SATGPA - Generative Adversarial Network (GAN)

Evaluation Synthetic Data Creation

Steffen Moritz, Hariolf Merkle, Felix Geyer, Michel Reiffert, Reinhard Tent (DESTATIS)

January 31, 2022

Executive Summary

We used mainly the `sdv` python libraries to employ GANs. We specifically used the `ctgan` (Conditional GAN) and `CopulaGAN` implementations. We also tested the R package `ganGenerativeData`. The GAN algorithms required quite a lot of computing power, which is a clear downside. From our main metrics the final GAN result seemed like a good trade-off between utility and privacy. Out of all different methods we tested (`FCS`, `IPSO`, `GAN`, `Simulation`, `Minutemen`) it was ranked second for utility and third privacy. But, a closer look into more utility measures weakens this first impression. Only one variable has a similar distribution as its original according to the Kolmologorov-Smirnov test. The `S_pMSE` for tables and for distributions is still very high. The Mahalanobis distance of the regression parameters **always exceeds the critical value**. There is **no reasonable usability** according to Mlodak's information loss criterion. From a privacy perspective the GAN looks quite good (also when looking at more detailed metrics). From our perspective it seemed like the GAN algorithms tend to extrapolate more than other algorithms (e.g. FCS).

USE CASE RECOMMENDATIONS

Releasing_to_Public	Testing_Analysis	Education	Testing_Technology
NO	NO	YES	YES

The utility of **GAN** has some flaws, thus we **don't** think it is a good idea to **release this data to the public**. Also scientists may be led to false conclusions when using this data for **testing analysis**. We can imagine GAN generated data in **education** or in **technology testing**. On first sight it seems like GAN data is somehow to computationally intensive to consider it for testing, but we also see an advantage in the fact, that they tend to extrapolate a little more. This could be beneficial for testing.

Dataset Considerations

When deciding, if data is released to the public it is of utmost importance to define, **which variables** are the **most relevant** in terms of **privacy and utility**. This process is very **domain and country specific**, since different areas of the world have different privacy legislation and feature specific overall circumstances. This step would require input and discussions with actual domain experts. Since we are foreign to US privacy law and there is no SAT equivalent in Germany, the assumptions made for the Synthetic Data Challenge are basically a **educated guess** from our side.

From a **utility perspective** it is important to know which variables and correlations are most interesting for actual users of the created synthetic dataset. Different use cases might require focus on different variables and correlations. We could not single out a most important variable, thus in our utility analysis we decided to focus on the overall SAT utility and **not to prioritize a specific variable**. From a data plausibility perspective it was essential to us, that the `sat_v + sat_m = sat_sum` stay consistent.

From a **privacy perspective** it has to be decided, which variables are **confidential** and which are **identifying**. As already mentioned, specifying this depends on multiple factors e.g. **regulations** or also **other public information**, that could be used for **de-anonymization**. For our analysis, we made the following assumptions: Feature `sex` is an identifying value. For the SAT percentiles (`sat_v`, `sat_m`, `sat_sum`) there can be argued in both directions, but we decided for it to be an identifying variable. The same holds for the grade point average `fy_gpa`. We assumed the grade point average to be more confidential than the SAT results. It is very likely for example that students who passed the SAT exchanged about their grades with their fellow students or that teachers know about the SAT results of their students. So these (older) information potentially can be used to **identify a student** within a dataset and find out about the **(newer) information of grade point average**. The calculus behind this decision is that the older information about a test, that is only used to get the college admission, is **less confidential than the newer information** about actual grades, which might give information about a student's current situation.

Method Considerations

GANs are in comparison to the other methods a **relatively recent method** (Goodfellow, et al., 2014). For our experiments we used the `sdv` python libraries. We employed `ctgan` (Conditional GAN) and `CopulaGAN` for our analysis. We also tried using the R package `ganGenerativeData`, but because we were not able to use all variable types as input, the results with ganGenerativeData were not on the same level as with svd. For the **SAT** dataset computing time with GAN seemed reasonable. Yet, already for SAT as we were using ctgan with `epochs` settings of over 1000, the **process already took hours**. For **ACS** we had to cancel several runs, after not completing after 8 hours computing time. This clearly shows, that the GAN method has it's **limits** when dealing with **large and complex datasets**. Dedicated computing infrastructure would be quite useful in these cases. Data utility wise (especially for the lower epoch settings) the resulting synthetic dataset differed significantly from the original data. In comparison to the other methods the GAN seemed to create **implausible values** more often. E.g. negative values for variables, that are known to be only positive. While the tendency to slightly **extrapolate** might seem negative at first, this can also be a positive e.g. when the original dataset itself is only a sample or when the method is used to create technical test data.

