WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Analyzing broad spectrum company data using synthetic datasets

The Team

11801716	Felix	Stockhammer
11813383	Kerstin	Scharinger
11838239	Robin	Kulha
11839876	Maroan	El Sirfy

Supervisor: PD Dr. Ronald Hochreiter

Data Coaches: Gabriel Matejka, Harald Wimmer

Data Science Lab

Winter 2020/21

Table of Content

1. Introduction and Project Overview

The goal of this project is to create a synthetic data set that contains information about employees and to perform simple exploratory data analysis. As the original dataset includes sensible information, the aim is to establish an algorithm that keeps valuable information for further analysis in the data while anonymizing private information. The explorative analysis conducted on the synthesized data should then reflect the results if the same analysis is run on the original data.

The project is being conducted in partnership with BDO, an economic and tax counselling firm located in Vienna, Austria.

1.1. What is a Synthetic Dataset?

Synthesizing something means “to form (a material or abstract entity) by combining parts or elements. In tangible terms spoken, it means that something new is being formed by using already existing elements. In our case, we are creating a new entity, which is the synthetic dataset out of an original dataset.

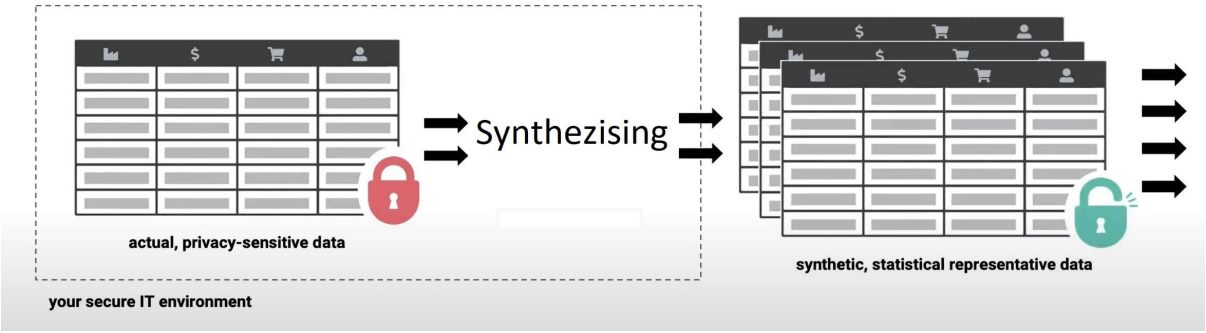

Figure 1: Synthetic Data (Platzer, M; Mostly.ai, 2020)

An important key factor when synthesizing data is that the new set should have all the properties of the dataset used for this creation. Properties can be expressed as (but not limited to):

- ❏ attributes being similarly deviated
- ❏ correlations for variables staying roughly the same
- ❏ summary of the data (mean, min, max) should be almost the same

A synthetic dataset, therefore, is a statistically almost identical version of the dataset with privacy in mind. In the process of synthetization, the algorithm extracts the global structure, patterns, and correlations from the original dataset, to then generate a completely new synthetic dataset. As the synthetic data is sampled from scratch from a probability distribution, the generated synthetic data has no identical relationship to any real data subject.

There are several use cases for synthetic data:

- ❏ Big Open Data: granular data can be shared with researchers
- ❏ Development: as-good-as-real data can be provided to developers and testers
- ❏ Data Monetization: sharing data assets without privacy risk
- ❏ Customer Centricity: Enables broad customer understanding of a company
- ❏ SaaS: providers often need data, the synthetic dataset can be shared for testing
- ❏ Open APIs: enables external developers access to realistic, representative data

In general, a well-made synthesized dataset provides increased data privacy on one side, while maintaining a high data utility on the other side. Synthetization should ensure that no re-identification of real individuals is possible (eg. in the case of synthesizing census data) while preserving usable information for analytical purposes.

1.2. High-Level Problem Statement

Our Data Coaches are facing the problem of the restrictions of data usability by privacy restrictions. They are in possession of a large data set containing a lot of private data which may be valuable for them by gaining some inside from the data. Unfortunately, the data can't be provided to data analysts or us to perform analysis, so the challenge is to implement an algorithm to synthesize before unseen data and perform an exploratory data analysis on the synthetic data. Due to the very high sensibility of the data the synthetization has to ensure the privacy restrictions of the European privacy regulations are supported while all the necessary insights provided by the data remain. For example, the data structure, relationships, and correlation of the original data should be reflected by the synthesized dataset.

1.3. Difference to anonymization

On the surface, synthesized data is also anonymized because it is hard, depending on how strict the synthetization is chosen, to retrace a single, individual entry (which could be a person). Classic anonymization, however, tries to disguise personal information by destroying and changing personal data, which works just fine for small data. However, the data cannot be used for analysis properly as statistical parameters (eg. correlations) are lost during the process. Even then, with enough anonymized data of an individual, it is still possible to completely identify this individual by backwards-engineering.

This is where synthetization comes in. The synthetization process allows us to create a big amount of fake data very similar to the real one, by using the real data to train the machine learning model. Because this data is fake, any re-identification is considered to be impossible. Nevertheless, statistical parameters are maintained with a slight variation for further analysis.

This is advantageous because you do not need a lot of real data to get a huge amount of data to work with. For example, if you need to get a lot of bank transaction data for testing an App,

this method could save a lot of money as you do not need to buy the data or ensure privacy-related issues are being dealt with.

1.4. Anonymization legislation

Modern-day privacy regulations (GDPR, CCPA, etc.) provide strict definitions of anonymity. The European data privacy regulation introduced with the DSGVO is considered the toughest privacy law in the world. To ensure data privacy, three main issues according to Article 29 Working Party of the European Data Protection Board have to be fulfilled. Privacy requires that it is impossible to single out an individual from a single dataset or to link records relating to an individual or inferred information from a dataset concerning an individual.

According to GDPR §4.1 data is considered anonymous if and only if none of the subjects is re-identifiable, neither directly nor indirectly, neither by the data controller nor by any third party. The definition is softened, requiring that the re-identification attack needs to be reasonably likely to be performed. Although it is possible to provide useful anonymization for a single dataset according to the GDP Regulations, the danger of a so-called linkage attack, when a 3rd party successfully joins data to re-identify individuals should be considered. (Platzer, 2020) (Platzer, 2020) (European Parlament, 2018)

2. Data and Overview

2.1 BDO Data

For this project, we are working on a dataset provided by BDO, although “provided” might be the wrong word since we have never worked on the original dataset. Nevertheless, we have gained some insights into the contents of the dataset in various meetings with our Data Coaches. In these meetings, the most important columns have been discussed, to allow us to focus on a small excerpt (attribute wise) from the dataset at first and increase the number of columns included in our algorithm as the project progresses. Furthermore, to get a general idea of the structure of the original dataset, our Data Coaches have provided us with a small excerpt (20 non-original entries) of the original data, which allows us to increase our understanding of the possible contents of each attribute.

In general, the dataset deals with the merits of employees. It is highly complex with attributes between 100 and 500 and between 100.000 and 500.000 entries. The dataset included various categorical and numerical variables which were synthesized and analyzed in the project. As the project progressed, our Date Coaches iteratively provided us with the synthesized dataset created with our code based on the original data. This allowed us to try out our algorithm on a larger dataset on our machines. Later during the project, BDO also sent us an unrelated dummy variable dataset, where we could use more synthesizers to test the general performance and quality.

2.2 External Datasets

After the intermediate consultation, we decided to streamline the process and branch out to different synthesizers to create a comparison. Primarily to measure performance, we also used the large openly available dataset with policy data. The dataset contains data about police employees and comprises various categorical and numerical variables. Results of our attempts to synthesize this dataset can be found further below.

2.3 Process

Tailoring an algorithm to a completely unknown dataset raises some challenges. This holds even for the first steps such as loading the data and determining the structure of the included columns. For example, assigning attributes to be treated as factors or as numeric variables requires some basic knowledge of the dataset. However, such details are very important to be able to accurately synthesize the data. Our approach for handling possible misunderstandings of the data was to gain further knowledge of the dataset in meetings with our Data Coaches. We are aware that this approach leads to a synthetization algorithm that serves the purpose to synthesize this specific dataset rather than a synthetization algorithm that can be used with various input datasets (Overfitting). It is one of the key issues in this field as the nature of synthesizers requires knowledge of the originals to ensure the quality of the data.

