INSTITUTE FOR EMPLOYMENT RESEARCH
The Research Institute of the Federal Employment Agency

AnigeD

GENERATING SYNTHETIC DATA IS COMPLICATED: KNOW YOUR DATA AND KNOW YOUR GENERATOR

PSD2024: Privacy in Statistical Databases 2024,
26. September, 2024

Jonathan Latner, PhD
Dr. Marcel Neuenhoeffer
Prof. Dr. Jörg Drechsler

SECTION 1: INTRODUCTION

OVERVIEW

- Common perception that making synthetic data is easy
- We to show that its complicated
  - You need to know your data
    - Missing values, messy data, etc.
  - You need to know your synthetic data generator (SDG)
    - Compare 3 SDGs: DataSynthesizer, CTGAN, Synthpop
    - How does it deal with missing values?
    - How computationally efficient is it (in terms of duration in time)?
- Conclusion
  - On the one hand, generating synthetic data is easy (even I can do it!)
  - On the other hand, users need to be aware of challenges
  - Every SDG has advantages/disadvantages (no one, correct solution)
  - One should be skeptical about automated processes for generating synthetic data

THE GOOD NEWS – MAKING SYNTHETIC DATA IS EASY

- `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 (Ping et al., 2017): 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 BAD NEWS – MAKING SYNTHETIC DATA IS HARD

- According to the Alan Turing Institute (Jordan et al., 2022)
- Synthetic data is not a replacement for real data. It is a distorted version of the real data.
- We must ask:
  - Why are we creating synthetic data?
  - How different should it be?
  - How do we measure the difference?
  - How does the synthesizer work?
- Answering these questions will help us make decisions about which synthesizer is the right choice
  - Hyperparameters: from complete black box to complete user choice
  - Variables: not just with categorical, dichotomous, and continuous variables, but with messy values
  - Computationally efficiency (i.e. duration in time): is important and often ignored. The algorithm should scale well with the dimension of the data space in a relational way, not exponential way.

OUR GOAL IS TO ILLUSTRATE THE CHALLENGES

- Know your data (1 dataset)
  - Social Diagnosis 2011 (SD2011) - Cleaning/pre-processing (most evaluations use clean data)
- Know your generator
  - Compare 3 synthetic data generators (SDGs): DataSynthesizer, CTGAN, Synthpop
- 3 utility measures
  - Compare univariate frequency
  - Propensity score mean-squared error (pMSE) - Append the original and synthetic datasets. Create an indicator variable for original/synthetic datasets. The probability of being in the synthetic dataset is computed for each record in the combined dataset (n); this is the propensity score (p). Lower scores are better. ($pMSE = \frac{1}{N}\sum_{i=1}^{N}[\hat{p}_i - c]^2$)
  - Computationally efficient with respect to duration in time
- In this talk, we are focusing on utility. Relative to the challenge of achieving privacy, utility challenges receive less attention.

SECTION 2: KNOW YOUR DATA (SD2011)

REAL DATA

- Social Diagnosis 2011 (SD2011)
- Loads with Synthpop
  - `http://www.diagnoza.com/index-en.html`
  - Not entirely clear how original data is created or cleaned to create data in Synthpop
    - No
- Like real data, has ‘quirks’ or unusual values/variables
  - Includes missings
    - Informative (i.e. for never worked abroad, `wkabdur` is missing)
    - Non-informative
  - Includes ‘errors’
    - `smoke` - Does smoke is NO, but `nociga` - 20/22 cigarettes per day
    - `bmi` = 451, but `height`(cm) = 149 and `weight`(kg) = NA (999)
  - Includes generated variables (Can be problematic for SDGs)
    - `bmi`, `agegr`

DATA (SD2011)