Privacy and Risk Evaluation

Disclosure Risk (R-Package: synthpop with own Improvements)

Our starting point was the **matching of unique records**, as described in the disclosure risk measures chapter of the starter guide. The synthpop package provides us with an easy-to-use implementation of this method: `replicated.uniques`. However, one downside of just using `replicated.uniques` is that it does **not consider almost exact matches in numeric variables**. Imagine a data set with information about the respondents' income. If there is a matching data point in the synthetic data set for a unique person in the original data set, that only differs by a slight margin, the original function would not identify this as a match. **Our solution** is to borrow the notion of the **p% rule** from **cell suppression methods**, which identifies a data point as critical, if one can guess the original values with **some error of at most p%**. Thus, **our improved risk measure** is able to evaluate disclosure risk in numeric data. Our Uniqueness-Measure for **"almost exact"** matches provides us with the following outputs:

- **Replication Uniques** | Number of unique records in the synthetic data set that replicates unique records in the original data set w.r.t. their quasi-identifying variables. In brackets, the proportion of replicated uniques in the synthetical data set relative to the original data set size is stated.
- **Disclosure in >= 1 CVar** | Number of replicated unique records in the synthetical data set that have a real disclosure risk in at least one confidential variable, i.e. there is at least one confidential variable where the record in the synthetical data set is "too close" to the matching unique record in the original data set. We identify two records as "too close" in a variable, if they differ in this variable by at most p%. In brackets, the described number is given in proportion to the original data set size.
- **Disclosure in 2 CVars** | Number of replicated unique records in the synthetical data set that have a real disclosure risk in both confidential variables, i.e. in both of the confidential variables the record in the synthetical data set is "too close" to the matching unique record in the original data set. We identify two records as "too close" in a variable, if they differ in this variable by at most p%. In brackets, the described number is given in proportion to the original data set size.

For our selected best parametrized solution in this method-category, we got the following results:

Replication.Uniques	Disclosure.in.>=1.CVar	Disclosure.in.2.CVars
10 (1%)	2 (0.2%)	1 (0.1%)

Perceived Disclosure Risk (R-Package: synthpop)

Unique records in the synthetic dataset may be **mistaken for unique records** based on the fact, that **only the identifying variables match**. This can lead to problems, even if the associated confidential variables significantly differ from the original record. E.g. people might assume a certain income for a person, because they believe to have identified him from the identifying variables. Even if his real income **is not leaked** (as the confidential variables are different), this assumed (but wrong)

information about him **might lead to disadvantages**. The **perceived risk** is measured by matching the unique records among the identifying variables, in our case `sex`, `sat_v`, `sat_m` and `sat_sum`. We applied the method `replicated.uniques` of the synthpop package. There is no fixed threshold that must not be exceeded in this measure, however, a smaller percentage of unique matches (referred to as Number Replications) is preferred to minimize the perceived disclosure risk.

These are the results variables for perceived disclosure risk:

- **Number Uniques** | Number of unique individuals in the original data set.
- **Number Replications** | The number of matching records in the synthetic data set (based only on identifying variables). This is the number of individuals, which might perceived as disclosed (real disclosures would also count into this metric)
- **Percentage Replications** | The calculated percentage of duplicates in the synthetic data

For our selected best parametrized solution in this method-category, we got the following results:

