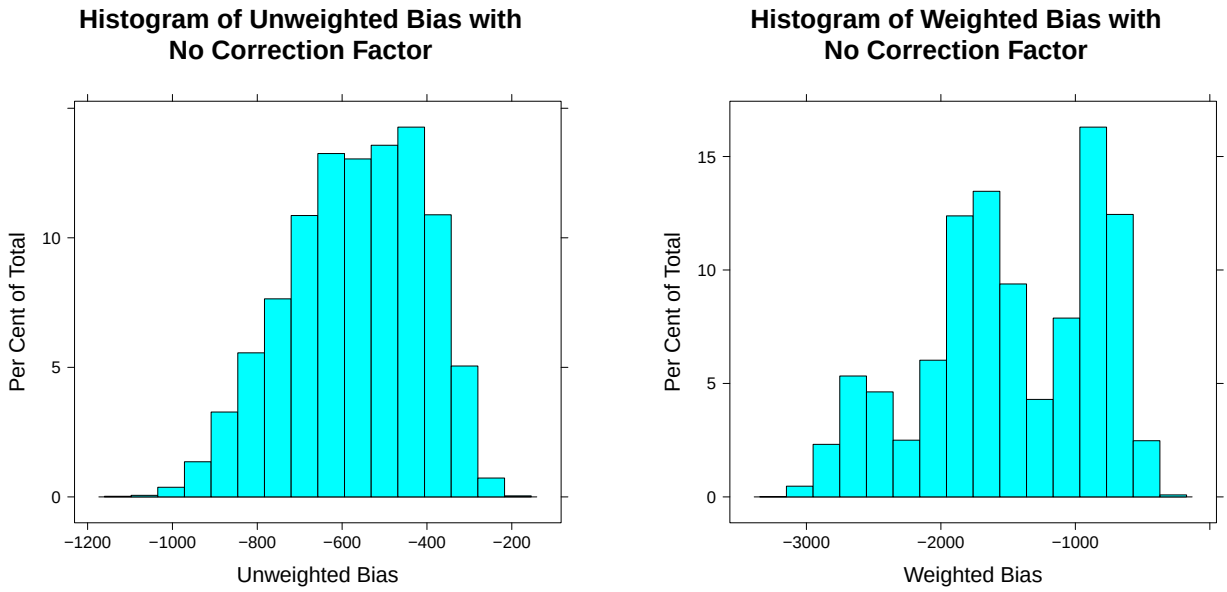


Figure 23: Histograms of bias (unweighted on left, weighted on right) with no correction factor, for the bootstrap samples for cash-at-hand imputation.

