EES2019 Stacking Process

Synthetic Variables Evaluation (Pt.1)

Giuseppe Carteny

20.09.2021

Contents

1 Introduction **1**

2 Evaluation of synthetic variable estimations and workarounds implementation **1**

2.1 Evaluation procedure 2

2.2 Modifying the synthetic variables estimation procedure 5

2.3 Which Variables 6

3 Who Does What **7**

4 The Deadline **7**

1 Introduction

Our next tasks consist in the *evaluation* of the synthetic variables estimated during the last weeks and the creation of workarounds for dealing with problematic estimates.

This task will be based on the existing workflow, but it will require the creation of (1) a new set of scripts for evaluating synthetic variables estimation, (2) changes in the scripts dedicated to said estimations, and (3) a set of R `markdown` scripts summarizing these steps.

This tutorial addresses points (1) and (2). Point (3) will be the topic of a second tutorial.

2 Evaluation of synthetic variable estimations and workarounds implementation

Evaluating and summarizing synthetic variables estimation is needed since, differently from other SDM variables, synthetic ones are not created with simple transformations of existing variables but through estimation

procedures (regression models) that the reader cannot evaluate without additional information.

We already evaluated the synthetic variables regression models, but we did it without a common framework. Moreover, we did not address the possible workarounds for dealing with problematic models.

Consequently, during the last days I have been working for implementing a standardized workflow for both evaluations and workarounds, that is presented below, using as an exemplary case the synthetic variables estimation for the Cypriot EES 2019 voter study (`EES2019_cy_synteval.R`). The new scripts for said tasks are going to be stored in a new subdirectory (`~/Scripts/synteval_scripts/country_spec_scripts/`).

From a general point o view, both OLS and logistic models are going to be evaluated using essentially two sets of information:

1. Regression tables;
2. Model fit statistics.

We might be interested in providing additional information, (such as model diagnostics, or statistics concerning the predictions, like RMSE for OLS models and accuracy, specificity, and sensitivity for the logit models) but I would prefer to reporting them without providing tables or figures[1].

2.1 Evaluation procedure

The workflow for extracting relevant Information about synthetic variables estimation is presented in the exemplary script mentioned a few lines above. First, *in each script*, we run all the steps of the main and country-specific workflows until the estimation of the distance/proximity generic variables (lines 1-99 of the exemplary script).

Then, in the following section of the script ("*Syntvars evaluation: Functions, variables and data frames*"), a set of auxiliary functions are loaded , and some auxiliary data frames and vectors are created, in particular: a list[2] containing the country-specific data frames, a list that includes regression models' data frames, another list containing dependent variables and predictors of our regression models, and a data frame summarising the relevant party codes and names.
All said datasets and vectors can be used for evaluating the regression models, but only a few of these will be included in the summary documents created with R `markdown` (as explained in the next tutorial).

The following five sections of the script, then, represent the first evaluations of the models. First (in the "*Syntvars evaluation: Null and full regression models*" section) the full regression models (as estimated in the country-specific scripts developed last week) and the null models are estimated.
The following two sections ("*Syntvars evaluation: OLS models summary*" and "*Syntvars evaluation: logit models summary*"), then, using the `stargazer` package, allow to print the regression tables (and, as we will see in the following tutorial, to format them in a pdf document created with R `markdown`).
Finally, in the next two sections ("*Syntvars evaluation: OLS models fit stats*" and "*Syntvars evaluation: logit*

[1] Also because in more than a few models, especially ligistic ones dedicated to small parties, we might not be even able to estimate the confusion matrix in a proper way.

[2] Essentially all the relevant data for fitting and evaluating the regression models, investigate the source of the eventual issues, are organized using lists for avoiding manual coding errors in the following steps in which said data frames or vectors will be used repeatedly.

models fit stats") some model fit statistics from both full and null models are estimated and summarised in R data frames.

If the regression models converge, and the results summarised above do not show any anomalous behaviour, then we have just to modify the country-specific 'genvars' script including the *new variables* that are listed few pages below ('Which variables' section of this document) and then move to the following country-specific script in our list.

If (some of) the regression models do show anomalous results/parameters, then we need to identify the source of non-convergence or misfit and deal with them. In the Cypriot case we have at least two logit regression models characterized by coefficients with huge standard errors (see Table 1).

