HALMSTAD UNIVERSITY

Synthetic Data Vault

Emmanuella Budu

2024-02-26

Overview

- What is SDV
- Data Preparation
- Modelling
- Sampling
- Evaluation

The need for synthetic data

Why would you want synthetic data?

Is synthetic data the hot new thing or an act of desperation?

Cassie Kozyrkov · Follow

3 min read · Jul 21, 2023

IDEAS MADE TO MATTER | DATA

What is synthetic data — and how can it help you competitively?

by **Brian Eastwood** | Jan 23, 2023

What Is Synthetic Data Generation and Why Is It Useful

Nikolaj Buhl • July 25, 2023 • 6 min read

Synthetic data could be better than real data

Machine-generated data sets have the potential to improve privacy and representation in artificial intelligence, if researchers can find the right balance between accuracy and fakery.

By Neil Savage

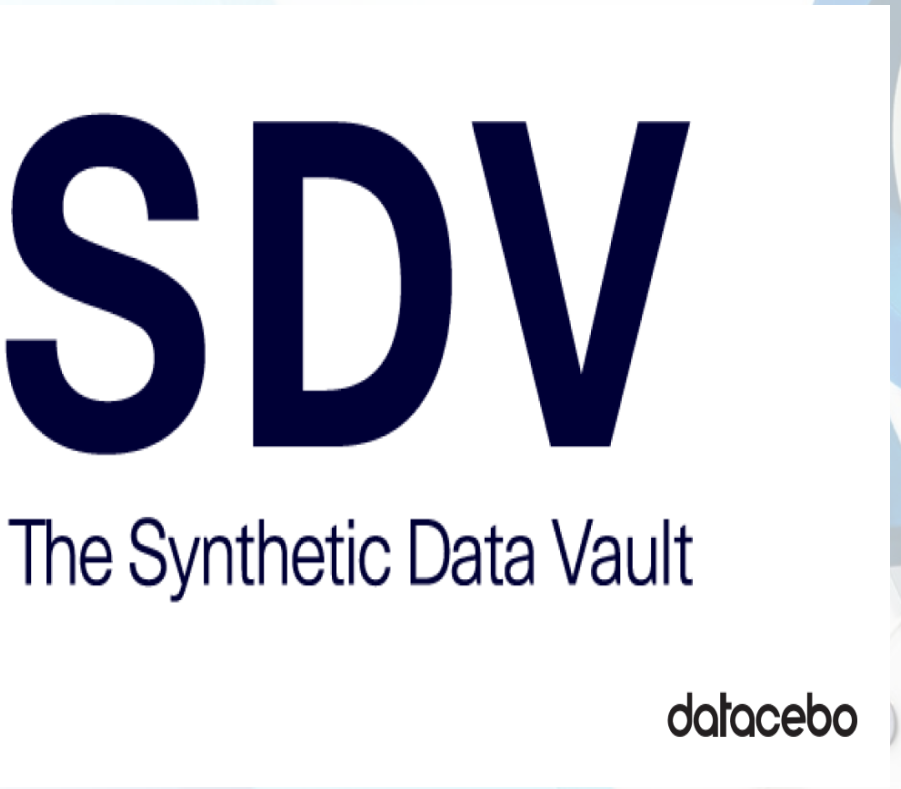
What and Why?

Data generated based on the statistical and structural properties of real data.

- Data Augmentation
- Privacy preservation
- Software testing

What is SDV

- Python library for creating synthetic data
- An ecosystem of synthetic data tools comprising data models, benchmark models and evaluation metrics.



https://docs.sdv.dev/sdv

HALMSTAD UNIVERSITY

Key Features

🧠 Train your own Generative AI Model

Choose from a variety of AI models meant for tabular data. Browse options for single table and multi-table (relational) data.

📊 Evaluate & Visualize Synthetic Data

Diagnose problems and measure statistical quality. For even more insight, visualize synthetic vs. real data.

⚙️ Customize your Synthesizer

Add business logic, control data the data pre-processing rules, and select anonymization options for sensitive values.