Table 1: Spikey values

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		
				…					
10	income	Personal monthly net income	numeric	5000	407	683	603		
11	marital	Marital status	factor	5000	7	9	0		
12	mmarr	Month of marriage	numeric	5000	13	1350	0		
14	msepdiv	Month of separation/divorce	numeric	5000	13	4300	0		
15	ysepdiv	Year of separation/divorce	numeric	5000	51	4275	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
				…					
28	wkabdur	Total time spent on working abroad	numeric	5000	33	0	4875		Yes
				…					
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

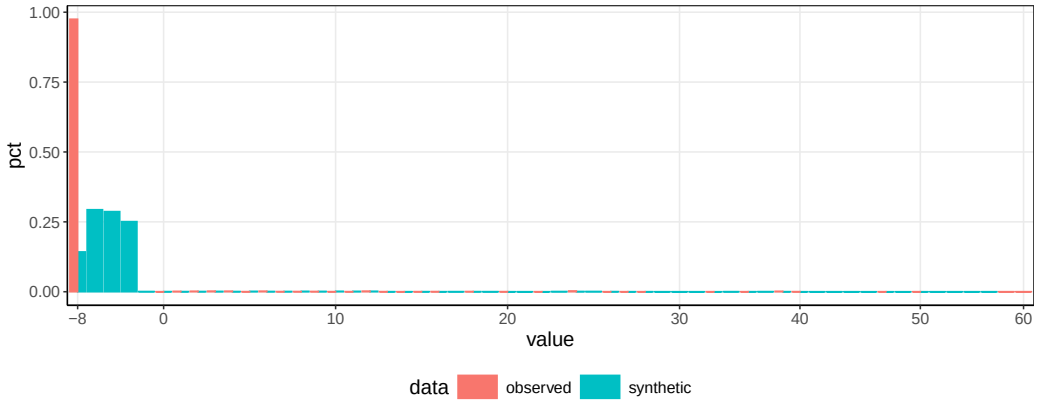
SECTION 3a): KNOW YOUR GENERATOR (DATASYNTHESIZER)

DataSynthesizer is a Python package that implements the PrivBayes algorithm to address the challenges associated with a differentially private (DP) method for releasing synthetic data outputs from high-dimensional real data inputs. To do this, the package implements a Bayesian network model to estimate the joint distribution of the data. The Bayesian network only works with discrete variables. One way to discretize continuous variables is by binning them. This can be a problem.

- Hyperparameters
  - ϵ Differential Privacy (DP): we turn it off (default 0.1)
  - k-degree Bayesian network (parents): 2 (default is 'greedy')
- Utility measure - compare univariate frequency between original and synthetic data

