Draft Research Report

Daniëlle Remmerswaal

15-11-2021

Contents

1. Introduction

1.1. Problem

In many cases when official statistics are produced, only a subset from a population is selected. An example is a statistic on the energy consumption per economic sector. From a register containing all buildings in the Netherlands, only establishments are selected, and dwellings discarded. Information on energy consumption per building and on the type of building and, if applicable, the economic sector, is recorded in multiple data sources. To produce the statistic, the data sources, which are not error-free, are combined. In a composite dataset, a combination of registers and surveys on the same statistic, errors become noticeable. In this type of situation we can distinguish between selection errors (regarding the type of building) and classification errors (of the economic sectors). Possible causes of errors are typos during data entry, mismatching definitions (between the data collectors and the CBS researchers), or not having up-to-date answers (Groen, 2012). It is important to estimate and correct errors as to identify and minimize bias in statistics.

1.2. short literature (from proposal)

Several methods have been developed to estimate and correct for errors in categorical data. To estimate errors in data at one time-point, an important method is the latent variable model approach by Biemer (2011), with Latent Class Analysis (LCA) being specifically for errors in categorical variables. For categorical data measured multiple times, Hidden-Markov Models (HMM) can be used (Pavlopoulos & Vermunt, 2015; Pankowska, 2020). In survey research, systematic (caused by survey design) and random measurement error can be simultaneously studied with the multitrait-multimethod (MTMM) approach (Cernat and Oberski, 2018). The latent variable model is applicable on composite datasets under the assumption that the sources are locally independent. Those measurements are then used as the 'indicators' of a latent, unobserved, 'true' variable. The Multiple Imputing Latent Class (MILC) method uses a latent class model to can also be used to correct the true value for categorical errors, and sets the amount of latent classes equal to the number of categories in the variable of interest (Boeschoten, 2019).

1.3. Gap (from proposal)

For the estimation of categorical errors of one particular type, or in combination with 'method error' (MTMM situation), multiple methods are available to researchers. However, on approaches for situations where misclassifications can be identified as being caused by different error types (e.g. selection or classification error), not much research has been conducted. An unexplored potential solution for this situation, is the use of a latent class tree in combination with the MILC method. In this thesis this new latent class tree approach to the MILC method will be compared to the "normal" MILC method, in which an extra class is added for the wrongly selected units. The structure crucial to the two methods is visualised in section 2.2. with the described energy consumption per economic sector example.

1.4. Research Question, hypotheses and expectations (from proposal)

This project aims to investigate whether the combination of the MILC-method with a latent class tree approach is an appropriate alternative to the MILC-method, to measure and correct for a combination of selection and classification errors. How well those approaches perform to estimate and correct the errors, will be investigated with a simulation study. Expectation is that the latent class tree structure gives more accurate estimates and more insight in the structure of data with those two types of errors, but is less straightforward to apply.

To perform the simulations and analyse the results, the freely-available opensource statistical software R will be used. In particular, the package poLCA will be used for the data simulation and the latent class analysis. (kopie zin)

2. Literature

2.1. LCA

Latent Class models are finite mixture models (fmm). To be more specific, latent class analysis is binomial fmm, latent profile analysis is a gaussian fmm.

- Assumptions Two key model assumptions (X is the latent class variable)

1. The mixture assumption.
   Joint distribution mixture of 2 class-specific distributions: P(scorepattern)=P(x=1)P(scorepattern|X=1)+ P(x=2)P(scorepattern|X=2). Mixture of C classes.

2. The local independence assumption:
   Within class X=x, responses are independent. P(scorepattern|X=1)=P(Q1=A1|X=1) * P(Q2=A2|X=1)* P(Q3=A3|X=1). K manifest variables.

The calculations of the posterior probabilities are based on the mixture and local independence assumption.

Classification works via modal assignment: assign unit to latent class with highest P(X=x|Y=y)=P(Class|scorepattern).

