




Generating synthetic data is complicated: Know your data and know your generator

Jonathan Latner¹(✉), Marcel Neunhoeffer¹, and Jörg Drechsler¹

Institute for Employment Research, Nuremberg, Germany {jonathan.latner,
marcel.neunhoeffer,jorg.drechsler}@iab.de

Abstract. The abstract should briefly summarize the contents of the paper in 150–250 words.

Keywords: First keyword · Second keyword · Another keyword.

1 Introduction

- [Gretel.ai](#): The synthetic data platform for developers. Generate artificial datasets with the same characteristics as real data, so you can develop and test AI models without compromising privacy.
- [Mostly.ai](#): Synthetic Data. Better than real. Still struggling with real data? Use existing data for synthetic data generation. Synthetic data is more accessible, more flexible, and simply...smarter.
- [Statice.ai](#): Generating synthetic data comes down to learning the joint probability distribution in an original, real dataset to generate a new dataset with the same distribution. The more complex the real dataset, the more difficult it is to map dependencies correctly. Deep learning models such as generative adversarial networks (GAN) and variational autoencoders (VAE) are well suited for synthetic data generation.
- [hazy.com](#): Synthetic data does not contain any real data points so can be shared freely. Say goodbye to lengthy governance processes associated with real data. Specifically, Hazy data is designed to preserve all the patterns, statistical properties and correlations in the source data, so that it can be used as a drop-in replacement for it.
- DataSynthesizer: The distinguishing feature of DataSynthesizer is its usability — the data owner does not have to specify any parameters to start generating and sharing data safely and effectively.

The idea of releasing synthetic data instead of actual data is often understood to have begun in the early 1990s [10, 17].¹ Here, when we refer to data, we are referring to microdata (i.e. one observation per individual unit) as opposed to tabular data (i.e. summary data). Releasing synthetic data means fitting a model to the original data and then generating new data based on this model so that the released data do not contain information on a real individual unit (person, firm, etc.) that would reveal information that could be used to identify the unit. The appeal is that the synthetic data mimic the statistical properties of the original data while maintaining the privacy of individual records.

There is a general perception that making synthetic data are easy. Our goal is to explain why making synthetic data are complicated. There are two reasons. First, one needs to know the data to be synthesized. Second, one needs to understand the synthetic data generator (SDG). We illustrate the problems that can result when there is a disconnect between knowledge of the generator and knowledge of the data by using three SDGs and one data set. In contrast to the simplicity of the original idea, and despite decades of progress, the creation of synthetic data include steps that are neither common knowledge nor described in one place. Our conclusion: Generating synthetic data is easier than ever. But, to generate good synthetic data, scientists need to have a detailed understanding how and for what purpose the synthetic data are generated.

¹ We note that several years before Rubin’s seminal paper [17], an approach was developed for statistical disclosure protection that we might recognize as synthetic data, but they never called it that way [8].

2 Study design

We compare four measures of utility from three synthetic data generators (SDGs) on one data set. In section 3, we describe the data. In section 4, we describe the SDGs. Following Taub et al. [19] and Little et al. [9], we assess the utility of the synthetic data using three measures: propensity mean squared error (pMSE), the ratio of estimates (ROE), and confidence interval overlaps (CIO). These utility measures are implemented in R from the Synthpop package, which we describe later. We add a fourth measure of utility, the duration in time (in seconds) required to create the synthetic dataset. Finally, while we do not quantitatively evaluate the privacy of the synthetic data, we qualitatively describe how each SDG approaches privacy in the appropriate subsection. To obtain consistent results, five fully synthetic datasets were generated using different random seeds for the each original data set.

JPL Note: below is direct copy from Little et al., 2022

According to Little et al. [9], the propensity score, or pMSE, developed in Woo et al. [20] and Snoke et al. [18] is a measure of data utility designed to determine how easy it is to discern between two datasets based upon a classifier. It is calculated by appending the original and synthetic datasets and creating an indicator variable T , where $T = 1$ for the synthetic dataset and $T = 0$ for the original dataset. For each record in the combined dataset the probability of being in the synthetic dataset is computed; this is the propensity score. The propensity score can be computed via logistic regression. The distributions of the propensity scores for the original and synthetic data are compared; where these are similar data utility should be high. In summary:

$$pMSE = \frac{1}{N} \sum_{i=1}^N [\hat{p}_i - c]^2 \quad (1)$$

where N is the number of records in the merged dataset, \hat{p}_i is the estimated propensity score for record i , and c is the proportion of data in the merged dataset that is synthetic (which is $1/2$). A pMSE score close to 0 would indicate high utility (a score of 0 indicates the original and synthetic data are identical). pMSE can be estimated on the complete data set or subsets of the variables, e.g., all one-way, two-way or three-way combinations.

ROE is calculated by taking the ratio of the synthetic and original data estimates, where the smaller of these two estimates is divided by the larger one. Thus, given two corresponding estimates (e.g. totals, proportions), where y^1 is the estimate orig from the original data and y^1 is the corresponding estimate from the synthetic synth data, the ROE is calculated as:

$$ROE = \frac{\min(y_{orig}^1, y_{synth}^1)}{\max(y_{orig}^1, y_{synth}^1)} \quad (2)$$

if $y_{orig}^1 = y_{synth}^1$ then the $ROE = 1$. The ROE will be calculated over univariate data, and takes a value between 0 and 1. For each numerical variable,

the ratio of estimates are averaged across binned categories. For each categorical variable, the ratio of estimates are averaged across categories to give an overall ratio of estimates.

The CIO, proposed by Karr et al. [7], is defined as:

$$J_k = \frac{1}{2} \left(\frac{U, k - L, k}{U_{orig}, k - L_{orig}, k} + \frac{U, k - L, k}{U_{syn}, k - L_{syn}, k} \right) \quad (3)$$

where U, k and L, k denote the respective upper and lower bounds of the intersection of the confidence intervals from both the original and synthetic data for estimate k , U_{orig}, k and L_{orig}, k represent the upper and lower bounds of the original data, and U_{syn}, k and L_{syn}, k of the synthetic data. Confidence interval overlap from two regression models, one with a dependent variable that is continuous (OLS) and one that is dichotomous (GLM).

Our fourth utility measure is the duration in time (in seconds) required to run one single synthetic dataset, which we refer to as computational efficiency.

3 Know your data

The data we use are called, Social Diagnosis 2011 - Objective and Subjective Quality of Life in Poland (SD2011). The data are included as part of the Synthpop package.² As shown in table 1, there are 35 variables and 5.000 observations. The data contain 21 variables are categorical (or factor) and 14 variables are numeric. There are two main reasons why SD2011 data are helpful to illustrate the potential problems that may exist in the disconnect between knowledge of the data and knowledge of the synthetic data generator (SDG).

One is data dimensionality. Compared to previous evaluations [2, 9], SD2011 contain the highest dimensional data set with respect to rows and columns. While previous research used data with more columns (> 40), the data contained far fewer rows (< 1.000) or if data contained more rows (32.000), then they contained fewer columns (25). Further, the data contain a factor variable (**eduspec**) and a continuous variable (**wkabdur**) with a large number of unique values. Data dimensionality affects the duration in time required to generate the synthetic data, which we refer to in this paper as computational efficiency.

The second reason is that SD2011 contain many properties of real data, including missing, ‘messy’ or extreme values, and generated variables. These properties are both independent and related. Previous evaluations use clean data from Census or machine learning libraries (Kaggle, UCI, OpenML). Without denying the value of clean data for evaluation purposes, real data illustrate advantages and disadvantages of SDGs that would otherwise not be understood.

Missing values present several challenges and come in two types: ‘missing at random’ and ‘informed missings’. One general problem is that some SDGs cannot

² We note that the data are publicly available <http://www.diagnoza.com/index-en.html>. However, we are as yet unable to replicate data made available from the Synthpop package from the raw data.

be used on data with missing values. For example, early versions of CTGAN, which we describe later, could not be used on data with missing values.³ The ability to synthesize missing values is a stage of development that SDGs must pass through before they can be applied to real data.