HALMSTAD UNIVERSITY

DATA PREPARATION

Data Types

Single Table Data

guest_email	has_rewards	room_type	amenities_fee	checkin_date	checkout_date	room_rate	billing_address	credit_card_number
michaelsanders@shaw.net	False	BASIC	37.89	27 Dec 2020	29 Dec 2020	131.23	49380 Rivers Street Spencerville, AK 68265	4075084747483975747
randy49@brown.biz	False	BASIC	24.37	30 Dec 2020	02 Jan 2021	114.43	88934 Boyle Meadows Conleyberg, TN 22063	180072822063468
webermelissa@neal.com	True	DELUXE	0.00	17 Sep 2020	18 Sep 2020	368.33	0323 Lisa Station Apt. 208 Port Thomas, LA 82585	38983476971380
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Multi-Table Data

table: hotels

hotel_id	city	rating
HID_000	Boston	4.8
HID_001	Boston	4.1
HID_002	San Francisco	3.8
⋮	⋮	⋮

table: guests

guest_email	hotel_id	checkin_date	checkout_date	room_type
awolf@phillips.com	HID_001	27 Dec 2020	28 Dec 2020	BASIC
harriskathleen@goodwin.com	HID_001	30 Dec 2020	02 Jan 2021	DELUXE
fclark@mayo.com	HID_000	17 Sep 2020	19 Sep 2020	BASIC
brenda70@miller.com	HID_002	05 Apr 2020	10 Apr 2020	SUITE
⋮	⋮	⋮	⋮	⋮

Sequential Data

Patient ID	Address	Smoker	Time	Heart Rate	Systolic BP
ID_000	41 King St.	True	01/02/2020	ELEVATED	110
ID_000	41 King St.	True	01/14/2020	ELEVATED	108
ID_000	41 King St.	True	01/24/2020	RESTING	119
...	...	...	...	...	...
ID_001	371 3rd Ave.	False	01/03/2020	RESTING	120
ID_001	371 3rd Ave.	False	01/23/2020	Missing	118
...	...	...	...	...	...

Loading Data: Demo data

- The SDV library offers a variety of demo datasets for users to begin with.
- Use the demo module to access the different datasets

```
from sdv.datasets.demo import get_available_demos

get_available_demos(modality='single_table')
```

```
dataset_name        size_MB         num_tables
adult               3.6             1
alarm               4.6             1
census              141.2           1
...                 ...             ...
```

Loading Data: Demo data

- Specify and load a particular dataset

```
from sdv.datasets.demo import download_demo

data, metadata = download_demo(
    modality='single_table',
    dataset_name='fake_hotel_guests'
)
```

Loading Data: Local data

- Use this method to load any datasets that are stored as CSVs

```
from sdv.datasets.local import load_csvs

# assume that my_folder contains a CSV file named 'guests.csv'
datasets = load_csvs(
    folder_name='my_folder/',
    read_csv_parameters={
        'skipinitialspace': True,
        'encoding': 'utf_32'
    })
```

Loading Data: Other Formats

- SDV relies on pandas for data manipulation and synthesis.
- Data stored in other formats must be loaded as pandas DataFrame objects before using it with SDV.

```
import pandas as pd

data = pd.read_excel('file://localhost/path/to/table.xlsx')
```

Metadata

- Demo datasets have predefined metadata
- SDV allows you to specify the metadata for your local files
- Metadata can be manually specified or automatically detected

```
from sdv.metadata import SingleTableMetadata

metadata = SingleTableMetadata()
```