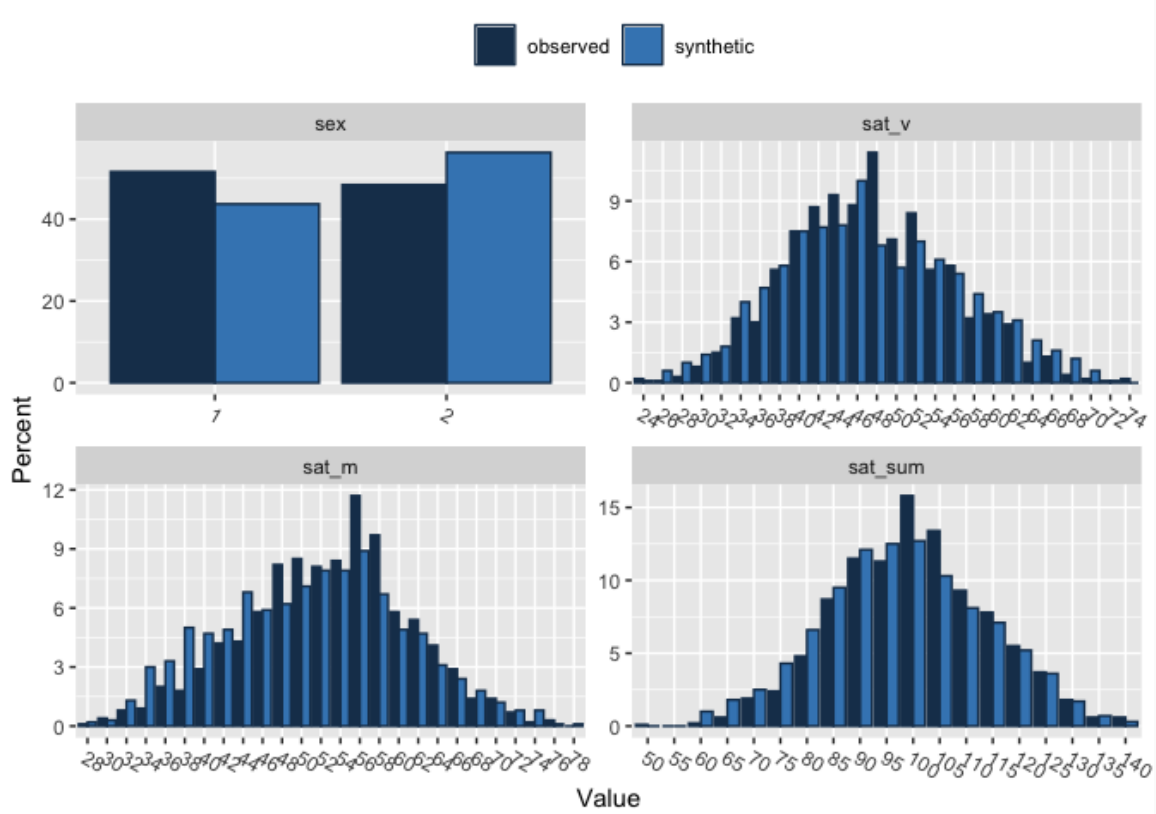
Metric	Number.Uniques	Number.Replications	Percentage.Replications
Perceived Risk	551	10	1