Another general problem is that it is not always clear what values are missing. For example, factor variables contain missing values and numeric variables contain negative values that we assume are missing. However, income is a numeric variable that contains both missing and negative values.

Informed missings present their own challenges. In its original form [17], the idea of creating synthetic data was to treat all data not in a sample as missing data from a population to be filled in using multiple imputation. The missing data are filled in using models fit with the original data to create synthetic data. While this approach is not a problem if values are missing at random, it is a problem if missing values are informative. For example, SD2011 contains a variable for total time spent working abroad (`wkabdur`) that is missing if an individual did not work abroad (`wkab = Y/N`). The correct coding of missing values when creating synthetic data requires knowledge of the data that is not always available and may not be easy to detect or follow simple rules.

Generated variables are variables created from the original values. In SD2011, there are two generated variables. The variable `agegr` is generated from `age` and the variable `bmi` is generated from `height(cm)` and `weight(kg)`. Given that synthesizing any one variable is a challenge, the challenge increases for each additional variable being synthesized, by definition. As we will show, the problem is exacerbated when generated variables contain messy or extreme values. It is not necessary to synthesize generated variables, but generated variables cannot be detected by the SDG and require knowledge of the data that is not always possessed by those with knowledge of the SDG.

Messy variables also exist. The variable for `wkabdur` or ‘total time spent working abroad’ is a numeric variable that Synthpop treats as a factor variable. Because this variable has a high number of unique values, the treatment of this variable of categorical or continuous affects the duration in time required to synthesize the entire data set. Another example of messy values are the variables: `smoke` and `nociga`. Two individuals indicate they do not smoke, but also indicate they smoke 20 or 22 cigarettes per day, respectively.

Finally, while it is easy to label variables as categorical or continuous, some continuous variables can look like categorical variables and visa versa. For example, the variable for `nofriend` is a continuous variable for the number of friends a survey respondent has. As we will show later, values below 10 are normally distributed, but then cluster at 10, 20, and so on. While it is understood that certain SDGs are more or less appropriate for certain variable types, variation exists within variable types that can also affect utility of a given SDG. In summary, SD2011 is a useful data set to compare and contrast the advantages and disadvantages of SDGs in a real setting.

³ <https://github.com/sdv-dev/CTGAN/issues/39>

We use three versions of the original data to create synthetic data and compare utility. SD2011(a) is the raw data. SD2011(b) codes all negative values in numeric variables as missing and all empty values in character variables as missing. SD2011(c) drops the two generated variables: `agegr` and `bmi`.

4 Know your generator

In the beginning, the idea of creating synthetic data was a theoretical process [17]. To create synthetic data from actual data, all non-respondents in a survey are treated as missing values to be replaced with ‘imputed’ values. Given a dataset (D) with observations (n), there is a two-stage process. First, a model for predicting a given outcome variable (y) given a set of background variables (x) is trained on the actual data. Second, values of the original outcome variable are replaced by random draws from the model of the outcome variable (y^*). This process is then repeated so that all actual values are replaced by imputed values for every variable. The result is a multiply-imputed synthetic dataset (D^*) that contains no values in the actual data, but looks structurally identical to the actual data, not only with respect to mean and variation within a given variable, but also correlations between variables.

Despite the simplicity, implementing the theoretical process is more complicated than the description might suggest. One primary question is what method to use. While the original application may have assumed some sort of parametric estimation using a linear or non-linear model, more recent research borrows from the machine learning literature, including Classification and Regression Trees (CART) [16], Random Forests [1], Support Vector Machines [4], Bayesian Networks [22], and General Adversarial Networks (GANs) [6], among others.

A related question is what package to use. A statistical package that can be loaded into R or Python or another statistical software program that contains a synthetic data generator (SDG) is not the same thing as the method used by the SDG. For example, the Python package Synthcity [14] contains multiple types of GANs (CTGAN, AdsGan, etc.) and Bayesian (PrivBayes, Bayesian Network) SDGs. It is important to distinguish the method from the package in order to reduce confusion when evaluating what SDG is right for what type of research question or data. In this section, we describe tuning the synthetic data generators (SDG). The focus is on finding optimal utility within a given SDG. Given the detail, most of the supporting figures are in the respective online appendix.

4.1 DataSynthesizer

DataSynthesizer (Version 0.1.11) was used [13], which implements the PrivBayes algorithm [22]. DataSynthesizer is designed to address the challenges associated with creating synthetic data outputs from high-dimensional real data inputs. To achieve this goal, the package implements a Bayesian network model, which assumes that all variables are discrete. When learning a Bayesian network, continuous variables are encoded or discretized and the default value is 20 bins.

Afterward, values are re-encoded within a given bin. Therefore, the ability to maintain output scalability with dimensionality is partially derived by sacrificing some amount of utility in continuous variables.

There are a number of hyperparameters that might be tuned for DataSynthesizer, but we focus on two, with all other values set to default. One hyperparameter is differential privacy (ϵ -DP). This is a number that quantifies the amount of noise that is introduced into the original data to create synthetic data. In DataSynthesizer, the default is 0.1 and 0 turns off DP. We turn off DP in order to compare to the other synthesizers that do not have a similar hyperparameter.

A second hyperparameter are the number of attributes any one variable can have, i.e. parents (k). In a Bayesian Network, there are three possible relationships between X and Y . Direct dependence where each variable can only have direct attributes (1 parent), weak conditional independence where each variable can have both direct and indirect attributes (2+ parents), and strong conditional independence where all variables are independent (0 parents).

As a first step, we tune DataSynthesizer based on the number of parents in the Bayesian network using pMSE as a utility measure using SD2011(a), as shown in figure A.1 in the online appendix A. The utility measure is the pMSE and pMSE = 0.25 for 0 and 1 parents; pMSE = 0.2 with 2 parents; and pMSE = 0.21 for 3 parents. Therefore, 2 parents represents the optimal level of utility.

Next, we estimate pMSE for all two-way combinations of variables in SD2011(a), as shown in figure A.2a. While most values are close to 0, it is clear that synthetic values do not match well with original values for the variable, **wkabdur** (work abroad duration). In the original data, 0.975% of all values are -8, as shown in figure A.3a. The problem is that DataSynthesizer not only creates synthetic values of -8, but also values between -8 and 0 that do not exist in the original data. Therefore, we code all negative values in numeric variables as missing and all empty values in character variables as missing, this is SD2011(b).

Next, we re-estimate pMSE for all two-way combinations of variables in SD2011(b), as shown in figure A.2b. While most values are close to 0, it is clear that synthetic values do not match well with original values for the variable, **bmi** (body mass index). In the original data, the variable **bmi** contains a single value of 450, as shown in figure A.3b. This is an extreme value that is nearly 6 times larger than the next closest value of 76. Similar to the variable, **wkabdur**, the problem is that DataSynthesizer not only creates synthetic values of 450, but also values between 76 and 450 that do not exist in the original data. Therefore, we drop both generated variables (**bmi** and **agegr**) to create SD2011(c).

In summary, this exercise illustrates two key points. First, one must know the generator. The Bayesian network implemented by DataSynthesizer discretizes continuous variables into bins and then re-encodes values within a given bin. Second, one must know the data. Missing/negative or extreme values as well as the presence of generated variables affect the similarity or difference between the synthetic and original data. If we compare pMSE across the three versions of SD2011, then pMSE = 0.21 for SD2011(a), pMSE = 0.13 for SD2011(b), and

pMSE = 0.07 for SD2011(c), as shown in figure A.4. Therefore, preprocessing the data is at least as important, if not more than tuning the synthesizer.