Since we did not have access to the original dataset, we were building this algorithm based on our understanding of the data and feedback from previous versions of our algorithm. To assess the performance of the algorithm, the usual process included a meeting with our Data Coaches

in which we presented our changes to the algorithm which was then tested on the original data. Helpful feedback included for example detecting cases in the synthesized dataset which cannot (for structural reasons) occur in the real dataset. For example, we were made aware that a certain project number is an indicator of a project being internal. Such internal projects have certain article numbers that always start with the same number. A case in the synthesized data with an internal project having an article number not starting with these numbers would therefore be an indicator for a faulty synthetization. This again highlights the importance of insights in the original data.

3. Libraries and Packages

Libraries are used as a collection of precompiled routines, that a program can use and are moreover a collection of code (a package or module) that includes useful functions.

The list below represents the libraries and packages in use in the final version of our code.

Since we are both using R and Python for our project, we listed all the required packages for both programming languages.

3.1. Usual Candidates

The most common packages like Tidyverse and Pandas are required. Tidyverse is for R and Pandas for Python. Both are highly advisable for Data Science Project, as they offer useful tools for preparing and analyzing data.

Dplyr as part of one of the most common packages in R “Tidyverse” enables user-friendly dataframe manipulation.

Nearly every scientist working in Python draws on the power of NumPy.

It brings the computational power of languages like C and Fortran to Python, a language much easier to learn and use.

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. Seaborn is in our opinion one of the best and easiest libraries for visualizations in Python.

3.2. Synthetization Algorithms

In order to synthesize our data, we were looking for different packages and algorithms which have been developed before, to get an overview of state-of-the-art methods and to try which algorithm works best in our specific case. Since our Data coaches prefer to work with R, we were looking for R-packages as a first step. We discovered the package “Synthpop” which we used for our first trials of synthetization. In addition, we were searching for alternatives, outside the R-world, and came across the Synthetic Data Vault (SDV) framework for Python. This library comes with several synthetization methods and built-in performance measures, so we gave this a try as well. Moreover, as part of our search for synthetization algorithms, we found several online synthetization platforms (such as mostly.ai or statice.ai), which compete in this growing sector of synthesized data.

It was important for our data coaches to not only achieve good synthetization results with our code but also to get a general overview of the process in case somebody wants to synthesize data from a customer without having access to the data. We made an effort to compare the different algorithms not only regarding synthetization quality but also regarding usability.

3.3. R: Synthpop

The R-Package Synthpop was originally developed to generate synthetic versions of confidential individual-level data for use by researchers interested in making inferences about the population that the data represent. (Synthpop, 2020)

Replacements are generated by drawing from conditional distributions fitted to the original data using parametric or classification and regression tree models. The data is synthesized via the function syn() which can be largely automated if default settings are used, or with methods defined by the user. Optional parameters can be used to influence the disclosure risk and the analytical quality of the synthesized data.

The R-package Synthpop provides some simple methods for synthesizing data. By default, Synthpop comes with a variety of methods, which are listed below. These methods can be defined for each column in order to synthesize the data.

Method	Description	Data type
Non-parametric		
`ctree, cart`	Classification and regression trees	Any
`surv.ctree`	Classification and regression trees	Duration
Parametric		
`norm`	Normal linear regression	Numeric
`normrank`*	Normal linear regression preserving the marginal distribution	Numeric
`logreg`*	Logistic regression	Binary
`polyreg`*	Polytomous logistic regression	Factor, > 2 levels
`polr`*	Ordered polytomous logistic regression	Ordered factor, > 2 levels
`pmm`	Predictive mean matching	Numeric
Other		
`sample`	Random sample from the observed data	Any
`passive`	Function of other synthesised data	Any

Table 1: Built-in synthetization methods in the Synthpop-package (Nowok, Raab, & Dibben, 2016)

3.3.1. Synthpop - Code Complexity

After Synthpop is installed via cran, it is ready to be used in R. However, Synthpop cannot be easily used as a black box to just paste your data in and get a high-quality synthesized dataset out of it. Synthpop per default estimates some correlations between columns of the original data (for example it recognizes collinearity) and then draws samples from the original data. However, without further specification of the synthetization methods, the results are not satisfactory. To use synthpop efficiently, further understanding of the input data is needed. Knowledge about basic relations between the columns helps to set a sequence in which the columns will be synthesized as well as to specify the different methods presented in the table

above for each column where synthpop should not draw a sample out of the original data. Moreover, it is possible to let synthpop determine which predictors will be used for which synthesized column and to adapt these predictors manually before the final synthetization. In addition, users can add their own methods (i.e. other parametric or non-parametric methods) and use them within the synthetization function.

Moreover, the package comes with built-in methods that allow the users to compare the synthesized data with the original data. These methods include linear regressions which are run on the original as well as the synthesized data to compare the coefficients. This function also comes with an appealing visualization of the results.

3.3.2. Synthpop - Quality

Synthpop requires a lot of iterations to improve the quality of the synthesized data. As described above, there are many ways in which the individual user (programmer) can adapt the synthetization algorithm to arrive at satisfying results, eg. changing the sequence, the methods used for each column, or playing with the prediction matrix. However, the larger the original data set is, the more time-consuming the synthetization gets. Therefore, there will always be a trade-off between the quality of the synthetization and the time/efficiency side.

3.4 Python: Synthetic Data Vault Package

The Synthetic Data Vault (SDV) is a Synthetic Data Generation ecosystem of libraries that allows users to easily learn single-table, multi-table, and time-series datasets to, later on, generate new Synthetic Data that has the same format and statistical properties as the original dataset. (Pypi: SDV)

3.4.1. Synthesizer Benchmarking

Since the Synthetic Data Generation ecosystem provides a wide range of Synthesizers, the aim is to choose the most promising one. To evaluate the quality of the Synthetic Data generated with a Synthesizer, the machine learning efficacy by training a machine learning algorithm with synthetic data is used. The performance of the trained model is evaluated by a mixture of different measures like F1 score, accuracy, and R^2 test. The accuracy and F1 evaluate the performance of classification tasks and R^2 is used to evaluate regression tasks. The benchmarking uses eight different commonly used machine learning datasets from different sources and reports the average performance of multiple prediction models. (Xu, Skoularidou, Cuesta-Infante, & Veeramachaneni, 2019)

As depicted in the ‘Comparison of Synthetization methods on real-world datasets' table, there are three Synthesizers clearly standing out, namely CTGAN, CopulaGAN, and TVAESynthesizer.

Comparison of Synthetization methods on real world data sets								
Datasets	adult	census	credit	covtype	intrusion	mnist12	mnist28	news
Synthesizer	f1	f1	f1	macro_f1	macro_f1	accuracy	accuracy	r2
CLBNSynthesizer	0.305	0.286	0.441	0.330	0.385	0.720	0.163	-6.472
CTGAN	0.608	0.329	0.660	0.317	**0.541**	0.122	0.138	-0.070
CTGANSynthesizer	0.602	0.379	0.523	0.330	0.511	0.147	0.112	-0.513
CopulaGAN	0.607	**0.388**	**0.717**	0.327	0.517	0.159	0.149	**-0.056**
GaussianCopulaCategorical		0.000	0.000	0.140	0.177	0.125		-5.029
GaussianCopulaCategoricalFuzzy	0.258	0.054	0.010	0.139	0.256	0.168	0.176	-8.655
GaussianCopulaOneHot	0.198	0.133	0.000	0.182	0.331	0.485	0.493	-36.943
MedganSynthesizer	0.243	0.136	0.024	0.091	0.275	0.377	0.105	-5.601
PrivBNSynthesizer	0.429	0.245	0.011	0.215	0.383			
TVAESynthesizer	**0.619**	0.382	0.000	**0.456**	0.433	**0.779**	**0.769**	-0.245
TableganSynthesizer	0.353	0.272	0.270	0.000		0.087	0.083	-5.821
VEEGANSynthesizer	0.162	0.052	0.166	0.096	0.181	0.375	0.156	-3.10E+08

Table 2: Comparison of Synthetisation methods (Carles & GitHub, 2020)

These three Synthesizers are further explained in this report and exploit on the BDO dataset.