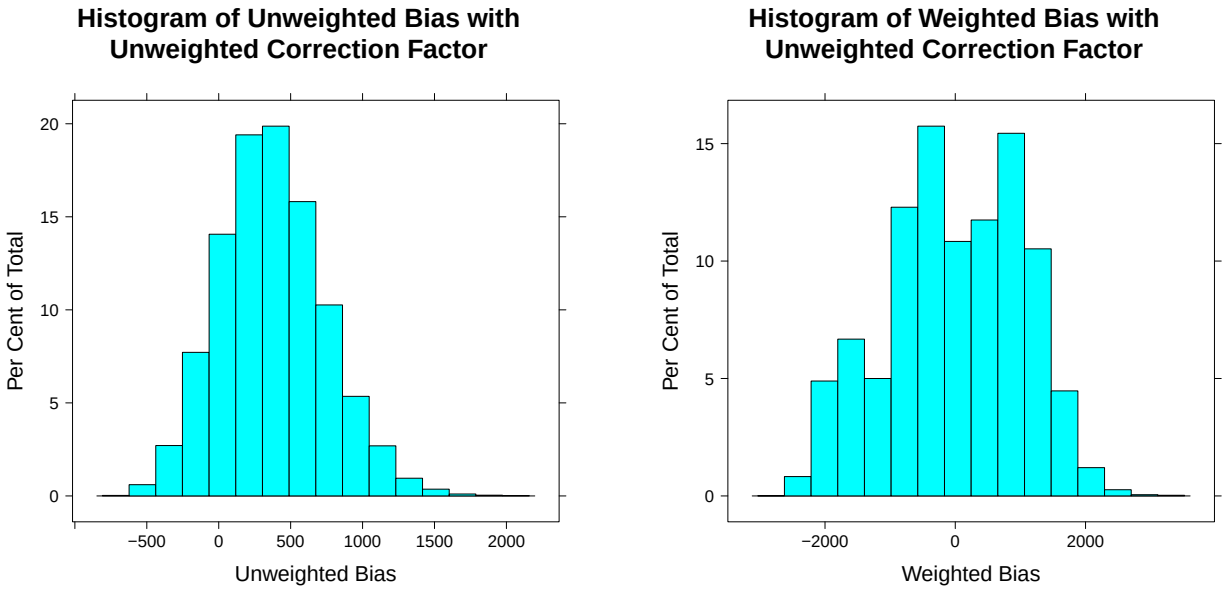


Figure 24: Histograms of bias (unweighted on left, weighted on right) with the unweighted method-of-moments correction factor, for the bootstrap samples for cash-at-hand imputation.

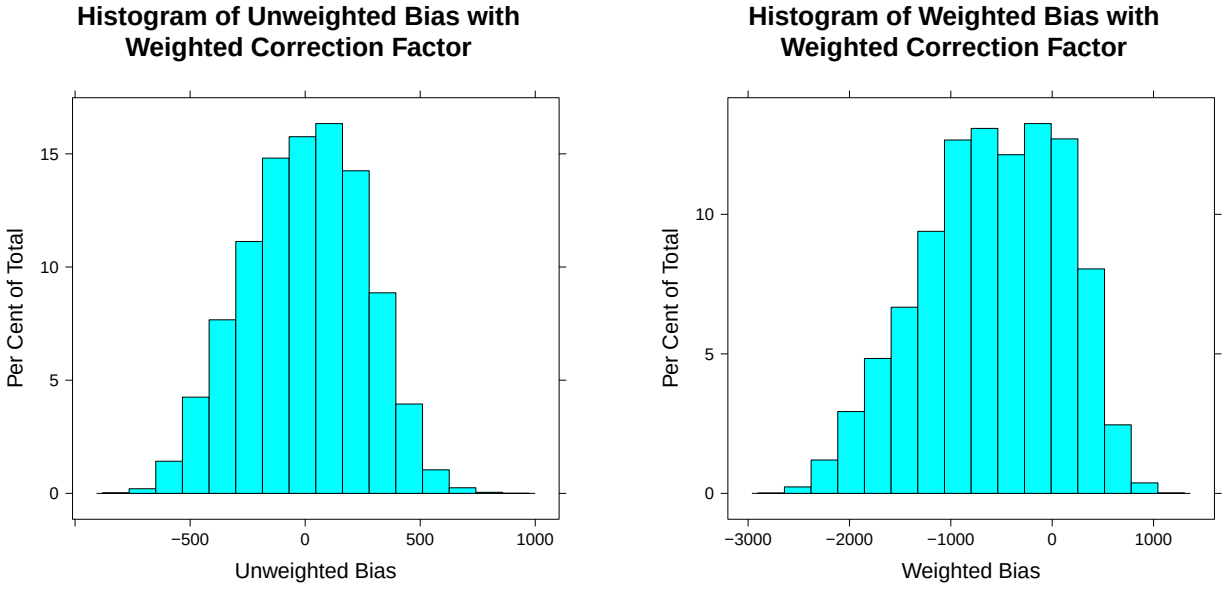


Figure 25: Histograms of bias (unweighted on left, weighted on right) with the weighted method-of-moments correction factor, for the bootstrap samples for cash-at-hand imputation.

