We propose a general, flexible, and scalable framework \textit{dpart}, an open source Python library for differentially private synthetic data generation. Central to the approach is autoregressive modelling—breaking the joint data distribution to a sequence of lower-dimensional conditional distributions, captured by various methods such as machine learning models (logistic/linear regression, decision trees, etc.), simple histogram counts, or custom techniques. The library has been created with a view to serve as a quick and accessible baseline as well as to accommodate a wide audience of users, from those making their first steps in synthetic data generation, to more experienced ones with domain expertise who can configure different aspects of the modelling and contribute new methods/mechanisms. Specific instances of \textit{dpart} include Independent, an optimized version of PrivBayes, and a newly proposed model, dp-synthpop.

\textbf{Main Contributions:}

\begin{itemize}
  \item Implement and open source a general and flexible framework for DP synthetic data generation, based on autoregressive modelling, to serve as a quick and efficient baseline.
  \item As a specific use case of \textit{dpart}, we modify and improve the speed of PrivBayes (Zhang et al., 2017) by 20x.
  \item As another use case, we propose a DP version of synthpop (Nowok et al., 2016), which we name dp-synthpop.
\end{itemize}

\textbf{Code:} https://github.com/hazy/dpart.

\textbf{Accepted at the Theory and Practice of Differential Privacy (TPDP) 2022, part of ICML 2022.}
2. Overview

Framework summary.

dpart is a general and flexible framework for building an effective DP generative model. The overall training flow relies on an autoregressive generative model (Fig. 1) and could be broken down to two main steps. First, identifying/specifying a visit order or a prediction matrix that describes how the joint distribution is broken down to a series of lower-dimensional conditionals (the dependency). In other words, if the dataset \( D \) is a collection of \( k \)-dimensional datapoints \( x \) then:

\[
P(x) = \prod_{i=1}^{k} P(x_i|x_1, x_2, ..., x_{i-1}) = \prod_{i=1}^{k} P(x_i|x_{<i})
\]

Second, given the series of conditionals, they are sequentially estimated by fitting predictive models (sampler methods). To generate synthetic data, the fitted sampler methods are used to generate one column at a time.

Installation.

dpart is written in Python due to the language popularity among data scientists as well as machine learning researchers and practitioners. It can be installed using pip:

```bash
generate: Differentially Private Autoregressive Tabular

Figure 1: dpart framework.

| Data type support | Method                      |
|-------------------|-----------------------------|
| numerical only    | DP linear regression        |
| categorical only  | DP logistic regression,     |
|                   | DP decision tree,           |
|                   | DP random forest            |
| both              | DP conditional distribution,|
|                   | DP histogram sample         |

Table 1: Methods.

At most one of visit_order and prediction_matrix could be used, as the two arguments conflict with each other.

2) methods arguments:
- \( \text{methods} \): a dictionary specifying the method each column should be modeled by. Columns must match the data type support of the selected method. A list with currently available methods is shown in Tab. 1 and further explanations provided below.

The methods can be split into the following three categories.

Numerical only methods. These methods can be applied on target columns with numerical data type (i.e., float, integer, date time, and timedelta):
- regression method: it relies on fitting a DP regression model to predict the target column. In order to allow for non-deterministic behavior, the standard deviation of the residuals is captured in a DP way using the Laplace mechanism. During generation, new values are sampled by adding appropriate noise to the prediction from the trained regression. Currently available regression methods are: \( \text{DP linear regression} \).

Categorical only methods. The methods below can be used on categorical columns with either an object, category, or boolean data type:
- classifier method: it fits a DP classification model that can output a conditional distribution. The available classification methods are: \( \text{DP logistic regression, DP decision tree, and DP random forest} \).

Both numerical and categorical methods:
- \( \text{DP conditional distribution} \): it captures and samples from a discretized joint distribution. Numerical data is binned using uniform binning to allow for a discrete representation and DP is satisfied by adding a Laplace noise to the counts before converting to a distribution.
- \( \text{DP histogram sampler} \): this method captures the marginal distribution of the target column without taking into account any input features. It is a specific use case of \( \text{DP conditional distribution} \).

3) privacy budget arguments:
- \( \epsilon \): a positive real number which defines the over-
**dpart: Differentially Private Autoregressive Tabular**

Notes

| Model         | Notes                                                                 |
|---------------|----------------------------------------------------------------------|
| Independent   | simple baseline model                                                 |
|               | (Ping et al., 2017; Tao et al., 2021)                                |
|               | (Stadler et al., 2022)                                               |
| PrivBayes     | optimized model                                                       |
|               | (Zhang et al., 2017; Ping et al., 2017)                               |
| dp-synthpop   | new model, DP version                                                 |
|               | (Nowok et al., 2016)                                                 |

*Table 2: Specific instances of dpart.*

all privacy budget to be used across the fitting step.

- Alternatively, a dictionary describing how the privacy budget can be split between the dependency and the methods steps could be provided. Furthermore, the user can break down the privacy budget between the methods for each column.
- **bounds:** a dictionary specifying the range (minimum and maximum) for all numerical columns as well as the distinct categories for categorical columns. This prevents further privacy leakage. Alternatively, **PrivacyLeakWarning** is displayed (see below).

**Troubleshooting.**

Inspired by diffprivlib, we adopt specific privacy (and other) warnings messages:

- **PrivacyLeakWarning:** this warning is raised when privacy related input is missing. A good example is **bounds** which must be provided to ensure that no further privacy leakage is incurred. However, if the **bounds** are not provided, the algorithm will run and infer the missing bound values but will raise a warning (if epsilon has been provided).
- **UserWarning:** this warning is raised when a method is not explicitly specified for a given column. The warning displays the default method which is used.

### 3. Specific Instances

While *dpart* allows for a wide range of flexibility and customization, we configure and present three specific instances of the framework, which could easily be imported with a single line of code. They are also summarized in Tab. 2.

**Independent.**

This specific use case models all columns independently by using **DP histogram sampler**. The model has also been used as a baseline by (Tao et al., 2021; Stadler et al., 2022) and while it looks very simple and naive, it has been shown that it could perform better than far more sophisticated models. The dependency graph is presented in Fig. 2(a). The code excerpt below demonstrates how one could initiate, fit *Independent*, and generate 1,000 rows for given privacy budget, dataset, and dataset bounds:

```python
>>> from dpart.engines import Independent
>>> dpart_ind = Independent(epsilon, bounds).fit(X)
>>> synth_df = dpart_ind.generate(1000)
```

**PrivBayes.**

**PrivBayes** could also be seen as a sub case of *dpart*. We speed up the implementation offered by (Ping et al., 2017) by 20x by re-implementing the dependency-inference step. Further performance improvements could be achieved by proposing alternative, more efficient dependency-inference approaches. A possible dependency graph produced by **PrivBayes** is shown in Fig. 2(b), while a code example could be found below:

```python
>>> from dpart.engines import PrivBayes
>>> dpart_pb = PrivBayes(epsilon, bounds).fit(X)
>>> synth_df = dpart_pb.generate(1000)
```

**dp-synthpop.**

Yet another instance of our framework, alas not DP, is synthpop. The original model expects either an explicit depen-
differentially private autoregressive tabular (dpart) models. For the dependency step, we achieve DP by utilizing the automatic “infer” option in our framework, i.e., we iteratively select the column maximizing the mutual information with the already chosen columns in a noisy way using the Exponential mechanism. As for estimating the conditional distributions, we rely on the DP predictive models from diffprivlib. A possible dependency graph is visualized in Fig. 2(c) and a call to the model is presented below:

```python
>>> from dpart.engine import DPsynthpop
>>> dpart_dpsp = DPsynthpop(epsilon, bounds=X_bounds).fit(X)
>>> synth_df = dpart_dpsp.generate(1000)
```

4. Comparison

In order to assess the quality and performance of the specific instances presented in Sec. 3, we run a quick experiment on a simplified version of the Adult dataset (Dua & Graff, 2017) (we choose a subset of the columns). The dataset comes with a binary classification task to predict whether the income of an individual is in excess of $50k given some demographic information.

All models are trained with privacy budgets (or epsilon) ranging from 0.01 to 1,000 (where smaller value means tighter privacy guarantees). For each epsilon value, the models are trained 5 times and for each trained model 5 synthetic datasets of size equivalent to the input are generated, leading to 25 datapoints per model per epsilon value. We evaluate the resulting synthetic datasets from two standard angles: 1) mean marginal similarity (the average marginal distribution similarity across all columns between real and synthetic data) and 2) accuracy (accuracy of a decision tree classifier trained on real/synthetic dataset and evaluated against a held-out test dataset). We demonstrate that even these simple baseline models could achieve competitive results.

4. Comparison

In order to assess the quality and performance of the specific instances presented in Sec. 3, we run a quick experiment on a simplified version of the Adult dataset (Dua & Graff, 2017) (we choose a subset of the columns). The dataset comes with a binary classification task to predict whether the income of an individual is in excess of $50k given some demographic information.

All models are trained with privacy budgets (or epsilon) ranging from 0.01 to 1,000 (where smaller value means tighter privacy guarantees). For each epsilon value, the models are trained 5 times and for each trained model 5 synthetic datasets of size equivalent to the input are generated, leading to 25 datapoints per model per epsilon value. We evaluate the resulting synthetic datasets from two standard angles: 1) mean marginal similarity (the average marginal distribution similarity across all columns between real and synthetic data) and 2) accuracy (accuracy of a decision tree classifier trained on real/synthetic dataset and evaluated against a held-out test dataset). We demonstrate that even these simple baseline models could achieve competitive results.

## Similarity

Overall, looking at the similarity in Fig. 3(a), all models improve as a larger privacy budget is allocated. Interestingly, Independent outperforms the other two models for small privacy budgets (epsilon < 1) but this can be explained by the fact that it is particularly suited to capture marginal distributions and it does not “waste” its budget on irrelevant steps. PrivBayes seems to capture the marginals better than dp-synthpop for all privacy budgets.