4.2 CTGAN

CTGAN (Version 1.9.0) was used [21]. CTGAN is part of the Synthetic Data Vault package [12]. GANs [6] simultaneously train two neural networks: a generator and a discriminator. The generator aims to create synthetic data that becomes increasingly indistinguishable from original data. The discriminator tries to get better at distinguishing between original and synthetic data. This adversarial process goes back and forth until the discriminator cannot distinguish between the original data and the generated data.

In their original application, GANs were designed to create synthetic images, but the approach was adapted to create synthetic microdata. One advantage is that GANs were created to deal with continuous variables (unlike DataSynthesizer). The disadvantage is that cells in micro data are not informative of neighbors in the same way that pixel cells in a photograph. As a result, GANs can struggle modeling relationships between cells and columns [3].

CTGAN contains a number of hyperparameters that one can use to tune the model. Following the package authors, we refer to these as ‘primary’ and ‘advanced’. The primary one is the number of steps contained in the model, which is a function of the number of observations in the original data and two hyperparameters: Epochs and Batch size. In the default, batch size is 500 and epochs is 300. With 5,000 observations in the original data, this means that it takes 10 steps within each Epoch to examine the entire data frame ($5000/500=10$) and the total number of steps is 3,000 ($10*300$). Too few epochs may result in underfitting and too many epochs may result in overfitting.

Advanced hyperparameters include dimensionality and embedding dimension, but others exist (i.e learning rate, and weight decay, etc.). The dimensionality is the number of layers in the generator/discriminator networks. A fully connected layer will be created for each one of the values provided (default is [256,256]). Technically, one can vary the values for both the dimensionality of the discriminator and generator, but we set the same value for both. The embedding dimension influences how much the information in the original data set is compressed. For example, an embedding dimension of 1 is equivalent to storing 34 columns of data in SD2011(c) in a single number (default is 128).

As shown in figure B.1 in online Appendix B, we tune the hyperparameters in three main ways. First, we maintain a constant number of steps (3,000), but allow the batch size to vary (figure B.1a). Second, we maintain a constant batch size (500), but allow the steps to vary (figure B.1b). Third and finally, we vary the dimensionality of the generator/discriminator networks and the embedding dimension (figure B.1c). Surprisingly, tuning these hyperparameters makes little difference with pMSE around 0.16. Given that, we maintain default values for all hyperparameters with the exception of epochs, which we increase to 600.

In summary, this exercise illustrates two key points. First, tuning CTGAN made little difference in the utility of the synthetic data. We are aware of other,

similar research (as yet unpublished) where tuning CTGAN did make a difference. One possible explanation is that our data contain a relatively small number of observations (5,000). Second, we note that pMSE is about twice as high as our optimized SDG using DataSynthesizer. The interpretation is that CTGAN is not a very good synthesizer. However, as described earlier, CTGAN is not the only GAN one can use to create synthetic microdata. Evidence (unpublished) suggests it is possible to create a better GAN.

4.3 Synthpop

Synthpop (Version 1.8.0) [11] was used. Synthpop is an R package that implements parametric and CART models (classification and regression trees) to generate synthetic data. CART models are the default. According to the online description,⁴ Synthpop follows the two-stage process described by Rubin [17], where the data are synthesized one at a time using sequential regression modelling.

Evaluations often find that Synthpop/CART models appear to create synthetic data that are statistically more similar to original data [2, 5, 9]. However, Synthpop, like all SDGs contains advantages and disadvantages [9]. The advantage is one can easily create synthetic data with high levels of utility and little tuning, but the disadvantage is that CART and tree-based models in general struggle computationally with variables that contain many categories. For this reason, we use default values of Synthpop.

5 Comparing SDGs

In this section, we compare our three synthetic data generators (SDGs) on our four measures of utility. We begin by comparing pMSE across our optimized SDGs, as shown in figure 1a. $\text{pMSE} = 0.15$ for CTGAN, 0.07 for DataSynthesizer, and 0.02 for Synthpop. More detail is visible in figure 1b that compares two-way pMSE for all variables in the data. In Synthpop, values are close to zero for most variables. In CTGAN, values are close to 0.075 for most variables. In DataSynthesizer, values are close to zero for most variables except the variable, `nofriend` (number of friends). The overall interpretation is that synthetic data from Synthpop is much closer to the original data (SD2011c) than synthetic data from CTGAN or DataSynthesizer.

Next, we examine one-way frequency statistics and the ratio of estimates (ROE) for select variables to better understand how different SDGs estimate synthetic values, as shown in figure 2. We begin with the variable, `nofriend` (number of friends), as shown in 2a. In the original data, the variable `nofriend` is a continuous variables that are normally distributed below 10, but then clusters at values of 10, 15, 20 and groups of 10 up to the maximum 99. In values below 30, Synthpop and CTGAN perform similarly well, but DataSynthesizer does

⁴ <https://www.synthpop.org.uk/about-synthpop.html#methodology>

not capture clustering or discontinuity in the variable. Instead, DataSynthesizer replaces real values with values that are normally distributed. In values above 30, CTGAN and DataSynthesizer are similarly bad. If we examine ROE values for the variable `nofriend` from figure 2d, then $ROE = 0.4$ for Synthpop, $ROE = 0.28$ for CTGAN, and 0.17 for DataSynthesizer (higher values are better). In both values above and below 30, Synthpop is the only SDG to capture the discontinuity in the continuous variable.

We get a different picture of SDG quality when we look at the variable, `bmi` (Body mass index), as shown in figure 2b. BMI is normally distributed and $ROE = 0.21$ for both CTGAN and DataSynthesizer, and $ROE = 0.24$ for Synthpop. Therefore, even if Synthpop is slightly better, all three SDGs similarly capture the normally distributed variable of BMI.

The last variable we examine is `wkabdur` (Work abroad duration). Values refer to the number of weeks a survey respondent worked abroad between 2007 and 2011, as shown in figure 2c. Nearly 98% of all values are missing, indicating that a respondent did not work abroad. Interestingly, both Synthpop and CTGAN overestimate the median as 14, but in the original data the mean is 14 and the median is 10. However, Synthpop is still better at capturing the distribution of the small number of nonmissing values than DataSynthesizer.

Figure ?? displays parameter estimates and confidence interval overlap (CIO) for two regression models, an OLS model where the dependent variable is `LN(Income)` and a GLM model where the dependent variable is `smoke`. In both models, the independent variables are `gender`, `age`, `education`, and `bmi`. In both models, Synthpop has the highest percent of CIO with CTGAN and DataSynthesizer perform similarly badly. However, when we examine the parameter estimates from the GLM model, Synthpop, like CTGAN and DataSynthesizer, over estimates the effect of education level on smoking for two values: less than secondary and secondary. Therefore, while synthetic data from Synthpop better capture parametric relationships in the original data, they are not a substitute for real data.

Finally, we compare the duration in time required to create synthetic data from the SDGs, i.e. computational efficiency, as shown in table 2. The first row displays results from the raw data. CTGAN requires 331 seconds (5 minutes and 30 seconds) and DataSynthesizer requires 245 seconds (4 minutes). By contrast, duration times from Synthpop are much longer. If we load SD2011 into R as a .csv file as we do with DataSynthesizer and CTGAN, then Synthpop requires 2.132 seconds (35 minutes and 30 seconds), but we if load SD2011 into R from the Synthpop package, then Synthpop requires 5.474 seconds (91 minutes).

The difference in duration within Synthpop and between Synthpop and the other SDGs is explained by two variables: `eduspec` and `wkabdur`. Both variables are variables with a large number of unique values (28 and 33, respectively). For some reason when SD2011 is loaded into R from the Synthpop package, then `wkabdur` is treated as a character variable, not a numeric variable, as it actually is. However, if SD2011 is loaded into R from a .csv file, then `wkabdur` is correctly treated as a numeric variable. The second problem is that Synthpop struggles

computationally with variables that contain many categories. A suggestion is to place this variables at the end prior to synthesizing the data [15]. When we treat `wkabdur` as a numeric variable and place `eduspec` at the end, then Synthpop requires less than 15 seconds, regardless of how SD2011 is loaded. Therefore, Synthpop is more computationally efficient with SD2011 than either CTGAN or DataSynthesizer with proper preprocessing.

