On the Utility of Synthetic Data: An Empirical Evaluation on Machine Learning Tasks

Markus Hittmeir
SBA Research
Vienna, Austria
mhittmeir@sba-research.org

Andreas Ekelhart
SBA Research
Vienna, Austria
aekelhart@sba-research.org

Rudolf Mayer
SBA Research
Vienna, Austria
rmayer@sba-research.org

ABSTRACT

With the recent advances and increasing activities in data mining and analysis, the protection of the privacy of individuals is crucial. Several approaches address this concern, from techniques like data anonymisation to secure, non-disclosive computation, all of which have their specific strengths and weaknesses, depending on the specific requirements. A slightly different approach is the generation of *synthetic data*, which tries to preserve the overall properties and characteristics of the original data without revealing information about actual individual data samples. The promise is that, for most purposes, models trained on the synthetic data instead of the real data do not show a significant loss of performance. In this paper, we give an overview on currently available approaches for synthetic data generation, and empirically evaluate the utility of the generated synthetic data by testing them on a number of supervised machine learning tasks on several publicly available datasets.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning**; • **Security and privacy** → **Data anonymization and sanitization**; *Usability in security and privacy*; *Privacy protections*;

KEYWORDS

Privacy-Preserving Data Mining, Synthetic Data, Machine Learning

ACM Reference Format:
Markus Hittmeir, Andreas Ekelhart, and Rudolf Mayer. 2019. On the Utility of Synthetic Data: An Empirical Evaluation on Machine Learning Tasks. In *Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019) (ARES '19), August 26–29, 2019, Canterbury, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3339252.3339281

1 INTRODUCTION

Due to technological advances both in collection and storing of data, an ever increasing amount of micro-data, i.e. data that contains information about individuals, is collected. This includes domains such as health care, employment, or social media. Along with these developments, rapid advances in data analysis entail an increasing interest in accessing and mining micro-level data. This is a serious threat to the privacy of the individuals affected. If not for ethical reasons, several regulations, such as the EU's General Directive on Data Protection (GDPR), which came into effect in May 2018, put restrictions on how data can be collected, shared and utilised. However, it is still often required to publish data bases, either to a limited number of recipients, or generally to the public, especially in research settings. Traditional approaches to comply with data protection and privacy aspects in these settings often include anonymisation of data before publishing or processing, such as in the approaches of k-anonymity [11] or *differential privacy* [3]. For a detailed overview on privacy-preserving methods, see [2, 4]. K-anonymity has been shown to be still prone to linkage attacks, when adversaries have background knowledge and access to other data sources. Differential privacy, when applied to the model or the output of the model, on the other hand is not applicable for all types of analysis techniques. Both approaches distort the data records to some extend, which has potentially negative effects on the utility of the data and the models subsequently trained upon, as e.g. demonstrated in [5]. An alternative to anonymisation is the generation of *synthetic data*. This is data that has been generated from an original dataset (that cannot easily be shared), with the aim of preserving the global properties and relations between the attributes in the dataset without revealing the individuals described by the data. This allows the synthetic data to be shared much more easily, while in theory it should remain possible to train machine learning models on the synthetic data that ultimately achieve effectiveness comparable to models trained on the real data. In this paper, we will provide an analysis of the effects on utility of the synthetic data, as well as the privacy implications.

The remainder of this paper is structured as follows. Section 2 will provide an overview on related work and will describe our selected, publicly available approaches for generating synthetic data. After explaining our experiment setup in Section 3, we discuss the utility and privacy aspects in our evaluation in Section 4. Finally, we provide conclusions and an outlook on future work in Section 5.