## Accuracy

When it comes to accuracy, displayed in Fig. 3(b), both PrivBayes and dp-synthpop are positively correlated with the increase of the privacy budget. PrivBayes approaches the real baseline for epsilon >= 1. dp-synthpop’s underperformance compared to PrivBayes is likely due to the use of DP linear and logistic regressions as underlying methods (while the default method behind the original synthpop is CART) as DP decision tree and DP random forest are significantly slower. Unsurprisingly, Independent does not perform well in this task.

5. Conclusion

In this work, we introduced dpart, a general open source framework for DP synthetic data generation. All readily available specific use cases of our framework (Independent, PrivBayes, and dp-synthpop) are DP allowing for the simple and safe use of a varied set of generative approaches. Furthermore, the dpart interface provides great flexibility ensuring that more complex approaches or customized behaviors and domain knowledge can be easily configured.

The framework can be extended through the implementation of new DP methods, the introduction of new dependency building strategies as well as new readily available instances to ease the use of well-known and documented models. We are looking forward to keep improving on this framework with the help of a growing community of researchers and practitioners.
References

Acs, G., Melis, L., Castelluccia, C., and De Cristofaro, E. Differentially private mixture of generative neural networks. *IEEE TKDE*, 2018.

Aydore, S., Brown, W., Kearns, M., Kenthapadi, K., Melis, L., Roth, A., and Siva, A. A. Differentially private query release through adaptive projection. In *ICML*, 2021.

Benedetto, G., Stanley, J. C., Totty, E., et al. The creation and use of the SIPP synthetic Beta v7.0. *US Census Bureau*, 2018.

Dua, D. and Graff, C. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/adult, 2017.

Holohan, N., Braghin, S., Mac Aonghusa, P., and Levacher, K. Diffprivlib: the IBM differential privacy library. arXiv:1907.02444, 2019.

Jordon, J., Yoon, J., and Van Der Schaar, M. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *ICLR*, 2018.

Jordon, J., Szpruch, L., Houssiau, F., Bottarelli, M., Cherubin, G., Maple, C., Cohen, S. N., and Weller, A. Synthetic data–what, why and how? arXiv:2205.03257, 2022.

Liu, T., Vietri, G., and Wu, S. Z. Iterative methods for private synthetic data: Unifying framework and new methods. *NeurIPS*, 2021.

McKenna, R., Miklau, G., and Sheldon, D. Winning the NIST Contest: A scalable and general approach to differentially private synthetic data. *Journal of Privacy and Confidentiality*, 2021.

NHS England. A&E Synthetic Data. https://data.england.nhs.uk/dataset/a-e-synthetic-data, 2021.

NIST. 2018 Differential Privacy Synthetic Data Challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2018-differential-privacy-synthetic, 2018.

NIST. 2020 Differential Privacy Temporal Map Challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2020-differential-privacy-temporal, 2020.

Nowok, B., Raab, G. M., and Dibben, C. synthpop: Bespoke creation of synthetic data in r. *Journal of Statistical Software*, 2016.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 2011.

Ping, H., Stoyanovich, J., and Howe, B. DataSynthesizer: Privacy-Preserving Synthetic Datasets. In *SSDBM*, 2017.

Stadler, T., Oprisanu, B., and Troncoso, C. Synthetic Data – Anonymization Groundhog Day. In *Usenix Security*, 2022.

Tao, Y., McKenna, R., Hay, M., Machanavajjhala, A., and Miklau, G. Benchmarking Differentially Private Synthetic Data Generation Algorithms. arXiv:2112.09238, 2021.

Vietri, G., Tian, G., Bun, M., Steinke, T., and Wu, S. New oracle-efficient algorithms for private synthetic data release. In *ICML*, 2020.

Xie, L., Lin, K., Wang, S., Wang, F., and Zhou, J. Differentially private generative adversarial network. arXiv:1802.06739, 2018.

Zhang, J., Cormode, G., Procopiuc, C. M., Srivastava, D., and Xiao, X. Privbayes: Private data release via bayesian networks. *ACM TODS*, 2017.

Zhang, Z., Wang, T., Li, N., Honorio, J., Backes, M., He, S., Chen, J., and Zhang, Y. Privsyn: Differentially private data synthesis. In *USENIX Security*, 2021.