2.2. Total error framework for Registers

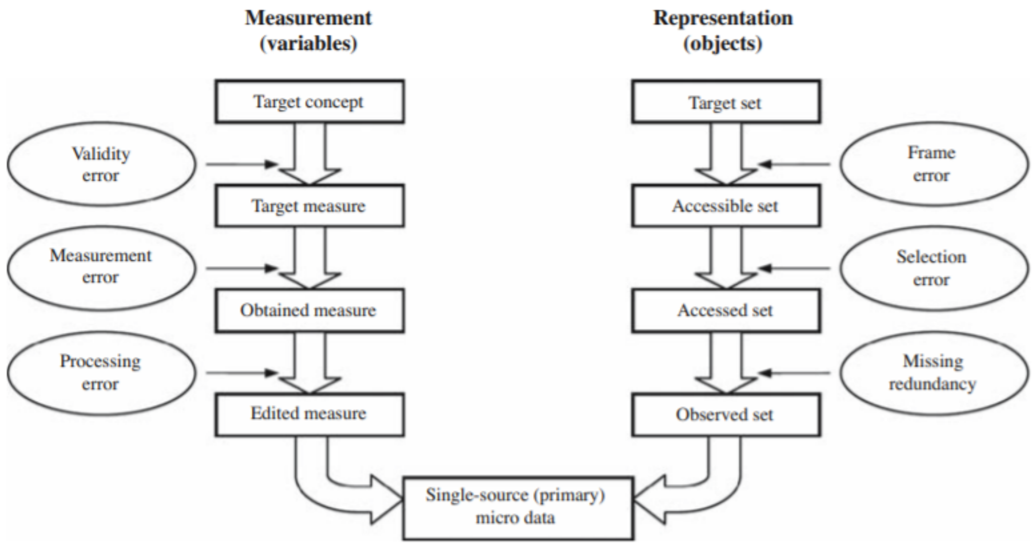
Total Survey Error is "any error arising from the survey process that contributed to the deviation of an estimate from its true parameter value" (Biemer, 2016). TSE all errors that can occur during design, collection, processing, and analysis of survey data. We can divide TSE in sampling and non-sampling errors. Sampling errors are due to sampling a sample instead of a whole population. non-sampling errors include measurements errors (specification(of concept in question), sampling frame, nonresponse, measurement, data processing, and modelling/estimation errors).

Registers are generally not created and maintained by statistical agencies themselves, but by external owners (e.g. governments or businesses) for administrative purposes. This can be problematic for statistical agencies because there can be errors without knowing. The household is an important statistical unit, but does not exist in administrative registers, and thus has to be created by statistical agencies. Errors are thus unavoidable.

Zhang (2009) developed a unit-error theory to provide "a framework for evaluating the statistical accuracy of these register-based household statistics." In general, there is a lack of theories to evaluate the quality of register statistics, especially as compared to survey methodology. Related is the flawed assumption that registers are error-free, which they are not. While sampling errors are of course absent, there are other non-sampling errors: "over- and under-coverage, lack of relevance, misclassification, delays and mistakes in the data registration process, inconsistency across the administrative sources, and not the least missing data." This is important as this allows for the "the conceptualization and measurement of the statistical accuracy in register statistics, which will enable us to apply rigorous statistical concepts such as bias, variance, efficiency and consistency, e.g. as one is able to do when it comes to survey sampling."

"The sources of error under Phase 1 of Zhang's model are a result of the initial data collection and processing, and will flow through into any use of the data in the production of a statistical output. Phase 2 errors relate to using these source data sets to produce a particular statistical output. They depend on the desired outputs and the design under consideration."(Reid et al.)

Zhang (2012) extended, or generalised, the Total Survey Error framework so that it would be applicable to administrative datasources.



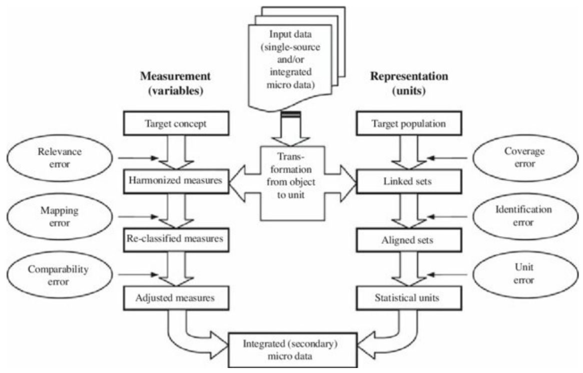


Figure 1: quality assessment framework phase 2 (Zhang 2012)

This framework also enables me to explain the difference between problems that can be solved with the multitrait-multimethod (MTMM) approach and with the MILC-tree method. MTMM is used for situations in survey research where multiple errors on the measurement side are studied. The MILC-tree method attempts to solve situations where there is one type of error on the measurement side (meaurement error, in our case misclassification) and one on the representation side (coverage error), as visible in the figure above called "quality assessment framework phase 1."