3.4.2. CTGAN

CTGAN, part of SDV, is a GAN-based Deep Learning synthesizer that can generate tabular data by model tabular data distribution and sample rows from distribution. GAN means **generative adversarial network.** The model works with a generator and a discriminator neural network. The generative network generates new data while the data gets evaluated by the discriminative network. The generator trains based on whether it succeeds in fooling the discriminator. The discriminator tries to find out if data points are real or synthesized, so it forms the back-testing part. After the model training process, the learned model can be saved as a pkl file containing information to create synthetic data. Furthermore, the saved model can be loaded, and the desired amount of synthetic data can be created to start the data analysis. (Neha, Roy, & Veeramachaneni, 2016)

3.4.2.1 CTGAN - Code Complexity

It is recommended to install the SDV Package using pip and it runs on Linux, macOS, and Windows systems with Python 3.6, 3.7, and 3.8 installed. The package provides some sample data that can be loaded to evaluate the performance of the synthetization method. Despite that, any dataset can be used to train the model. To use it, the CTGAN function has to be loaded and a model has to be created. The CTGAN model learns from the data and can be saved for further data generation. Unfortunately, data evaluations for us show that some correlations between columns have not remained in the synthetization process. To overcome this problem, the package provides some constraints that can be used to prevent the correlation of specific columns.

Fortunately, the package provides a built-in evaluation function that measures some data quality parameters that can be interpreted and compared.

3.4.2.2 CTGAN - Quality

It can be said that there is a perceptible trade-off between the time of model training and the synthetic data quality. The standard model trains with 300 iterations and two lawyers for the generator and discriminator respectively, per default. Therefore, the training time is really low but unfortunately, the model lacks recognising correlations between variables. By enhancing the training parameters and setting manual constraints for specific variables the correlation of the original data is retained in the model training process. The retention of correlated variables within the synthetic data can be evaluated with a correlation matrix comparing the original and synthetic datasets. As correlated variables may constitute valuable insights in the data analytics process, the retention of correlation can be seen as a quality measure. To ensure data privacy in the synthetization process we identified possible duplicates within the synthetic data and the original data to remove the possible identical data.

3.4.3. CopulaGAN

The CopulaGAN is a variation of the CTGAN Model which enriches the data learning process from the original data. Therefore, the model learns the data with the cumulative distribution function based on the GaussianCopulas. A Copula is a mathematical function that allows one to describe the joint distribution of multiple random variables by analyzing the dependencies between their marginal distributions. As a result, the learning process of the model should be easier to perform and correlations between variables are likely to remain in the newly generated data. The CopulaGAN model provides the same save function for model saving and evaluation function as CTGAN and can therefore be compared with other synthetization models included in the Synthetic Data Vault Python package. (Neha, Roy, & Veeramachaneni, 2016)

3.4.4. TVAE

TVAE is a Variational AutoEncoder based (VAE-based) Deep Learning synthesizer. This method has become very popular for learning the distribution of given data as it performs generally better than other methods.

VAE also has two components in its architecture, the encoder and decoder. The encoder encodes input data into hidden states. In other words, the encoder projects input data onto the lower-dimension vector of which each element has its distribution. Then a vector is sampled from such distributions to obtain a specific input (the hidden state) to the decoder. Last, the decoder tries to decode the input to the input data of the encoder. Note that, different from GAN, whose hidden state distributions are predefined, VAE learns the hidden state distributions during the process.

With this tool, one can load and fit data to create a model. Using this model, synthetic data can be created, of course, also having more entries than the original data. The primary key can be added if this column is only allowed to have unique entries. Another very useful feature is relatively easy to use anonymization of personal information. For instance, if the data contains addresses those can be replaced by fake addresses from a huge list provided by so-called

'Faker Provides'. Basically, the same features as SVD CTGAN are included. (Neha, Roy, & Veeramachaneni, 2016)

3.5 Python: SDMetrics

The SDMetrics library provides a set of dataset-agnostic tools for evaluating the quality of a synthetic database by comparing it to the real database that it is modelled after. It includes a variety of metrics such as:

- Statistical metrics which use statistical tests to compare the distributions of the real and synthetic distributions.
- Detection metrics which use machine learning to try to distinguish between real and synthetic data.
- Descriptive metrics which compute descriptive statistics on the real and synthetic datasets independently and then compare the values.

The evaluation function provides the following synthetization quality indicators:

The '*BayesianNetwork'* Metrics fit a BayesianNetwork to the distribution of the original dataset and evaluates the likelihood of the synthetic data having been sampled from that distribution.

The '*GaussianMixture'* Metrics fit a GaussianMixture model to the distribution of the real data and evaluate the likelihood of the synthetic data having been sampled from that distribution.

The '*DetectionMetrics'* is a trained Machine Learning Classifier that tries to distinguish between the real and synthetic data. The Metrics include a 'LogisticDetection' using a Logistic Regression Classifier to detect whether each row is real or synthetic and evaluate the performance using an Area under the ROC curve metrics. The 'SVCDetection' provides the same procedure using a Support Vector Classifier.

The '*Machine Learning Efficacy Metrics'* evaluates the score obtained by a Machine Learning model fitted on the synthetic data in comparison to those scores obtained by the original data. The Metrics provide a wide range of Machine Learning Algorithm like a 'BinaryDecisionTreeClassifier', 'BinaryAdaBoostClassifier', 'BinaryLogisticRegressionClassifier', 'BinaryMLClassifier', 'MulticlassDecisionTreeClassifier', 'LinearRegressionClassifier', 'MLPRegressor' and 'MLEfficacy'.

The '*MultiSingleColumn'* Metrics computes the average across all columns that are compatible and applies a Single metric on all those columns.

The '*MultiSingleClumn'* Metrics includes the 'CSTest,' a Chi-Squared test for categorical columns that returns the average of the p-values obtained across all columns and the 'KSTest', an inverted Kolmogorov-Smirnov D. statistic für numerical columns using the empirical CDF and returns the average of the KS statistic values.

The '*MultiColumnPairs'* Metrics computes the average across all column pairs that are compatible with a ColumnPairs metric. Including a 'ContinuousKLDivergence' and a 'DiscreteKLDivergence'. (MIT Data To AI Lab, 2018)

These metrics provide a tool to evaluate the synthetic data quality and the usability of the data for further data analysis performed on synthetic data to derive insights from the original data.

3.6 Comparing the synthesizers

In order to compare the different packages and estimate the performance, we used a dataset which included data about police officers in the United States to determine how well certain relations in the dataset were captured in the synthesized data. This was mainly done because we had access to this original data, and we didn't have to meet up with our Coaches for every single iterative change.

It is easy to compare the synthesized dataset with the original dataset as the used packages do usually have built-in functions to assess the performance of the algorithms. In addition, we wanted to compare synthesized datasets from different packages, so we decided to use some more common methods to take a look at possible differences between the performance of the various synthetization methods.

To compare the data without facing our initial challenges (see 5.2 Framework Challenges), we decided, among other things, to build a linear model to compare how accurate numeric values are synthesized and a Chi-Square Test for categorical variables.