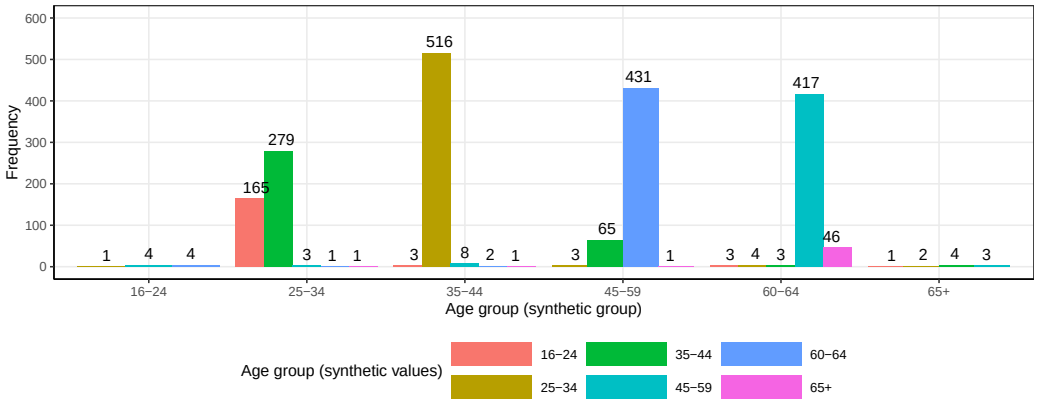
PREPROCESS MISSING VALUES

Figure 1: Captures values < 0 as continuous, not missing/categorical



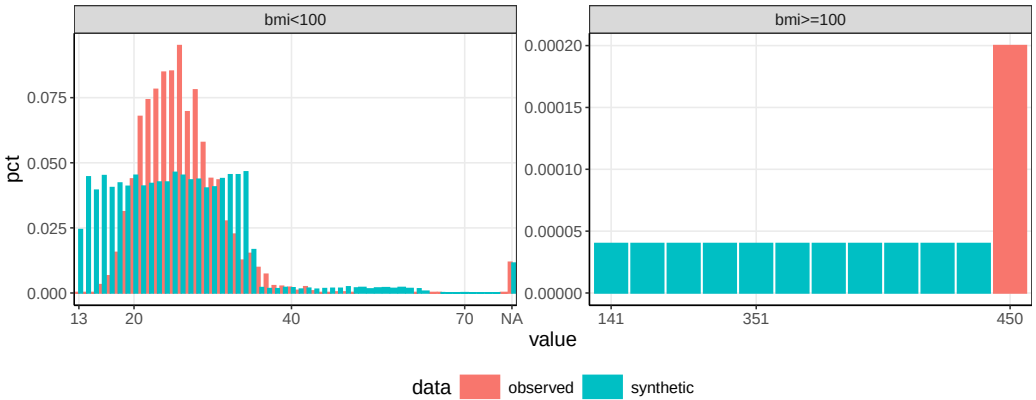
DROP GENERATED VARIABLES (1 OF 2)

Figure 2: Age group: Number of misclassified



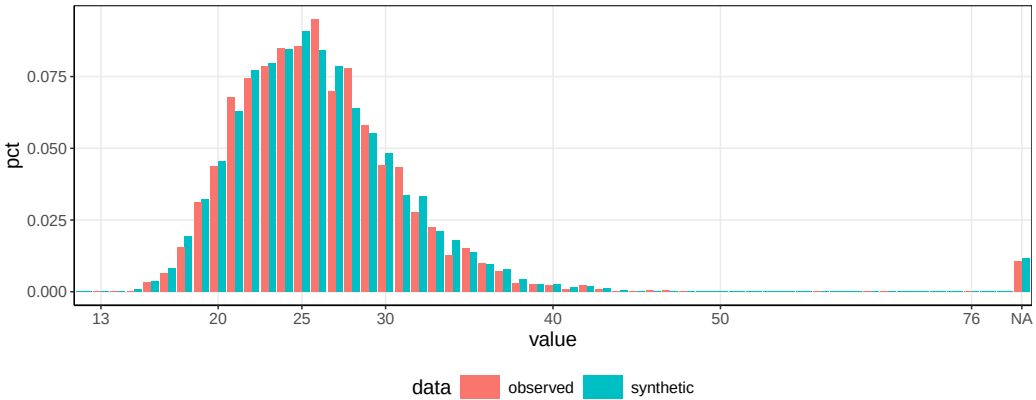
DROP GENERATED VARIABLES (2 OF 2)