2 RELATED WORK

One of the earliest usages of synthetic data was in the partial synthetic data approach by [9], where certain columns are generated synthetically. An overview on more than 20 approaches is given in [10], categorising the approaches into fully or partially synthetic, as well as identifying whether they are based on the original data. Commercial solutions for data synthetisation also gain momentum, e.g. by *mostly.ai* [1]. Due to licensing issues, we could not include

[1] https://mostly.ai/synthetic-data-engine.html

ARES '19, August 26–29, 2019, Canterbury, United Kingdom

ACM ISBN 978-1-4503-7164-3/19/08…$15.00
https://doi.org/10.1145/3339252.3339281

commercial tools in our evaluation. Naturally, details about the synthetization process are not so readily available in this case. However, specifically for *mostly.ai*, it is stated to be built on neural network models, such as auto-encoders (see e.g. [8] for a discussion on auto-encoders). One other example of the use of auto-encoders for synthetic data is reported in [1]. They have also been used extensively in the generation of synthetic *image* data, e.g. [12].

For selecting the synthetic data generation tools included in our analysis, we focused on recent methods that are open-source implementations and utilise a powerful generative model to ensure high-quality data. Both approaches we identified, the *Synthetic Data Vault* and the *DataSynthesizer*, are developed in the *Python* programming language.

2.1 Synthetic Data Vault

The Synthetic Data Vault (SDV) has been developed by Patki et al. in 2016 [6]. The source code and documentation of the package can be found at [2]. The modelling and sampling process of SDV consists of three steps. First, the *DataNavigator* extracts all relevant information from the dataset and transforms the contents into numerical values. The *Modeler* then creates generative models of the input, and is subsequently passed to the *Sampler* for generating synthetic rows of data. Prior to this process, the user has provide the data in Comma-Separated-Value (CSV) files. Furthermore, basic information about its structure and data types should be specified, in a Javascript-Object-Notation (JSON) file.

While the SDV is able to build models for relational databases with multiple tables, we use the standalone table model for the purposes of the present paper. The descriptive model of such a table is obtained from the distribution of the values of each column and the covariances between the attributes. In order to find a good estimate for the distribution of a column, a Kolmogorov-Smirnov test is applied. This test returns p-values for the likelihood that the data matches common distributions, such as the truncated Gaussian, the uniform, the beta or the exponential distribution. The highest p-value determines the shape of the chosen Cumulative Distribution Function (CDF) for the column. Next, the covariance between the columns is calculated. In order to prevent the shape of different distributions from influencing the covariance estimates, a multivariate version of the Gaussian Copula is applied. The synthetic data is then generated via the resulting model, which consists of the parameters for the column distributions and the covariance matrix of the table after the transformation by the copula.

2.2 DataSynthesizer

The DataSynthesizer (DS) has been developed by Ping et al. in 2017. We briefly describe the approach and available implementation, and refer the reader to [7] for additional information. The source code is publicly available at [3]. The approach of the DS is based on earlier work by [13].

The high-level system architecture of the DS consists of two modules, namely the DataDescriber and the DataGenerator. The *DataDescriber* takes a dataset in first normal form as input (in the form of a CSV file), and infers both the data types and the domain of the attributes. However, the system also allows the users to specify data types. The DataDescriber creates a description file of the distributions and types of attributes in the input table. The *DataGenerator* then samples synthetic data based on the description. Both the DataDescriber and the DataGenerator may be invoked in one of three modes. In 'random mode', the tool generates type-consistent random values for each attribute. In 'independent attribute mode', the DataSynthesizer performs a frequency-based estimation and preserves the unconditioned probability distribution of each attribute, but not the dependencies between them. In 'correlated attribute mode', a Bayesian network is constructed to model the correlated attributes. The user may specify the maximum number of parents of each node, which may become a runtime (efficiency) requirement for large datasets. Data privacy concerns are addressed by providing *ε-differential privacy* (cf. [3]). The user of the DS is able to determine the value of ε and may also set it to 0 for disabling differential privacy completely.

3 EXPERIMENT SETUP

In this section, we discuss the datasets used in our experiment as well as our methods and the strategy for the utility evaluation. Our main goal in the experiments is to provide an unbiased evaluation, for both the models trained on the original baseline, as well as the models on the synthetic data. Therefore, we refrained from overly optimising certain aspects (such as parameter settings for the models and data generation, data preparation, etc.) for specific models.

3.1 Datasets

Our experiments are based on a total of five standard benchmark datasets, taken from the UCI repository[4]and Kaggle[5] (see Table 1).