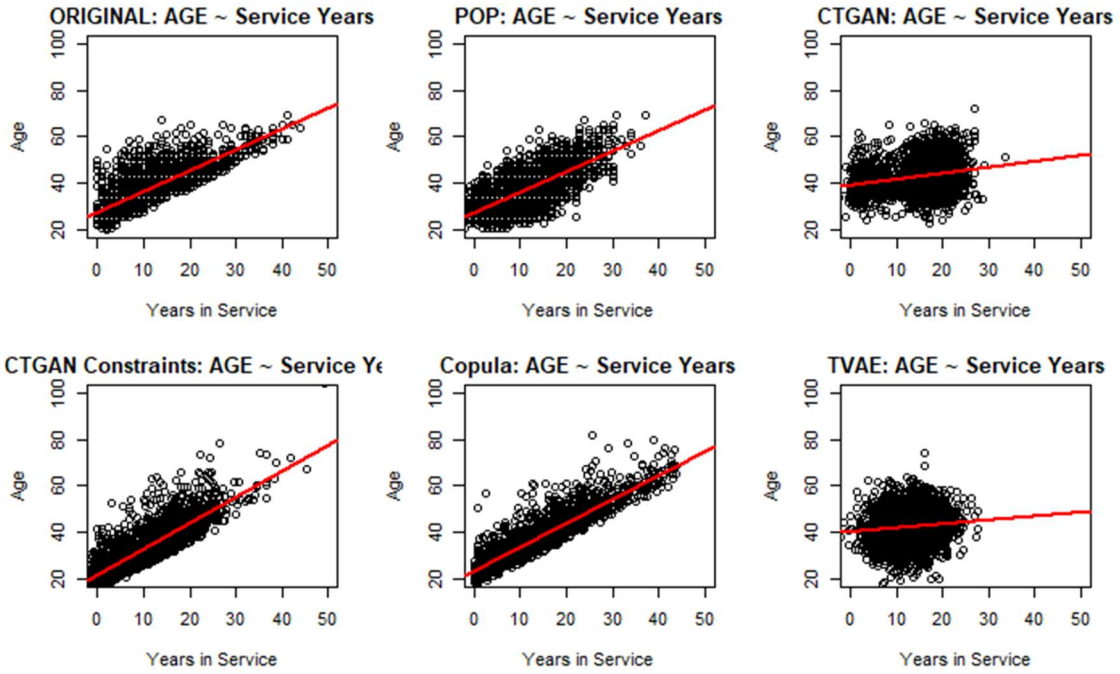

Figure 2: Linear models used on the different synthesized datasets

First, we looked at a linear model reflecting the relationship between the age of a person and the years in service, in other words, the years spent working. Intuitively, the number of service years should be smaller than the age, as can be seen in the original dataset where the lowest age where people started to work was approximately 20. Taking a look at the results, it shows us that our idea to try out different synthesizers was good, taking into account the quantitative accuracy captured in the synthesized data. It allowed us to visualize the differences (at least with numeric variables) in a clear and understandable way. It can be seen that while the

synthpop package captured the regression coefficient nearly perfectly, it did not optimally synthesize the distribution of the two columns, since it delivered negative values. Nonetheless, further improvements regarding the selected methods or sequence might have decreased the error. The GAN-based methods delivered better results by default, although the coefficients were estimated to be greater than in the original dataset.

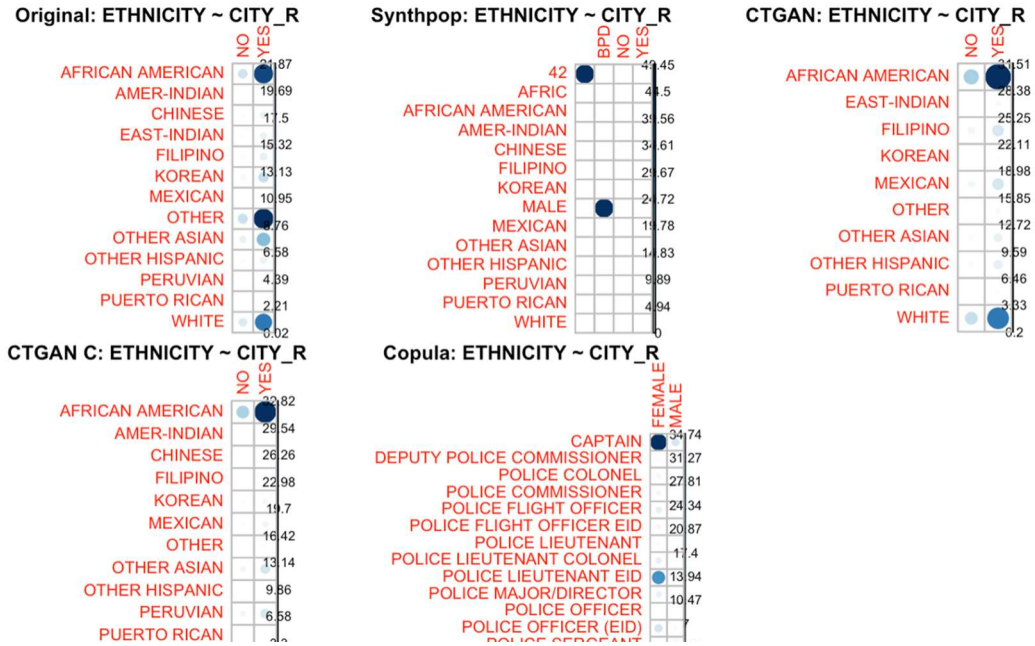

Figure 3: Evaluation of categorical variables in synthesized data

Here, the categorical variables Ethnicity and City Residential of a dummy were used to show the differences of the methods visually. There is a clear similarity of the CTGAN with and without constraints and the original data set, which is a hint for the efficiency of those methods. However, this test should be done with the original dataset. If achievable, it would be helpful to compare all variables with each other and get an average p-value of each method. Though, it remains to be seen if this will be helpful.

To visualize and compare the correlations in the Original data and the Synthetic Data a correlation matrix serves to show correlation coefficients between variables. The figure can be found on the following page. The matrix demonstrates in each cell the correlation between two variables and can be used to summarize the data. A strong correlation between variables is expressed with the maximum value 1 for a positive correlation and -1 for a negative correlation and decreases to no correlation expressed with 0.

According to Xu et al. (2019), TVAE does outperform CTGAN in some cases. However, GANs do have several advantages related to achieving differential privacy. This method includes a generator network that does not have access to the real data during the training process, which makes it beneficial with regards to achieving differential privacy in comparison to TVAE.

Using this information, we were confident to use those methods for the project.

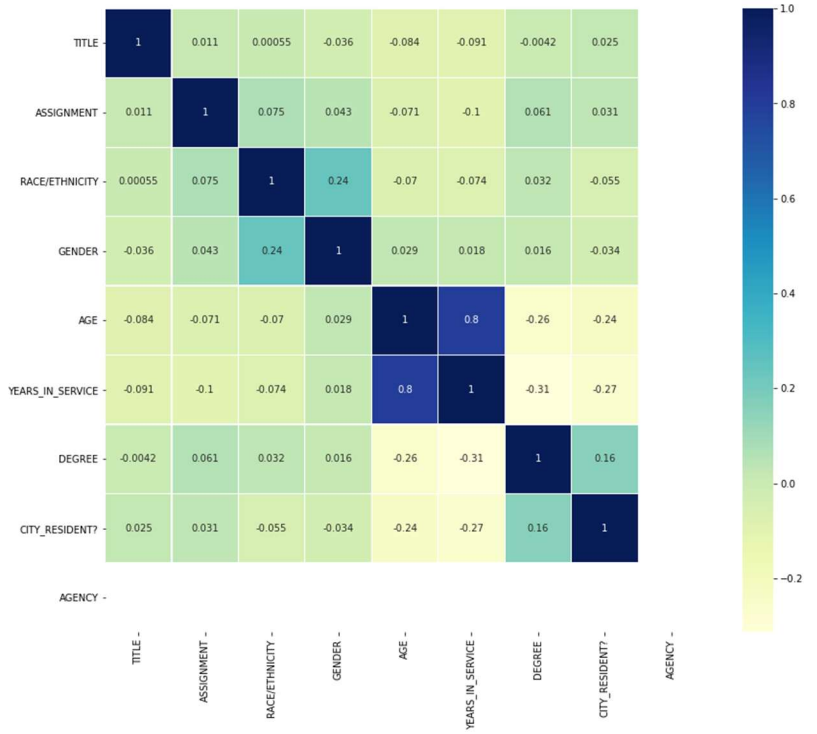

Figure 4: Correlation Matrix of Original Data - Police Data Set

Figure 5: Correlation Matrix of Synthesized Data - Police Data Set (CTGAN with enhanced parameter settings)

4. R - Code

4.1 Synthetisation with Synthpop

Before the synthetization can begin, the original dataset needs to be imported. We provided a code that was tested on our machines with mock-data to our Data Coaches, who then let the code run with the original data. In consultation with our Data Coaches, we decided to select the most important columns of the otherwise massive dataset, to limit the computational complexity of the synthetization algorithm. After that, we assigned the correct data type to each column (eg. factor or numerical).

