The Journal of Open Source Software

Metasyn: Transparent Generation of Synthetic Tabular Data with Privacy Guarantees

Raoul Schram [1*], **Samuel Spithorst** [1], and **Erik-Jan van Kesteren** [1,2*¶]

1 Utrecht University, The Netherlands **2** ODISSEI: Open Data Infrastructure for Social Science and Economic Innovations, The Netherlands ¶ Corresponding author ***** These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Synthetic data is a promising tool for improving the accessibility of datasets which are too sensitive to be shared publicly. To this end, we introduce `metasyn`, a Python package for generating synthetic data from tabular datasets. Unlike existing synthetic data generation software, `metasyn` is built on a simple generative model that omits multivariate information. This choice enables transparency and auditability, keeps information leakage to a minimum, and enables privacy guarantees through a plug-in system. While the analytical validity of the generated data is thus intentionally limited, its potential uses are broad, including exploratory analyses, code development and testing, and external communication and teaching (van Kesteren, 2024).

Figure 1: Logo of the `metasyn` project.

Statement of need

`Metasyn` is aimed at owners of sensitive datasets such as public organisations, research groups, and individual researchers who want to improve the accessibility of their data for research and reproducibility by others. The goal of `metasyn` is to make it easy for data owners to share the structure and an approximation of the content of their data with others while keeping privacy concerns to a minimum.

With this goal in mind, `metasyn` distinguishes itself from existing software for generating synthetic data (e.g., Nowok et al., 2016; Ping et al., 2017; Templ et al., 2017) by strictly limiting the statistical information from the real data in the synthetic data. `Metasyn` explicitly avoids generating synthetic data with high analytical validity; instead, the synthetic data has realistic structure and plausible values, but multivariate relations are omitted ("augmented plausible synthetic data"; (Bates et al., 2019)). Moreover, our system provides an **auditable and editable intermediate representation** in the form of a `.json` metadata file from which new data can be synthesized.

These choices enable the software to generate synthetic data with **privacy and disclosure**

guarantees through a plug-in system, recognizing that different data owners have different needs and definitions of privacy. A data owner can define under which conditions they would accept open distribution of their synthetic data — be it based on differential privacy (Dwork, 2006), statistical disclosure control (Hundepool et al., 2012), k-anonymity (Sweeney, 2002), or another specific definition of privacy. As part of the initial release of `metasyn`, we publish a plug-in following the disclosure control guidelines from Eurostat (Bond et al., 2015).

Software features

At its core, `metasyn` has three main functions: **estimation**, to fit a model to a properly formatted tabular dataset; **generation**, to synthesize new datasets based on a fitted model; and **(de)serialization**, to create a file from the model for auditing, editing, and saving.

Estimation

Model estimation starts with an appropriately pre-processed data frame, meaning it is tidy (Wickham, 2014), each column has the correct data type, and missing data are represented by a missing value. Accordingly, `metasyn` is built on the `polars` data frame library (Vink et al., 2024). As an example, the first records of the "hospital" data built into `metasyn` are printed below:

```
┌────────────┬───────────────┬───────────────┬──────┬──────┬──────────────┐
│ patient_id ┆ date_admitted ┆ time_admitted ┆ type ┆ age  ┆ hours_in_room │
│ ---        ┆ ---           ┆ ---           ┆ ---  ┆ ---  ┆ ---          │
│ str        ┆ date          ┆ time          ┆ str  ┆ i64  ┆ f64          │
╞════════════╪═══════════════╪═══════════════╪══════╪══════╪══════════════╡
│ A5909X0    ┆ 2024-01-01    ┆ 10:30:00      ┆ IVT  ┆ null ┆ 3.633531     │
│ B4025X2    ┆ 2024-01-01    ┆ 11:23:00      ┆ IVT  ┆ 59   ┆ 6.932891     │
│ B6999X2    ┆ 2024-01-01    ┆ 11:58:00      ┆ IVT  ┆ 77   ┆ 1.970654     │
│ B9525X2    ┆ 2024-01-01    ┆ 16:56:00      ┆ MYE  ┆ null ┆ 1.620047     │
│ …          ┆ …             ┆ …             ┆ …    ┆ …    ┆ …            │
└────────────┴───────────────┴───────────────┴──────┴──────┴──────────────┘
```

Note that categorical data are encoded as `cat` (not `str`) and missing data is represented by `null` values. Model estimation with `metasyn` is then performed as follows:

```python
from metasyn import MetaFrame
mf = MetaFrame.fit_dataframe(df_hospital)
```

The generative model in `metasyn` makes the simplifying assumption of *marginal independence*: each column is considered separately, similar to naïve Bayes classifiers (Hastie et al., 2009). For each column, a set of candidate distributions is fitted (see Table 1), and then `metasyn` selects the one that fits best (usually having the lowest BIC (Neath & Cavanaugh, 2012)). Key advantages of this approach are transparency and explainability, flexibility in handling mixed data types, and computational scalability to high-dimensional datasets.

Table 1: Candidate distributions associated with data types in the core `metasyn` package.

Data type	Candidate distributions
Categorical	Categorical, Constant
Continuous	Uniform, Normal, LogNormal, TruncatedNormal, Exponential, Constant
Discrete	Poisson, Uniform, Normal, TruncatedNormal, Categorical, Constant
String	Regex, Categorical, Faker, FreeText, Constant
Date/time	Uniform, Constant

JOSS
The Journal of Open Source Software

From this table, the string distributions deserve special attention as they are not common probability distributions. The regex (regular expression) distribution uses the package `regexmodel` to automatically detect structure such as room numbers (A108, C122, B109), identifiers, e-mail addresses, or websites. The FreeText distribution detects the language (using lingua) and randomly picks words from that language. The Faker distribution can generate specific data types such as localized names and addresses pre-specified by the user.