3. Methoden

poLCA uses EM and Newton-Raphson algorithms to maximize the latent class model log-likelihood function (Linzer and Lewis 201?). All considered covariates should be included in the latent class model, to avoid biased estimates of the relationship between an omitted covariate and the latent variable.

On simulation studies. "Simulation studies are computer experiments that involve creating data by pseudo-random sampling from known probability distributions. They are an invaluable tool for statistical research, particularly for the evaluation of new methods and for the comparison of alternative methods."(Morris, White and Crowther, 2018) ## 3.1. Analytical strategy Overview (from proposal)

The simulated dataset consists of the following features: . N=5.000 . 3 indicator variables with o two levels of selection error (5% and 20%) o two levels of classification error (5% and 20%) o generated using a tree structure with selection error first as specified in Figure 2. The proportions of each latent class (in the figure the numbers 1, 2A, 2B and 2C) add up to the total of 1. . 2 categorical covariates, one with strong relation with selection error, and weak relation with classification error, another with the reverse relations.

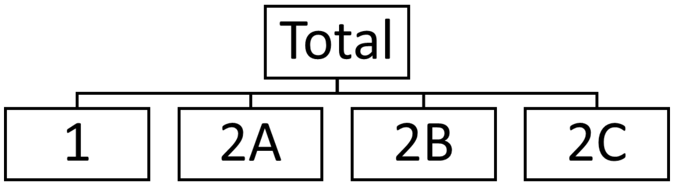


Figure 2: Figure 1: MILC

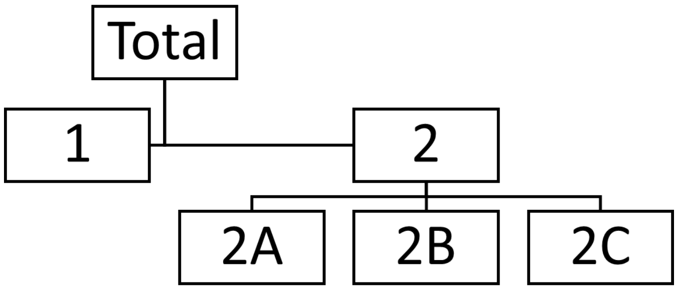


Figure 3: Figure 2: TREE MILC

Next, tree-MILC and MILC are both applied on the simulated datasets, and compared to the original simulated datasets on basis of their classification performance. The performance of the methods will be compared in terms of bias and confidence validity of the proportion of the classes, and their relations with the covariates.

. Variables: measurements of different datasources on same attribute . Classes: the categories in variable of interest . Variation: in the amount of selection and classification errors, and their relative proportions

To be able to evaluate the performance of the beforementioned extensions of the MILC method to estimate and correct errors, simulated data will be used since herewith ‘true values’ are available. To perform the simulations and analyse the results, the freely-available opensource statistical software R will be used. In particular, the package poLCA will be used for the data simulation and the latent class analysis.

The simulated composite datasets will contain both selection and classification errors (of varying levels). In the simulated dataset the latent classes are the groups that are potentially misselected or misclassified, and each variable contains information on the categories from one data source. Next, the MILC method and the MILC-method with a latent class tree approach are both applied on the simulated datasets. The performance of the classifications of both methods is evaluated by comparing it to the original simulated datasets. The accuracy of the methods will be compared in terms of bias and confidence validity of the statistic under investigation, the proportion of the classes, and on how well they estimate the (relative) size of the two errors.

the latent class tree approach allows for the selection errors (building type) and classification errors (economic sectors) to be distinguished. Without this tree structure, the misclassified buildings will simply get allocated to an additional class.