Table 1: Logit regression models for the 2019 EES Cypriot voter study y-hats

	Vote choice					
	Model 1	**Model 2**	**Model 3**	**Model 4**	**Model 5**	**Model 6**
D3_rec2	0.947*	−0.002	0.021	−0.592	0.627	−0.801
	(0.370)	(0.288)	(0.393)	(0.533)	(0.976)	(0.653)
D8_rec1	−0.434	−0.591	−0.444	−0.119	17.416	0.710
	(0.405)	(0.342)	(0.458)	(0.695)	(4662.207)	(0.852)
D5_rec1	0.599	−0.044	0.133	−0.231	18.293	0.490
	(0.447)	(0.365)	(0.494)	(0.650)	(4185.871)	(0.836)
EDU_rec2	−0.539	0.071	0.445	−0.711	18.623	0.493
	(0.428)	(0.406)	(0.541)	(0.728)	(4402.527)	(0.835)
EDU_rec3	−0.564	0.564	0.815	−0.146	19.541	−2.007
	(0.530)	(0.475)	(0.655)	(0.749)	(4402.527)	(1.283)
D1_rec1	0.070	0.369	0.486	0.398	−18.687	1.477*
	(0.412)	(0.322)	(0.420)	(0.613)	(4663.119)	(0.625)
D7_rec1	−1.162**	0.867*	−0.607	1.069	−0.672	0.977
	(0.374)	(0.339)	(0.404)	(0.690)	(0.993)	(0.727)
D7_rec2	−0.681	0.851	−15.514	1.761	−19.124	0.895
	(0.810)	(0.674)	(830.420)	(0.986)	(9079.926)	(1.317)
D4_age	0.029*	0.035***	0.034*	0.004	0.049	−0.022
	(0.012)	(0.010)	(0.014)	(0.017)	(0.039)	(0.020)
D10_rec	−0.191	0.257**	0.030	0.022	−0.307	−0.228
	(0.110)	(0.097)	(0.126)	(0.165)	(0.299)	(0.205)
Constant	−2.479**	−4.712***	−4.190***	−3.493**	−59.265	−3.200*
	(0.894)	(0.790)	(1.084)	(1.250)	(7657.671)	(1.357)
N	395	395	395	395	395	395
Log Likelihood	−123.939	−167.536	−105.246	−63.872	−20.109	−47.003
AIC	269.878	357.071	232.492	149.744	62.219	116.006

***p < .001; **p < .01; *p < .05

Model 3 shows a very high standard error on one coefficient (`D7_rec2`). This fact, however, seems to do not affect the remaining ones. In cases like this one, we might keep the regression model as it is. Model 5, on the contrary, shows six problematic coefficients with huge standard errors (`D8_rec1`, `D5_rec1`, `EDU_rec2`, `EDU_rec3`, `D1_rec1`, `D7_rec2`) that in the end affect also the constant term. In this case we must, first, identify the possible sources of misfit and then develop a workaround.

About the possible sources of misfit, 99.9999% of times, such problems can be identified using cross-

Table 2: Long format cross-tabulation: Recalled vote for party 505 by respondents' area of residency

stack_505	D8_rec	Freq
0	0	84
1	0	0
0	1	354
1	1	5

tabulations between the dependent variable and those independent variables whose coefficients show very high standard errors in the regression tables. As you can see below, for instance, the huge SE of the `D8_rec` variable coefficient in Model 5 is likely determined by the fact that among the few respondents who recall to have voted for the party coded 505, namely the *Ecological and Environmental Movement (Cyprus Green Party)*, none of them live in rural areas. As a consequence we have an empty cell that brings havoc on the maximum likelihood estimator.

In the exemplary script such cross-tabulations can be easily obtained by picking the data frame of the model under examination (data frames that we obtained in the earlier chunks of our script with the `regdf.auxfun` function) and examining them with the `table` base function[3].

Once identified the source of misfit, then we can move to the evaluation of the **workarounds**. In the exemplary script I first tried a sharper approach, dropping the problematic variables for Model 5 in *all* the models (section "*Syntvars evaluation: partial logit models*"). Then I evaluated whether the partial models' and the full models' fit are significantly different using a set of likelihood-ratio chi-squared tests (with the `anova` base function). In two cases (Models 3 and 4) the null hypothesis H_0 (namely, the nested/constrained model fits better than the full/unconstrained one) cannot be rejected. In one case (Model 5, the model of interest) the null hypothesis can be only rejected at $p < 0.1$, in other two cases (Models 2 and 6) the null hypothesis can be rejected at $p < 0.05$, and in one case (Model 1) the null hypothesis can be rejected at $p < 0.001$.

With such results, dropping all the problematic variables for Model 5 across all the models might be represent a rather disruptive solution. Thus, since for Model 5 we can accept the alternative hypothesis H_a only at $p < 0.1$, then I decided to drop the problematic variables just for this model and implement this solution in the synthetic variable estimation script.

Overall, I would prefer that you apply a similar strategy when dealing with problematic models, namely:

1. Create constrained models for all those showing unusual coefficients/parameters;
2. Test whether the unconstrained models fit better than the constrained ones at least at $p < 0.05$;

If the full/unconstrained models do not fit better than the partial/constrained ones then we can modify the synthetic variables estimation dropping the problematic variables.

[3]If you want to obtain all the y/x_i of a regression model you might use the `yxconttab.auxfun` function that takes as arguments the data frame of interest and a logical argument in which you can specify whether you want a standard cross tab (set as 'TRUE') or a cross tab in long format (set as 'FALSE').