Figure 3: BMI: Preprocess extreme values



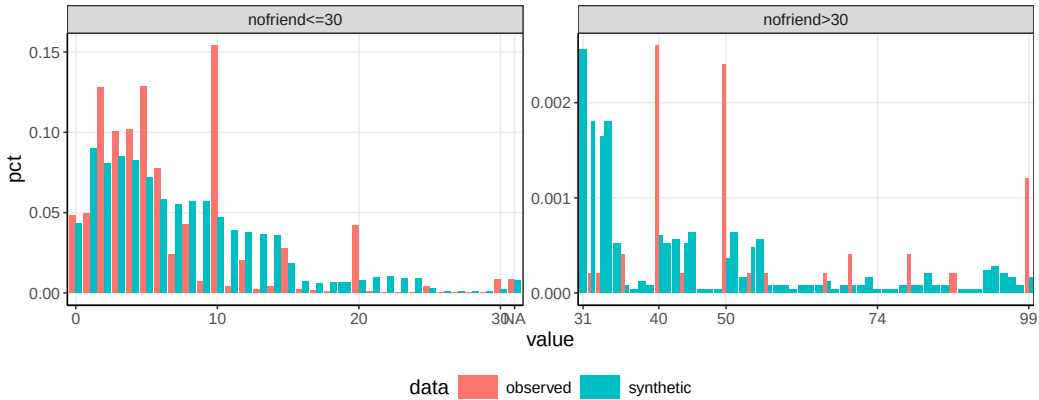
REGENERATE GENERATED VARIABLES FROM SYNTHETIC VALUES

Figure 4: Body mass index



CAN ONLY PREPROCESS SO MUCH

Figure 5: nofriend: 'Spikey' or discontinuous values



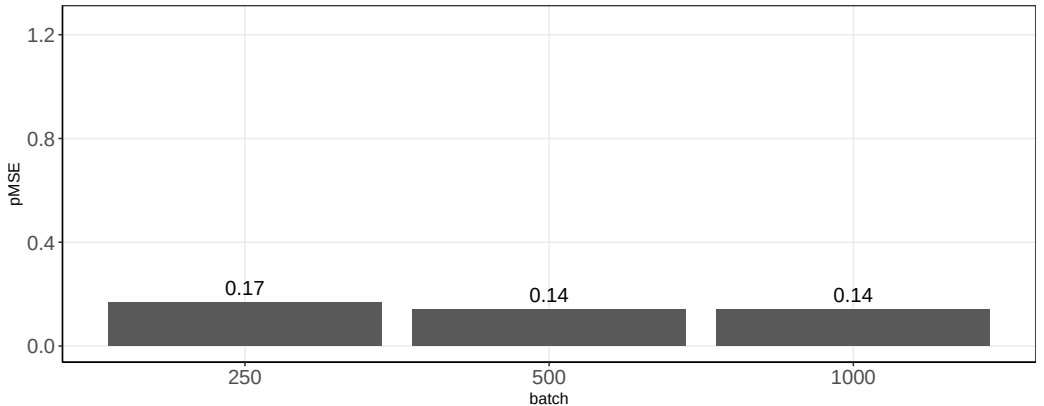
SECTION 3b): KNOW YOUR GENERATOR (CTGAN)

GANs simultaneously train two neural networks: a generator and a discriminator. The goal of the generator is to create synthetic data that becomes increasingly indistinguishable from original data. The goal of the discriminator is 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.

- Hyperparameters
  - Actual steps for observations (n) = f (batch * steps per epoch * epochs)
    - batch size = Number of samples to process in each step (default is 500)
    - epochs = Number of times the GAN gets to see the full dataset (default is 300).
  - Dimensions
    - Discriminator/Generator - defines the architecture of the Discriminator by specifying the number of neurons in each layer. Defaults to (256, 256)
    - Embeddings - Size of the random sample passed to the Generator. (Default 128)
- Utility - pMSE

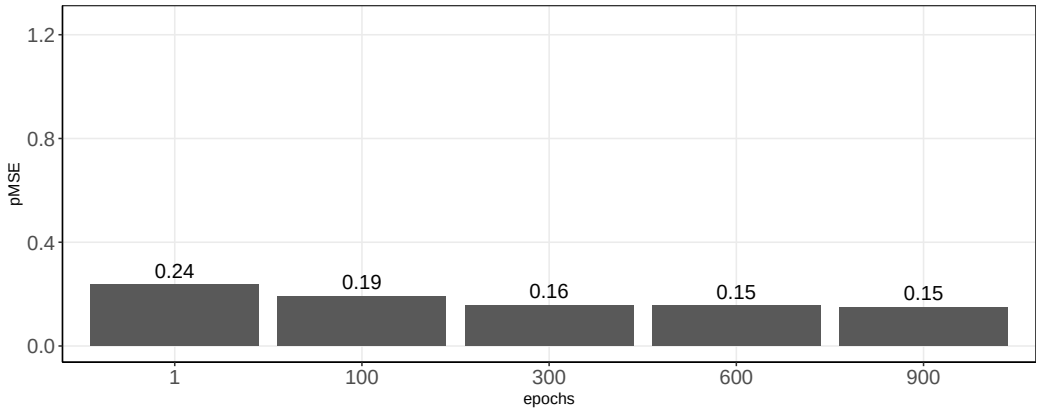
VARY BATCH SIZE, BUT STEPS ARE CONSTANT (3000)