Table 1: Dataset characteristics

Dataset	# Features	# Instances	# Classes
Adult[6] (Census Income[7])	15	48,842	2
Banknote Authentication[8]	5	1,372	2
Iris[9]	5	150	3
Social Network Ads[10]	5	400	2
Titanic[11]	12	891	2

We included datasets with personal identifying information as well as sensitive data (e.g., ethnicity and salary), as these are typical cases which require data privacy. However, we also included some other benchmark datasets for further analysis and comparisons. We aimed at a diverse set of characteristics in the dataset, from a range of different domains. The Social Network Ads dataset is used to predict if a user bought a certain product, based on their gender, age, and estimated salary. The Adult (sometimes called *Census Income*)

[2]https://github.com/HDI-Project/SDV
[3]https://github.com/DataResponsibly/DataSynthesizer
[4]http://archive.ics.uci.edu/ml
[5]https://www.kaggle.com/
[6]https://archive.ics.uci.edu/ml/datasets/Adult
[7]https://archive.ics.uci.edu/ml/datasets/census+income
[8]http://archive.ics.uci.edu/ml/datasets/banknote+authentication
[9]https://archive.ics.uci.edu/ml/datasets/iris
[10]https://www.kaggle.com/rakeshrau/social-network-ads
[11]https://www.kaggle.com/c/titanic/data

dataset shows whether a person's income exceeds $50,000 per year, according to census data such as age, work class, and education. The Titanic dataset tells if a passenger survived the sinking of the Titanic and provides passenger information such as ticket class, gender, age, and family members on board. In addition, we added the Banknote Authentication dataset for the task of identifying forged banknotes and the Iris dataset with three types of plants.

3.2 Data Generation

For each of the datasets, we performed the following procedure to synthesise and prepare the data for the utility evaluation.

(1) We deleted columns as part of standard feature cleaning.
(2) To ensure reliable results, we performed a repeated holdout method, i.e. we randomly generated ten different splits of the table into training and test data, such that the size of the latter is 20% of the original table.
(3) For each split, we applied the tools discussed in Section 2 to the training dataset to generate synthetic training data of equal length.

Since we wanted to investigate the effect of differential privacy for the Data Synthesizer, the tool is applied twice in Step 3. For each of the splits generated in Step 2, we therefore obtain the following data files: The test data, the original training data, the training data synthesised by the SDV, and two synthetic training data files from the DS, one with and one without differential privacy. The described procedure is applied to each original dataset except the Adult Census dataset, for which we adhered to the split published in the UCI repository.

We used the SDV with the default model type `copulas.multivariate.GaussianCopula` for the Modeler. The DS is applied in correlated attribute mode to preserve correlations in the synthetic data, which is important for realistic machine learning models. We used both the SDV and the DS with default settings and as explained in the respective GitHub repositories. We intentionally did not optimise the approaches for the selected datasets, as we wanted to investigate their general performance. The only parameter we changed was ε for differential privacy in the DS — its influence on the data generation will be discussed in Section 4.

3.3 Utility Analysis

When sanitising a dataset via anonymisation, synthetisation or other approaches, some sensitive information at the level of individual records is invariably removed [2]. The utility of such a dataset for researchers, economists or other data analysts, can thus be measured by the extent to which it preserves aggregate and statistical information. Given a candidate for synthetic data, a *utility metric* quantifies the utility of this release candidate (resp. the information loss due to the synthetisation process). There are in general two approaches for such an evaluation. One is to utilise one or more quantitative measures of information loss (see [2] for an overview). Given that these measures do not necessarily reflect the final utility of a machine learning model, we will employ the second approach, which is to directly use the synthetic dataset as an input to the model building, and evaluate the quality of the model.