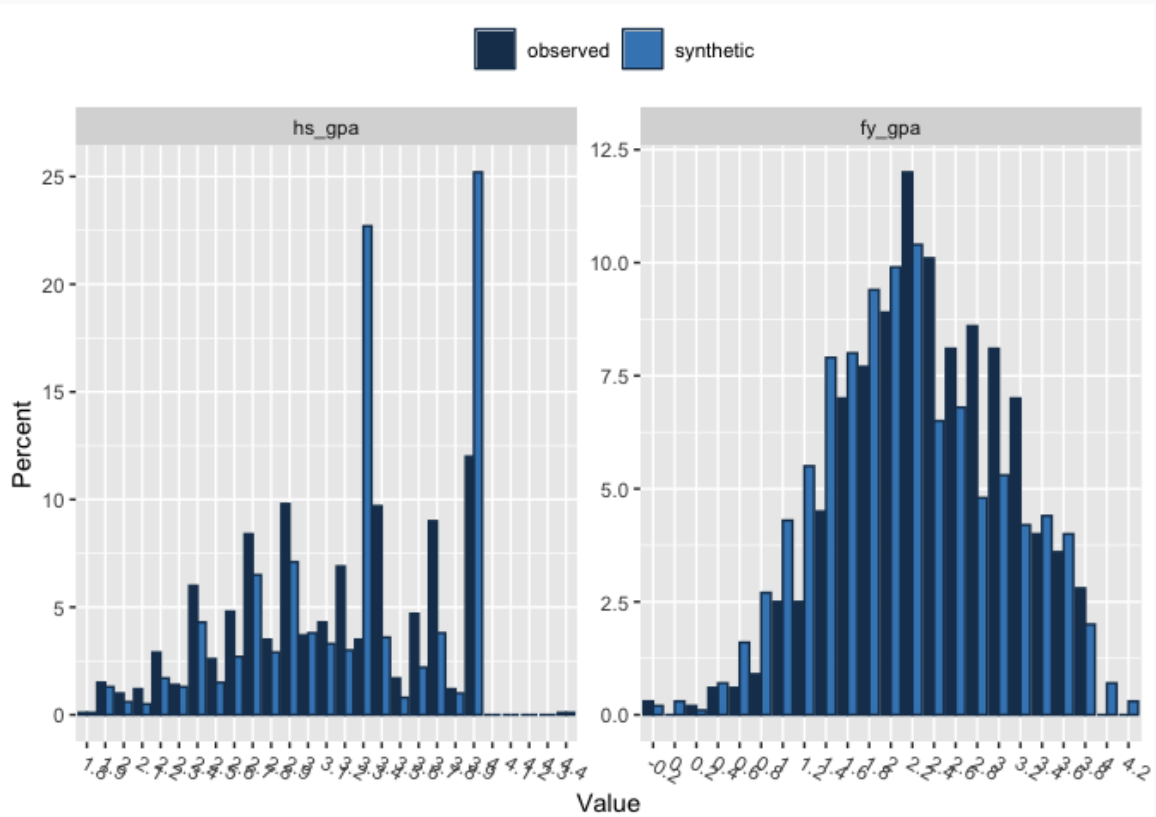
Utility Evaluation

Different utility measures are applied in this section. These utility measures are the basis of the utility evaluation of the generated synthetic dataset. These are the utility measures for our selected dataset after optimization and tuning.

Graphical Comparison for Margins (R-Package: synthpop)

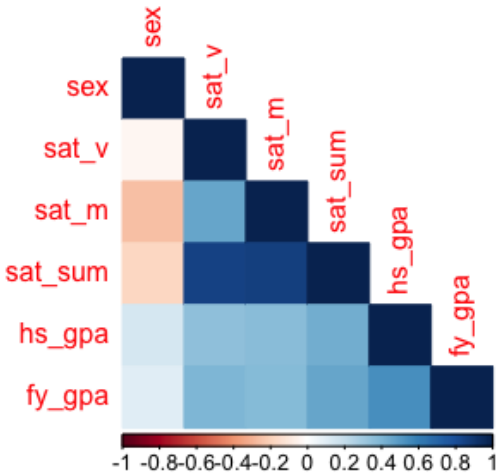
The following histograms provide an ad-hoc overview on the marginal distributions of the original and synthetic dataset. Matching or close distributions are related to a high data utility.

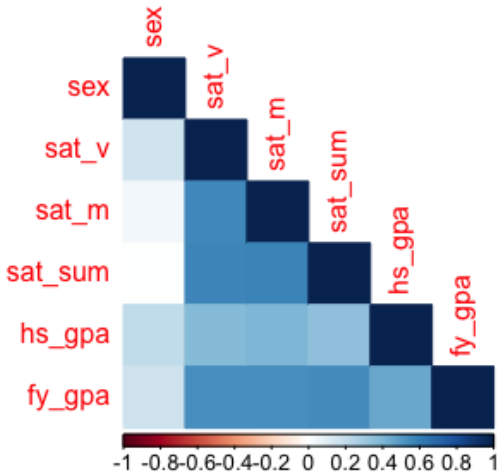




Correlation Plots for Graphical Comparison of Pearson Correlation

Synthetic Datasets should represent the dependencies of the original datasets. The following correlation plots provide an ad-hoc overview on the Pearson correlations of the original and synthetic dataset. The left plot shows the original correlation whereas the right plot provides the correlation based on the synthetic dataset.





Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov test is a classic way to compare (marginal) distributions. A significant result indicates that the two distributions are not identical. The following statistic show the share of variables in the synthetic dataset that have a p-value > 5%. (1: 100%, 0: 0%)

Mean_KS_not_signif
0.17

Mahalanobis Distance for Regression Parameters

To assess testing analysis, coefficients and standard errors calculated based on synthetic dataset should lead to the same results when calculated on the original data. The evaluate differences and taking the variance covariance matrix into account, we calculated the mahalanobis distance between the respective regression parameters. Since we used several regression models, we present the mean of cases in which the mahalanobis distance exceeded the critial value (0: excellent; 1: poor).

Mahalanobis.Distance
1

Distributional Comparison of Synthesised Data (R-Package: synthpop) by (S_)pMSE

Propensity scores are calculated on a combined dataset (original and synthetic). A model (here: CART) tries to identify the synthetic units in the dataset. Since both datasets should be identically structured, the pMSE should equal zero. The S_pMSE (standardised pMSE) should not exceed 10 and for a good fit

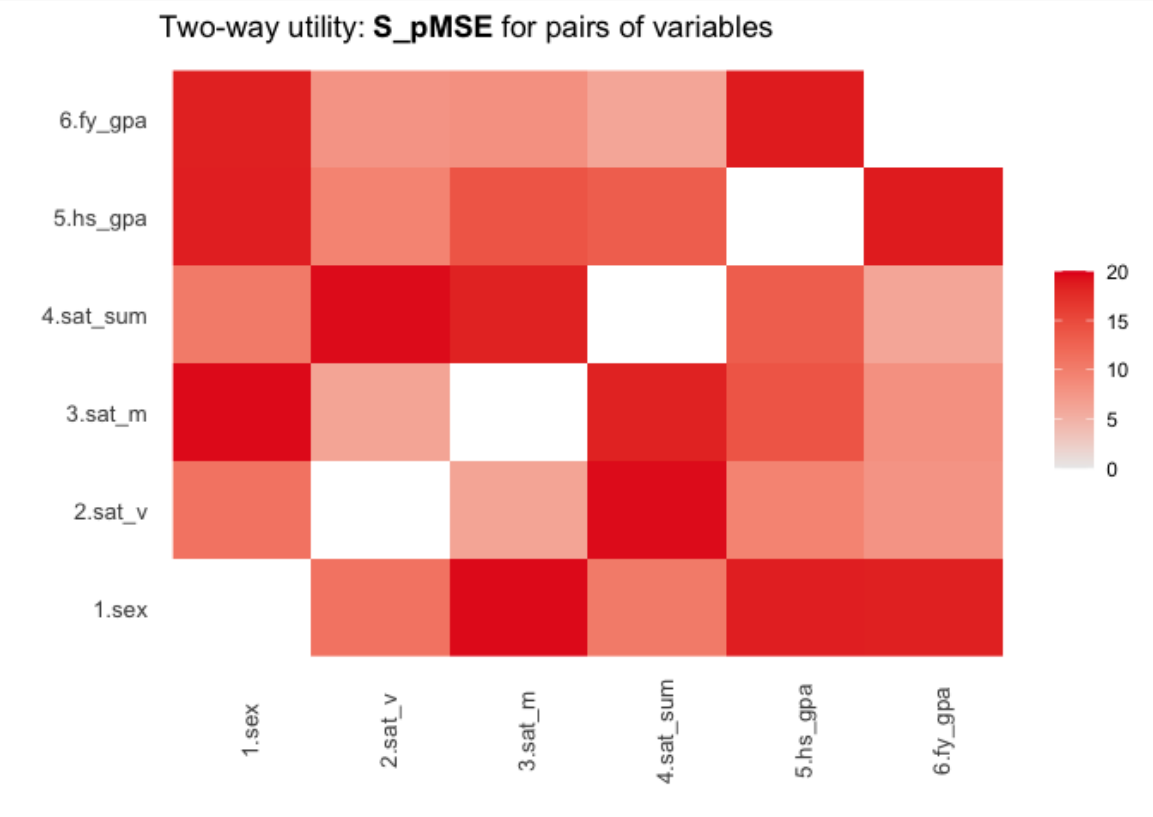
below 3 according to Raab (2021, https://unece.org/sites/default/files/2021-12/SDC2021_Day2_Raab_AD.pdf)

	pMSE	S_pMSE	df
sex	0.0015637	25.019268	1
sat_v	0.0019345	7.738043	4
sat_m	0.0046276	18.510251	4
sat_sum	0.0027942	11.176962	4
hs_gpa	0.0081659	32.663548	4
fy_gpa	0.0069875	27.950000	4

pMSE	S_pMSE
0.1186995	3.03237

Two-way Tables Comparison of Synthesised Data (R-Package: synthpop) by (S_)pMSE

Two-way tables are evaluated based on the original and the synthetic dataset. Here, tables/cells are also evaluated based on pMSE and S_pMSE (see above). We also present the results for the mean absolute difference in densities (MabsDD).