MILC

For this part, we will use MILC as developed by Boeschoten (2017). Step 1: take five bootstrap samples from a simulated dataset. Step 2: apply the latent class model to each of the bootstrapped samples, with as number of classes the number of categories in the variable of interest plus one for the misselected units. Step 3: Calculate posterior probabilities per profile (for each class and score pattern (variables and covariates)). Step 4: Impute dataset with the obtained posterior probabilies Step 5: Pool the 5 imputations with Rubins pooling rules. Step 6: Infer statistics of interest (proportions of classes and relations with covariates) and compare to original dataset

MILC with tree step

The same as above, except for step 2. The latent class model is applied twice in the following way: Tree step 2a: apply the latent class model to each of the bootstrapped samples with the number of classes being two. One class for the rightly, and one for the wrongly selected units (the first branching in figure 2, represented by the blocks with 1 and 2 in them). Tree step 2b: apply the latent class model to the rightly selected units only, with the number of classes being the number of categories in the variable of interest.

poLCA package

The poLCA package is an R software package for the estimation of latent class (regression) models for polytomous outcome variables. It is currently the only R package that allows the use of polytomous outcomes, and covariates. Other R packages for categorical outcome variables can only use binary outcomes.

Potential shortcomings: poLCA recodes NA's to a seperate category (zero's) and does not make use of imputation methods.

```r
#install.packages("poLCA") #install if not done yet
library(poLCA) #load the package into memory
```

```
## Loading required package: scatterplot3d
```

```
## Loading required package: MASS
```

- label problem "Because the latent classes are unordered categories, the numerical order of the estimated latent classes in the model output is arbitrary, and is determined solely by the start values of the EM algorithm. (Linzer and Lewis, 2011, poLCA, 4.6) (order depends on random starting values), A possibility to fix this is with poLCA.reorder(polcamodel$probs.start, c(1,3,2)) #example to switch order of second and third class. needed are the start probabilities of a model. So this is only a post-hoc fix, and not a general solution.

In latent class tree (LCT) modelling (see graphical representation below), the data is split and structured into classes by method of a sequential comparison of 1- and 2-class models. The top-down approach continues until the information criterion (e.g. BIC) no longer chooses 2-class models over 1-class models. The splits are performed based on the posterior class membership probabilities of each new class (child nodes) conditional on the class before the splitting (parent node). Here, the label switching problem is prevented by assigning the number to to the larger class each time (in the picture the left branch) and the smaller class number 2

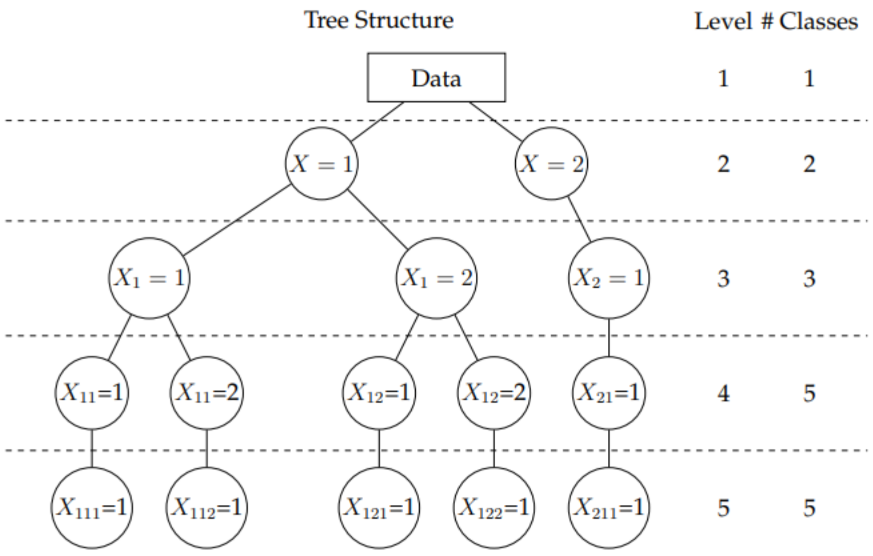


Figure 2.1: Graphical example of a LCT

(right branch).