We approached the classification tasks by applying five machine learning algorithms, namely Naïve Bayes, Support Vector Machines, K-Nearest Neighbours, Random Forests and Logistic Regression. All classifiers are implemented in the Python sklearn package[12], and we used standard parameters to avoid introducing bias. For each dataset and each split setup consisting of the files described in the previous subsection, we performed the following procedure:

(1) On the original training and test data, we applied label encoding and, if necessary, one-hot encoding. Subsequently, we used sklearn's StandardScaler for feature scaling.
(2) We fitted the model to the training data and predicted the results on the test data.
(3) We repeated Step 1 and 2 for all synthesised training datasets.

We present the resulting accuracy scores in the following tables of size 4×5 (number of different training datasets × number of different classifiers). As the procedure is repeated for every random split generated in the previous step of the experiment, this yields a total of ten such results. We aggregate those by reporting the average results plus the standard deviation per dataset (see Section 4.2).

4 EVALUATION

In this section, we discuss the results of our experiments and present them together with an analysis of differences between the original and the synthetic data.

4.1 Comparison of Original to Synthetic Data

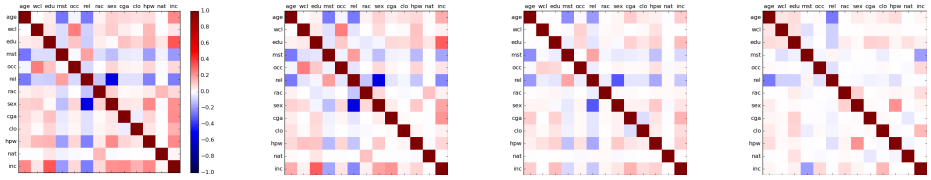
The first step of our evaluation is a statistical comparison of the synthesised and the original training data. For each of the datasets, we performed the following steps.

- For each attribute, we generated a histogram visualising both the distribution of the real and the synthetic data.
- We computed the correlation coefficients and generated a heat map to visualise dependencies between attributes.
- We measured the distance between the real and the synthetic data via row-by-row computations of nearest neighbours.

For the sake of brevity in this paper, we limit the presentation of the results to some exemplar and key findings. In the following, we discuss several aspects related to the Adult Census dataset, as representative for general tendencies. We first compare the heat maps in Figure 1. As can be seen in the Figure 1c and Figure 1b, the sign of the correlation coefficient is, in general, preserved both by the SDV and the DS. For this comparison, we did not inject any noise in form of differential privacy in the DS, i.e. $\varepsilon = 0$. The degree of preserved correlation between the attributes is notably higher for the DS. As a consequence, the SDV constructs data with larger differences to the original, which can also be observed by comparing histograms for the attributes. As an example, we visualise the distributions for the feature 'occupation' in Figure 2, which has a domain consisting of 14 categories. Synthesised datasets with larger differences to the original dataset will clearly affect both the data privacy and the performance of machine learning models. It is expected that there is a trade-off between these two.

No formal model for evaluating the privacy of synthetic datasets has yet been proposed. We thus employ a statistics analysis. Our basic assumption is that privacy is endangered if individuals like the ones in the original dataset appear also in the synthetic dataset.

[12] https://scikit-learn.org/stable (specifically, we used version 0.20.3)