Data generation

After creating a `MetaFrame`, `metasyn` can randomly sample synthetic datapoints from it. This is done using the `synthesize()` method:

```
df_syn = mf.synthesize(3)
```

This may result in the following data frame. Note that missing values in the age column are appropriately reproduced as well.

patient_id --- str	date_admitted --- date	time_admitted --- time	type --- cat	age --- i64	hours_in_room --- f64
B7906X1	2024-01-04	13:32:00	IVT	37	4.955418
B0553X2	2024-01-02	10:54:00	IVT	39	3.872872
A5397X7	2024-01-03	18:16:00	CAT	null	6.569082

Serialization and deserialization

`MetaFrames` can also be transparently stored in a human- and machine-readable `.json` metadata file. This file contains dataset-level descriptive information as well as variable-level information. This `.json` can be manually audited, edited, and after saving this file, an unlimited number of synthetic records can be created without incurring additional privacy risks. Serialization and deserialization with `metasyn` is done using the `save()` and `load()` methods:

```
mf.save("hospital_admissions.json")
mf_new = MetaFrame.load("hospital_admissions.json")
```

Privacy

As a general principle, `metasyn` errs on the side of privacy by default, aiming to recreate the structure but not all content and relations in the source data. For example, take the following sensitive dataset where study participants state how they use drugs in daily life:

participant_id --- str	drug_use --- str
OOWJAHA4	I use marijuana in the evening…
8CA1RV4P	I occasionally take CBD to hel…
FMSVAKPM	Prescription medication helps …
…	…

When creating synthetic data for this example, the information in the open answers is removed, and using our standard `FreeText` distribution this information is replaced by words from the detected language (English):

participant_id --- str	drug_use --- str
ZQJZQAB7	Lawyer let sort her yet line e…
7KDLEL0S	Particularly third myself edge…
QBZKGXC7	Put color against call researc…

Additionally, `metasyn` package allows for plug-ins which alter the estimation behaviour. Through this system, privacy guarantees can be built into `metasyn` and additional distributions can be supported. For example, the `metasyn-disclosure-control` plug-in implements output guidelines from Eurostat (Bond et al., 2015) through *micro-aggregation*.

Acknowledgements

This research was conducted in whole or in part using ODISSEI, the Open Data Infrastructure for Social Science and Economic Innovations (https://ror.org/03m8v6t10)

`metasyn` was supported by the Utrecht University FAIR Research IT Innovation Fund (March 2023)

References

Bates, A., Spakulová, I., Dove, I., & Mealor, A. (2019). *ONS methodology working paper series number 16—synthetic data pilot*. https://www.ons.gov.uk/methodology/methodologicalpublications/generalmethodology/onsworkingpaperseries/onsmethodologyworkingpaperseriesnumber16syntheticdatapilot

Bond, S., Brandt, M., & Wolf, P. de. (2015). *Guidelines for the checking of output based on microdata research*. Eurostat. https://web.archive.org/web/20160408145718/http://dwbproject.org/export/sites/default/about/public_deliveraples/dwb_d11-8_synthetic-data_cta-ecta_output-checking-guidelines_final-reports.zip

Dwork, C. (2006). Differential privacy. *International Colloquium on Automata, Languages, and Programming*, 1–12. https://doi.org/10.1007/11787006_1

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Vol. 2). Springer. https://doi.org/10.1007/978-0-387-84858-7

Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E. S., Spicer, K., & De Wolf, P.-P. (2012). *Statistical disclosure control*. Wiley & Sons, Chichester. https://doi.org/10.1002/9781118348239

Neath, A. A., & Cavanaugh, J. E. (2012). The bayesian information criterion: Background, derivation, and applications. *Wiley Interdisciplinary Reviews: Computational Statistics*, *4*(2), 199–203. https://doi.org/10.1002/wics.199

Nowok, B., Raab, G. M., & Dibben, C. (2016). Synthpop: Bespoke creation of synthetic data in r. *Journal of Statistical Software*, *74*, 1–26. https://doi.org/10.18637/jss.v074.i11

Ping, H., Stoyanovich, J., & Howe, B. (2017). Datasynthesizer: Privacy-preserving synthetic datasets. *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 1–5. https://doi.org/10.1145/3085504.3091117

Sweeney, L. (2002). K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *10*(05), 557–570. https://doi.org/10.1142/S0218488502001648

Templ, M., Meindl, B., Kowarik, A., & Dupriez, O. (2017). Simulation of synthetic complex data: The r package simPop. *Journal of Statistical Software*, *79*(10), 1–38. https://doi.org/10.18637/jss.v079.i10

van Kesteren, E.-J. (2024). To democratize research with sensitive data, we should make synthetic data more accessible. *Patterns*, *5*(9), 101049. https://doi.org/10.1016/j.patter.2024.101049

Vink, R., Gooijer, S. de, Beedie, A., Gorelli, M. E., Guo, W., Zundert, J. van, Peters, O., Hulselmans, G., nameexhaustion, Grinstead, C., Marshall, Burghoorn, G., chielP, Turner-Trauring, I., Santamaria, M., Heres, D., Mitchell, L., Magarick, J., ibENPC, … Brannigan, L. (2024). *Pola-rs/polars: Python polars* (py-1.4.1). Zenodo. https://doi.org/10.5281/zenodo.7697217

Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, *59*(10), 1–23. https://doi.org/10.18637/jss.v059.i10