Figure 6: DataSynthesizer (0.07) and Synthpop (0.02)



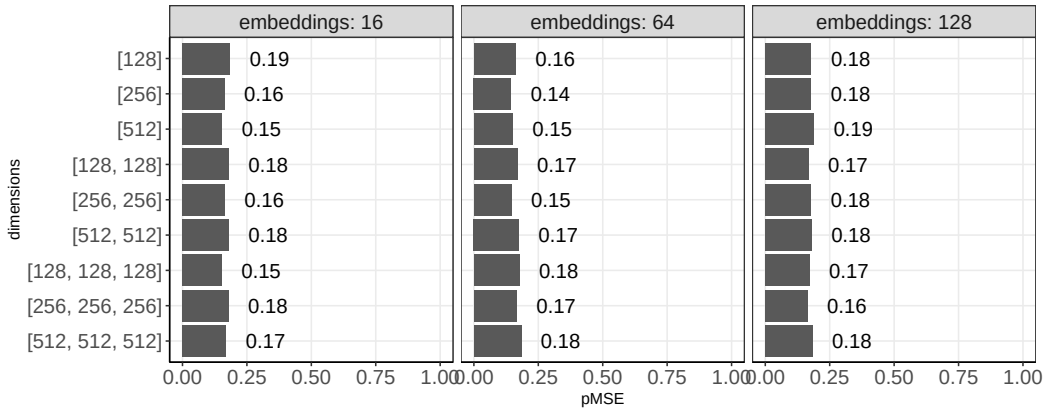
VARY EPOCHS, BUT BATCH SIZE IS CONSTANT (500)

Figure 7: DataSynthesizer (0.07) and Synthpop (0.02)



OPTIMIZE DIMENSIONS

Figure 8: DataSynthesizer (0.07) and Synthpop (0.02)



SECTION 3c): KNOW YOUR GENERATOR (SYNTHPOP)

Synthpop is an R package that implements parametric and ML based models (classification and regression trees (CART) and random forests) to generate synthetic data. Synthpop follows a sequential process, where the first variable to be synthesized is generated by drawing new values from the marginal distribution of this variable (either by drawing from a parametric distribution or by sampling from the empirical distribution), and the subsequent variables are synthesized one at a time, always conditioning on those variables that have been synthesized in earlier steps

- Hyperparameters (default)
- Utility measure - Computational efficiency

COMPUTATIONAL EFFICIENCY

Table 2: Synthpop may have lowest pMSE, but is not as efficient with high dimensional data

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	

SECTION 4: CONCLUSION

SUMMARY

- You need to know your data
  - Missing values, Extreme values, Generated variables, ‘Spikey’ or discontinuous values
- You need to know your synthesizer
  - CTGAN
    - Pro: high levels of computational efficiency
    - Con: GAN architecture may not produce synthetic data with high levels of utility, at least in low dimensional data (33 columns, 5000 rows).
  - DataSynthesizer uses a Bayesian network model.
    - Pro: high levels of computational efficiency
    - Con: algorithm assumes all variables are discrete, which reduces utility of synthetic continuous variables.
  - Synthpop
    - Pro: high levels of utility (pMSE)
    - Con: CART struggles with computational efficiency on datasets that contain variables with many categories.
- The people who know the data may not understand the synthesizer and vice versa