However, the issue reveals a more general problem: unlike CTGAN and DataSynthesizer, Synthpop is sensitive to the number (and order) of categorical variables with large values. To illustrate this, we added one, two, or three categorical variables with 20, 25, or 30 unique random values to SD2011. While duration times for CTGAN and DataSynthesizer are almost insensitive to the number of categorical variables or unique values, duration times for Synthpop increase with additional variable and number of unique values. Therefore, Synthpop is less computationally efficient as the data dimensions increase.

6 Conclusion

The key conclusion is that making synthetic data is complicated because it requires knowledge that may not always exist in one place.

There is no one size fits all solution and choosing the right synthetic data generator (SDG) is the result of different trade-offs

Acknowledgments Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper. Only add this information once your submission is accepted and deanonymized.

7 Tables

Table 1: Social Diagnosis 2011 (SD2011)

Number	Variable	Description	Type	Observations	Unique-Values	Missings	Negative-values	Generated	Messy
1	sex	Sex	factor	5000	2	0	0		
2	age	Age of person, 2011	numeric	5000	79	0	0		
3	agegr	Age group, 2011	factor	5000	7	4	0	Yes	Yes
4	placesize	Category of the place of residence	factor	5000	6	0	0		
5	region	Region (voivodeship)	factor	5000	16	0	0		
6	edu	Highest educational qualification, 2011	factor	5000	5	7	0		
7	eduspec	Discipline of completed qualification	factor	5000	28	20	0		
8	socprof	Socio-economic status, 2011	factor	5000	10	33	0		
9	unempdur	Total duration of unemployment in the last 2 years (in months)	numeric	5000	30	0	1556		
10	income	Personal monthly net income	numeric	5000	407	683	603		
11	marital	Marital status	factor	5000	7	9	0		
12	mnarr	Month of marriage	numeric	5000	13	1350	0		
13	ynarr	Year of marriage	numeric	5000	75	1320	0		
14	msepdiv	Month of separation/divorce	numeric	5000	13	4300	0		
15	ysepdiv	Year of separation/divorce	numeric	5000	51	4275	0		
16	ls	Perception of life as a whole	factor	5000	8	8	0		
17	depress	Depression symptoms indicator	numeric	5000	23	89	0		
18	trust	View on interpersonal trust	factor	5000	4	37	0		
19	trustfam	Trust in own family members	factor	5000	4	11	0		
20	trustneigh	Trust in neighbours	factor	5000	4	11	0		
21	sport	Active engagement in some form of sport or exercise	factor	5000	3	41	0		
22	nofriend	Number of friends	numeric	5000	44	0	41		
23	smoke	Smoking cigarettes	factor	5000	3	10	0		
24	nociga	Number of cigarettes smoked per day	numeric	5000	30	0	3737		Yes
25	alcahuase	Drinking too much alcohol	factor	5000	3	7	0		
26	alscol	Starting to use alcohol to cope with troubles	factor	5000	3	82	0		
27	workab	Working abroad in 2007-2011 (in months)	factor	5000	3	438	0		
28	wkabdur	Total time spent on working abroad	numeric	5000	33	0	4875		Yes
29	wkabint	Plans to go abroad to work in the next two years	factor	5000	4	36	0		
30	wkabintdur	Intended duration of working abroad	factor	5000	6	4697	0		
31	emcc	Intended destination country	factor	5000	18	4714	0		
32	englang	Knowledge of English language	factor	5000	4	15	0		
33	height	Height of person	numeric	5000	65	35	0		
34	weight	Weight of person	numeric	5000	91	53	0		
35	bmi	Body mass index (weight - kg/(height - cm ²)*10000)	numeric	5000	1396	59	0	Yes	Yes

Table 2: Duration (in seconds)

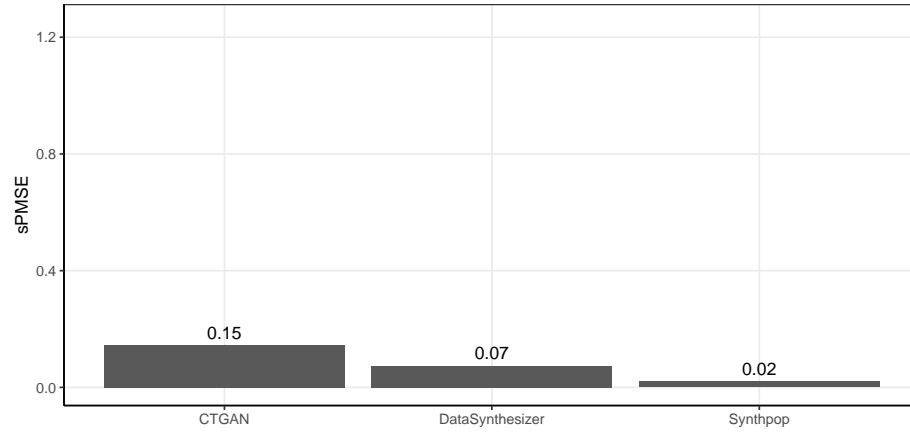
version	description	ctgan	datasynthesizer	synthpop (csv)	synthpop (package)
v00	Raw (SD2011)	331.01	245.37	2132.12	5474.39
v01	Without eduspec or wkabdur	290.30	264.43	10.99	8.45
v02	Without wkabdur	337.07	351.76	13.96	11.02
v03	Without eduspec	306.46	351.24	11.39	8.92
v04	Last variables: eduspec-wkabdur	374.57	344.02	14.23	287.85
v05	Last variables: wkabdur-eduspec	419.60	339.92	14.60	3657.55
v06	as.numeric(wkabdur) and last variable: eduspec	356.02	347.36	14.12	11.05
v07_1	20 + 1 factor variable (20 values)	339.05	264.96	42.23	
v07_1_25	+ 1 factor variable (25 values)	400.28	326.84	137.47	
v07_1_30	+ 1 factor variable (30 values)	339.73	269.72	363.18	
v07_2	20 + 2 factor variable (20 values)	369.74	339.45	74.96	
v07_2_25	+ 2 factor variable (25 values)	364.56	361.81	631.43	
v07_2_30	+ 2 factor variable (30 values)	373.25	346.15	1222.54	
v07_3	20 + 3 factor variable (20 values)	393.99	369.58	122.77	
v07_3_25	+ 3 factor variable (25 values)	401.03	383.40	881.53	
v07_3_30	+ 3 factor variable (30 values)	394.44	424.64	3654.59	

Notes: Synthpop (csv) indicates that SD2011 was loaded into R from a csv file. Synthpop (package) indicates that SD2011 was loaded into R from the Synthpop package.