Appendix B Tables

Entry	Sample Size	Respondents	Response Rate
All	34,252	799	0.023
All (without those excluded by address checks and screening phone calls)	27,384	799	0.029
Phase 1 All	27,011	650	0.024
Phase 1 (without those excluded by address checks and screening phone calls)	24,488	650	0.027
Phase 1 Wave 1	9,233	256	0.028
Phase 1 Wave 2 All	9,347	351	0.038
Phase 1 Wave 2A	4,607	169	0.037
Phase 1 Wave 2B	3,697	151	0.041
Phase 1 Wave 2C	1,043	31	0.030
Phase 1 Wave 3 All	5,907	43	0.007
Phase 1 Wave 3A	3,699	33	0.009
Phase 1 Wave 3B	2,208	10	0.005
Phase 2	7,241	149	0.021

Table 1: Table showing sample sizes and response rates for different parts of the survey. Businesses in stratum C, stratum jumpers to or from the C stratum, and businesses contacted by personal visits and regional representatives are not shown in this table.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	−5.4925	0.2789	−19.69	0.0000
phonenumberPresentTRUE	1.6425	0.2549	6.44	0.0000
faxnumberPresentTRUE	0.1680	0.0829	2.03	0.0428
webaddressPresentTRUE	0.4948	0.0911	5.43	0.0000
dbprescreenscoreHigh Risk	−0.5384	0.1050	−5.13	0.0000
dbprescreenscoreMissingValue	−0.2365	0.0986	−2.40	0.0165
age	0.0089	0.0021	4.25	0.0000
regionBC	0.2487	0.1243	2.00	0.0455
regionON	0.2874	0.1190	2.42	0.0157
regionPR	0.1999	0.1210	1.65	0.0986
regionQC	0.5249	0.1182	4.44	0.0000
hasPoboxTRUE	0.5138	0.1197	4.29	0.0000
primarynaics245	0.2098	0.1057	1.99	0.0471
primarynaics272	−0.6168	0.1000	−6.17	0.0000
primarynaics281	0.0688	0.1010	0.68	0.4955

Table 2: Table showing logistic regression summary for response probability prediction, including each of the coefficient values and significance p-values.

	acc_cash	acc_cheque	acc_credit	acc_debit	acc_prepaid
phonenumberPresentTRUE	0.6328	0.0405	0.1013	0.2133	0.3585
faxnumberPresentTRUE	0.0653	0.0001	0.0000	0.0000	0.2209
webaddressPresentTRUE	0.4857	0.7252	0.0000	0.0096	0.5523
dbprescreenscoreHigh Risk	0.6006	0.0371	0.0000	0.0000	0.2098
dbprescreenscoreMissingValue	0.2366	0.1556	0.0008	0.0002	0.0658
age	0.0311	0.0000	0.0061	0.0020	0.0422
regionBC	0.4881	0.8519	0.6102	0.9089	0.7972
regionON	0.1905	0.1173	0.4762	0.4576	0.3477
regionPR	0.1714	0.0251	0.1088	0.4606	0.8523
regionQC	0.0431	0.5922	0.6648	0.9797	0.0048
hasPoboxTRUE	0.9880	0.0007	0.5653	0.9305	0.0036
primarynaics245	0.5342	0.1636	0.0946	0.0455	0.8048
primarynaics272	0.4079	0.0000	0.1168	0.1178	0.4744
primarynaics281	0.1366	0.0427	0.0000	0.0000	0.0000

Table 3: Logistic regression significance (p-value) in terms of predicting select variables of interest (here the MOP acceptance variables), when each of the predictor variables is considered on its own.

	Number of Entries in Cluster	Number of Responses in Cluster	Proportion of Entries in Cluster (%)	Predicted Response Rate in Cluster (%)	Empirical Response Rate in Cluster (%)
1	11,101	93	32.410	0.875	0.838
2	12,884	250	37.615	1.973	1.940
3	6,794	237	19.835	3.460	3.488
4	2,886	178	8.426	5.551	6.168
5	587	41	1.714	8.920	6.985

Table 4: Table showing response rates within each cluster with a total of 5 clusters.