	pMSE	S_pMSE	MabsDD
1.sex:2.sat_v	0.0062636	11.135279	0.252

	pMSE	S_pMSE	MabsDD
1.sex:3.sat_m	0.0110205	19.592004	0.318
1.sex:4.sat_sum	0.0058635	10.424085	0.260
1.sex:5.hs_gpa	0.0104900	18.648892	0.350
1.sex:6.fy_gpa	0.0104128	18.511687	0.322
2.sat_v:3.sat_m	0.0094798	6.319897	0.308
2.sat_v:4.sat_sum	0.0293258	19.550519	0.522
2.sat_v:5.hs_gpa	0.0142756	9.517041	0.396
2.sat_v:6.fy_gpa	0.0117084	7.805609	0.376
3.sat_m:4.sat_sum	0.0276219	18.414620	0.502
3.sat_m:5.hs_gpa	0.0211410	14.093969	0.478
3.sat_m:6.fy_gpa	0.0122524	8.168250	0.328
4.sat_sum:5.hs_gpa	0.0198291	13.219372	0.458
4.sat_sum:6.fy_gpa	0.0092443	6.162844	0.316
5.hs_gpa:6.fy_gpa	0.0282962	18.864111	0.530

Information Loss Measure Proposed by Andrzej Mlodak (R-Package: sdcMicro)

The value of this information loss criterion is between 0 (no information loss) and 1. It is calculated overall and for each variable.

Information.Loss
0.6363493

Individual Distances for Information Loss:

sex	sat_v	sat_m	sat_sum	hs_gpa	fy_gpa
0.475	0.8464092	0.8570983	0.9093859	0.3230122	0.4071904

Tuning and Optimizations

For optimizing the results of the **GAN method** we tried to **optimize parameters** and tried **different types of GAN algorithms**.

Epochs parameter

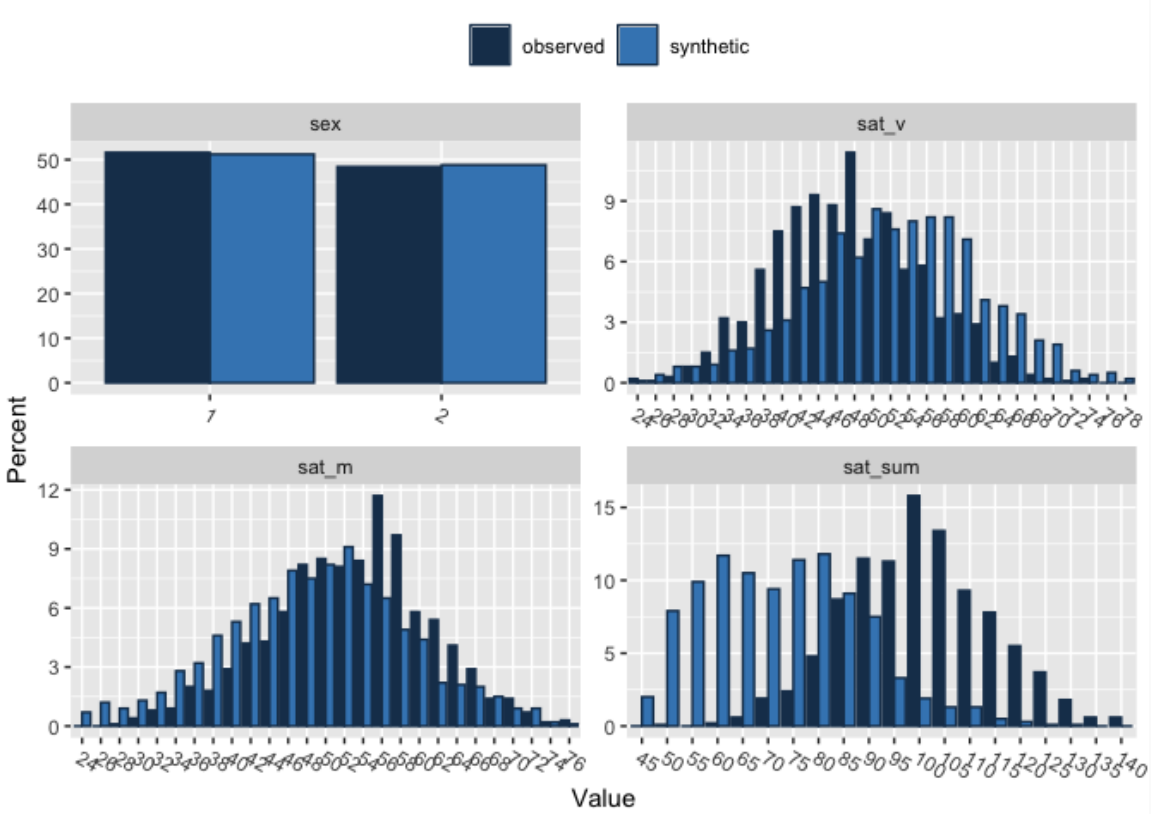
For parameter tuning we tried different parameters for the `epochs` argument of **ctgan**. Clearly it could be seen, the more `epochs` we used, the better the results got (of course with decreasing gains). Or final model was calculated with `epochs = 1000`. The results of the final model can be seen in the utility evaluation chapter. In comparison e.g. the utility result of using `epochs = 10` yielded way worse results.