8 Graphs

Fig. 1: Comparing Synthetic Data Generators (pMSE)

(a) pMSE



(b) Two-way pMSE

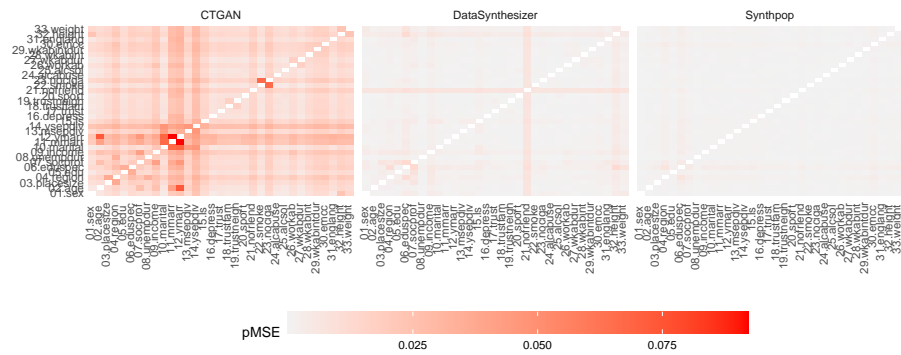
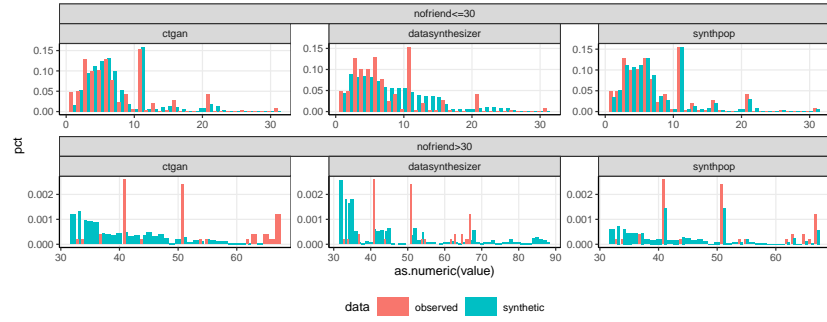
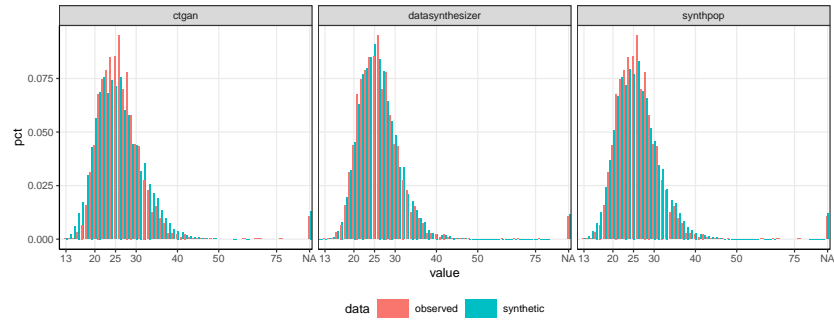


Fig. 2: Comparing one-way frequency of select variables and ROE

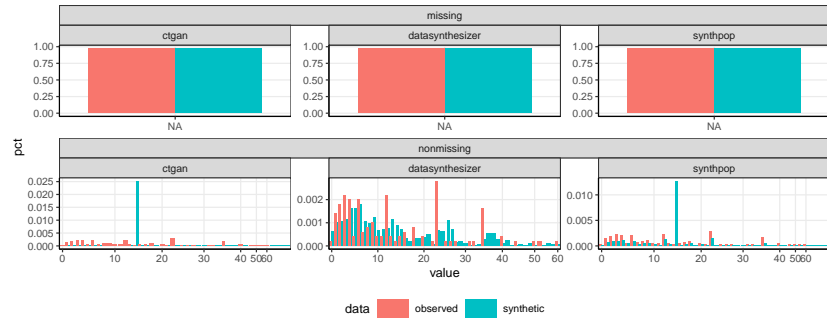
(a) Number of friends



(b) BMI



(c) Work abroad duration



(d) Ratio of estimates (ROE)

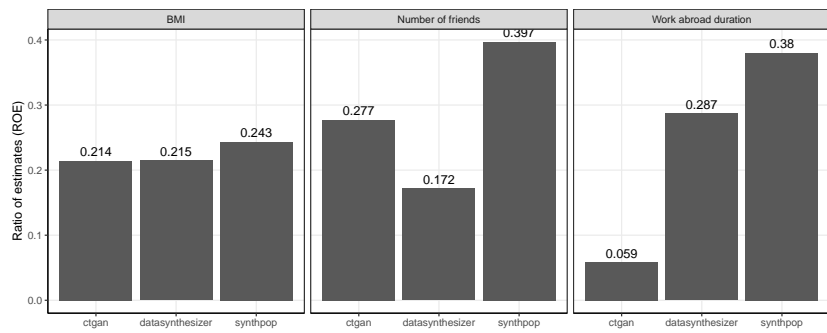
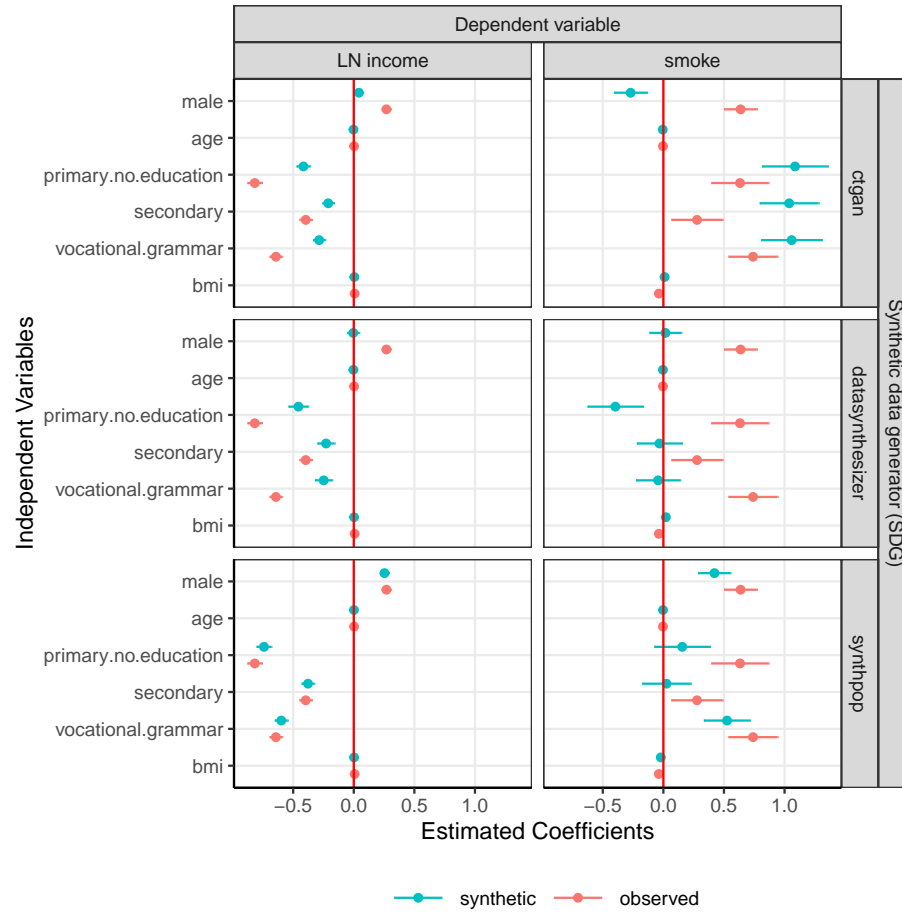


Fig. 3: Confidence interval overlap



sdg	LN income	smoke
synthpop	0.590403	0.4036399
ctgan	-1.479144	-0.4202185
datasynthesizer	-1.176236	-0.5550558