	Number of Entries in Cluster	Number of Responses in Cluster	Proportion of Entries in Cluster (%)	Predicted Response Rate in Cluster (%)	Empirical Response Rate in Cluster (%)
1	3,620	20	10.569	0.436	0.552
2	6,814	59	19.894	1.058	0.866
3	8,227	143	24.019	1.671	1.738
4	5,829	136	17.018	2.402	2.333
5	3,784	129	11.048	3.169	3.409
6	2,672	99	7.801	4.074	3.705
7	1,734	96	5.062	5.167	5.536
8	1,039	80	3.033	6.443	7.700
9	415	31	1.212	8.348	7.470
10	118	6	0.345	11.667	5.085

Table 5: Table showing response rates within each cluster with a total of 10 clusters.

Variable	Present Rate (%)	Missing Rate (%)	Empirical Acceptance Rate (%)
Cash at Hand	92.3	7.7	N/A
Accept Cash	98.8	1.2	96.0
Accept Cheque	97.3	2.7	67.7
Accept Credit Card	98.3	1.7	75.5
Accept Debit Card	98.1	1.9	76.7
Accept Prepaid Card	95.6	4.4	29.7

Table 6: Table showing present and missing rates for various questions in the survey. The empirical acceptance rate is the proportion of retailers who answered "yes" to the MOP acceptance question out of retailers who responded to the question.

	Number of Trees	Cross-Entropy	Weighted Cross-Entropy	AUROC	Weighted AUROC	Bias	Weighted Bias	Bootstrap SD
Accept Cash	320	0.14	0.16	0.76	0.73	0.01	0.01	0.07
Accept Cheque	240	0.52	0.54	0.78	0.77	0.03	−0.02	0.16
Accept Credit Card	410	0.35	0.42	0.89	0.87	−0.01	−0.02	0.14
Accept Debit Card	360	0.31	0.38	0.90	0.90	0.00	−0.01	0.15
Accept Prepaid Card	310	0.54	0.47	0.72	0.75	0.00	0.00	0.18

Table 7: Table showing MOP acceptance variable imputation results, with the number of trees used to obtain these results. The cross-entropy, area under the ROC curve (AUROC or ROC AUC), and bias are shown (both unweighted and weighted versions, unweighted unless otherwise specified). The "Bootstrap SD" is the average standard deviation of each entry among the bootstrap samples in the bootstrap imputation.

	No Correction Factor	Unweighted Correction Factor	Weighted Correction Factor
CF Value	1.00	4.24	2.95
CF Value SD	0.00	1.04	0.40
Unweighted Bias	−1,481.42	−6.14	−590.57
Unweighted Bias SD	641.41	1,038.05	687.91
Weighted Bias	−568.59	371.90	−1.86
Weighted Bias SD	157.79	365.02	255.33
LogMSE	3.12	5.17	4.28
LogMSE SD	0.30	0.83	0.51
Weighted LogMSE	3.42	5.56	4.66
Weighted LogMSE SD	0.42	1.00	0.70

Table 8: Table showing results with various correction factors for the back-transformation of the imputed cash at hand. In this table, SD means standard deviation and LogMSE refers to the mean squared error after a logarithmic transformation $x \mapsto \log(x+1)$ is performed.

Appendix C List of Predictor Variables for Imputation

The following variables are used in generating predictions for the imputed values (for the variables of interest, namely, cash-at-hand and the MOP acceptance variables):

Variable	Description
`age`	Age of the business, based on D&B (zero if missing).
`ageMissing`	Binary indicator if the age variable is missing (based on D&B).
`certificate`	Did the business request a certificate of appreciation (from the survey)?
`contacttitlePresent`	Is a contact title for the business owner present in the D&B dataset?
`dbprescreenscore`	The D&B prescreen score.
`detailed_report`	Did the business request a copy of the detailed report (from the survey)?
`doingbusinessasPresent`	Is the "doing business as" field present in the D&B dataset?
`durables`	Is the business in the durables sector (based on the D&B NAICS variable)?
`employeesatthislocation`	Number of employees at the particular location (from D&B).
`facilitysizesqft`	Facility size in square feet (from D&B).
`faxnumberPresent`	Is a fax number present for the business in the D&B dataset?
`final_report`	Did the business request a copy of the final report (from the survey)?
`gasoline`	Is the business in the gasoline sector (based on the D&B NAICS variable)?