To increase the performance of synthetization with Synthpop, it is important to have some basic understanding of the data. This includes logical dependencies as well as deciding on optimal methods to synthesize each column (eg. parametric or non-parametric methods). In several trials, we decided on a sequence of synthetization (which column should be synthesized first, second, …), which is important since values from columns that are synthesized first can be used for the models for synthesizing later columns, and on optimal methods for each column. As a result, we created a dataframe which displays the position in the sequence and the method for every column. A small excerpt of this table is displayed below.

	sequence	method
PersonenNr	1	sample
ProjektNr	6	sample
ArtikelNr	7	nested.ProejktID
Leistung	8	cart

Table 3: Excerpt of a table displaying sequence and method of synthetisation on BDO data

During our trials, it was shown that it makes sense to synthesize the person-identification IDs at first, which would then be incorporated to predict later features such as the cost rate. Moreover, treating certain project-related codes as subcategories of the higher-level project number (as can be seen in the method “nested.PROJEKTNR”) also proved to be more efficient than for example creating a decision-tree model for synthesizing the column ARTIKELNR.

Another parameter which can be changed for the synthetisation is the predictor matrix, which represents the columns which are used as a predictor in the model for another column. We have adapted this matrix mainly because of computational performance reasons, by for example stating that if columns A and B are used to generate column C, then column D should be generated by column C instead of A + B + C, but only in cases where this made sense intuitively.

```
#use in synthesis function
synthpop <- syn(df, visit.sequence = sequence.ini, method =
syn.rules$method, predictor.matrix = predictor_matrix, models = TRUE)
```

The actual synthetization happened with the function *syn()* from the Synthpop package. The sequence, method and predictor matrix, which were discussed above, are used within the function. The result of this function is a model which can be used to generate synthetic data. After checking some basic descriptive and distributional properties from the synthesized data in comparison to the input data, we decided to use some basic linear regression models to assess the similarity of the synthesized data with the original data.

```
#use built-in functions to determine the performance of the
synthetisation algorithm.
fit_syn <- lm.synds(SELBSTKOSTEN ~ VERRECHSATZ + SELBSTKOSTENSATZ, data
= synthpop)
#try out model that was fit on synthesized data on original data and
compare performance with Z values
compare.fit.synds(fit_syn, df)
```

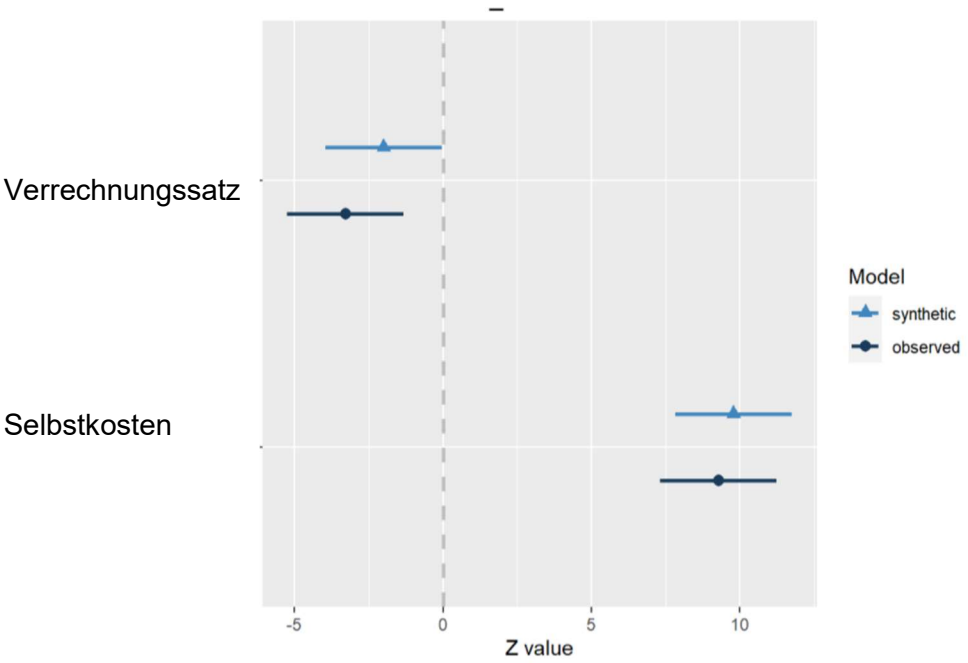

Figure 6: Evaluation of linear model on both original and synthesized data

From the plot, we can see that the Z values for the model on the synthetic data lie in the confidence interval of the Z value of the model on the original data, which shows that our synthetization algorithm captured the relationship between the variables in the model quite accurately.

4.2 Preprocessing for the Analysis

After getting a synthesized dataset, we performed some simple analysis on it. Since the synthesized dataset should provide the same insights as the original dataset, we wrote some code to analyse the synthesized data which could then be used by our Data Coaches on the original. data, with hopefully the same results. Since the dataset deals with the merits of employees, our Data Coaches provided us with some rough ideas of effects they think they might find in their original data. Specifically, they wanted to find out if people changed their behaviour (regarding holidays, sick leave, etc.) before they departed the company. To analyse this behaviour, it is important to determine whether a person has left the company or not. Since there is no entry (e.g. a certain project number) which displays a date when the person has left the company, we decided to determine a threshold date (1 June 2020) and looked whether a person has booked merits since that day or not. If not, we treated the person as someone who has left the company. For that, the last entry for every employee was found out and compared to the threshold date. The next step was to get rid of overrepresented employees and those who only have one entry in our excerpt of the synthesized dataset. Roughly 9.500 entries remained.

Then, we wanted to figure out the date that lays 12 months before the date of the last entry of employees that left the company. These 12 months would provide a sufficient time frame to compare the behaviour in this timeframe with previous entries.

```
# filter so that only employees that left the firm are in the df
left_old <- filtered %>% filter(left == "yes")

# calculate date that is 12 months before the last entry of the employee
months_prior <- lapply(max_date, FUN = function(x) {ymd(x) -
months(12)})

# split df by employee
split_list <- split(left, left$ID)
```

```
# apply: check if an entry happened 12 months before the person left the
company or not
for (i in unique(left$ID)){
  split_list[[i]]$prior <- ifelse(as.Date(split_list[[i]]$datum) <=
as.Date(months_prior[[i]]), "yes", "no")}
```

4.3.1 Analysis - Holidays & sick leave

With this new binary variable, describing whether an entry happened in the 12 months before an employee departed the company or not, we could analyze whether there are differences between these two categories. It is important to note that, as described in the code above, only employees that left the company were considered for this analysis.

First, we checked whether people tended to go on more holidays 12 months before they leave than usual. A cross-table was created with the binary variable described above and the indication whether an entry depicts a holiday (displayed by a certain article number) or not. Using these insights, a Chi-Squared test was performed. As the p-value is above 5% there is no significant result.

```
table(left_new$prior, left_new$ARTIKELNR == "9010")
chisq.test(table(left_new$prior, left_new$ARTIKELNR == "9010")) # no
significant difference, X-squared = 0.72653, df = 1, p-value = 0.394
```

The same concept was used for sick leave with again no significant result. This is primarily due to a low number of employees who had a sick-leave entry before they left.

4.3.2 Analysis - internal projects

With the final analysis, we wanted to find out whether employees usually book more internal projects before they leave which could be a hint for the reason to quit. We created a new column which shows if the project was internal or not (displayed by certain numbers as first digits of the article number) and then split the dataset by unique employees to continue the processing.