Bibliography

- [1] Caiola, G., Reiter, J.P.: Random forests for generating partially synthetic, categorical data. *Trans. Data Priv.* **3**(1), 27–42 (2010)
- [2] Dankar, F.K., Ibrahim, M.: Fake it till you make it: Guidelines for effective synthetic data generation. *Applied Sciences* **11**(5), 21–58 (2021)
- [3] Drechsler, J., Haensch, A.C.: 30 years of synthetic data. *arXiv preprint arXiv:2304.02107* (2023)
- [4] Drechsler, J.: Using support vector machines for generating synthetic datasets. In: *Privacy in Statistical Databases: UNESCO Chair in Data Privacy, International Conference, PSD 2010, Corfu, Greece, September 22–24, 2010. Proceedings.* pp. 148–161. Springer (2010)
- [5] Drechsler, J., Reiter, J.P.: An empirical evaluation of easily implemented, nonparametric methods for generating synthetic datasets. *Computational Statistics & Data Analysis* **55**(12), 3232–3243 (2011)
- [6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. *Advances in neural information processing systems* **27** (2014)
- [7] Karr, A.F., Kohnen, C.N., Oganian, A., Reiter, J.P., Sanil, A.P.: A framework for evaluating the utility of data altered to protect confidentiality. *The American Statistician* **60**(3), 224–232 (2006)
- [8] Liew, C.K., Choi, U.J., Liew, C.J.: A data distortion by probability distribution. *ACM Transactions on Database Systems (TODS)* **10**(3), 395–411 (1985)
- [9] Little, C., Elliot, M., Allmendinger, R.: Comparing the utility and disclosure risk of synthetic data with samples of microdata. In: *International Conference on Privacy in Statistical Databases.* pp. 234–249. Springer (2022)
- [10] Little, R.J., et al.: Statistical analysis of masked data. *Journal of official statistics* **9**, 407–407 (1993)
- [11] Nowok, B., Raab, G.M., Dibben, C.: synthpop: Bespoke creation of synthetic data in r. *Journal of statistical software* **74**, 1–26 (2016)
- [12] Patki, N., Wedge, R., Veeramachaneni, K.: The synthetic data vault. In: *IEEE International Conference on Data Science and Advanced Analytics (DSAA).* pp. 399–410 (Oct 2016). <https://doi.org/10.1109/DSAA.2016.49>
- [13] Ping, H., Stoyanovich, J., Howe, B.: Datasynthesizer: Privacy-preserving synthetic datasets. In: *Proceedings of the 29th International Conference on Scientific and Statistical Database Management.* pp. 1–5 (2017)
- [14] Qian, Z., Cebere, B.C., van der Schaar, M.: Synthcity: facilitating innovative use cases of synthetic data in different data modalities (2023). <https://doi.org/10.48550/ARXIV.2301.07573>, <https://arxiv.org/abs/2301.07573>
- [15] Raab, G.M., Nowok, B., Dibben, C.: Guidelines for producing useful synthetic data. *arXiv preprint arXiv:1712.04078* (2017)

- [16] Reiter, J.P.: Using cart to generate partially synthetic public use microdata. *Journal of official statistics* **21**(3), 441 (2005)
- [17] Rubin, D.B.: Statistical disclosure limitation. *Journal of official Statistics* **9**(2), 461–468 (1993)
- [18] Snoke, J., Raab, G.M., Nowok, B., Dibben, C., Slavkovic, A.: General and specific utility measures for synthetic data. *Journal of the Royal Statistical Society Series A: Statistics in Society* **181**(3), 663–688 (2018)
- [19] Taub, J., Elliot, M., Sakshaug, J.W.: The impact of synthetic data generation on data utility with application to the 1991 uk samples of anonymised records. *Trans. Data Priv.* **13**(1), 1–23 (2020)
- [20] Woo, M.J., Reiter, J.P., Oganian, A., Karr, A.F.: Global measures of data utility for microdata masked for disclosure limitation. *Journal of Privacy and Confidentiality* **1**(1) (2009)
- [21] Xu, L., Skoularidou, M., Cuesta-Infante, A., Veeramachaneni, K.: Modeling tabular data using conditional gan. In: *Advances in Neural Information Processing Systems* (2019)
- [22] Zhang, J., Cormode, G., Procopiuc, C.M., Srivastava, D., Xiao, X.: Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)* **42**(4), 1–41 (2017)

A Appendix: DataSynthesizer

Fig. A.1: Tuning DataSynthesizer across parents

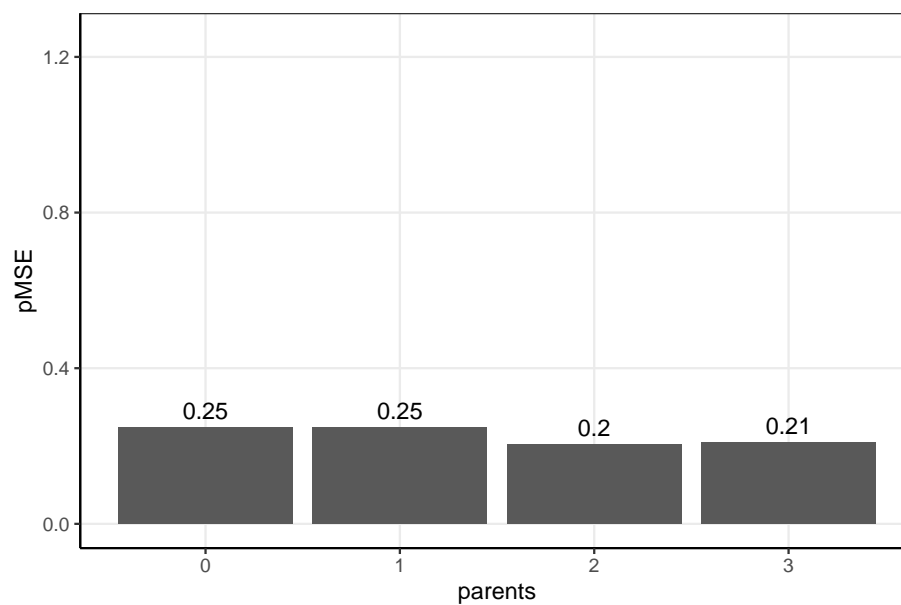
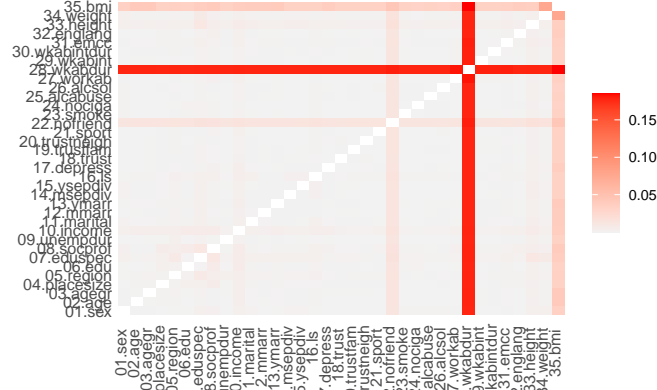
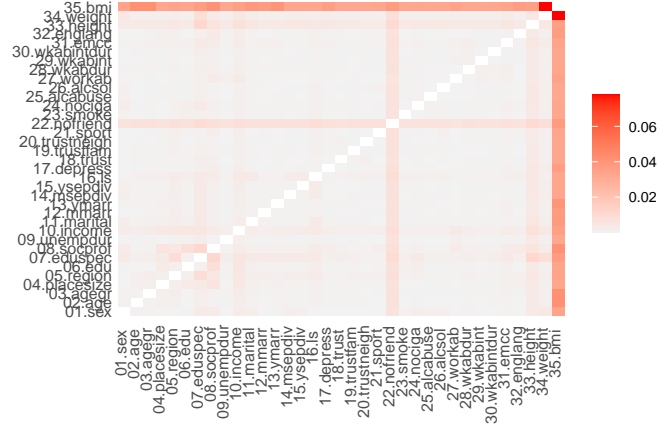


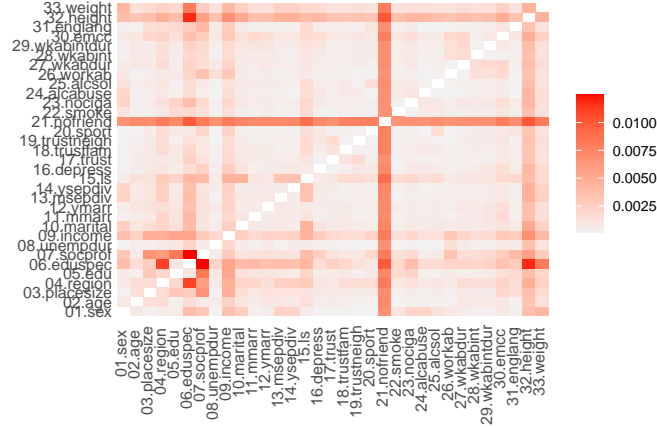
Fig. A.2: Datasynthesizer two-way correlation (pMSE)