genretail	Is the business in the general retail sector (based on the D&B NAICS variable)?
greg_employees	Number of employees (based on the survey).
greg_sales	Total sales (based on the survey).
greg_wages	Wages (based on the survey).
greg_year_of_founding	Year of founding (based on the survey).
groceries	Is the business in the grocery sector (based on the D&B NAICS variable)?
has_safe	Does the business have a safe (from the survey)?
insurance_cft	Does the business have insurance for counterfeit bank notes (from the survey)?
insurance_credit	Does the business have insurance for credit cards (from the survey)?
insurance_debit	Does the business have insurance for debit cards (from the survey)?
insurance_no	Does the business not have insurance for any method of payment (from the survey)?
insurance_premium	Insurance premium paid by the business (based on the survey).
insurance_prepaid	Does the business have insurance for prepaid cards (from the survey)?
ipad_draw	Did the business request to be placed in the draw for an iPad Air 2 (from the survey)?
is_cati	Did the business respond to the survey via a CATI phone call?
is_online	Did the business respond to the survey online?
isexporter	Is the business an exporter (from D&B)?

isimporter	Is the business an importer (from D&B)?
ismanufacturing	Is the business in manufacturing (from D&B)?
latitude	Latitude of the business (from D&B).
location_scope	Is the response for one particular sales location or the entire business (from the survey)?
locationtype_EDITED	Edited location type based on the survey (single location, branch, or headquarters).
longitude	Longitude of the business (from D&B).
MAIL	Did the business respond to the survey via mail?
netincomeusdollarsmillion	Net income in US dollars, millions (from D&B).
no_incentive	Did business not request any incentive (based on the survey)?
no_of_newvarsmissing	Number of new variables (based on survey data, such as ratios of survey variables) missing.
numemp	Stratum for the number of employees in the business (based on D&B).
ownsrents	Does the business own or rent its location (from D&B)?
pays_cit_fees	Does the business pay cash-in-transit (CIT) fees?
phase	Phase the business was contacted in.
phonenumberPresent	Is a phone number present for the business in the D&B dataset?
primarynaics_EDITED	Two-digit primary NAICS code for the business, from D&B and edited based on survey responses.
primarynaics_FRAME	Two-digit primary NAICS code for the business, from D&B.

primarynaics3	Three-digit primary NAICS code for the business, from D&B.
primarystate	Primary province of business (from D&B).
primaryussiccode	Primary USSI code for the business (from D&B).
regioncode_EDITED	Edited region code for the business (original based on the province in D&B, edited based on the survey response).
regioncode_FRAME	Region code for the business (based on the province in D&B).
restaurants	Is the business in the restaurants sector (based on the D&B NAICS variable)?
revenuecanadiandollarsmillion	Revenue in Canadian dollars, millions (from D&B).
revenueusdollarsmillion	Revenue in US dollars, millions (from D&B).
sales	Total annual sales made by the business (from the survey).
semidurables	Is the business in the semidurables sector (based on the D&B NAICS variable)?
services	Is the business in the services sector (based on the D&B NAICS variable)?
size_EDITED	Edited size stratum of the business, based on the survey and D&B.
size_FRAME	Size stratum of the business in the survey frame (based on D&B and missing entries filled in).
size_REPORTED	Size stratum of the business based on survey response.
stratum_jumper	Is the business a stratum jumper (computed based on the survey and D&B)?
totalemployees	The total number of employees in the business (from D&B).

wave	Which survey wave was the business contacted in?
webaddressPresent	Is a web address present for the business in D&B?
webinar	Did the business request a webinar invite (from the survey)?
yearoffounding	Year of founding of the business (from D&B).

These variables are of different types, some being numeric and others categorical. For many of the variables, some of the entries are missing.