(a) Original (b) DS (c) SDV (d) DS with DP

Figure 1: Heat maps for the Adult (Census Income) dataset

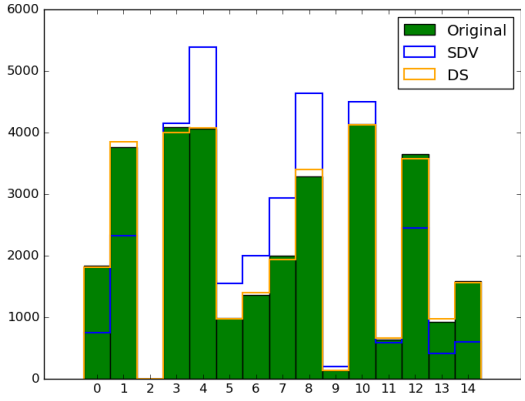


Figure 2: Distributions of 'occupation' attribute

Therefore, we require some kind of distance measure between datasets. We want to identify whether there are any individuals (i.e. rows) in the original dataset that also appear in the synthesised datasets. We are interested in both exact fits as well as similarities, as those might still contain some information that could lead to a privacy breach. For each row in the synthesised datasets, we therefore computed the nearest neighbour in the original, that is, the row to which the euclidean distance is minimal. To transform categorical attributes into comparable numerical values, we used a combination of label- and one-hot encoding, followed by a step of feature scaling. The results for Adult Census are summarised in Figure 3, where we can see the euclidean distance on the x-axis and the amount of rows on the y-axis (with bin size of 40). We observe that the dataset synthesised by the DS is significantly closer to the original compared to the dataset synthesised by the SDV. We can influence that by enabling differential privacy and setting, e.g., $\varepsilon = 0.05$ in the DS. We then obtain the graphs in Figure 1d and Figure 3c, respectively. The heat map now shows significant differences in the dependencies between attributes, and the correlations appear to be much weaker in general. We also see that the mean minimum distance to points in the original dataset has significantly shifted to the right, and now more resembles the distribution of distance that can also be observed in the Synthetic Data Vault, though still with a higher number of samples that exhibit a very low distance.

4.2 Utility on Machine Learning Tasks

The final step of our experiment is to train machine learning models on the real and the synthesised training datasets, and to evaluate these models by comparing their prediction scores on the test data. We computed the accuracy scores of all models, which is the relation between true predictions and all predictions that have been made.

Let us start with the results of the Social Network dataset. The columns show the scores for the classifiers we used, i.e. Naïve Bayes (NB), Support Vector Machine (SVM), K-nearest neighbour (KNN), Random Forest (RF) and Logistic Regression (LR). As described in

Table 2: Classification results on the Social Network dataset

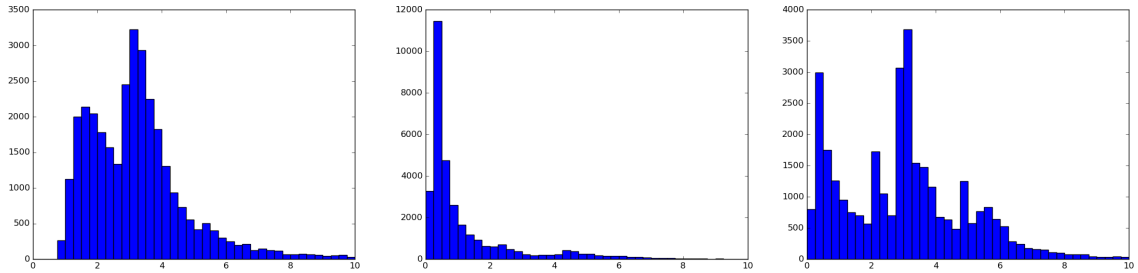
SocNet	NB	SVM	KNN	RF	LR
Real	89.6±2.5	83.6±3.1	90.6±2.1	88.6±2.7	84.0±2.9
DS 0	88.7±2.8	84.2±2.4	89.4±2.8	88.6±3.5	85.6±2.4
DS 0.1	87.6±2.5	83.4±3.4	85.1±3.4	81.3±5.7	83.3±3.3
SDV	77.9±5.8	74.5±8.6	75.1±5.3	73.1±5.3	78.9±5.1

Section 3.3, the scores in this table are the arithmetic means of the accuracy scores for ten different random splits of the original data. Additionally, the standard deviation is included. We can see that the values of the DS, without differential privacy, are high for each of the five models, sometimes even higher than the scores of the model trained with the original data. In this context, we have to stress the fact discussed in Section 4.1, namely that the dataset synthesised by the DS with $\varepsilon = 0$ is much closer to the original than the dataset synthesised by the SDV. Not surprisingly, the values of the DS are generally higher. Hence, we decided to also conduct the experiment with enabled differential privacy, and the third row shows the results for $\varepsilon = 0.1$. It has to be mentioned that, for $\varepsilon > 0$, we observed unstable performance scores of the DS, depending on its own random seeds utilised for differential privacy, which cannot be controlled by the user. The results range from scores as good as for $\varepsilon = 0$ to scores below those of the SDV. Therefore, the performances of DS with $\varepsilon > 0$ displayed in this table and in the following ones have to be interpreted with caution. Further investigation of this behaviour will be a subject for future work.