(a) SD2011(a)



(b) SD2011(b)



(c) SD2011(c)

Fig. A.3: Frequency values for original and synthetic data (DataSynthesizer)

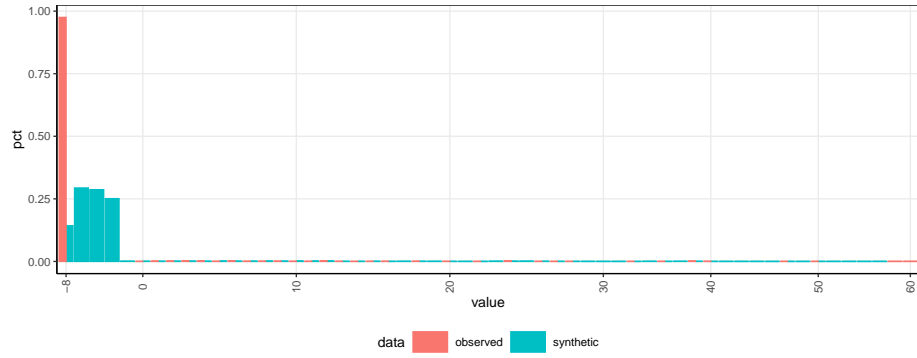
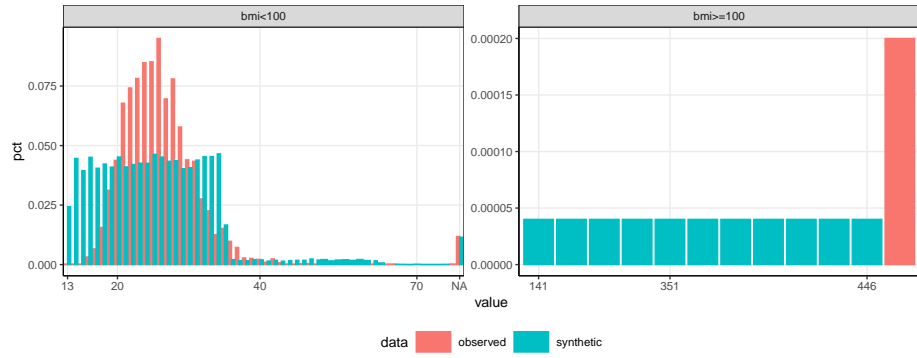
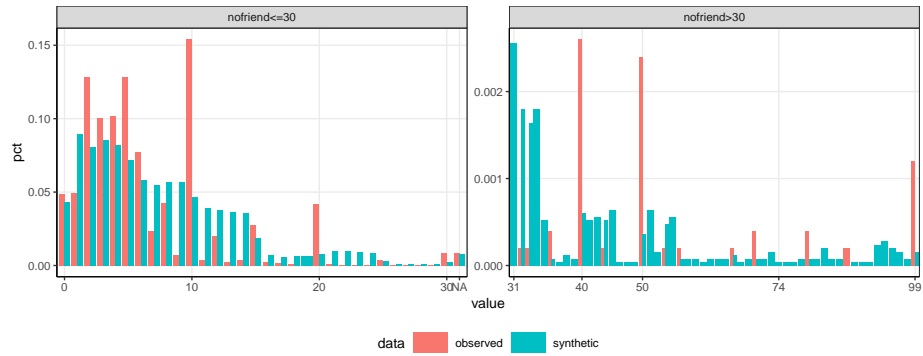
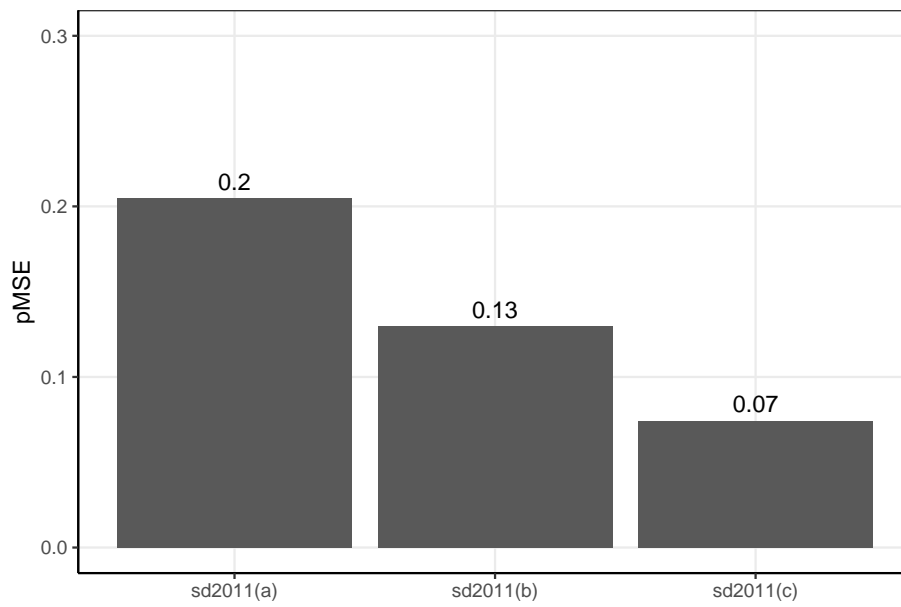
(a) Variable: `wkabdur` (Work abroad duration)(b) Variable: `bmi` (Body mass index)(c) Variable: `nofriend` (Number of friends)

Fig. A.4: Tuning DataSynthesizer across data sets (parents = 2)



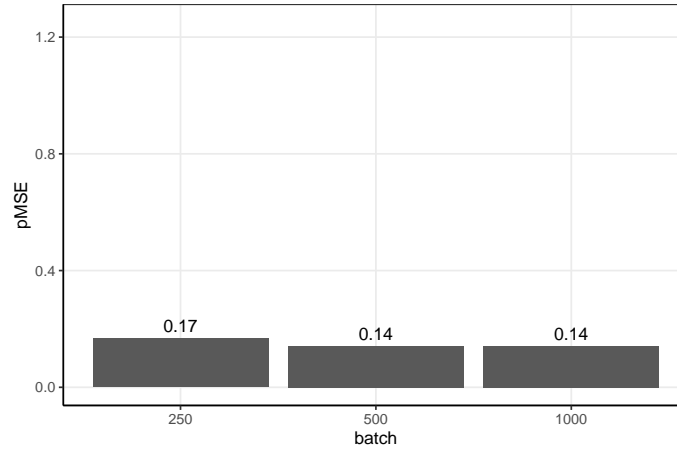
B Appendix: CTGAN

Table B.1: Batch size and epochs = total steps

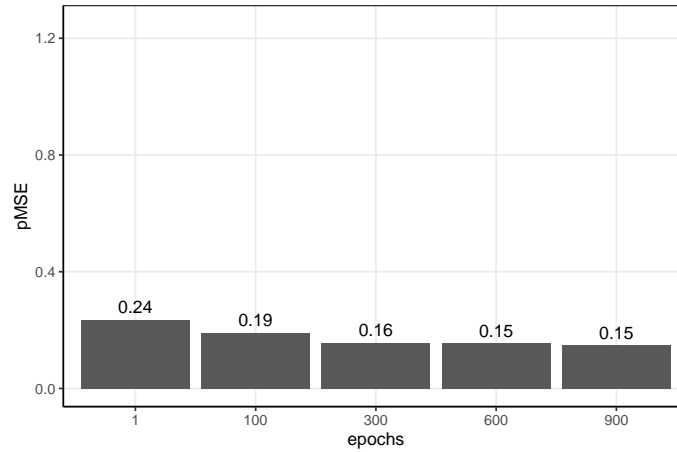
N	Batch size	Steps per Epoch	Epochs	Total Steps	Compare
5,000	500	10	100	1,000	Constant batch size, as shown in figure B.1a
5,000	500	10	300	3,000	
5,000	500	10	600	6,000	
5,000	500	10	900	9,000	
5,000	100	50	60	3,000	Constant batch size, as shown in figure B.1b
5,000	250	20	150	3,000	
5,000	500	10	300	3,000	
5,000	1,000	5	600	3,000	