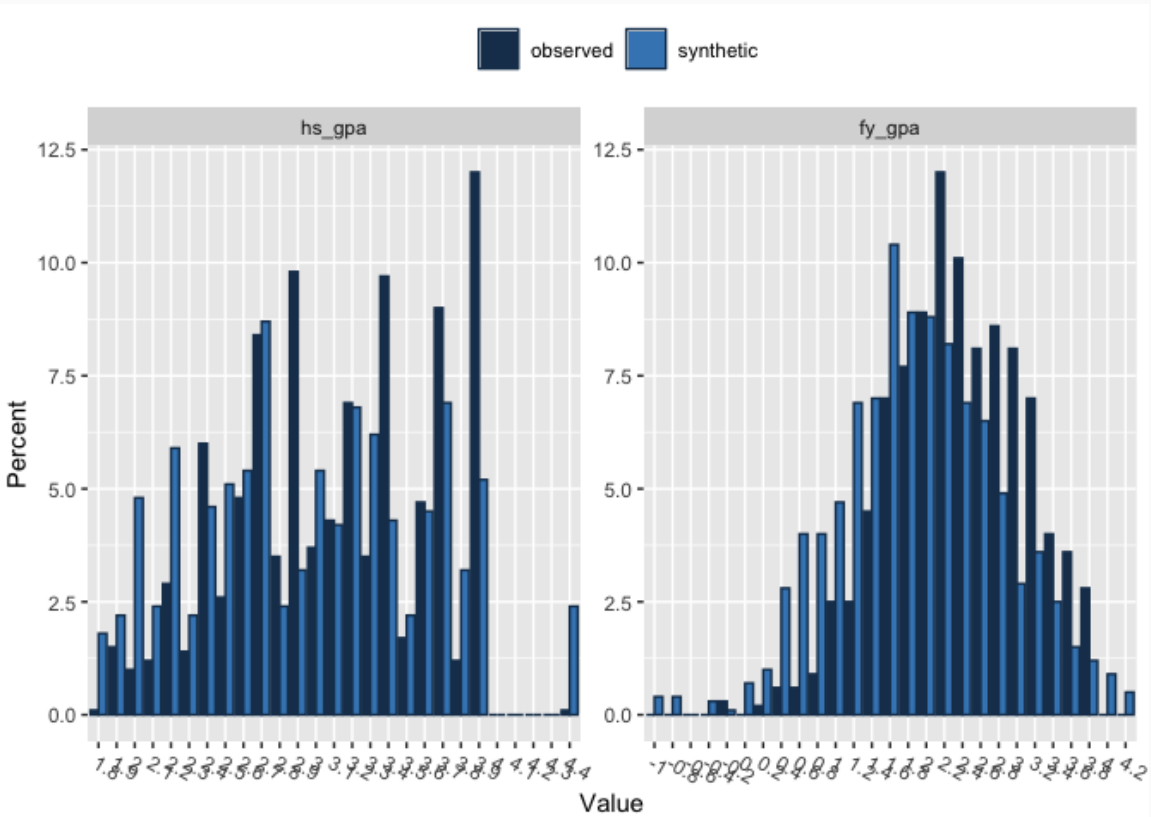
This can be clearly seen from the following **pMSE** and **S_pMSE** values for **ctgan** / `epochs = 10`.