Basics of poLCA simulation.

```
basic.sim <- poLCA.simdata(
  N=5000, #standard is 5000
  nclass = 3, #nr of latent classes, standard is 2
  ndv = 4, #nr of indicator variables, standard is 2
)
basic.sim$P
```

```
## [1] 0.32734494 0.08905452 0.58360054
```

```
head(basic.sim$dat) #the number of categories per indicator variable is drawn randomly
```

```
##   Y1 Y2 Y3 Y4
## 1  2  1  3  3
## 2  2  1  4  3
## 3  3  2  4  2
## 4  2  2  3  3
## 5  3  2  3  3
## 6  3  1  1  2
```

- proportions per class If not specified, the proportions per class are specified randomly (like in the example above). P can be specified with the proportions for each of the latent classes.

```
library(poLCA)
basic.sim2 <- poLCA.simdata(N=5000,nclass=3,ndv=4,P=c(0.4, 0.3, 0.3))
basic.sim2$P
```

```
## [1] 0.4 0.3 0.3
```

However, the specified proportions of the classes are 'overruled' when continuous covariates are present.

```
#two covariates added
set.seed(123)
basic.sim3 <- poLCA.simdata(N=5000,nclass=3,ndv=4,P=c(0.4, 0.3, 0.3), niv = 2)
head(basic.sim3$dat) #four manifest variables and two covariates
```

```
##   Y1 Y2 Y3 Y4         X1         X2
## 1  1  3  2  1  0.8377870  1.3424605
## 2  2  4  2  5  0.1533731  0.1235129
## 3  2  1  2  4 -1.1381369  0.8581478
## 4  1  4  2  5  1.2538149 -1.4054279
## 5  3  1  2  5  0.4264642 -0.0144571
## 6  2  4  3  1 -0.2950715  0.5455757
```

```
basic.sim3$P #due to the covariates, the specified class proportions are overruled
```

```
## [1] 0.4999831 0.1680110 0.3320059
```

```
basic.sim3$b #randomly drawn covariate structure with integers from -2 to 2
```

```
##      [,1] [,2]
## [1,]   -1   -1
## [2,]   -1   -2
## [3,]    0   -2
```

Categorical covariates do not have this problem, they allow for specification of the proportions of the classes. Due to this practical reason, and since we are interested in calculating the posterior probabilities per profile (combination of class and score pattern (variables and covariates)), we will add categorical covariates. Categorical covariates will be drawn as if it was another indicator variable but do not represent another data source but a covariate (e.g. gender, age group, or eligibility for rent benefit).

The class-conditional outcome probabilities of the indicator variables can be specified.

```
#specify outcome probabilities of the indicator variables
set.seed(123)
#probabilities for three classes for four indicator variables
probs <- list(matrix(c(0.9,0.05,0.05,
                        0.05,0.9,0.05,
                        0.05,0.05,0.9 ), ncol=3,   byrow=TRUE), # Y1
               matrix(c(0.9,0.05,0.05,
                        0.05,0.9,0.05,
                        0.05,0.05,0.9 ), ncol=3, byrow=TRUE), # Y2
               matrix(c(0.90,0.05,0.05,
                        0.05,0.9,0.05,
                        0.05,0.05,0.9 ), ncol=3, byrow=TRUE), #Y3
               matrix(c(0.05,0.05,0.90,
                        0.05,0.9,0.05,
                        0.90,0.05,0.05 ), ncol=3, byrow=TRUE))#Y4
basic.sim4 <- poLCA.simdata(N=5000,nclass=2,ndv=2,P=c(0.4, 0.3, 0.3), probs= probs)
basic.sim4$P
```

```
## [1] 0.4 0.3 0.3
```

```
basic.sim4$probs
```

```
## [[1]]
##      [,1] [,2] [,3]
## [1,] 0.90 0.05 0.05
## [2,] 0.05 0.90 0.05
## [3,] 0.05 0.05 0.90
##
## [[2]]
##      [,1] [,2] [,3]
## [1,] 0.90 0.05 0.05
## [2,] 0.05 0.90 0.05
## [3,] 0.05 0.05 0.90
##
## [[3]]
##      [,1] [,2] [,3]
## [1,] 0.90 0.05 0.05
## [2,] 0.05 0.90 0.05
## [3,] 0.05 0.05 0.90
##
## [[4]]
##      [,1] [,2] [,3]
## [1,] 0.05 0.05 0.90
## [2,] 0.05 0.90 0.05
## [3,] 0.90 0.05 0.05
```

As can be seen the probs argument overrules the specification of the number of classes and the number of manifest variables.