Fig. B.1: Tuning CTGAN (effect of steps)

(a) Effect of batch size with constant steps (3.000)



(b) Effect of epoch number with constant batch size (500)



(c) Effect of dimensionality

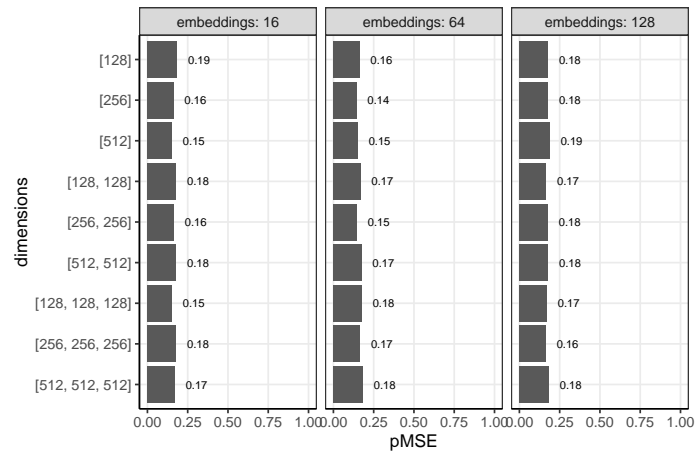
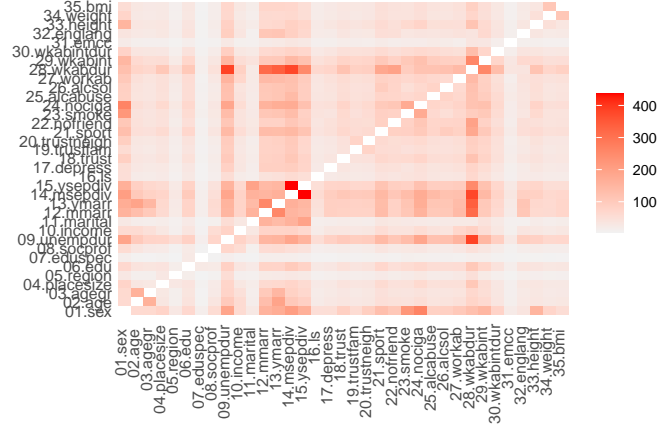
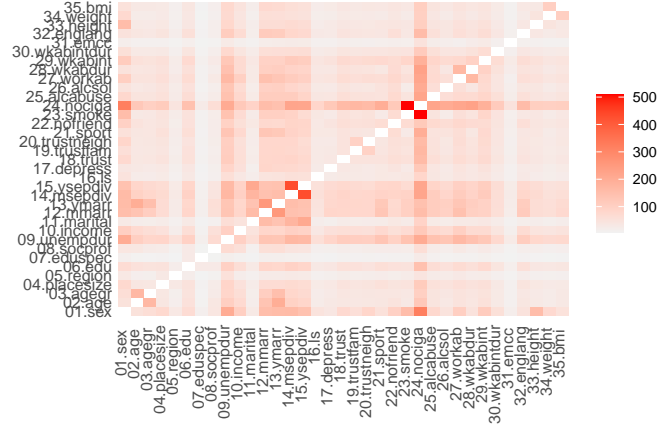


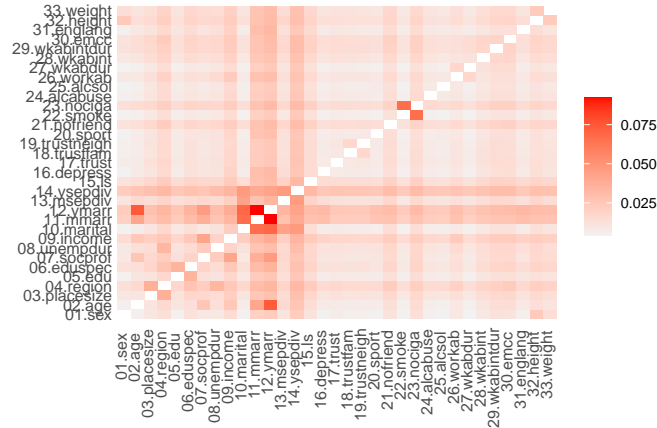
Fig. B.2: CTGAN two-way correlation (pMSE)



(a) SD2011(a)

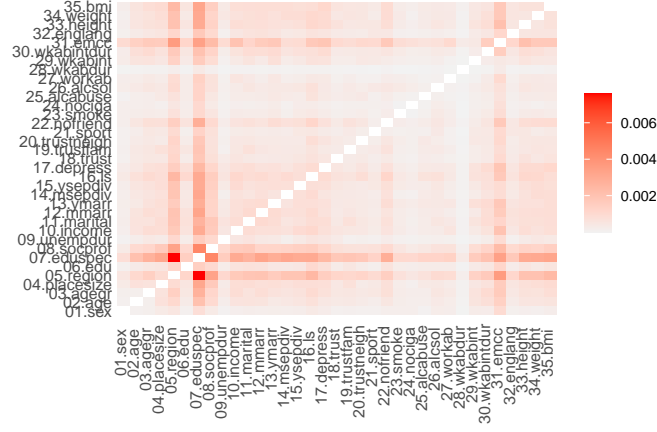


(b) SD2011(b)

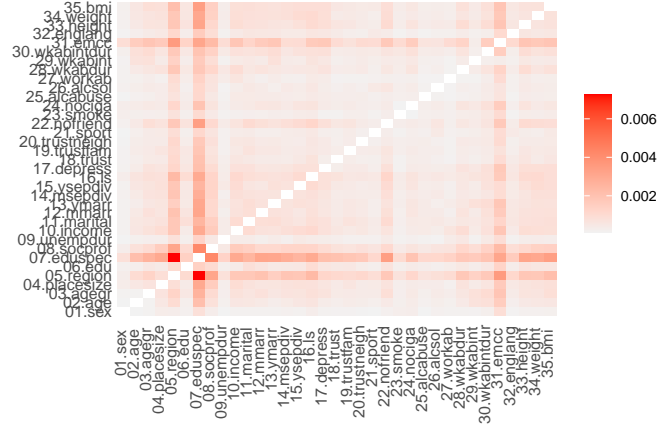


(c) SD2011(c)

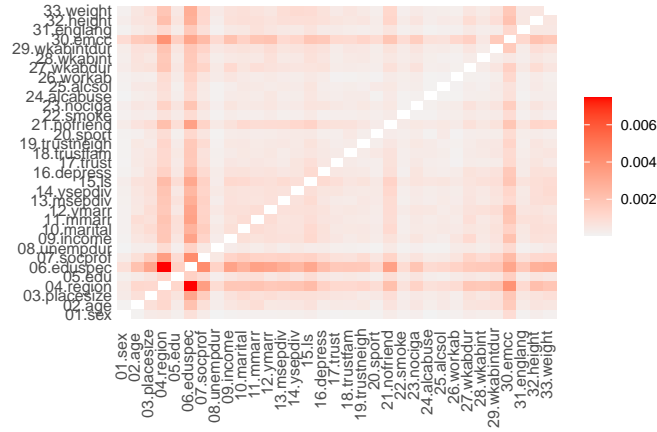
Fig. B.3: Synthpop two-way correlation (pMSE)



(a) SD2011(a)



(b) SD2011(b)



(c) SD2011(c)