2.2 Modifying the synthetic variables estimation procedure

Once concluded our evaluation then we can implement the workaround in the country-specific script dedicated to the generic variables estimation, that in our case is the one dedicated to the Cypriot 2019 EES voter study (`EES2019_cy_genvars.R` in the `~/Scripts/country_spec_scripts/` subdirectory). For applying the workaround chosen above I slightly modified the `gensyn.fun` function that we already used for estimating the synthetic variables last weeks. Now the function handles estimations dedicated to a limited set of values of our dependent variables (PTV or vote choice).

For modifying our country-specific script:

1. We run the main script (`EES2019_stack.R`) until line 62;
2. We run the country-specific script until until the second last section, that is w/o cleaning the environment.

At this point you should have a code that looks more or less like the following one:

```
EES2019_cy_stack %<>%
  left_join(.,
            lapply(data = EES2019_cy_stack,
                   cat.indvar =  c('D3_rec', 'D8_rec',  'D5_rec', 'EDU_rec'),
                   cont.indvar =  c('D4_age', 'D10_rec'),
                   yhat.name = 'socdem_synt',
                   regsum = F,
                   X = list('Q10_gen','Q7_gen'),
                   FUN = gensyn.fun) %>%
              do.call('left_join',.),
            by = c('respid', 'party')) %>%
  as_tibble()
```

```
## Joining, by = c("respid", "party")
```

At this point then you can insert an additional subsection like the following:

```
pred_505_cy <-
  gensyn.fun(data        = EES2019_cy_stack,
             depvar      = 'Q7_gen',
             cat.indvar  = c('D3_rec'),
             cont.indvar =  c('D4_age', 'D10_rec'),
             yhat.name   = 'socdem_synt',
             regsum      = F,
             stack_party = c('505')
             )
```

As you can see now the `gensyn.fun` function has an additional argument (by default set as missing), `stack_party`, in which you can insert the specific party codes (as a *character* vector) for which you want to estimate the y-hats with a constrained model.

After you can then filter the previously estimated y-hats and bind the newly estimated ones with the SDM, as shown below:

```
EES2019_cy_stack <-
  left_join(EES2019_cy_stack %>% dplyr::select(-c(socdem_synt_vc)),
            EES2019_cy_stack %>%
              dplyr::select(respid, party, socdem_synt_vc) %>%
              filter(party!=505) %>%
              rbind(pred_505_cy),
            by = c('respid','party'))
```

2.3 Which Variables

The list of variables that we are going to use for computing our synthetic variables is shown below.

Note that the list of predictors is changed, thus we must insert additional variables (`D1_rec`, and `D7_rec`) in the regression models for estimating our y-hats.

Dependent variables already included in previous models:

- **`Q7_gen`**: A variable measuring whether the respondent voted or not for the stack party;
- **`Q10_gen`**: Respondent's propensity to vote for the stack party;

Independent variables already included in previous models:

- **`'D3_rec'`**: Respondent's gender (0 = Male, 1 = Female), recoded from the original `D3` EES2019 variable (categorical);
- **`'D5_rec'`**: Whether the respondent is married/remarried/single living with a partner (1) or single/divorced/separated/widowed (0), recoded from the original `D5` EES2019 variable (categorical);
- **`'D8_rec'`**: Whether the respondent lives in a rural (0) or urban area (1), recoded from the original `D8` variable (categorical);
- **`EDU_rec`**: Respondent's years of formal education (1 = 15 years or less, 2 = 16-19 years, 3 = 20+);
- **`D4_age`**: Respondent's age, recoded from the original `D4_1` (year of birth) EES2019 variable (ordinal treated as continuous);
- **`D10_rec`**: Respondent's religiosity, recoded from the original `D10` EES2019 variable (ordinal treated as continuous). In particular, the values (min = 0, max = 6) are inverted, so that higher values indicate stronger religiosity and lower values indicate low/none religiosity.

Independent variables **not included** in previous models:

- **`'D1_rec'`**: Trade union membership (0 = not a member, 1 = member), recoded from the original `D1` EES2019 variable (categorical);

- `'D7_rec'`: Subjective social class (0 = working class or lower middle, 1 = middle class, 2 = upper middle or higher class), recoded from the original `D7` EES2019 variable (categorical).

3 Who Does What

The list of countries on which we will work is not changed. I will take care of Belgium, Bulgaria, Cyprus, and Italy, while you will take care of the following countries:

- **Willie**: Denmark, Estonia, Germany, Luxembourg, Malta, Netherlands, Spain, United Kingdom;
- **Julian**: Czech Rep., Finland, Greece, Hungary, Lithuania, Slovakia, Poland, Sweden;
- **Matthias**: Austria, Croatia, France, Ireland, Latvia, Portugal, Romania, Slovenia.

From now on I will take care of the codebook. Thanks all of you for your contribution!

4 The Deadline

The work is not particularly heavy but requires patience and precision. Moreover, I know that this period you are busy with your courses, so I believe that we will need a bit more than one week of work. So the next deadline is **22.10.2021**.