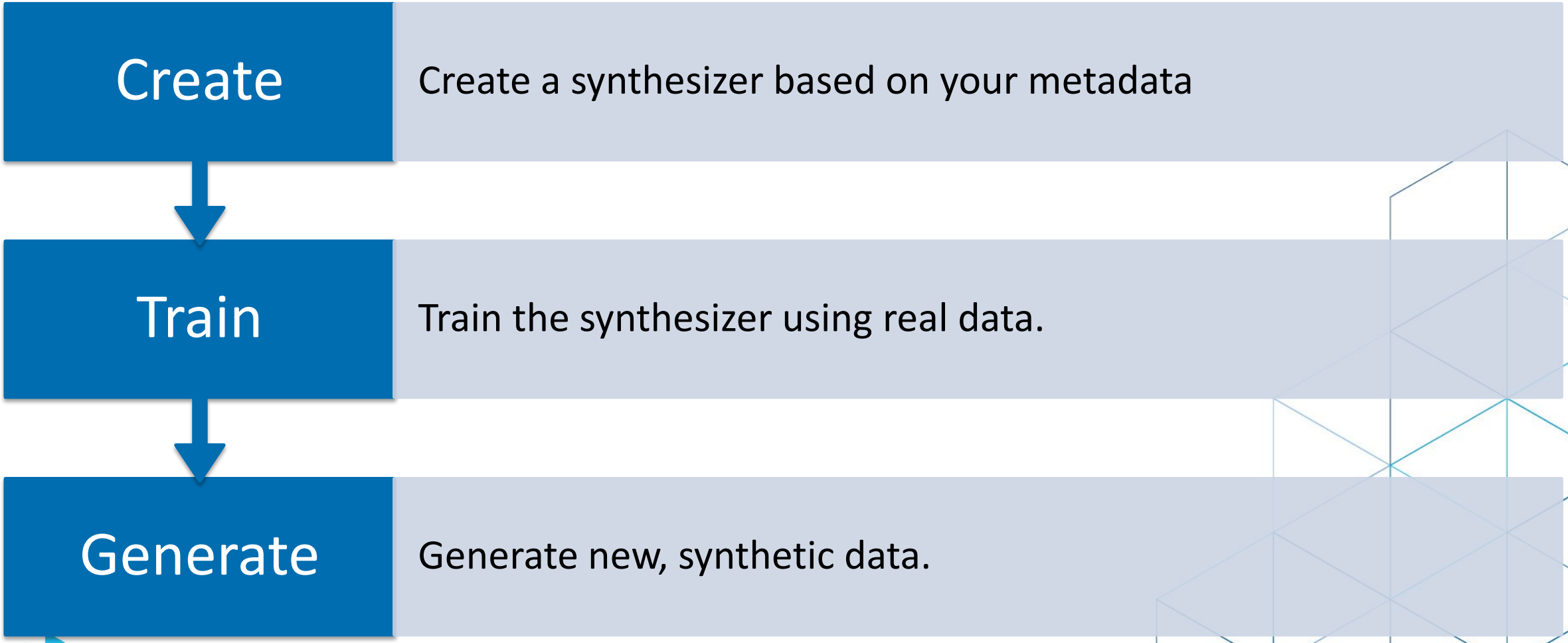
Pre-processing

- SDV provides reversible functionalities to pre-process your data through the transformer module
- Change data types
- Impute missing values
- Encode categorical variables
- Anonymize sensitive data

HALMSTAD UNIVERSITY

MODELLING

Modelling



HALMSTAD UNIVERSITY

Modelling: CTGAN

- **C**onditional **T**abular **G**enerative **A**dversarial **N**etwork
- Uses a conditional GAN based model
- Parameters:

  enforce_min_max_values

  enforce_rounding

  verbose

  epochs

  cuda
- Methods

  fit: train a model on data

  get_loss_values: access the loss values computed during each epoch and batch.

```python
from sdv.single_table import CTGANSynthesizer

synthesizer = CTGANSynthesizer(metadata)
synthesizer.fit(data)

synthetic_data = synthesizer.sample(num_rows=10)
```

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling tabular data using conditional GAN. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 659, 7335–7345

Modelling: TVAE

- **T**abular **V**ariational **A**uto**E**ncoder
- Uses a VAE-based model
- Parameters:

  enforce_min_max_values

  enforce_rounding

  epochs

  cuda
- Methods

  fit: train a model on data

  get_loss_values: access the loss values computed during each epoch and batch.

```python
from sdv.single_table import TVAESynthesizer

synthesizer = TVAESynthesizer(metadata)
synthesizer.fit(data)

synthetic_data = synthesizer.sample(num_rows=10)
```

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling tabular data using conditional GAN. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 659, 7335–7345