A loop over all employees was performed. In each iteration, the proportion of internal projects in the last five entries was calculated as well as the proportion during the employee's whole lifetime in the company. With that, a column was created representing if the proportion of the last five entries was bigger than the overall proportion. The small number of 5 (last entries) was used as the number of entries per employee, in general, was very small. For a more detailed analysis, this would need to be raised, and certain definitions (When does a person count as "left the company"? etc.) need to be re-considered.

```
# loop over employees
for (i in unique(filtered$ID)){

  # calculate proportion of internal projects for the last 5 entries
before the company was left
  last_five <- mean(tail(split_list_all[[i]], n = 5)$internal)

  # calculate proportion of internal projects the whole career of the
employee
  total <- mean(split_list_all[[i]]$internal)

  # check if the proportion in the last 5 entries is higher
  split_list_all[[i]]$higher <- last_five > total}
```

As the table shows, there are some differences.

```
# check table
table(filtered_new$left, filtered_new$internal)
       0    1
  no  6377 2154
  yes  698  223
```

However, the following Chi-Squared test did not reveal any significant result.

```
# calculate chi-squared-test
chisq.test(table(filtered_new$left, filtered_new$internal)) # not
significant
X-squared = 0.42077, df = 1, p-value = 0.5166
```

As the Analysis was not the main goal of the project, we kept the analysed cases and situations as well as the used methods quite simple. There was no pressure to find any significant result since the whole idea was that analyzing both the synthesized and the original data would lead to the same results. In our case, the results were that we did not find any significant effect in the synthesized data, and when our Data Coaches tried the same analysis on the original data, there was no significant result either.

5. Python - Code

5.1 SDV: GAN

We started using Python after the intermediate presentation to try out more synthetization methods. Initially, we used a dummy variable dataset that we synthesized and later on we trained the model on the original BDO data. We installed the required packages in a Jupyter notebook and started training a CTGAN model, a CopulaGAN model and a TVAE model.

The following code shows an excerpt of our python code, in particular the constraints handling, the model training, run time of the training process and the synthetic data generation.

```
model_modify = CTGAN(
    #primary_key='',
    epochs= 300,
    batch_size=100,
    log_frequency = True,
    generator_dim=(256, 256, 256), #change from 2 to 3 generator layer
    discriminator_dim=(256, 256, 256), #change from 2 to 3 discriminator
layer
    l2scale = 1e-07
)
model_modify.fit(orig_modify)
```

```
Wall time: 7min 37s
```

In this code piece, we adjusted CTGAN, included constraints and changed the standard two-layer neural network to three layers, so it was able to learn more about the data. While the training quality increases it is demonstrated that the code run time takes much longer.

```
#evaluation of the quality
evaluate(syn_3, orig_3, aggregate = False)
```

The CTGAN synthetization performance can be evaluated using a built-in function which returns various metrics. A list of all possible metrics is here (3.5 SDMetrics).

	metric	name	score	min_value	max_value	goal
0	BNLogLikelihood	BayesianNetwork Log Likelihood	-18.420681	-inf	0.0	MAXIMIZE
1	LogisticDetection	LogisticRegression Detection	0.230677	0.0	1.0	MAXIMIZE
2	SVCDetection	SVC Detection	0.053217	0.0	1.0	MAXIMIZE
11	GMLogLikelihood	GaussianMixture Log Likelihood	-40.577053	-inf	inf	MAXIMIZE
12	CSTest	Chi-Squared	1.000000	0.0	1.0	MAXIMIZE
13	KSTest	Inverted Kolmogorov-Smirnov D statistic	0.897000	0.0	1.0	MAXIMIZE
14	KSTestExtended	Inverted Kolmogorov-Smirnov D statistic	0.850893	0.0	1.0	MAXIMIZE
15	ContinuousKLDivergence	Continuous Kullback–Leibler Divergence	0.865911	0.0	1.0	MAXIMIZE
16	DiscreteKLDivergence	Discrete Kullback–Leibler Divergence	0.124603	0.0	1.0	MAXIMIZE

Table 4: Evaluation of CTGAN synthetisation using enhanced settings

Comparison of the correlation matrix between original data and CTGAN generated synthetic data:

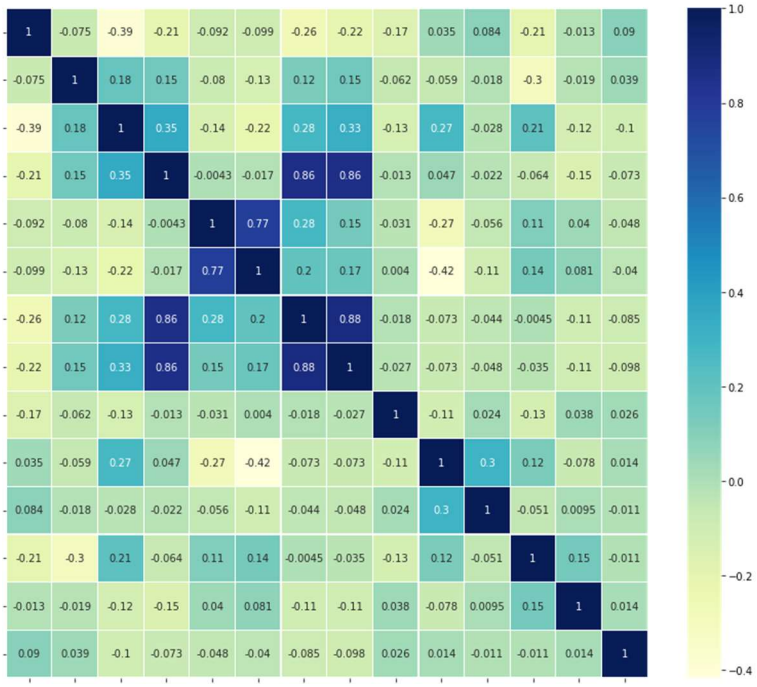

Figure 7: Correlation Matrix of Original Data (BDO Data)

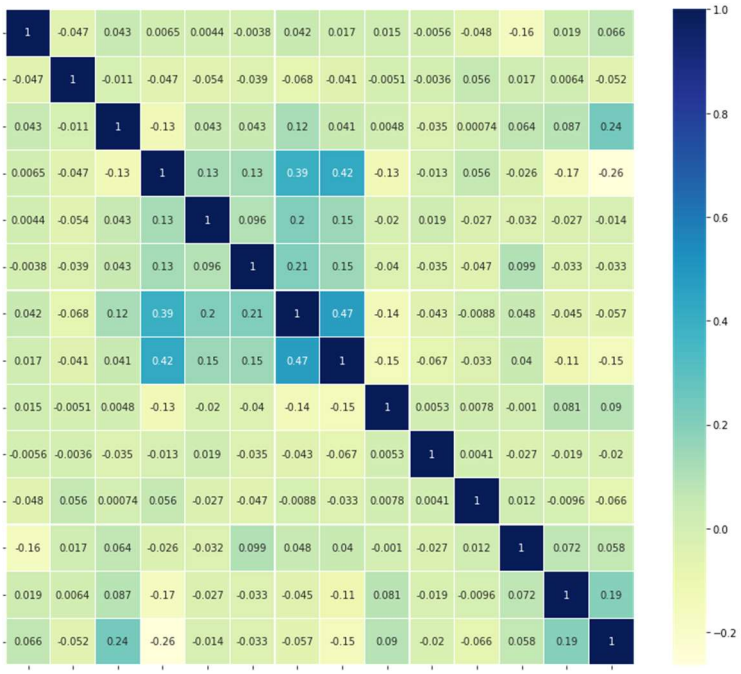

Figure 8: Correlation Matrix of CTGAN-synthesized BDO data, using enhanced parameter settings and 2000 rows of data

5.2 Copula GAN

After loading the original sample data with a length of 20.000, handling missing values and fitting data types, the synthetization was performed. Here, only 200 rows were used took about three and a half minutes to process and some parameters were set as well.

```
%%time
random.seed(123)

model_final = CopulaGAN(
    constraints=constraints,
    #primary_key='',
    epochs= 500,
    batch_size=100,
    log_frequency = False,
    generator_dim=(256, 256, 256), #change from 2 to 3 generator layer
    discriminator_dim=(256, 256, 256), #change from 2 to 3 discriminator layer
)

model_final.fit(orig_final)
# Wall time: 3min 39s
```

The Copular GAN synthetization is evaluated just with CTGAN, using the built-in function.

```
#evaluation of the quality
evaluate(syn_final, orig_final, aggregate = False)
```

	metric	name	score	min_value	max_value	goal
0	BNLogLikelihood	BayesianNetwork Log Likelihood	-18.420681	-inf	0.0	MAXIMIZE
1	LogisticDetection	LogisticRegression Detection	0.209849	0.0	1.0	MAXIMIZE
2	SVCDetection	SVC Detection	0.087088	0.0	1.0	MAXIMIZE
11	GMLogLikelihood	GaussianMixture Log Likelihood	-158.331651	-inf	inf	MAXIMIZE
12	CSTest	Chi-Squared	0.999966	0.0	1.0	MAXIMIZE
13	KSTest	Inverted Kolmogorov-Smirnov D statistic	0.659286	0.0	1.0	MAXIMIZE
14	KSTestExtended	Inverted Kolmogorov-Smirnov D statistic	0.800000	0.0	1.0	MAXIMIZE
15	ContinuousKLDivergence	Continuous Kullback–Leibler Divergence	0.386444	0.0	1.0	MAXIMIZE
16	DiscreteKLDivergence	Discrete Kullback–Leibler Divergence	0.097226	0.0	1.0	MAXIMIZE

Table 5: Evaluation metrics of CopulaGAN synthesizer

The results are not consistent – some are very good, others are at the lower end. However, most of the metrics weren't as good as with CTGAN.