pMSE	S_pMSE
0.2064707	5.131348

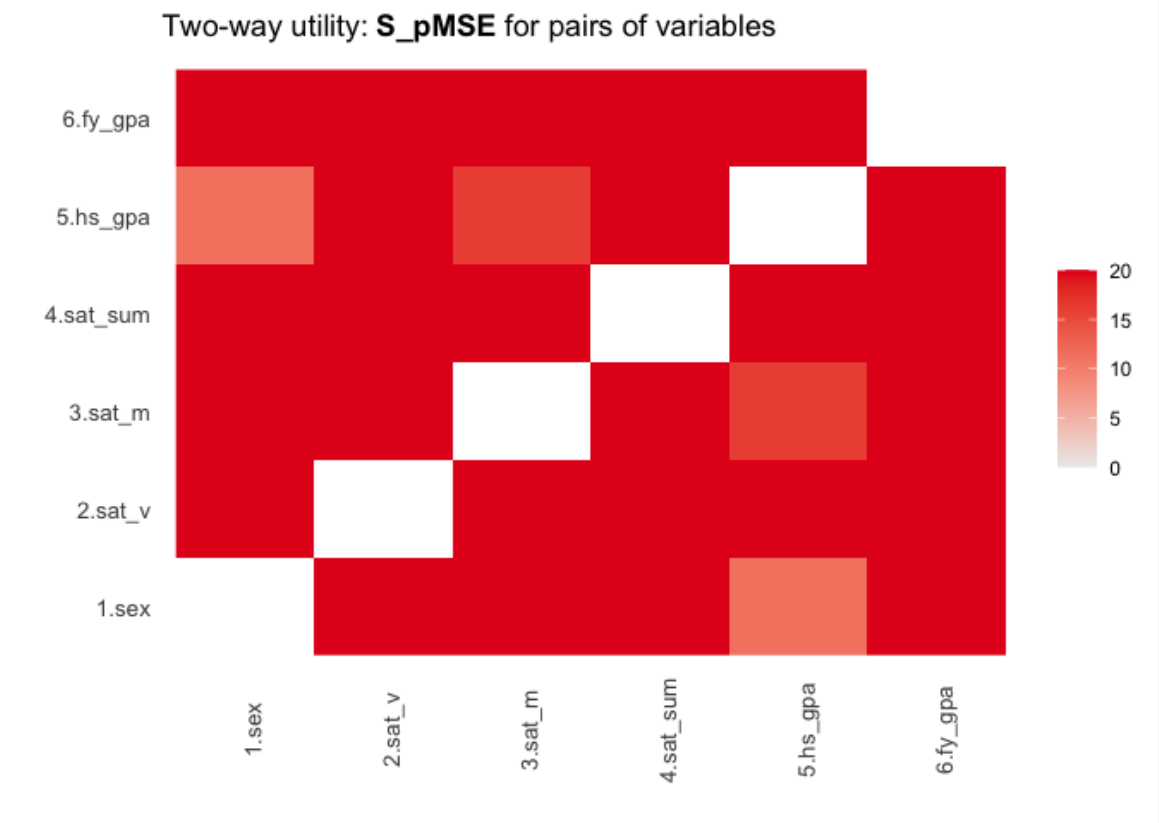
This is clearly worse than the 0.1187 and 3.03237 with epochs = 1000. In our experiments a clear trend of improving S_pMSE numbers with higher epoch values.

The following histograms shows that `epochs = 10` produces way worse results in comparison to our final model.






```
result2$ut$utility.plot
```



Since the risk measures **did not change significantly**, **higher epoch** values were a **clear benefit**. (at the cost of computation time)