Saving and Loading a Model

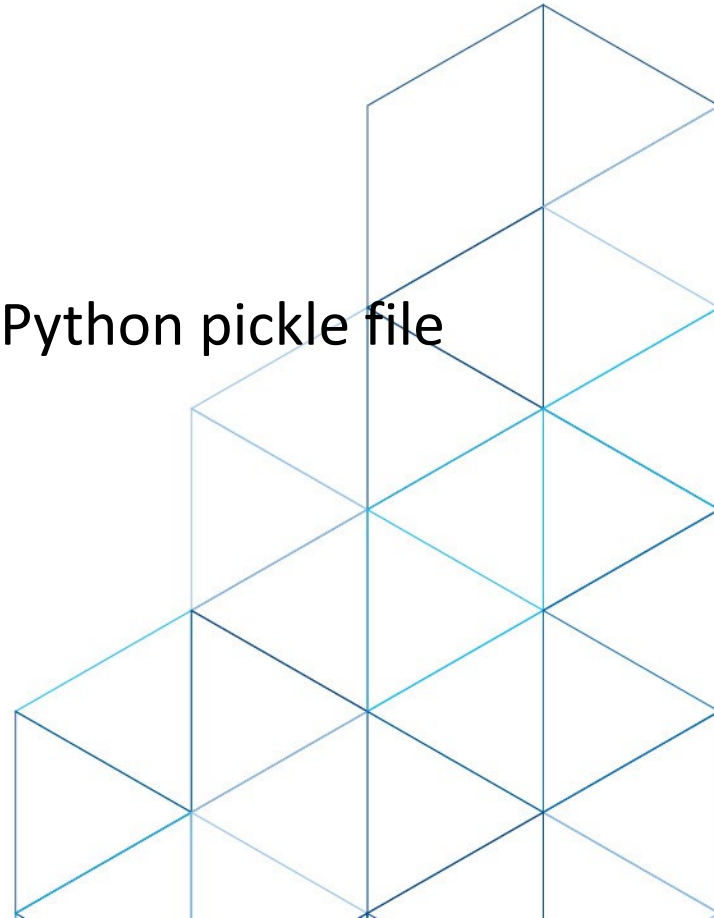
- SDV provides functions to save your trained synthesizer as a Python pickle file

```
synthesizer.save(
    filepath='my_synthesizer.pkl'
)
```

- In the same way, you can also load a trained synthesizer from a Python pickle file

```
from sdv.single_table import CTGANSynthesizer

synthesizer = CTGANSynthesizer.load(
    filepath='my_synthesizer.pkl'
)
```



HALMSTAD UNIVERSITY

Model Customizations

- Functionalities for hyperparameter tuning

batch_size: Number of data samples to process in each step. This value must be even, and it must be divisible by the `pac` parameter (see below). Defaults to `500`.

compress_dims: Size of each hidden layer in the encoder. Defaults to `(128, 128)`.

decompress_dims: Size of each hidden layer in the decoder. Defaults to `(128, 128)`.

embedding_dim: Size of the random sample passed to the Generator. Defaults to `128`.

batch_size: Number of data samples to process in each step. This value must be even, and it must be divisible by the `pac` parameter (see below). Defaults to `500`.

discriminator_dim: Size of the output samples for each one of the Discriminator Layers. A Linear Layer will be created for each one of the values provided. Defaults to `(256, 256)`.

discriminator_decay: Discriminator weight decay for the Adam Optimizer. Defaults to `1e-6`.

discriminator_lr: Learning rate for the discriminator. Defaults to `2e-4`.

discriminator_steps: Number of discriminator updates to do for each generator update. Default `1` to match the original CTGAN implementation

embedding_dim: Size of the random sample passed to the Generator. (Default `128`)

generator_decay: Generator weight decay for the Adam Optimizer. Defaults to `1e-6`

generator_dim: Size of the output samples for each one of the Residuals. A Residual Layer will be created for each one of the values provided. Defaults to `(256, 256)`.

generator_lr: Learning rate for the generator. Defaults to `2e-4`.

HALMSTAD UNIVERSITY

HALMSTAD UNIVERSITY

SAMPLING

Sampling

- Create new data
- **num_rows**: number of rows to synthesize (required)
- **batch_size**: number of rows to sample at a time. Defaults to the same as num_rows.
- **output_file_path**: filepath for writing the synthetic data.

```
synthetic_data = synthesizer.sample(
    num_rows=1_000_000,
    batch_size=1_000
)
```