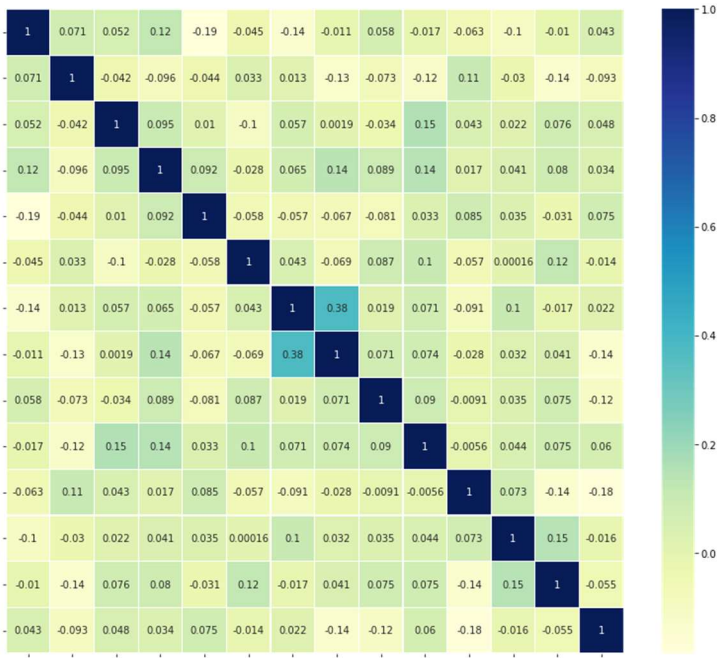

Figure 9: Correlation Matrix of Copula GAN-synthesized BDO data with enhanced parameter settings and constraints

5.3 TVAE Model

Synthetization & Evaluation

Just as with Copular GAN some preprocessing was performed. This time, only 2.000 rows were used as that alone took six minutes to process and some parameters were set as well.

```
orig_3 = orig_data[:2000]
```

```
model3 = TVAE(
    epochs=500,
    batch_size=100,
)
model3.fit(orig_3)
```

The TVAE synthetization performance can be evaluated just like with Copular GAN.

	metric	name	score	min_value	max_value	goal
0	BNLogLikelihood	BayesianNetwork Log Likelihood	-18.137605	-inf	0.0	MAXIMIZE
1	LogisticDetection	LogisticRegression Detection	0.430553	0.0	1.0	MAXIMIZE
2	SVCDetection	SVC Detection	0.147044	0.0	1.0	MAXIMIZE
11	GMLogLikelihood	GaussianMixture Log Likelihood	-43.089870	-inf	inf	MAXIMIZE
12	CSTest	Chi-Squared	1.000000	0.0	1.0	MAXIMIZE
13	KSTest	Inverted Kolmogorov-Smirnov D statistic	0.823000	0.0	1.0	MAXIMIZE
14	KSTestExtended	Inverted Kolmogorov-Smirnov D statistic	0.857321	0.0	1.0	MAXIMIZE
15	ContinuousKLDivergence	Continuous Kullback–Leibler Divergence	0.787863	0.0	1.0	MAXIMIZE
16	DiscreteKLDivergence	Discrete Kullback–Leibler Divergence	0.182706	0.0	1.0	MAXIMIZE

Table 6: Evaluation Metrics of TVAE synthesizer

Some of the tests are outstanding like the Chi-Squared test, others however only reached a low score.

Privacy Test

A Personally Identifiable Information (PII) test was performed to check whether the synthetic data is safe so that no personal information of the original data is exposed or even used in the synthetic dataset. For that, a pandas function was used.

```
pd.concat([orig_data, syn_3]).duplicated().sum()
```

Fortunately, there are no identical entries in the original sample data and the synthesized, just as it should be.

Also, if there should be duplicates in the synthetic data they shall be deleted. For having an overview, a data frame was created including an extra column whether a row is a duplicate.

```
#delete duplicates from Synthetic data
a=len(orig_data)
b=len(syn_3)
df=pd.concat([orig_data, syn_3])
df['is_duplicate']=df.duplicated()
syn_dup=df[a:]
syn_tested = syn_dup[syn_dup.is_duplicate == False]
```

In our case, the newly gathered data had no duplicates from the beginning, probably because our dataset has enough variables and not too many entries. Perhaps this could occur with some data with fewer columns though.

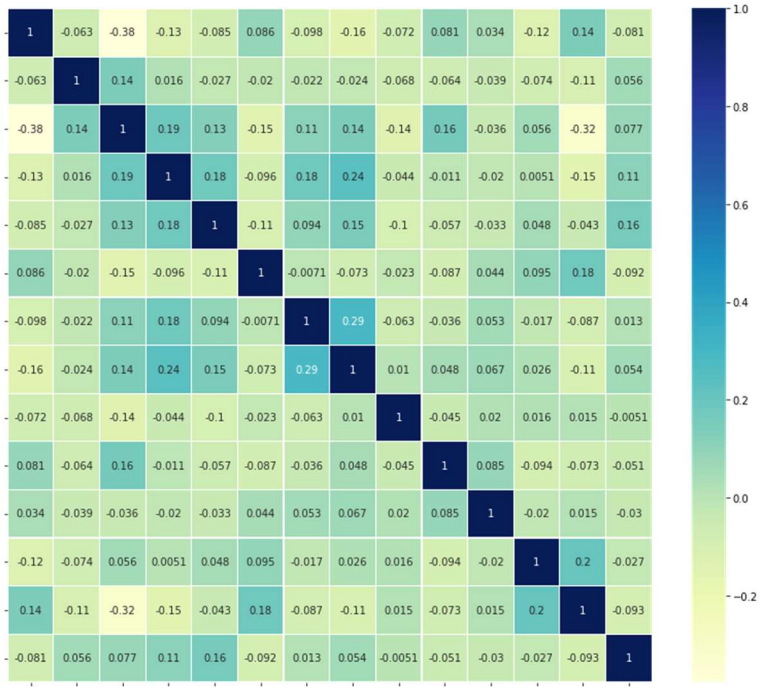

Figure 10: Correlation Matrix of Synthetic Data (TVAE)

6.Challenges

6.1 Project Challenges

Trying to find appropriate literature for our project was not an easy task, as this research area is quite new. We managed to obtain working papers from the R Package Synthpop and links

from blog posts of the Austrian Start-up Mostly.ai. For other packages, we relied mostly on GitHub and other package documentation platforms where we could find explanations.

For the Synthpop Package, we observed performance issues with our algorithm synthesizing the huge data set between 100.000 and 500.000 observations. As the algorithm had not stopped calculating the process had to be interrupted. This is an issue we had to consider in our programming and had to come up with a solution. So far, we tried performance tweaking the code for synthpop.

Another challenge was that the GAN package library (CTGAN) is most commonly used in python, while synthpop is used in R. To compare the packages equally, the synthesized datasets have to be transferred and must be compared using similar methods.

6.2 Framework Challenges

We had no access to the original data but still wanted to test the performance of our algorithm regularly. A solution to mitigate this problem was to use a different data set. It had to be a large dataset for experimenting on the general accuracy of our synthetization algorithm as well as performance.