We continue with discussing the Adult (Census Income) dataset, and recall that these are the results for the single split published by UCI. For this table, we computed both the accuracy and the

Table 3: Classification results on the Adult (Census Income) dataset

Adult	NB	SVM	KNN	RF	LR
Real	79.7 | 41.5	81.0 | 41.4	81.4 | 56.2	82.2 | 57.3	82.3 | 53.3
DS 0	81.1 | 38.4	79.5 | 23.6	79.4 | 47.4	80.0 | 47.6	81.7 | 47.3
DS 0.1	78.8 | 36.8	79.4 | 39.7	77.6 | 46.6	78.4 | 41.8	79.4 | 44.9
DS 0.05	78.8 | 20.3	76.4 | 0.0	75.8 | 39.1	76.1 | 30.3	77.7 | 20.8
SDV	77.2 | 7.1	76.4 | 0.0	76.9 | 11.6	76.4 | 2.0	77.0 | 5.7



(a) SDV (b) DS (c) DS with DP

Figure 3: Minimum distances for the Adult (Census Income) dataset

F1 score, the latter of which is the harmonic mean of precision and recall. Note that each cell is of the format 'Accuracy | F1-Score'. Furthermore, we included the results for DS with $\varepsilon = 0.05$ to observe the effects of varying degrees of differential privacy. Also, $\varepsilon = 0.05$ is the parameter choice that has been investigated in Figure 1d and Figure 3c. At first glance, it appears that both the accuracy scores of DS and SDV are close to the original dataset. We can see the repeating behaviour that the DS without differential privacy scores better than the SDV. The difference is smaller than one might expect after the analysis of the two datasets in the Section 4, in particular after comparing the two graphs in Figure 3. However, this can be explained by taking a closer look at the distribution of the target variable 'income' in the training data and the metrics of SDV besides the accuracy score. A dummy classifier that always predicts the most frequent class of the 'income' attribute (sometimes referred to as the "Zero Rule" classifier) already achieves 76.4% accuracy. Considering the scores of the Synthetic Data Vault, we can see that they are only slightly better for NB, KNN and LR. Indeed, an inspection of the confusion matrices reveals that the models trained on the training data synthesized by SDV tend to strongly prefer the more frequent class over the less frequent one. As a consequence, the SDV shows rather low F1 scores.

Table 4: Classification results on the Banknotes dataset

Bank	NB	SVM	KNN	RF	LR
Real	83.7±2.4	98.5±0.5	100.0±0.1	98.4±0.7	98.3±0.5
DS 0	83.7±3.0	97.3±0.9	95.5±2.6	93.5±1.9	95.7±0.8
DS 0.1	70.0±4.1	87.5±7.4	85.8±3.6	81.6±4.7	80.2±6.9
SDV	81.3±1.0	96.7±0.9	92.8±1.6	92.1±2.4	96.9±1.2

Table 5: Classification results on the Iris dataset

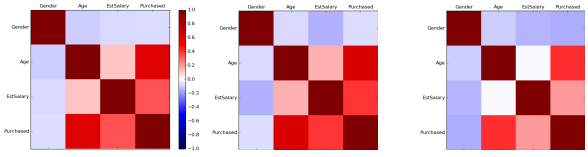
Iris	NB	SVM	KNN	RF	LR
Real	93.7±5.1	97.7±2.6	96.0±3.6	94.4±4.0	96.0±4.2
DS 0	93.0±6.4	97.0±2.8	95.4±4.0	95.0±4.8	95.3±5.0
DS 0.1	67.3±19.3	49.3±17.5	72.7±9.9	63.7±8.5	50.3±14.6
DS 0.25	88.7±3.4	83.7±10.3	88.3±7.8	86.0±6.3	79.0±10.4
SDV	94.0±4.7	96.0±3.9	92.0±5.6	90.7±4.9	95.0±4.3