Data simulation

The data with the two types of errors, the beforementioned selection and classification errors.

```
library(poLCA)
options(scipen = 999) # remove scientific notation
set.seed(123) #set.seed for replicability
```

N = 5000 will be used. Boeschoten (2017) demonstrated that N=10.000 does not yield different/better/… estimates/results.

simulation overview

2x2: selection error (5 or 20%), classification error (5 or 20%) For the simulation study I will simulate n=1000 datasets. Now I will simulate one dataset to demonstrate the method. The datasets have to be simulated in two parts, first the selection error, and then the classification error has to be set, and the datasets have to be combined to result in a dataset with a tree-like structure.

For all variants, the following aspects will be the same: * N=5000 * the same probabilities of the classes * four indicator variables with each … *

variant A

5% selection error, 5% classification error

in probs2 0.05 (5% selection error) in probs3 twee keer 0.025 (5% classification error)

```r
set.seed(123)
#probabilities for two classes for four indicator variables
probs2 <- list(matrix(c(0.95, 0.05,
                       0.05, 0.95), ncol=2, byrow=T),
              matrix(c(0.95, 0.05,
                       0.05, 0.95), ncol=2, byrow=T),
              matrix(c(0.95, 0.05,
                       0.05, 0.95), ncol=2, byrow=T),
              matrix(c(0.05, 0.95,
                       0.95, 0.05), ncol=2, byrow=T))
dat1 <- poLCA.simdata(5000,nclass=2,probs= probs2, P= c(0.2,0.8), missval = F)
df1 <- cbind(dat1$dat,dat1$trueclass)
head(df1)
```

```
##   Y1 Y2 Y3 Y4 dat1$trueclass
## 1  2  2  2  2              2
## 2  2  2  2  1              2
## 3  2  2  2  1              2
## 4  2  2  2  1              2
## 5  2  2  2  1              2
## 6  1  1  1  2              1
```

```r
#probabilities for three classes
probs3 <- list(matrix(c(0.95,0.025,0.025,
                        0.025,0.95,0.025,
                        0.025,0.025,0.95 ), ncol=3,   byrow=TRUE), # Y1
               matrix(c(0.95,0.025,0.025,
                        0.025,0.95,0.025,
                        0.025,0.025,0.95 ), ncol=3, byrow=TRUE), # Y2
               matrix(c(0.95,0.025,0.025,
                        0.025,0.95,0.025,
                        0.025,0.025,0.95 ), ncol=3, byrow=TRUE), #Y3
               matrix(c(0.025,0.025,0.95,
                        0.025,0.95,0.025,
                        0.95,0.025,0.025 ), ncol=3, byrow=TRUE))#Y4
set.seed(125)
dat_sectoren <- poLCA.simdata(N=5000, P=(c(0.4,0.35,0.25)), probs=probs3)
df2 <- cbind(dat_sectoren$dat[,1:4]+1,sectors=dat_sectoren$trueclass+1) #+1 zodat '1' hier niet meer vo


#elke Y-waarde die 2 is (=bedrijf) vervangen door een nieuwe waarde (die de klasse/sector) aangeeft
for(i in 1:ncol(df1)){
  to_replace = which(df1[,i] == 2)
  df1[,i][to_replace] <- df2[,i][to_replace]
}

head(df1)
```

```
##   Y1 Y2 Y3 Y4 dat1$trueclass
## 1  4  4  4  2              4
## 2  2  2  2  1              2
## 3  2  2  2  1              2
## 4  2  2  2  1              2
## 5  4  4  4  1              4
## 6  1  1  1  2              1
```

```
summary((as.factor(df1[,5]))) #how many per class
```

```
##    1    2    3    4
## 1021 1615 1355 1009
```

variant B

5% selection error, 20% classification error