Conditional Sampling

- SDV provides functionalities to sample specific values in the synthetic data
- First, define conditions
- Sample with conditions

```
from sdv.sampling import Condition

suite_guests_with_rewards = Condition(
    num_rows=250,
    column_values={'room_type': 'SUITE', 'has_rewards': True}
)

suite_guests_without_rewards = Condition(
    num_rows=250,
    column_values={'room_type': 'SUITE', 'has_rewards': False}
)
```

```
synthetic_data = custom_synthesizer.sample_from_conditions(
    conditions=[suite_guests_with_rewards, suite_guests_without_reward
    output_file_path='synthetic_simulated_scenario.csv'
)
```

HALMSTAD UNIVERSITY

EVALUATION

Evaluation: Diagnostics

- Run some basic checks for data format and validity. Ensures that the synthetic data is valid.

```
from sdv.evaluation.single_table import run_diagnostic

diagnostic_report = run_diagnostic(
    real_data=real_data,
    synthetic_data=synthetic_data,
    metadata=metadata)
```

```
Generating report ...
(1/2) Evaluating Data Validity: : 100%|██████████| 17/17 [00:00<00:00,
(2/2) Evaluating Data Structure: : 100%|██████████| 1/1 [00:00<00:00,

Overall Score: 100.0%

Properties:
- Data Validity: 100.0%
- Data Structure: 100.0%
```

Evaluation: Data Quality

- Check statistical similarity between the real and the synthetic data.

```
from sdv.evaluation.single_table import evaluate_quality

quality_report = evaluate_quality(
    real_data=real_data,
    synthetic_data=synthetic_data,
    metadata=metadata)
```

Evaluation: Visualization

- Visualize real and generated data in 1-2 dimensions
- Offers column plots and pair plots

```
from sdv.evaluation.single_table import get_column_plot

fig = get_column_plot(
    real_data=real_data,
    synthetic_data=synthetic_data,
    metadata=metadata,
    column_name='amenities_fee'
)

fig.show()
```

```
from sdv.evaluation.single_table import get_column_pair_plot

fig = get_column_pair_plot(
    real_data=real_data,
    synthetic_data=synthetic_data,
    metadata=metadata,
    column_names=['room_rate', 'room_type'],
    )

fig.show()
```

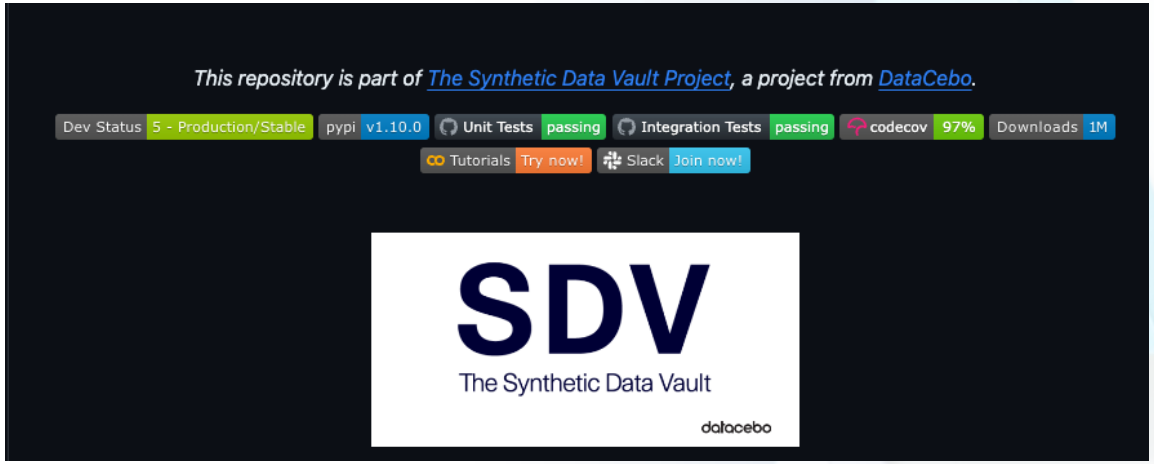
Advanced features

- Anonymizing Personally Identifiable Information (PII)
- Setting Bounds and Specifying Rounding for Numerical Columns
- Setting Distributions for Individual Variables
- Setting constraints on modelling

Where to get help

slack

See what sdv is up to



https://github.com/sdv-dev/SDV/issues

HALMSTAD UNIVERSITY

HALMSTAD UNIVERSITY

The End