Better results of the SDV may be observed on the Banknotes and the Iris datasets. In particular, the results for Naïve Bayes, SVMs and LR are very close to the model trained on the real dataset, while for k-NN and Random Forests, there is a noticeable difference. These datasets have a structural similarity, as they both have five attributes of the same type, namely a continuous variable. However, there are still differences. The Banknotes dataset is significantly larger, while the task on Iris is non-binary, as there are three different classes. On Iris, enabling differential privacy causes the DataSynthesizer to achieve rather poor and unstable results with high standard deviation. For this dataset, we considered a value of 0.25 for ε, which means that the amount of noise injected by differential privacy is in fact less than the amount of $\varepsilon = 0.1$ we have used as a reference point in all other experiments. This might be due to the rather compact range of feature values and the small number of instances per class. Of course, the Iris dataset itself doesn't contain any personal or sensitive information; still, the data synthetisation is agnostic of that, and is not expected to work worse on non-sensitive data.

Larger differences between DS and SDV, more similar to those in the Social Network dataset, can be observed for the Titanic dataset. Here, the results on the SDV data are 12% off from the performance on the real data for the SVM, k-NN and Random Forests. The DataSynthesizer achieves much better results, even with differential privacy enabled.

Table 6: Classification results on the Titanic dataset

Titanic	NB	SVM	KNN	RF	LR
Real	76.6±3.0	77.6±3.1	78.9±2.9	79.9±2.9	79.1±3.6
DS 0	73.9±5.0	77.5±3.0	74.6±2.7	72.5±2.8	76.9±3.7
DS 0.1	72.7±4.3	75.6±7.2	73.0±4.3	72.5±4.5	75.5±6.1
SDV	71.8±4.7	65.3±7.7	66.8±5.2	66.0±6.4	71.8±5.3



(a) Original (b) DS (c) SDV

Figure 4: Heat maps for the Social Network dataset

For Social Network and Titanic, the differences between the DS with $\varepsilon = 0$ and the SDV are rather significant. SDV's loss of accuracy of 10% or more can be explained by taking a closer

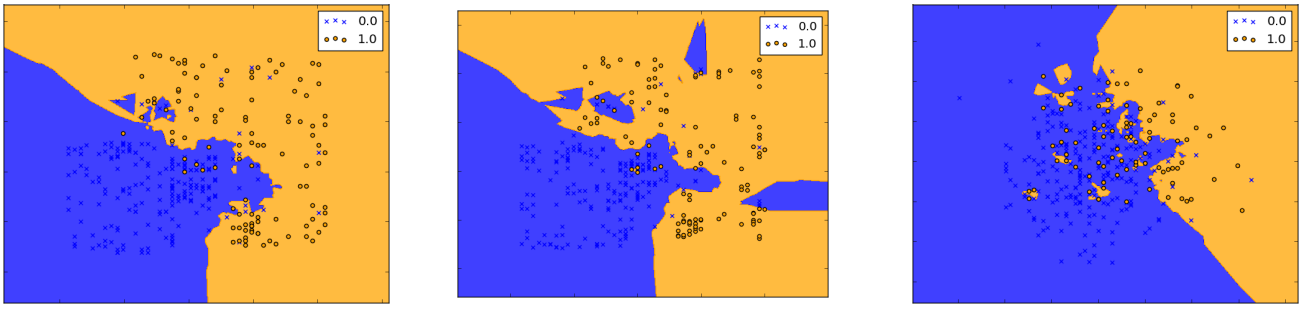


Figure 5: K-NN model on the Social Network dataset: Original (left), DS (middle), SDV (right)

look at certain correlations. For example, let us consider the heat maps for the Social Network dataset in Figure 4. Here, the most important observation is that the SDV does not preserve the positive correlation between the attributes 'EstimatedSalary' and 'Age'. Let us consider the effect on the performance of K-Nearest Neighbours by generating a scatter plot. We try to predict the class 'Purchased' ('0' and '1') based on the mentioned features. Each point represents an individual, where the x-axis shows the age and the y-axis the estimated salary; note that both features already have been scaled to standard normal. Individuals who purchased are represented by orange points, the others by blue cross marks. The regions are also coloured orange and blue depending on what the model has learned for points in the respective area.