Overall, the challenges regarding the general framework of the project can be put into three main bullet points:

- Technical requirements of the client

Depending on which programming language(s) we used for the final synthetization, our customer, in this case, BDO, needed to have relatively easy access to that language and packages. Also, the packages we were going to use, should work on their computer. Furthermore, the running time of the synthetization code had to be within limits.

- Data cleaning

Another very important aspect was the simplicity for the customer to run the code. The best thing would be to upload/set a dataset, press a button and get a synthetic dataset. However, for that data cleaning is of greater importance as well.

As this was not the main focus of this project, we did not focus on that topic. Thus, we only provided a simple data cleaning algorithm (mostly for the synthesized data in the python packages).

- Data synthetization method

Of course, this step was the crucial one. Choosing a synthetization method was not easy. After analysing those methods, Synthpop and GAN-based synthesizer we can say that they have different advantages, which are also dependent on the dataset itself.

All in all, we have learned that the process of synthesizing an unknown dataset is a very iterative process. Our example showed that it is hard to just “throw” a synthetisation algorithm

on a dataset and expect a useful synthesized dataset. In our understanding, a deeper understanding of the original dataset, its background and possible correlations between attributes is helpful or even needed, to achieve satisfying synthetization results.

7. Recap & Future Work

7.1 Reflection

During the beginning of the project, we spent a lot of time researching synthetic data and possibilities to synthesize a dataset most efficiently. Moreover, as we dealt with a large, unknown to us dataset, we had to understand the structure and variables before.

As part of the iterative process with our data coaches, we had a lot of ideas concerning the methods on how to streamline the approach of using anonymous data and creating synthetic data. Some of them were later going to be an integral part of our project (Using GAN methods and other datasets for example), while others were disregarded as being either too impractical or time-consuming to stay in-scope of our project.

Since the intermediate presentation, we tried to figure out which algorithm or method would work best for our problem. We started with the R Synthpop Package since this package was used and recommended by many researchers and the Census bureau. We wanted to program our algorithm in R as it was also the preferred interface for our client. But we were not sure if the methods of Synthpop with dependent distributions of variables and random sampling and decision trees would be able to deliver our desired results to create synthetic data. Therefore, we tried to dive into deep generative models (GAN) to generate synthetic but realistic data by training a machine learning model on the original data to replicate its structure and content information.

This worked well for the most part, as we tested it with several built-in methods. We also built handcrafted linear models for all the synthesized datasets to be able to compare numeric variables and their dependencies. From there on until the end of the project we tried to finetune parameters and get GAN as well as synthpop to be more accurate for categorical variables while maintaining somewhat acceptable performance.

Since analyzing the data was in-scope of our project, there was linearly more time spent on it as the project progressed. Initial analysis included the understanding of the dataset and comparing methods and parameters. Upon reaching an acceptable level of representativeness and accuracy in the synthetic datasets, we also spent time on data analysis to discover possible effects with regards to people that have left the company.

7.2 Future Work

As we only had one semester for this project, there were a few things that were out of scope. From the beginning, it was clear that we could not write our own synthesizer from scratch for various reasons. Secondly, during the project we thought about different ways on how to “anonymize” columns of the dataset, however, we quickly realized it would be too time-consuming. Moreover, because of the time constraint, there was a limit on how in-depth we could compare the packages we used.

As we focused our analysis on the correlations of numeric variables and used Chi-Squared for the categorical ones, it would be interesting to use different methods to compare the synthetic data. One could use various statistical methods, other than the ones we used, to compare the datasets. Based on this, different parameters for the synthesizers could be worked out.

In addition to improving the methods of a synthesizer, engineering could improve by:

- Thinking about how pre-processing would need to change as datasets scale beyond a single machine (synthesize big data)
    - Using platforms such as Apache Spark for Python to deal with larger data
- Thinking about how processing and algorithms would need to change with big data
    - Generally, changes that must be made to accommodate a larger dataset, for example splitting the data, only taking a percentage of the data etc.
- dealing with data sparsity, while not knowing the original data

Beyond that, synthetic data is a very active field of research and there are many more interesting techniques to discover (from active learning algorithms, reinforcement learning with GAN to other methods for creating synthetic data), which could be investigated to deliver well-rounded synthetic datasets.

Last but not least, for the analysis, we only used attributes we thought (in coordination with BDO) seemed important. A broader analysis of the original dataset would therefore need to include more attributes than the ones we have worked with.

Future work could include how to deal with this issue as well as the others mentioned above.

8. Team and Responsibilities

We as a team managed to have a meeting each week to discuss our research, coding and new ideas. We discussed our ideas as a group and tried to split up the work according to the strengths of each team member. Although we agreed, a good project depends on good teamwork and for the most part, we did work as a team, we assigned the following responsibilities to the team members to ensure each important task is properly supervised:

Felix: Synthetization, minutes writer

Kerstin: Synthetization, evaluation, coordination

Maroan: Explorative Analysis, visualisation of descriptives

Robin: Explorative Analysis, report structuring

9. References

Carles, S., & GitHub. (2020, October 17). *SDGYM Leaderboard*. Retrieved January 26, 2021, from https://github.com/sdv-dev/SDGym/blob/master/sdgym/leaderboard.csv

European Parlament. (2018, April 10). *Datenschutzkonferenz Online.* Retrieved January 12, 2021, from Article 29 Data Protection Working Party: https://www.datenschutzkonferenz-online.de/media/wp/20180416_Article29WPGuidelinesonConsent_publishpdf.pdf

MIT Data To AI Lab. (2018). *Singel Table Metrics*. (MIT Data To AI Lab) Retrieved from SDV: https://sdv.dev/SDV/user_guides/evaluation/single_table_metrics.html

Neha, P., Roy, W., & Veeramachaneni, K. (2016, October 17). *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Montreal, QC.* Retrieved December 11, 2020, from The Synthetic Data Vault: https://dai.lids.mit.edu/wp-content/uploads/2018/03/SDV.pdf

Nowok, B., Raab, G., & Dibben, C. (2016). *Synthpop: Bespoke Creation of Synthetic Data in R*. Retrieved October 21, 2020, from https://cran.r-project.org/web/packages/synthpop/vignettes/synthpop.pdf

Platzer, M. (2020, 11 04). *Truly anonymous synthetic data - evolving legal definitions and technologies (Part I)*. (Mostly.ai) Retrieved 01 12, 2020, from https://mostly.ai/2020/11/04/truly-anonymous-synthetic-data-legal-definitions-part-i/

Platzer, M. (2020, 11 04). *Truly anonymous synthetic data - evolving legal definitions and technologies (Part II)*. (Mostly.ai) Retrieved 01 12, 2020, from https://mostly.ai/2020/11/04/truly-anonymous-synthetic-data-legal-definitions-part-ii/

Platzer, M; Mostly.ai. (2020, September 25). *The Worlds Most Accurate Synthetic Data Platform?* Retrieved October 07, 2020, from https://mostly.ai/2020/09/25/the-worlds-most-accurate-synthetic-data-platform/

Pypi: SDV. (n.d.). Retrieved December 3, 2020, from https://pypi.org/project/sdv/

Synthpop. (2020). *About Synthpop.* Retrieved from https://www.synthpop.org.uk/index.html

Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019, October 28). *Modeling Tabular Data using Conditional GAN*. Retrieved December 11, 2020, from https://arxiv.org/pdf/1907.00503.pdf

10. List of Tables and Figures

11. Appendix

M. Platzer, "The World's Most Accurate Synthetic Data Platform? Let's check the Numbers!," Mostly.ai, 25 09 2020. [Online]. Available: https://mostly.ai/2020/09/25/the-worlds-most-accurate-synthetic-data-platform/. [Accessed 07 10 2020].

B. Nowok, G. Raab and C. Dibben, "Guidelines for producing useful synthetic data," 2016. [Online]. Available: https://arxiv.org/pdf/1712.04078.pdf. [Accessed 17 October 2020].

Pasieka, M; Mostly.ai;, "The evaluation of synthetic data: a comparison of three data generation methods," 28 September 2020. [Online]. Available: https://mostly.ai/2020/10/28/comparison-of-synthetic-data-types/. [Accessed 14 November 2020].

"SDV," [Online]. Available: https://sdv.dev/SDV/index.html. [Accessed 13 December 2020].