In Figure 5 we observe that, for the original data, the model does well for separating the two classes on the training data, obtaining an accuracy score of 92.5% on the test data for the specific one of the ten splits we worked with to generate the graphs. The scatter plot for the DS and, in particular, the colouring of regions looks rather similar, and the accuracy score is 90%. For the SDV, however, we can see that the distribution of points is different in general, and so are the regions learned by the model on the training data. Of course, this affects the accuracy score on the test data, which is 72.5% in this case. This particularly drastic example demonstrates the general effect of noises injected by the SDV.

5 CONCLUSIONS AND FUTURE WORK

We may conclude that the data synthesised by DS with disabled differential privacy is very much suitable for classification tasks in supervised machine learning. However, the analysis in Section 4.1 raises privacy concerns. For sensitive data, it is definitely recommended to use the SDV or the DS with a small value for ε. On the Banknotes and the Iris dataset, the SDV produces synthetic data with relatively large differences to the original tables' records, whereas the loss of utility for machine learning is comparably small. The latter does not appear to be true for the Social Network, the Adult and the Titanic dataset. We plan to investigate the reasons for this in greater detail, as well as the effect of the size of ε for the DS. A further aspect of future research will be in defining and quantifying the privacy levels and guarantees achieved by synthetic data. We have the impression that, in current work, it is rather assumed than shown that privacy will be preserved by synthetic data generation. We thus will compare synthetic data with other privacy-preserving methods, such as anonymisation.

ACKNOWLEDGMENTS

This work was partially funded by the KIRAS program (No 860663) of the Austrian Research Promotion Agency (FFG), the EU Horizon 2020 research and innovation programme under grant agreement No 732907. The competence center SBA Research (SBA-K1) is funded within the framework of COMET — Competence Centers for Excellent Technologies by BMVIT, BMDW, and the federal state of Vienna, managed by the FFG.

REFERENCES

[1] Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. 2018. Privacy preserving synthetic data release using deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 510–526.

[2] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, and Ashwin Machanavajjhala. 2009. Privacy-Preserving Data Publishing. *oundations and Trends in Databases* 2, 1–2 (Jan. 2009), 1–167.

[3] Cynthia Dwork. 2006. Differential Privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming ICALP (Lecture Notes in Computer Science)*, Vol. 4052. Springer, Venice, Italy, 1–12. https://doi.org/10.1007/11787006_1

[4] Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. 2010. Privacy-preserving Data Publishing: A Survey of Recent Developments. *Comput. Surveys* 42, 4, Article 14 (June 2010), 14:1–14:53 pages.

[5] Bernd Malle, Peter Kieseberg, and Andreas Holzinger. 2017. DO NOT DISTURB? Classifier Behavior on Perturbed Datasets. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*.

[6] Neha Patki, Roy Wedge, and Kaylan Veeramachaneni. 2016. The Synthetic Data Vault. In *Proceedings of the IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Montreal, QC, Canada, 399–410.

[7] Haoyue Ping, Julia Stoyanovich, and Bill Howe. 2017. DataSynthesizer: Privacy-Preserving Synthetic Datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*. Chicago, IL, USA.

[8] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Eric P. Xing and Tony Jebara (Eds.), Vol. 32. PMLR, Bejing, China, 1278–1286.

[9] Donald B. Rubin. 2004. *Multiple Imputation for Nonresponse in Surveys*. Wiley-Interscience.

[10] H Surendra and H. S. Mohan. 2017. A Review Of Synthetic Data Generation Methods For Privacy Preserving Data Publishing. *International Journal of Scientific & Technology Research* 6, 3 (March 2017), 95–101.

[11] Latanya Sweeney. 2002. K-anonymity: A Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 5 (Oct. 2002), 557–570.

[12] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. 2016. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems (NIPS)*. 4790–4798.

[13] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. PrivBayes: Private Data Release via Bayesian Networks. *ACM Transactions on Database Systems* 42, 4, Article 25 (October 2017), 41 pages.